

Bài 5: Phân tích cú pháp (Parsing)

About the lecturer

- Phd Nguyen Kiem Hieu
- Computer science department, School of Information and Communication Technology, HUST
- Email: hieunk@soict.hust.edu.vn

Nội dung

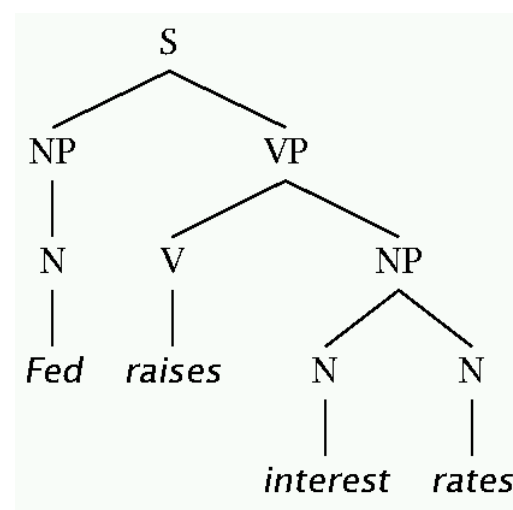
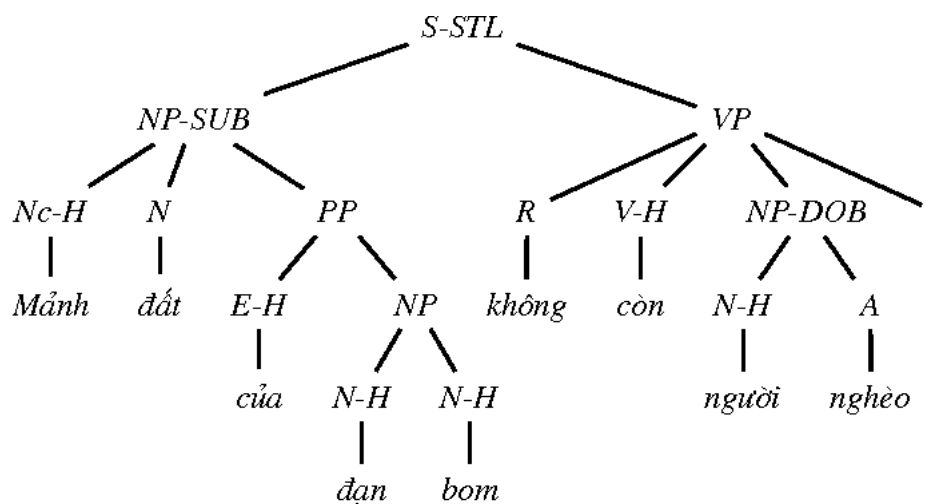
1. Phân tích cú pháp là gì
2. Tiếp cận phân tích cú pháp thành phần và phụ thuộc
3. Ý nghĩa và ứng dụng của phân tích cú pháp
4. Văn phạm CFG và phân tích cú pháp
5. Văn phạm PCFG
6. Cấu trúc phụ thuộc và phân tích

Luyện tập và ứng dụng

1. Sử dụng thư viện trích chọn thông tin cú pháp
2. Phân tích cú pháp tiếng Anh và kiểm lỗi

Phân tích cú pháp là gì?

- Mục tiêu của phân tích cú pháp là xác định thông tin về cấu trúc của một câu, tức là xác định mối quan hệ giữa các thành phần trong câu.
- Structure of a sentence, it means the syntactic information.



Hai tiếp cận trong phân tích cú pháp

- Phân tích cú pháp thành phần (cú pháp cấu trúc cụm): **Constituency parsing = Phrase structure grammar**
- Phân tích cú pháp phụ thuộc: **Dependency parsing**

Phân tích thành phần (cấu trúc cụm)

Constituency = grammar phrase structure

- **Starting unit: words**

the, cat, cuddly, by, door

- **Words combine into phrases**

the cuddly cat, by the door

- **Phrases can combine into bigger phrases**

- the cuddly cat by the door

Phân tích thành phần (cấu trúc cụm)

Constituency = grammar phrase structure

Phrase structure organizes words into nested constituents

Can represent the grammar with CFG rules

Starting unit: words are given a category (part of speech = pos)

the,	cat,	cuddly,	by,	door
Det	N	Adj	P	N

Words combine into phrases with categories

the cuddly cat,	by the door
$NP \rightarrow Det\ Adj\ N$	$PP \rightarrow P\ NP$

Phrases can combine into bigger phrases recursively

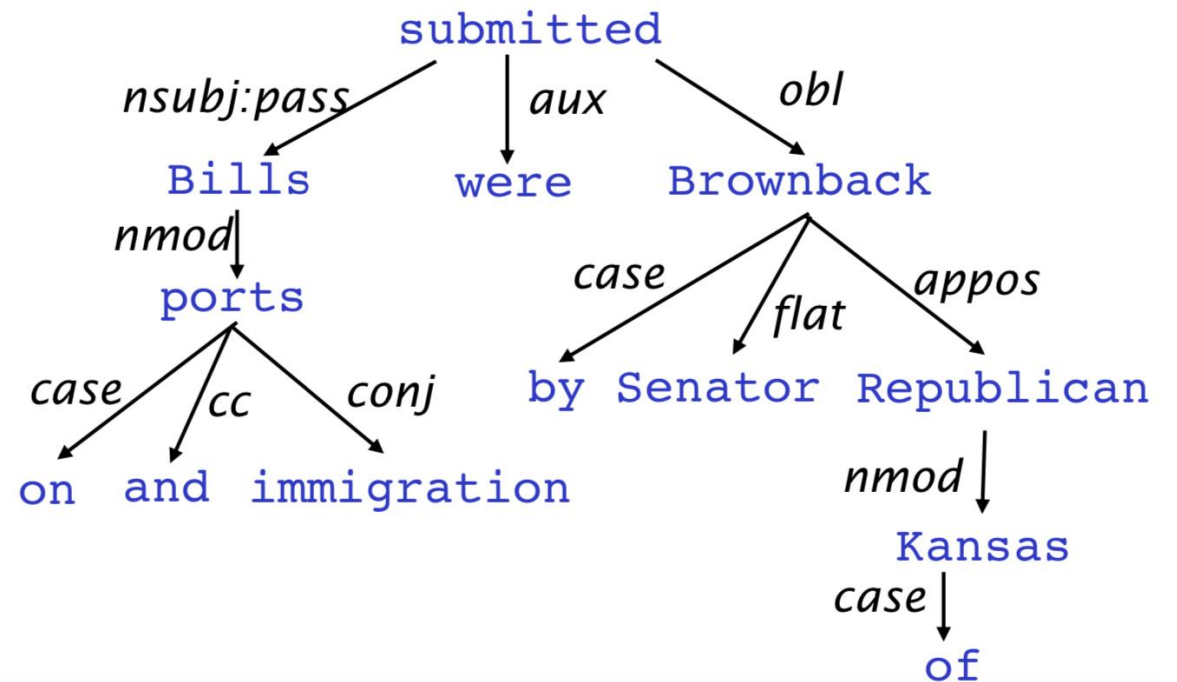
the cuddly cat by the door

$$NP \rightarrow NP\ PP$$

Cú pháp phụ thuộc và cấu trúc phụ thuộc (dependency grammar and dependency structure)

- Dependency syntax postulates that syntactic structure consists of relations between lexical items, normally binary asymmetric relations (“arrows”) called dependencies

- The arrows are commonly **typed** with the name of **ports** grammatical relations (subject, prepositional object, apposition, etc.)



Ý nghĩa và ứng dụng của PTCP

- Lấy thông tin cú pháp và ứng dụng cho bài toán hiểu ngôn ngữ
- Cấu trúc tiền đề cho phân tích ngữ nghĩa
- Ứng dụng:
 - Phân tích câu hỏi trong Question Answering
 - Machine Dịch trong tiếp cận dựa trên Rule
 - Các ứng dụng trích chọn thông tin (information extraction)
 - Kiểm lỗi ngữ pháp (grammar checker)

Thách thức trong phân tích cú pháp (Challenge in parsing)

- Đối với ngôn ngữ tự nhiên, phân tích cú pháp có tính nhập nhằng rất cao.
- Ví dụ:

I saw a man on a hill with a telescope.

Thách thức trong phân tích cú pháp (Challenge in parsing)

- Đối với ngôn ngữ tự nhiên, phân tích cú pháp có tính nhập nhằng rất cao.
- Ví dụ:

I saw a man on a hill with a telescope
N V Det N In Det N In Det N

1. There's a man on a hill, and I'm watching him with my telescope.
2. There's a man on a hill, who I'm seeing, and he has a telescope.
3. There's a man, and he's on a hill that also has a telescope on it.
4. I'm on a hill, and I saw a man using a telescope.
5. There's a man on a hill, and I'm sawing him with a telescope.

Parsing

Context Free Grammar
and Syntactic Parsing

CFG and Constituency Parsing

- Văn phạm phi ngữ (CFG) cảnh biểu diễn cấu trúc cụm
- Cây cú pháp và luật
- Thuật toán phân tích CKY
- Thuật toán phân tích Earley

A Phrase Structure Grammar

$S \rightarrow NP VP$

$VP \rightarrow V NP$

$VP \rightarrow V NP PP$

$NP \rightarrow NP NP$

$NP \rightarrow NP PP$

$NP \rightarrow N$

$NP \rightarrow e$

$PP \rightarrow P NP$

$N \rightarrow \text{people}$

$N \rightarrow \text{fish}$

$N \rightarrow \text{tanks}$

$N \rightarrow \text{rods}$

$V \rightarrow \text{people}$

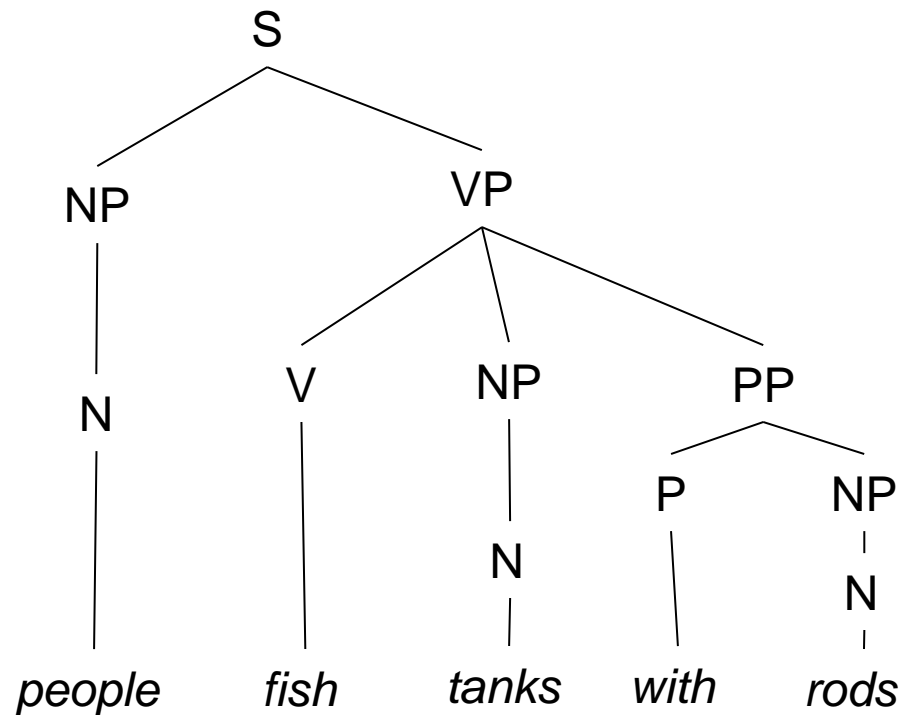
$V \rightarrow \text{fish}$

$V \rightarrow \text{tanks}$

$P \rightarrow \text{with}$

Sentence: "people fish with rods"

Phân tích cú pháp



$S \rightarrow NP VP$

$VP \rightarrow V NP$

$VP \rightarrow V NP PP$

$NP \rightarrow NP NP$

$NP \rightarrow NP PP$

$NP \rightarrow N$

$NP \rightarrow e$

$PP \rightarrow P NP$

$N \rightarrow \text{people}$

$N \rightarrow \text{fish}$

$N \rightarrow \text{tanks}$

$N \rightarrow \text{rods}$

$V \rightarrow \text{people}$

$V \rightarrow \text{fish}$

$V \rightarrow \text{tanks}$

$P \rightarrow \text{with}$

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 .

Phân tích cú pháp

- Là bài toán với văn phạm đã cho và một câu input, sinh cây cú pháp (parsed tree).
- Các vấn đề cần giải quyết:
 - Xây dựng văn phạm.
 - Thuật toán phân tích (time complexity)
 - Giải quyết tính nhập nhằng trong phân tích cú pháp.

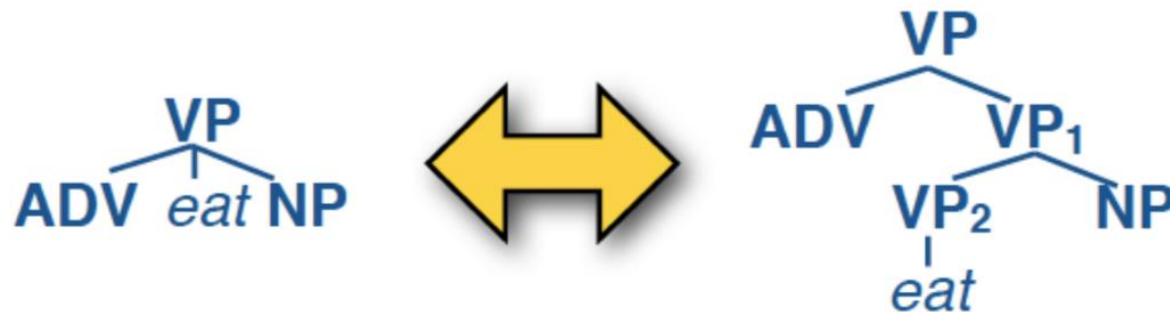
Các tiếp cận phân tích cú pháp

- Top-down vs. bottom-up:
 - Top-down: (goal-driven): from the start symbol down.
 - Bottom-up: (data-driven): from the symbols up.
- Naive vs. dynamic programming:
 - Naive: enumerate everything.
 - Backtracking: try something, discard partial solutions.
 - Dynamic programming: save partial solutions in a table.
- Examples:
 - CKY: bottom-up dynamic programming.
 - Earley parsing: top-down dynamic programming.

Thuật toán CYK và văn phạm chuẩn Chomsky (Chomsky Normal Form)

- Luật trong CNF chỉ thuộc một trong 2 dạng:
 - Two non-terminals: (e.g., $VP \rightarrow ADVVP$)
 - One terminal: $VP \rightarrow eat$
- Mỗi luật CFG có thể viết về dạng tương đương của luật CNF

Ví dụ:

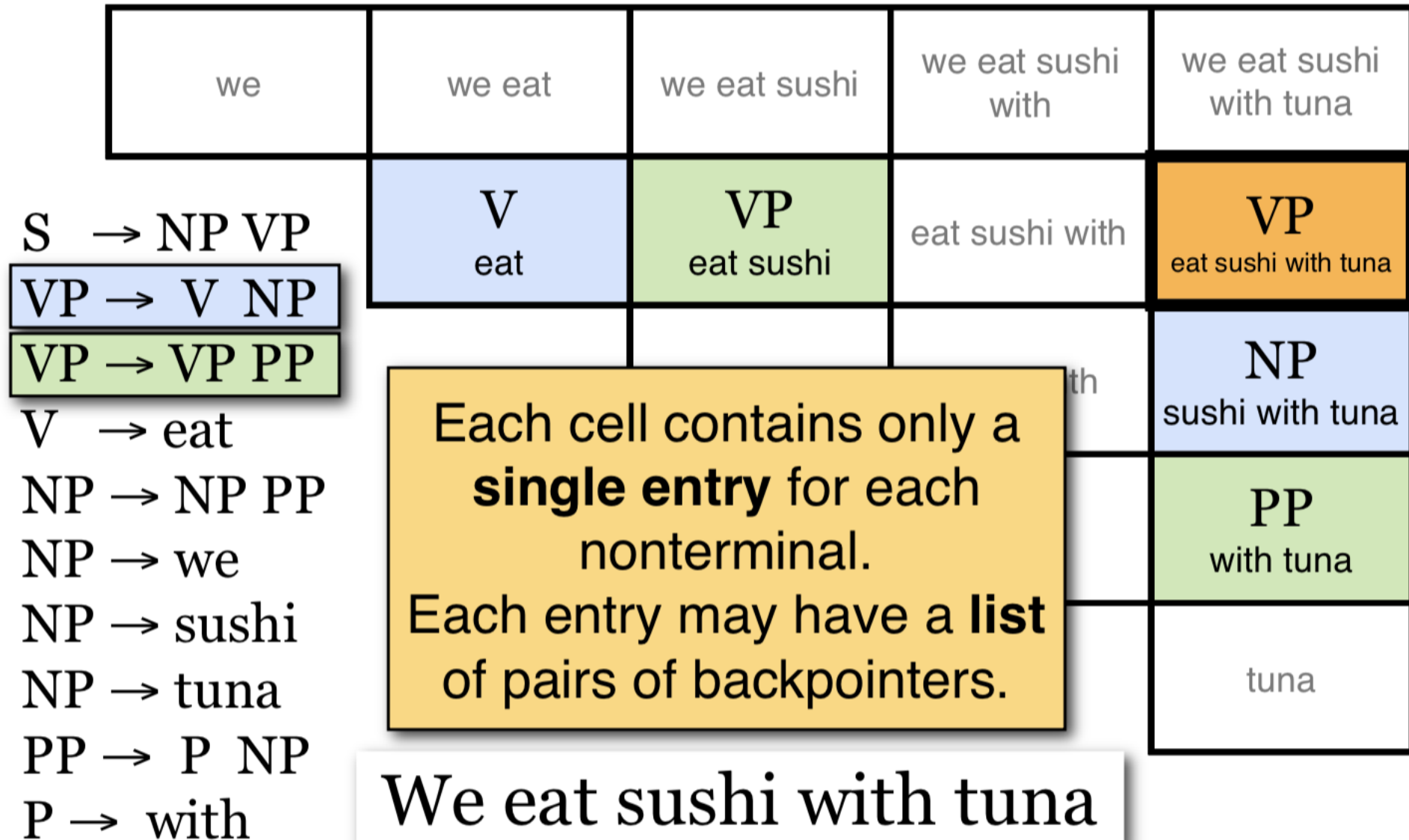


Chuyển về dạng CNF?: $VP \rightarrow VBD\ NP\ PP\ PP$

CKY chart parsing algorithm

- Bottom-up parsing: start with the words
- Dynamic programming:
 - save the results in a table/chart
 - re-use these results in finding larger constituents
- Complexity: $O(n^3 / |G|)$
 n : length of string, $|G|$: size of grammar)

Ví dụ



Bài tập 1

$$S \rightarrow NP VP$$
$$VP \rightarrow V NP$$
$$VP \rightarrow VP PP$$
$$PP \rightarrow P NP$$
$$NP \rightarrow NP PP$$

I eat sushi with chopsticks

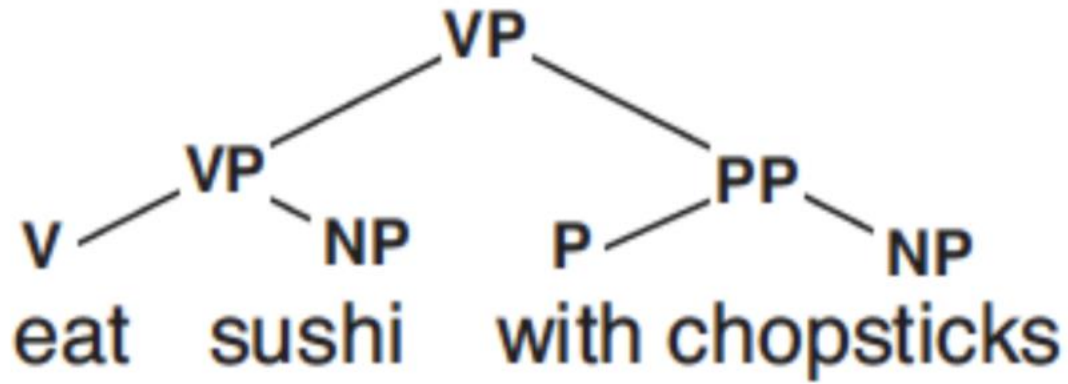
NP V

NP

P

NP

Đáp án



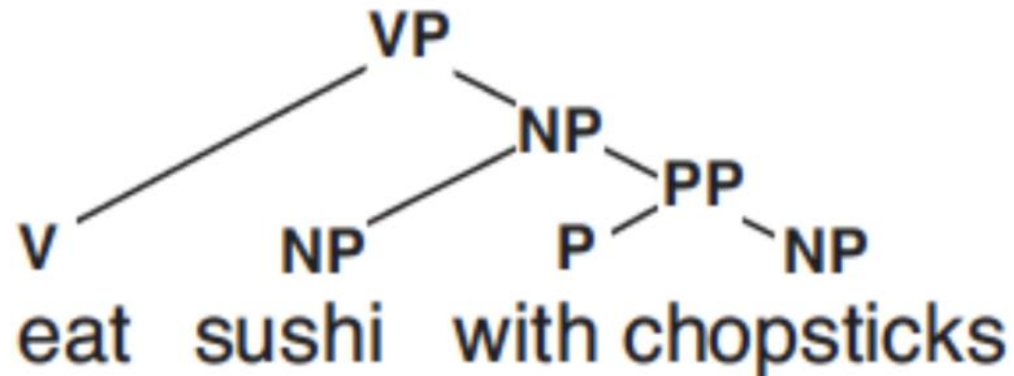
$S \rightarrow NP VP$

$VP \rightarrow V NP$

$VP \rightarrow VP PP$

$PP \rightarrow P NP$

$NP \rightarrow NP PP$



Thuật toán CYK

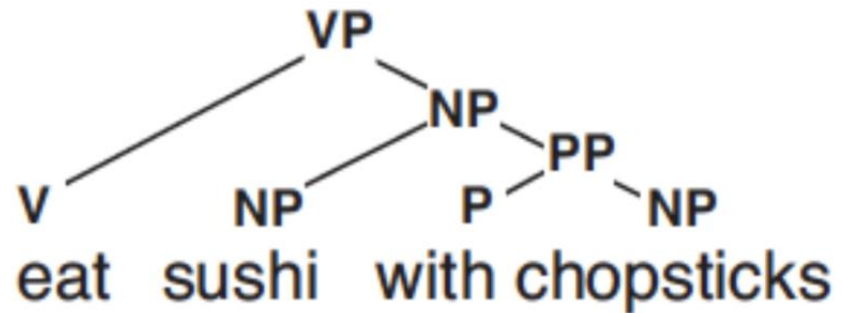
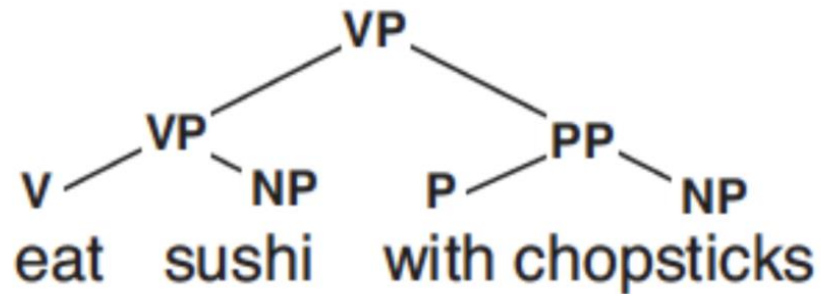
```
CYK (  $\mathcal{G}, w$  )  
     $\mathcal{G} = (\mathcal{V}, \Sigma, \mathcal{R}, S), \Sigma \cup \mathcal{V} = \{X_1, \dots, X_r\}, w = w_1 w_2 \dots w_n$ .  
begin  
    Initialize the 3d array  $B[1 \dots n, 1 \dots n, 1 \dots r]$  to FALSE  
    for  $i = 1$  to  $n$  do  
        for  $(X_j \rightarrow x) \in \mathcal{R}$  do  
            if  $x = w_i$  then  $B[i, i, j] \leftarrow \text{TRUE}$ .  
    for  $i = 2$  to  $n$  do /* Length of span */  
        for  $L = 1$  to  $n - i + 1$  do /* Start of span */  
             $R = L + i - 1$  /* Current span  $s = w_L w_{L+1} \dots w_R$  */  
            for  $M = L + 1$  to  $R$  do /* Partition of span */  
                /*  $x = w_L w_{L+1} \dots w_{M-1}, y = w_M w_{M+1} \dots w_R$ , and  $s = xy$  */  
                for  $(X_\alpha \rightarrow X_\beta X_\gamma) \in \mathcal{R}$  do  
                    /* Can we match  $X_\beta$  to  $x$  and  $X_\gamma$  to  $y$ ? */  
                    if  $B[L, M - 1, \beta]$  and  $B[M, R, \gamma]$  then  
                         $B[L, R, \alpha] \leftarrow \text{TRUE}$  /* If so, then can generate  $s$  by  $X_\alpha$ ! */  
    for  $i = 1$  to  $r$  do  
        if  $B[1, n, i]$  then return TRUE  
return FALSE
```

Parsing

Probability Context Free
Grammar and Syntactic
Parsing

Grammars are ambiguous

A grammar might generate multiple trees for a sentence:



What's the most likely parse τ for sentence S ?

We need a model to compute $P(\tau)$

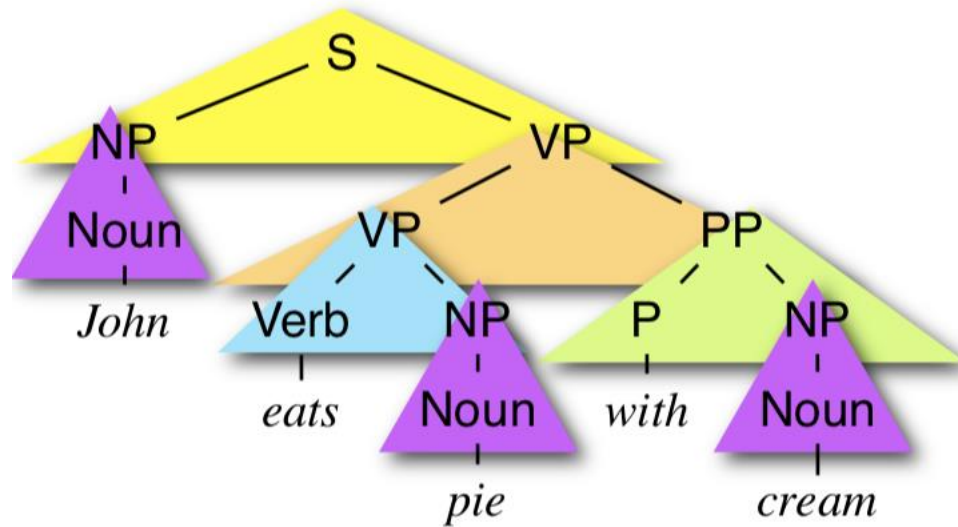
Probabilistic Context-Free Grammars

For every nonterminal X , define a probability distribution $P(X \rightarrow \alpha \mid X)$ over all rules with the same LHS symbol X :

S	\rightarrow	$NP \ VP$	0.8
S	\rightarrow	$S \ conj \ S$	0.2
NP	\rightarrow	$Noun$	0.2
NP	\rightarrow	$Det \ Noun$	0.4
NP	\rightarrow	$NP \ PP$	0.2
NP	\rightarrow	$NP \ conj \ NP$	0.2
VP	\rightarrow	$Verb$	0.4
VP	\rightarrow	$Verb \ NP$	0.3
VP	\rightarrow	$Verb \ NP \ NP$	0.1
VP	\rightarrow	$VP \ PP$	0.2
PP	\rightarrow	$P \ NP$	1.0

Computing $P(\tau)$ with a PCFG

The probability of a tree τ is the product of the probabilities of all its rules:



$$P(\tau) = 0.8 \times 0.3 \times 0.2 \times 1.0 \times 0.2^3$$
$$= 0.00384$$

S	→ NP VP	0.8
S	→ S conj S	0.2
NP	→ Noun	0.2
NP	→ Det Noun	0.4
NP	→ NP PP	0.2
NP	→ NP conj NP	0.2
VP	→ Verb	0.4
VP	→ Verb NP	0.3
VP	→ Verb NP NP	0.1
VP	→ VP PP	0.2
PP	→ P NP	1.0

Ví dụ

Input: POS-tagged sentence

John_N eats_V pie_N with_P cream_N

John	eats	pie	with	cream	
N NP 0.2	S 0.8*0.2*0.4	S 0.8*0.2*0.08		S 0.2*0.0024*0.8	John
	V VP 0.4	VP 0.3*0.2		VP max(0.008*0.2, 0.06*0.2*0.2)	eats
		N NP 0.2		NP 0.2*0.2*0.2	pie
			P	PP 1*0.2	with
				N NP 0.2	cream

S	→	NP VP	0.8
S	→	S conj S	0.2
NP	→	Noun	0.2
NP	→	Det Noun	0.4
NP	→	NP PP	0.2
NP	→	NP conj NP	0.2
VP	→	Verb	0.4
VP	→	Verb NP	0.3
VP	→	Verb NP NP	0.1
VP	→	VP PP	0.2
PP	→	P NP	1.0

Bài tập

$S \rightarrow NP VP$	1.0
$VP \rightarrow V NP$	0.7
$VP \rightarrow VP PP$	0.3
$PP \rightarrow P NP$	1.0
$NP \rightarrow NP PP$	1.0

I eat sushi with chopsticks

NP V

NP

P

NP

Weaknesses of PCFGs

- Lack of sensitivity to lexical information
- Lack of sensitivity to structural frequencies

(a)

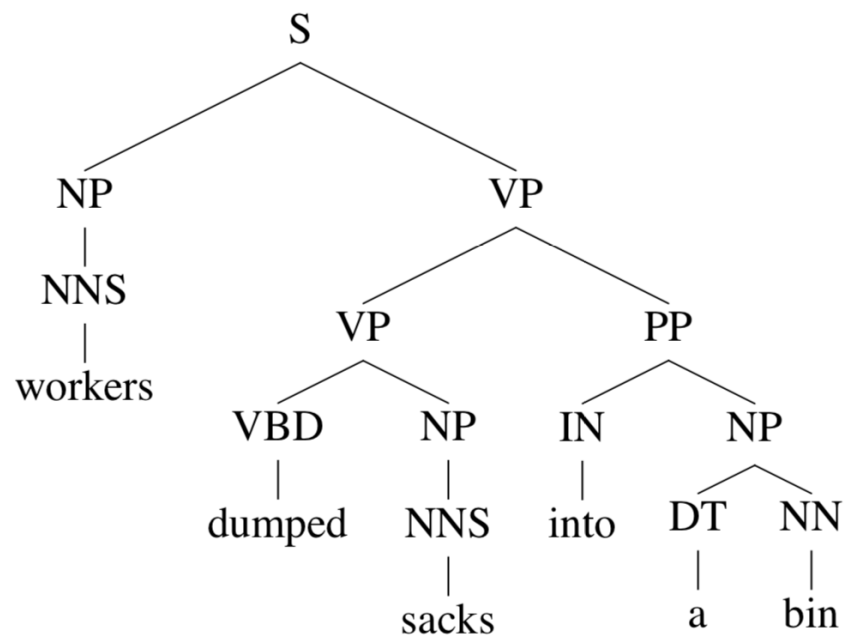
Rules
$S \rightarrow NP VP$
$NP \rightarrow NNS$
$VP \rightarrow VP PP$
$VP \rightarrow VBD NP$
$NP \rightarrow NNS$
$PP \rightarrow IN NP$
$NP \rightarrow DT NN$
$NNS \rightarrow workers$
$VBD \rightarrow dumped$
$NNS \rightarrow sacks$
$IN \rightarrow into$
$DT \rightarrow a$
$NN \rightarrow bin$

(b)

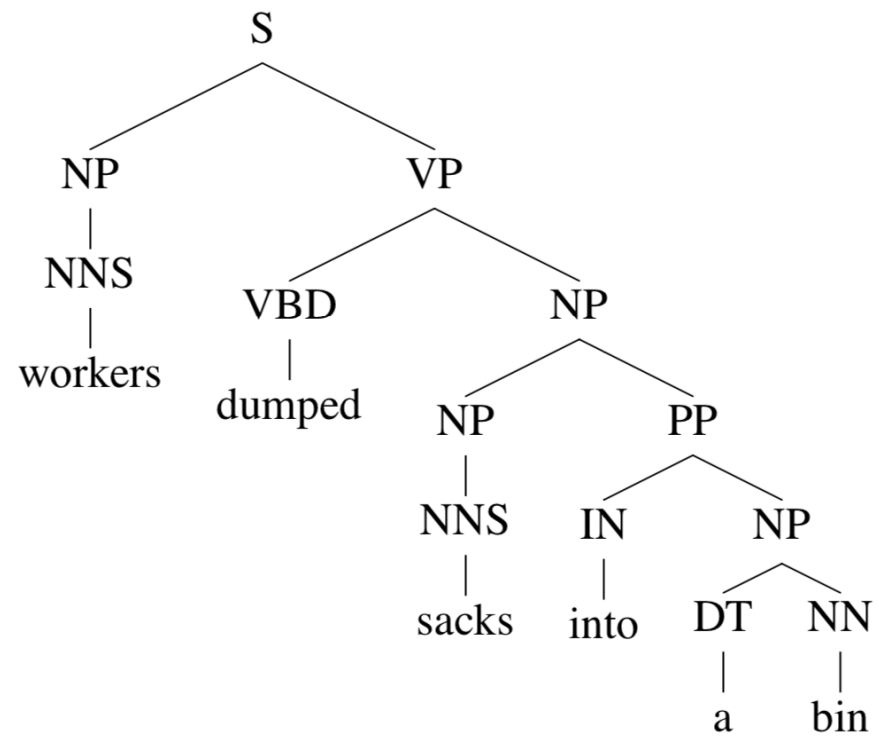
Rules
$S \rightarrow NP VP$
$NP \rightarrow NNS$
$NP \rightarrow NP PP$
$VP \rightarrow VBD NP$
$NP \rightarrow NNS$
$PP \rightarrow IN NP$
$NP \rightarrow DT NN$
$NNS \rightarrow workers$
$VBD \rightarrow dumped$
$NNS \rightarrow sacks$
$IN \rightarrow into$
$DT \rightarrow a$
$NN \rightarrow bin$

If $P(NP \rightarrow NP PP \mid NP) > P(VP \rightarrow VP PP \mid VP)$ then (b) is more probable, else (a) is more probable.

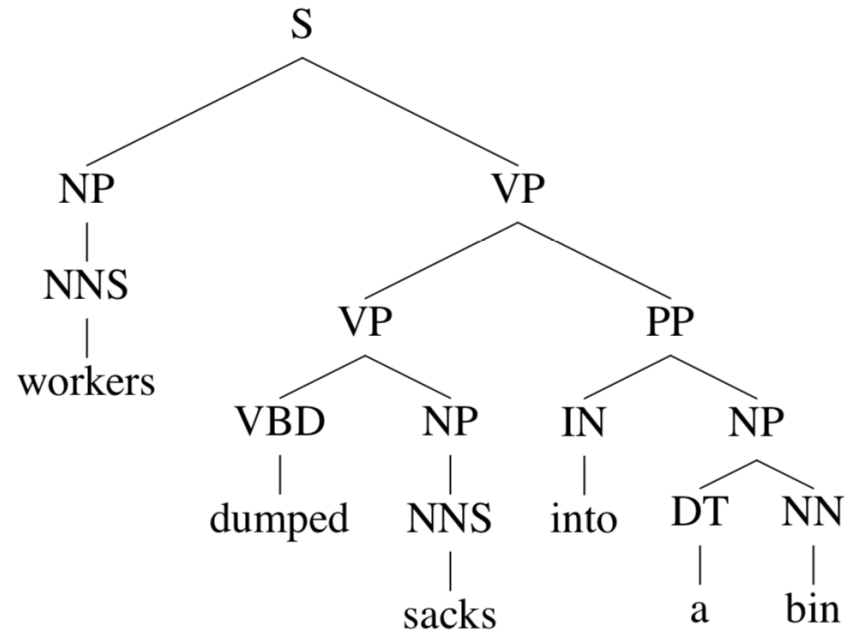
(a)



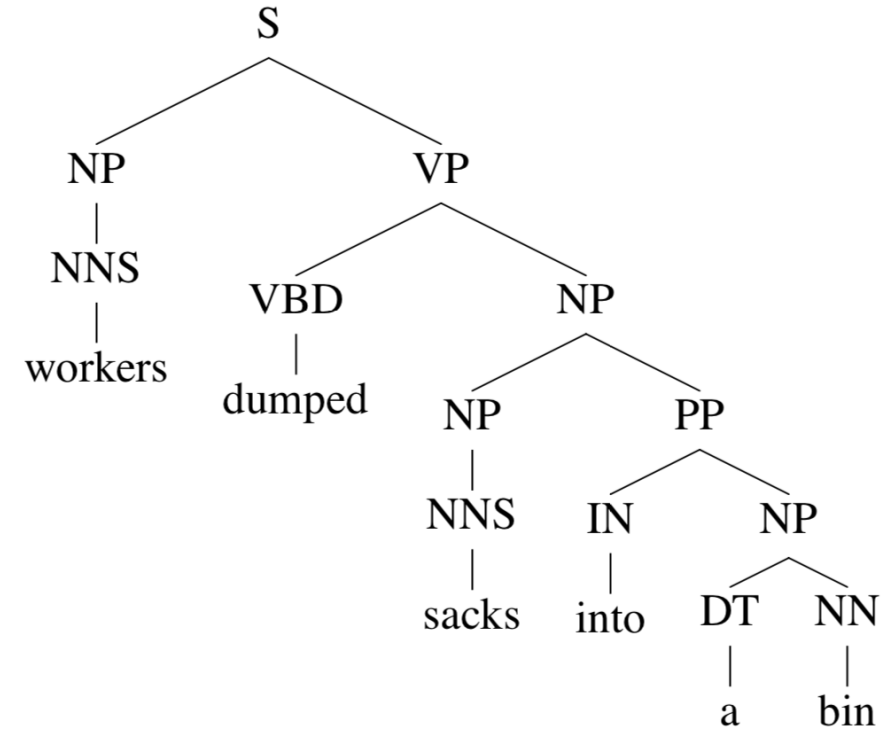
(b)



(a)



(b)



If $P(\text{NP} \rightarrow \text{NP PP} \mid \text{NP}) > P(\text{VP} \rightarrow \text{VP PP} \mid \text{VP})$ then (b) is more probable, else (a) is more probable.

Attachment decision is completely independent of the words

Solution: Head-Driven Phrase Structure Grammar

Add annotations specifying the “**head**” of each rule:

S	⇒	NP	VP
VP	⇒	Vi	
VP	⇒	Vt	NP
VP	⇒	VP	PP
NP	⇒	DT	NN
NP	⇒	NP	PP
PP	⇒	IN	NP

Vi	⇒	sleeps
Vt	⇒	saw
NN	⇒	man
NN	⇒	woman
NN	⇒	telescope
DT	⇒	the
IN	⇒	with
IN	⇒	in

Note: S=sentence, VP=verb phrase, NP=noun phrase, PP=prepositional phrase, DT=determiner, Vi=intransitive verb, Vt=transitive verb, NN=noun, IN=preposition

Head-Driven Phrase Structure Grammar

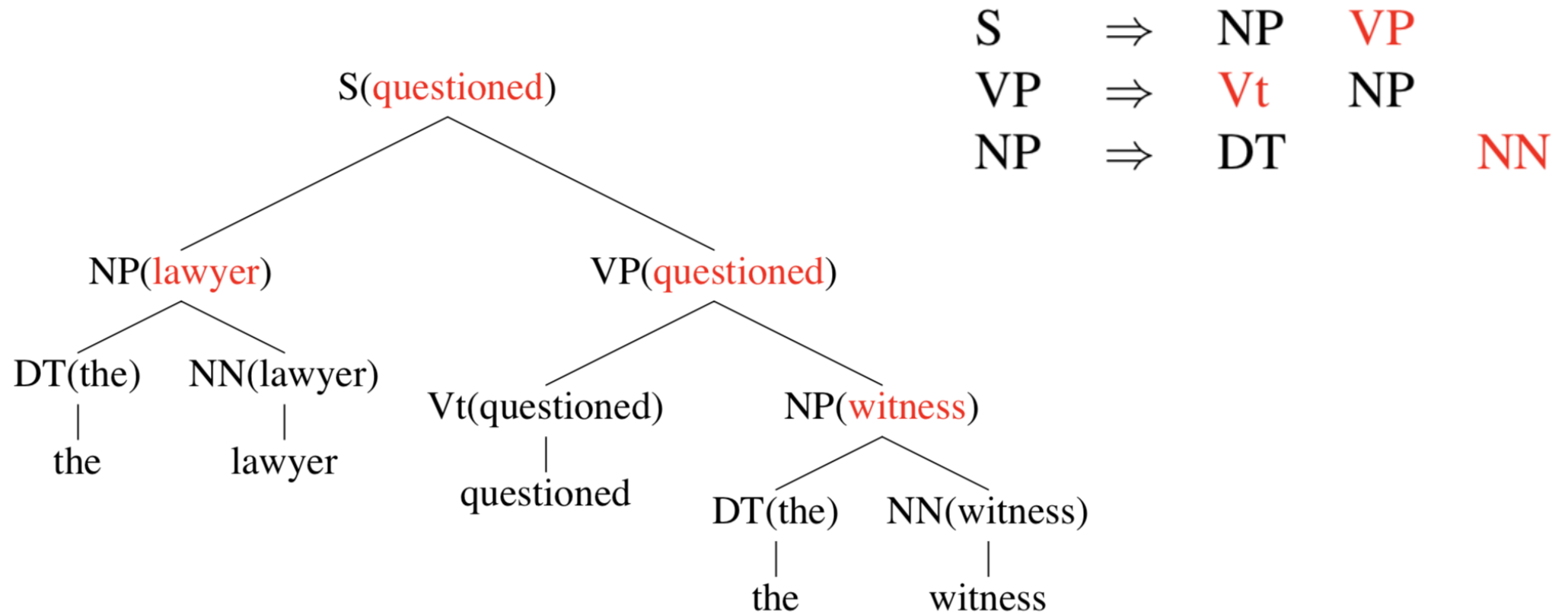
Each context-free rule has one “special” child that is the head of the rule. e.g.,

S	⇒	NP	VP	(VP is the head)	
VP	⇒	Vt	NP	(Vt is the head)	
NP	⇒	DT	NN	NN	(NN is the head)

Some intuitions:

- The central sub-constituent of each rule.
- The semantic predicate in each rule.

Adding Headwords to Trees



A constituent receives its **headword** from its **head child**.

Lexicalized PCFGs

- The basic idea in lexicalized PCFGs will be to replace rules such as

$$S \rightarrow NP VP$$

with lexicalized rules such as

$$S(\text{examined}) \rightarrow NP(\text{lawyer}) VP(\text{examined})$$

Example

In this case the parse tree consists of the following sequence of rules:

$S(\text{questioned}) \rightarrow_2 NP(\text{lawyer}) VP(\text{questioned})$

$NP(\text{lawyer}) \rightarrow_2 DT(\text{the}) NN(\text{lawyer})$

$DT(\text{the}) \rightarrow \text{the}$

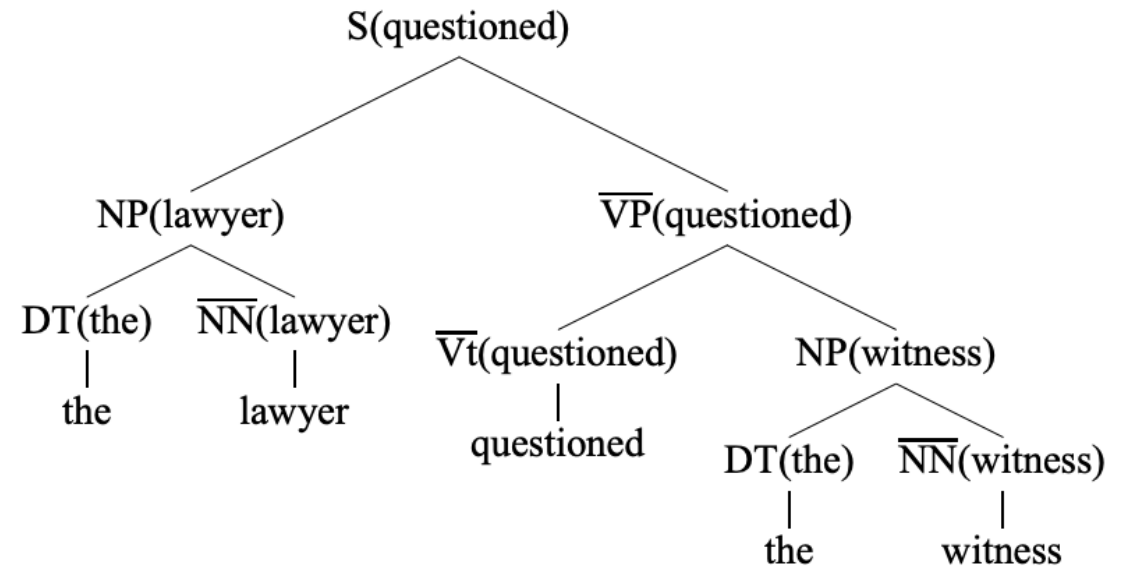
$NN(\text{lawyer}) \rightarrow \text{lawyer}$

$VP(\text{questioned}) \rightarrow_1 \bar{V}t(\text{questioned}) NP(\text{witness})$

$NP(\text{witness}) \rightarrow_2 DT(\text{the}) NN(\text{witness})$

$DT(\text{the}) \rightarrow \text{the}$

$NN(\text{witness}) \rightarrow \text{witness}$



Parameter Estimation in Lexicalized PCFGs

The number of rules (and therefore parameters) in the model is very large. However with appropriate smoothing—using techniques described earlier in the class, for language modeling—we can derive estimates that are robust and effective in practice.

$S(\text{examined}) \rightarrow NP(\text{lawyer}) VP(\text{examined})$

$X = S$

$H = \text{examined}$

$R = S \rightarrow NP VP$

$M = \text{lawyer}$

$$\begin{aligned} & q(S(\text{examined}) \rightarrow_2 NP(\text{lawyer}) VP(\text{examined})) \\ = & P(R = S \rightarrow_2 NP VP, M = \text{lawyer} | X = S, H = \text{examined}) \end{aligned}$$

Parameter Estimation in Lexicalized PCFGs

$$\begin{aligned} & P(R = S \rightarrow_2 \text{NP VP}, M = \text{lawyer} | X = S, H = \text{examined}) \\ = & P(R = S \rightarrow_2 \text{NP VP} | X = S, H = \text{examined}) \quad (3) \\ & \times P(M = \text{lawyer} | R = S \rightarrow_2 \text{NP VP}, X = S, H = \text{examined}) \quad (4) \end{aligned}$$

$$\begin{aligned} q_{ML}(S \rightarrow_2 \text{NP VP} | S, \text{examined}) &= \frac{\text{count}(R = S \rightarrow_2 \text{NP VP}, X = S, H = \text{examined})}{\text{count}(X = S, H = \text{examined})} \\ q_{ML}(S \rightarrow_2 \text{NP VP} | S) &= \frac{\text{count}(R = S \rightarrow_2 \text{NP VP}, X = S)}{\text{count}(X = S)} \end{aligned}$$

Penn Treebank

The Penn Treebank (PTB) project selected 2,499 stories from a three year Wall Street Journal (WSJ) collection of 98,732 stories for syntactic annotation. These 2,499 stories have been distributed in both Treebank-2 ([LDC95T7](#)) and Treebank-3 ([LDC99T42](#)) releases of PTB. Treebank-2 includes the raw text for each story. Three "map" files are available in a compressed file (pennTB_tipster_wsj_map.tar.gz) as an additional download for users who have licensed Treebank-2 and provide the relation between the 2,499 PTB filenames and the corresponding WSJ DOCNO strings in TIPSTER.

```
(S (NP-SBJ I)
   (VP wish
      (SBAR 0
         (S (NP-SBJ I)
            (VP was
               (NP-MNR-PRD that way))))))
.
E_S))
```