6.891: Lecture 3 (September 10, 2003)
Stochastic Parsing I

Overview

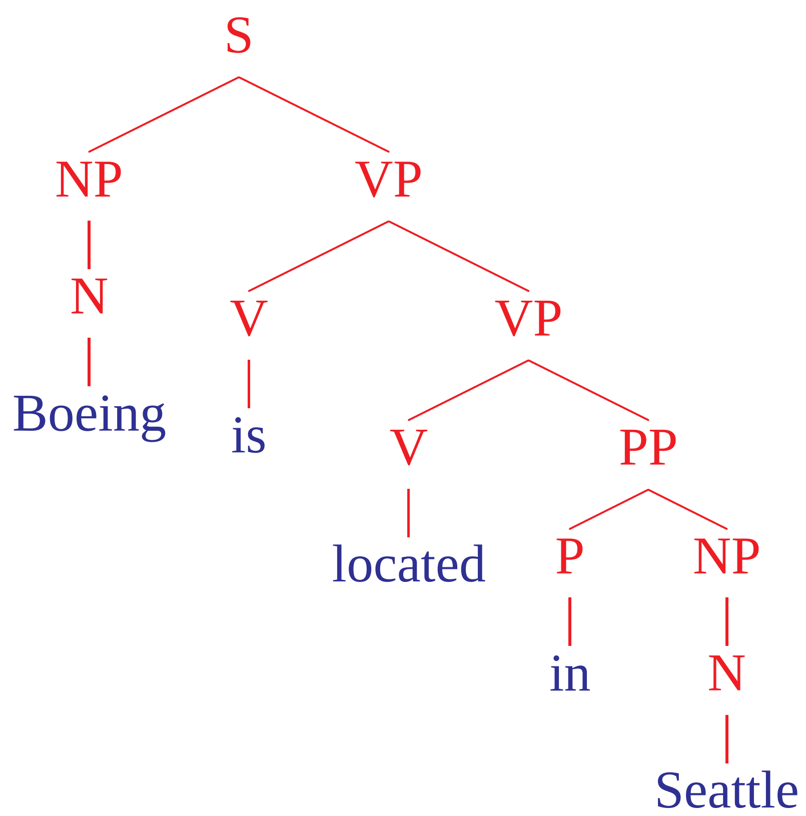
- An introduction to the parsing problem
- Context free grammars
- A brief(!) sketch of the syntax of English
- Examples of ambiguous structures
- PCFGs, their formal properties, and useful algorithms
- Weaknesses of PCFGs

Parsing (Syntactic Structure)

INPUT:

Boeing is located in Seattle.

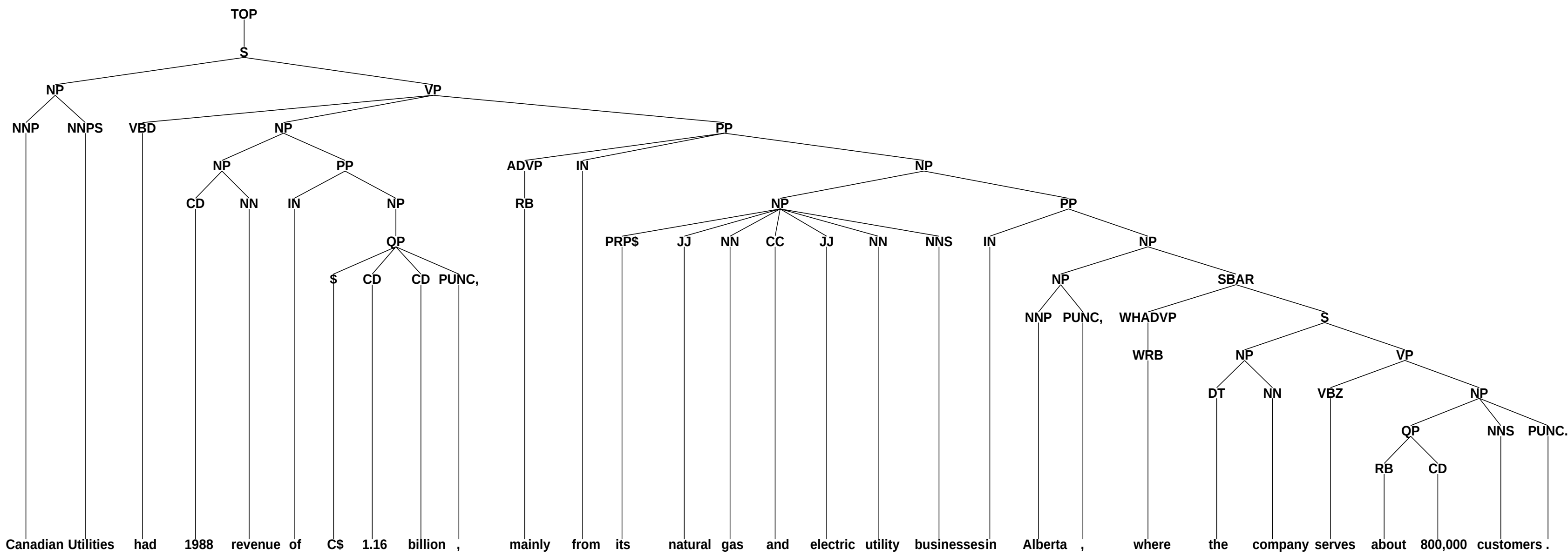
OUTPUT:



Data for Parsing Experiments

- Penn WSJ Treebank = 50,000 sentences with associated trees
- Usual set-up: 40,000 training sentences, 2400 test sentences

An example tree:

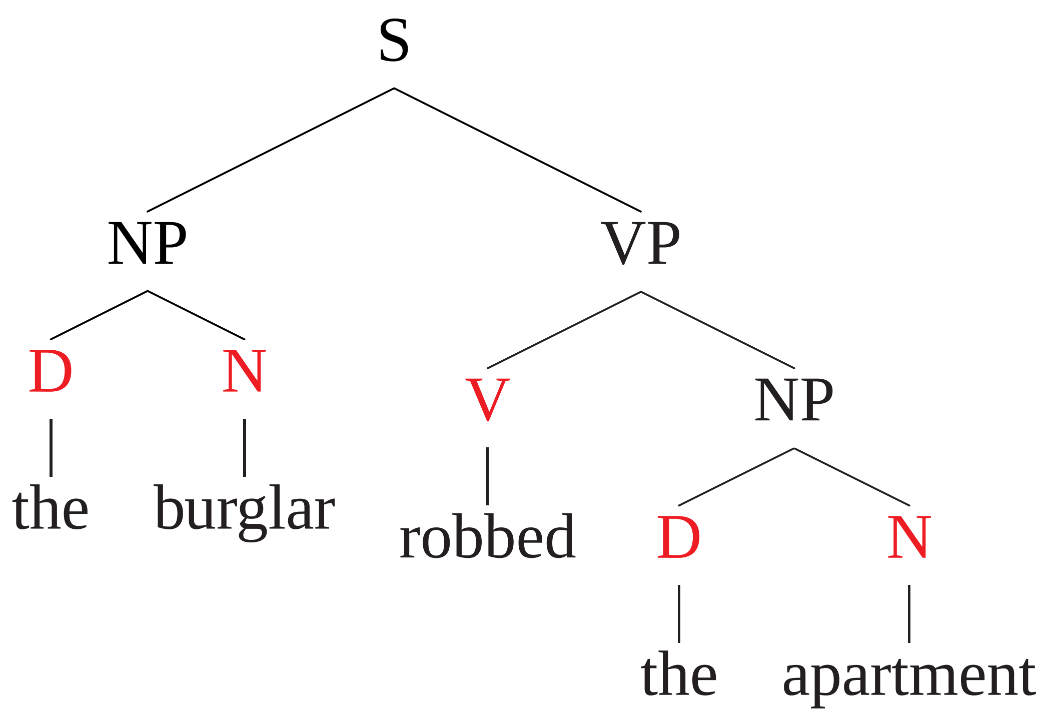


Canadian Utilities had 1988 revenue of C$ 1.16 billion , mainly from its natural gas and electric utility businesses in Alberta , where the company serves about 800,000 customers .

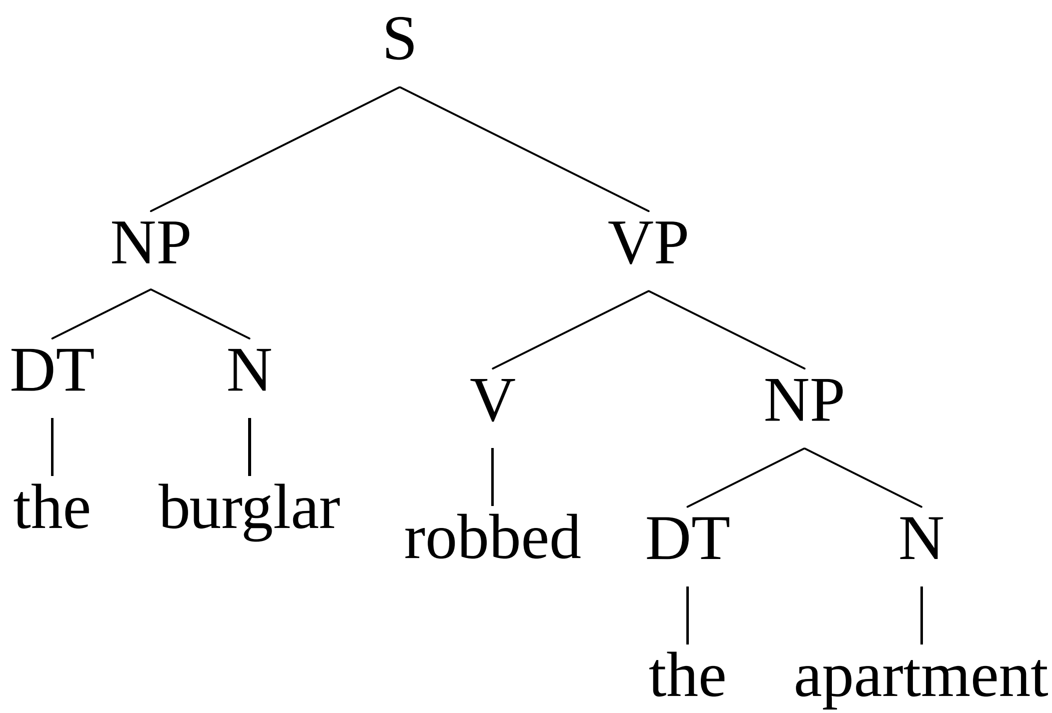
The Information Conveyed by Parse Trees

1) Part of speech for each word

(N = noun, V = verb, D = determiner)



2) Phrases

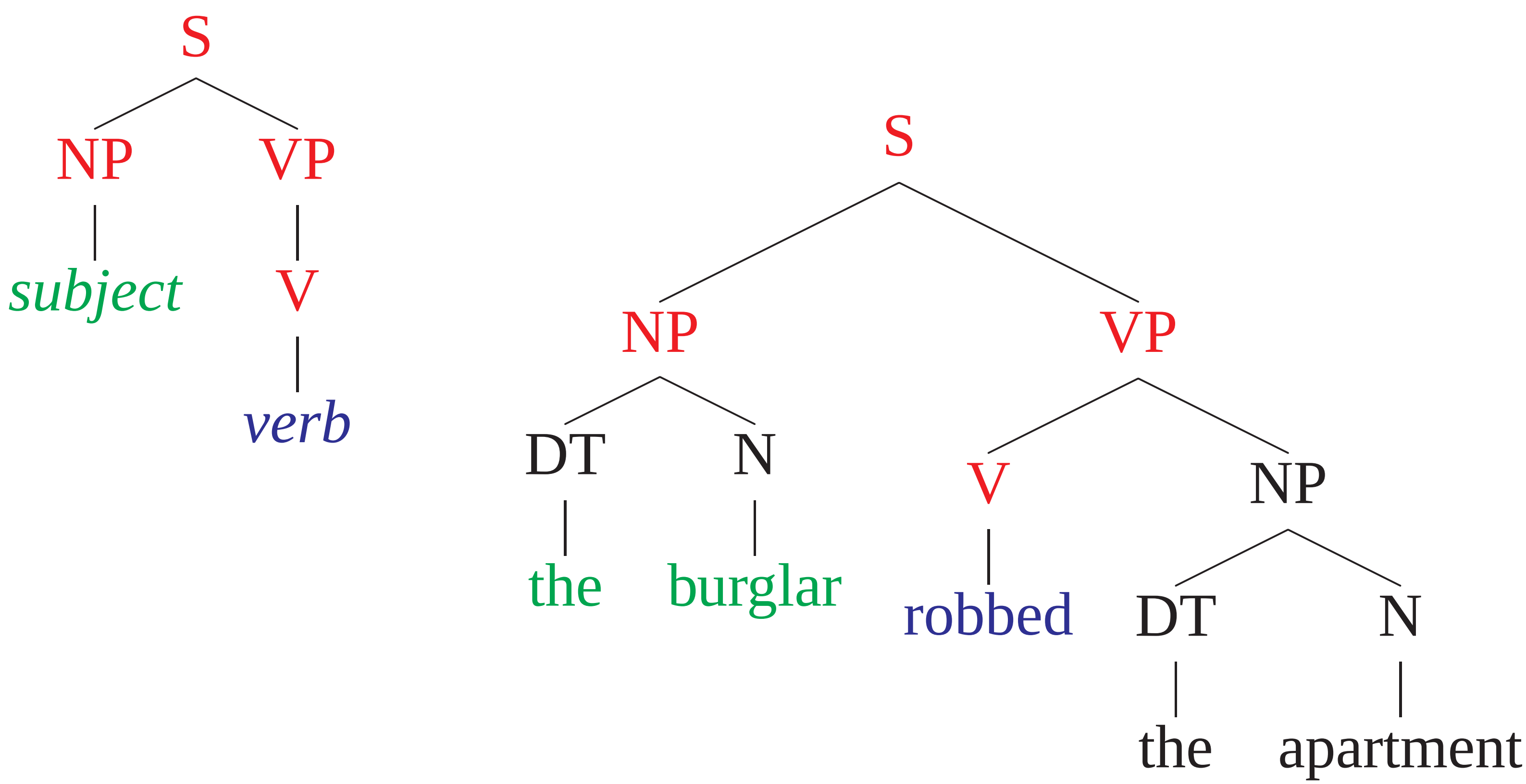


Noun Phrases (NP): “the burglar”, “the apartment”

Verb Phrases (VP): “robbed the apartment”

Sentences (S): “the burglar robbed the apartment”

3) Useful Relationships



⇒ "the burglar" is the subject of "robbed"

An Example Application: Machine Translation

- English word order is *subject – verb – object*
- Japanese word order is *subject – object – verb*

English:	IBM bought Lotus
Japanese:	*IBM Lotus bought*

English:	Sources said that IBM bought Lotus yesterday
Japanese:	*Sources yesterday IBM Lotus bought that said*

Context-Free Grammars

[Hopcroft and Ullman 1979]
A context free grammar $G = (N, \Sigma, R, S)$ where:

- N is a set of non-terminal symbols
- Σ is a set of terminal symbols
- R is a set of rules of the form $X \rightarrow Y_1 Y_2 \dots Y_n$ for $n \geq 0$, $X \in N$, $Y_i \in (N \cup \Sigma)$
- $S \in N$ is a distinguished start symbol

A Context-Free Grammar for English

N = {S, NP, VP, PP, DT, Vi, Vt, NN, IN}
S = S
Σ = {sleeps, saw, man, woman, telescope, the, with, in}

R =

S	$\Rightarrow$	NP	VP
VP	$\Rightarrow$	Vi	
VP	$\Rightarrow$	Vt	NP
VP	$\Rightarrow$	VP	PP
NP	$\Rightarrow$	DT	NN
NP	$\Rightarrow$	NP	PP
PP	$\Rightarrow$	IN	NP

Vi	$\Rightarrow$	sleeps
Vt	$\Rightarrow$	saw
NN	$\Rightarrow$	man
NN	$\Rightarrow$	woman
NN	$\Rightarrow$	telescope
DT	$\Rightarrow$	the
IN	$\Rightarrow$	with
IN	$\Rightarrow$	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

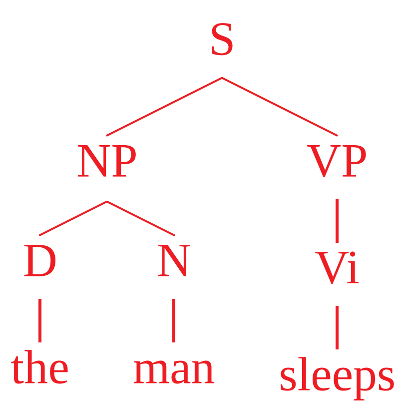
Left-Most Derivations

A left-most derivation is a sequence of strings $s_1 \dots s_n$, where

- $s_1 = S$, the start symbol
- $s_n \in \Sigma^*$, i.e. s_n is made up of terminal symbols only
- Each s_i for $i = 2 \dots n$ is derived from s_{i-1} by picking the left-most non-terminal X in s_{i-1} and replacing it by some β where $X \rightarrow \beta$ is a rule in R

For example: [S], [NP VP], [D N VP], [the N VP], [the man VP], [the man Vi], [the man sleeps]

Representation of a derivation as a tree:



DERIVATION	RULES USED
S	

DERIVATION	RULES USED
S	S $\rightarrow$ NP VP
NP VP	

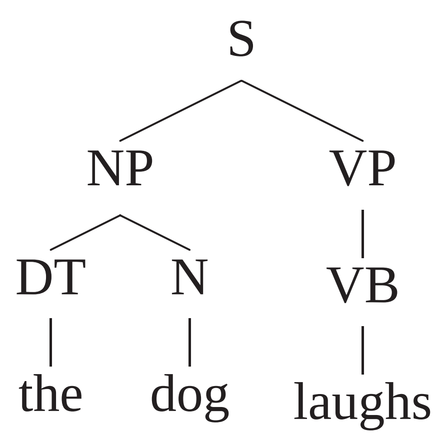
DERIVATION	RULES USED
S	$S \rightarrow NP\ VP$
NP VP	$NP \rightarrow DT\ N$
DT N VP	

DERIVATION	RULES USED
S	S $\rightarrow$ NP VP
NP VP	NP $\rightarrow$ DT N
DT N VP	DT $\rightarrow$ the
the N VP	

DERIVATION	RULES USED
S	S $\rightarrow$ NP VP
NP VP	NP $\rightarrow$ DT N
DT N VP	DT $\rightarrow$ the
the N VP	N $\rightarrow$ dog
the dog VP	

DERIVATION	RULES USED
S	S $\rightarrow$ NP VP
NP VP	NP $\rightarrow$ DT N
DT N VP	DT $\rightarrow$ the
the N VP	N $\rightarrow$ dog
the dog VP	VP $\rightarrow$ VB
the dog VB	

DERIVATION	RULES USED
S	S $\rightarrow$ NP VP
NP VP	NP $\rightarrow$ DT N
DT N VP	DT $\rightarrow$ the
the N VP	N $\rightarrow$ dog
the dog VP	VP $\rightarrow$ VB
the dog VB	VB $\rightarrow$ laughs
the dog laughs	



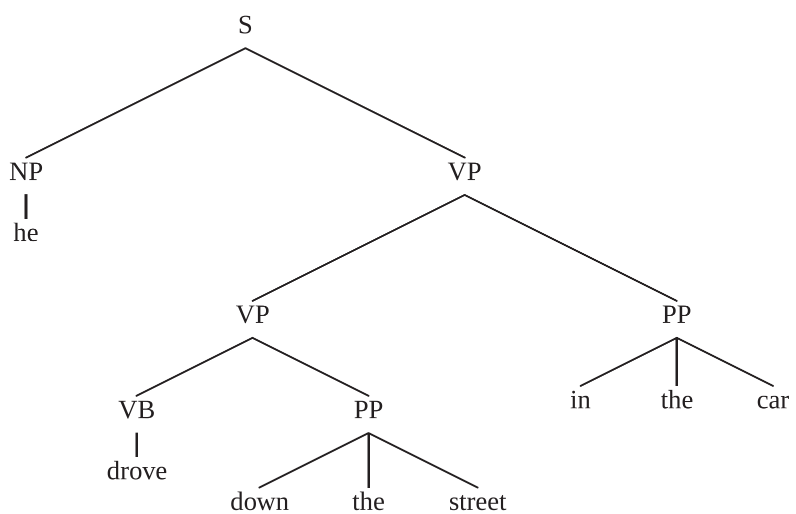
Properties of CFGs

- A CFG defines a set of possible derivations
- A string $s \in \Sigma^*$ is in the *language* defined by the CFG if there is at least one derivation which yields s
- Each string in the language generated by the CFG may have more than one derivation (“ambiguity”)

DERIVATION

S

RULES USED



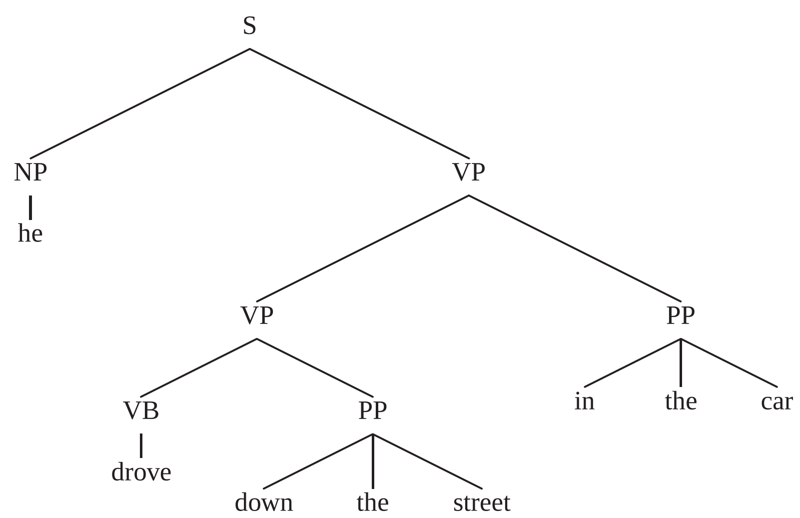
DERIVATION

S

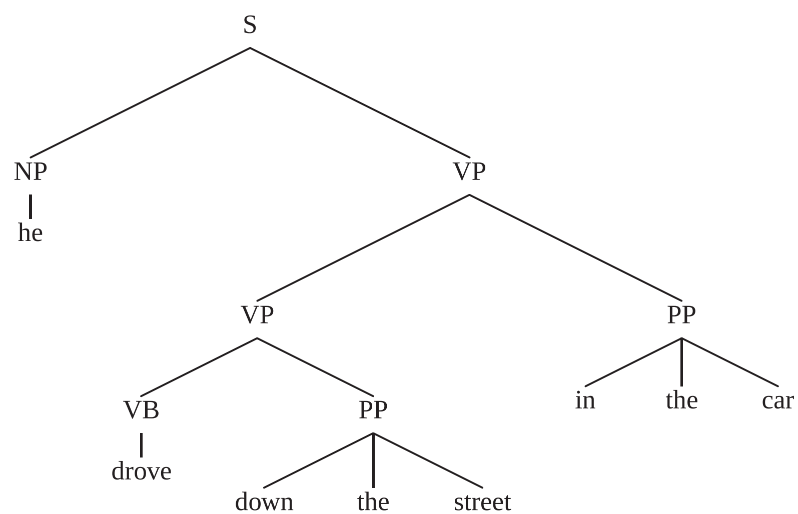
NP VP

RULES USED

$S \rightarrow NP\ VP$



DERIVATION	RULES USED
S	S $\rightarrow$ NP VP
NP VP	NP $\rightarrow$ he
he VP	



DERIVATION

S

NP VP

he VP

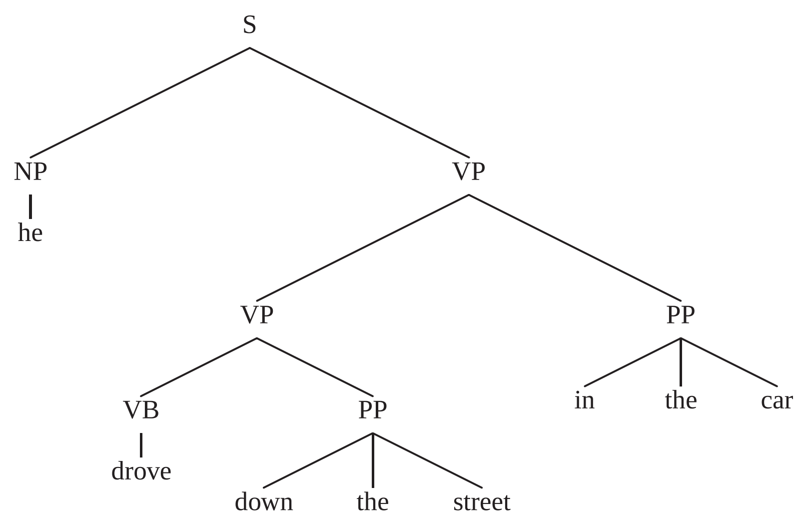
he VP PP

RULES USED

S $\rightarrow$ NP VP

NP $\rightarrow$ he

VP $\rightarrow$ VP PP



DERIVATION

S

NP VP

he VP

he VP PP

he VB PP PP

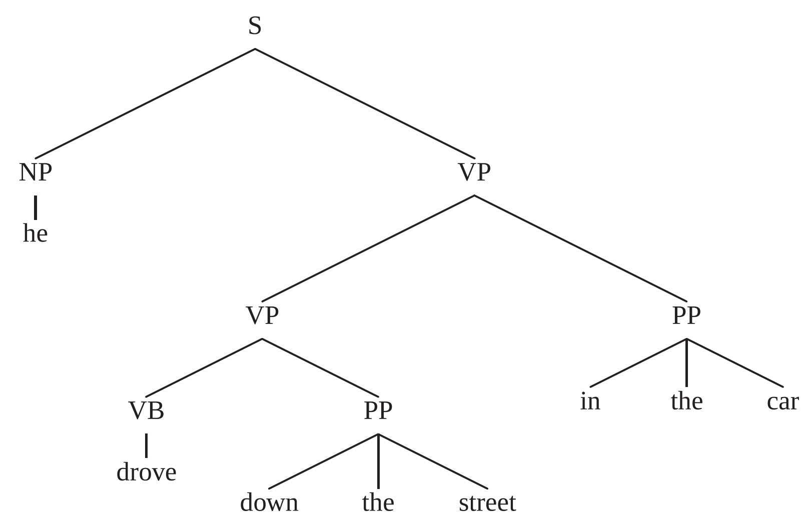
RULES USED

$S \rightarrow NP\ VP$

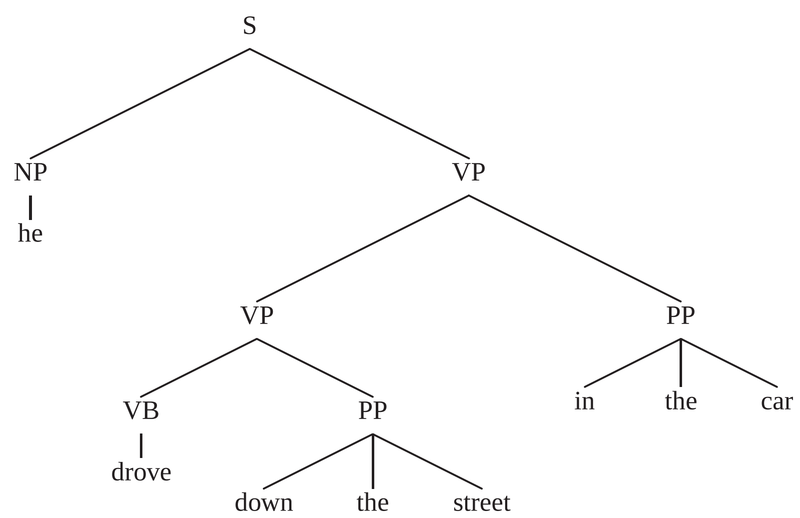
$NP \rightarrow he$

$VP \rightarrow VP\ PP$

$VP \rightarrow VB\ PP$



DERIVATION	RULES USED
S	S $\rightarrow$ NP VP
NP VP	NP $\rightarrow$ he
he VP	VP $\rightarrow$ VP PP
he VP PP	VP $\rightarrow$ VB PP
he VB PP PP	VB $\rightarrow$ drove
he drove PP PP	



DERIVATION

S

NP VP

he VP

he VP PP

he VB PP PP

he drove PP PP

he drove down the street PP

RULES USED

S $\rightarrow$ NP VP

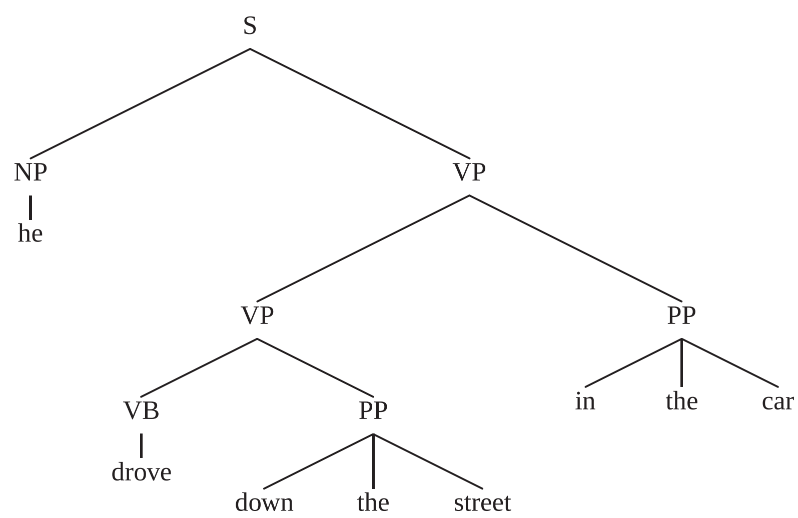
NP $\rightarrow$ he

VP $\rightarrow$ VP PP

VP $\rightarrow$ VB PP

VB$\rightarrow$ drove

PP$\rightarrow$ down the street



DERIVATION

S

NP VP

he VP

he VP PP

he VB PP PP

he drove PP PP

he drove down the street PP

he drove down the street in the car

RULES USED

S $\rightarrow$ NP VP

NP $\rightarrow$ he

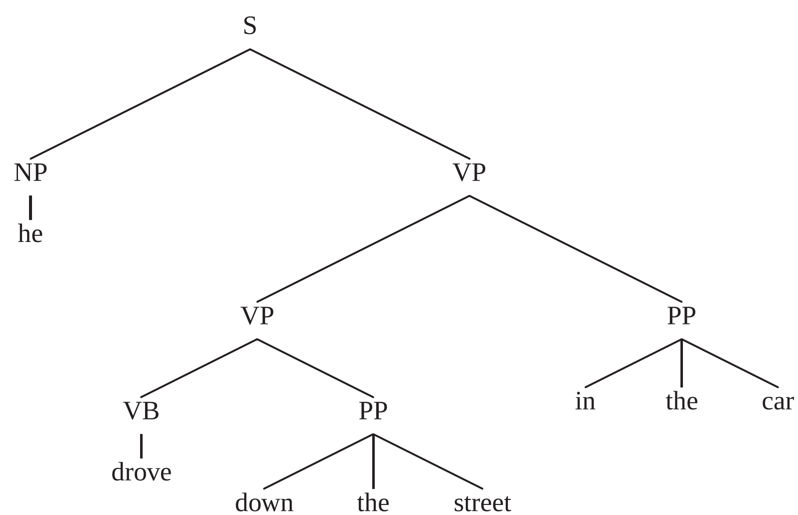
VP $\rightarrow$ VP PP

VP $\rightarrow$ VB PP

VB$\rightarrow$ drove

PP$\rightarrow$ down the street

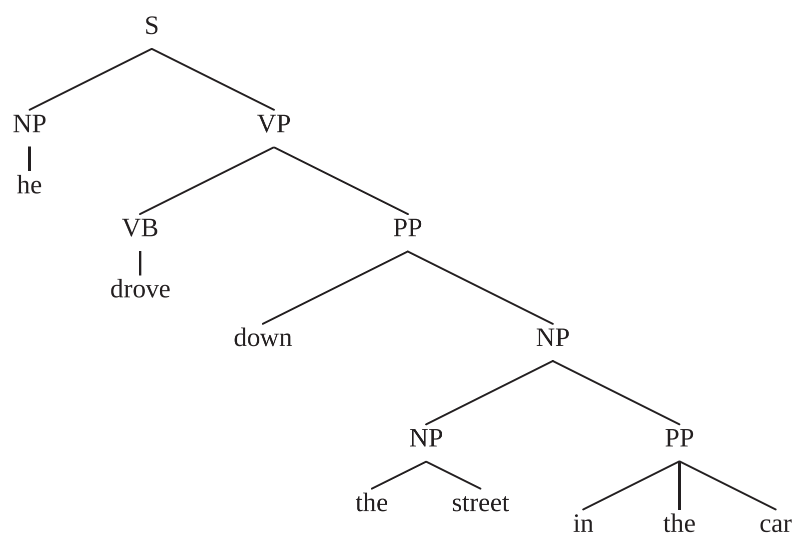
PP$\rightarrow$ in the car



DERIVATION

RULES USED

S



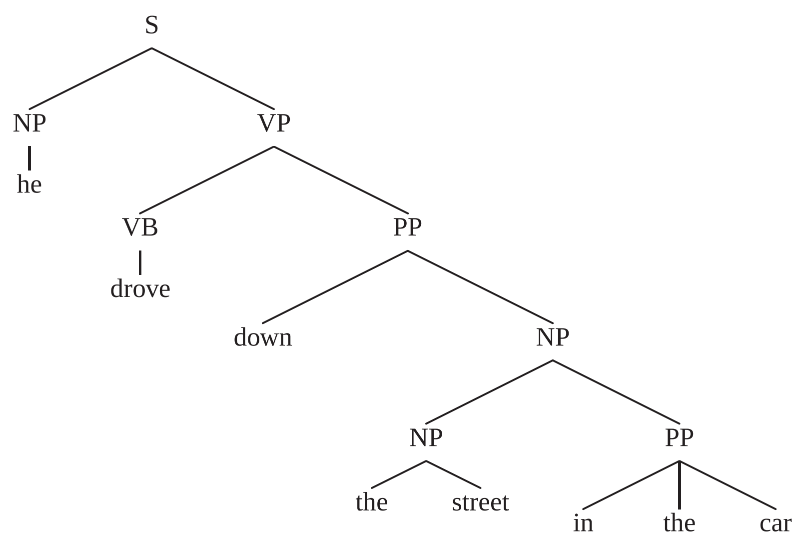
DERIVATION

S

NP VP

RULES USED

S $\rightarrow$ NP VP



DERIVATION

S

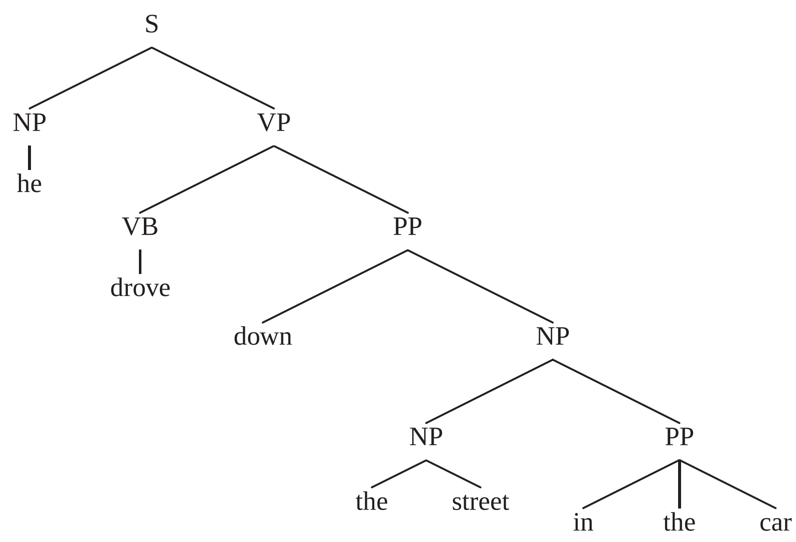
NP VP

he VP

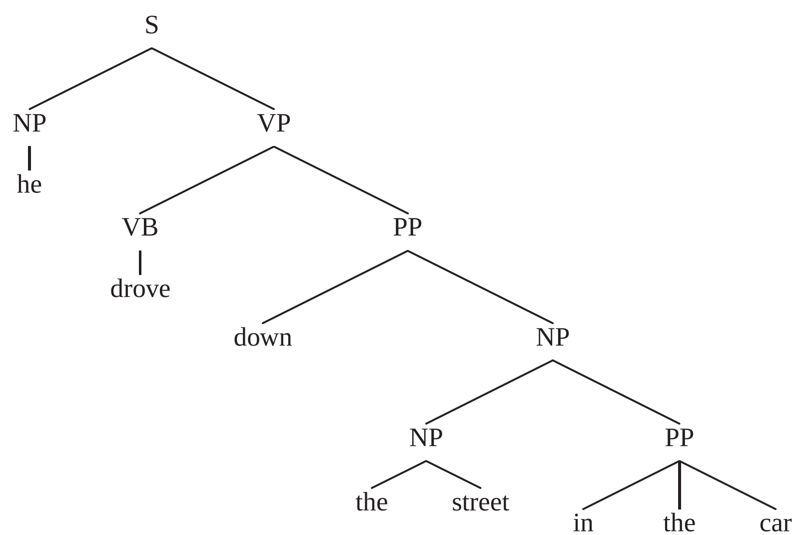
RULES USED

S $\rightarrow$ NP VP

NP $\rightarrow$ he



DERIVATION	RULES USED
S	$S \rightarrow NP\ VP$
NP VP	$NP \rightarrow he$
he VP	$VP \rightarrow VB\ PP$
he VB PP	



DERIVATION

S

NP VP

he VP

he VB PP

he drove PP

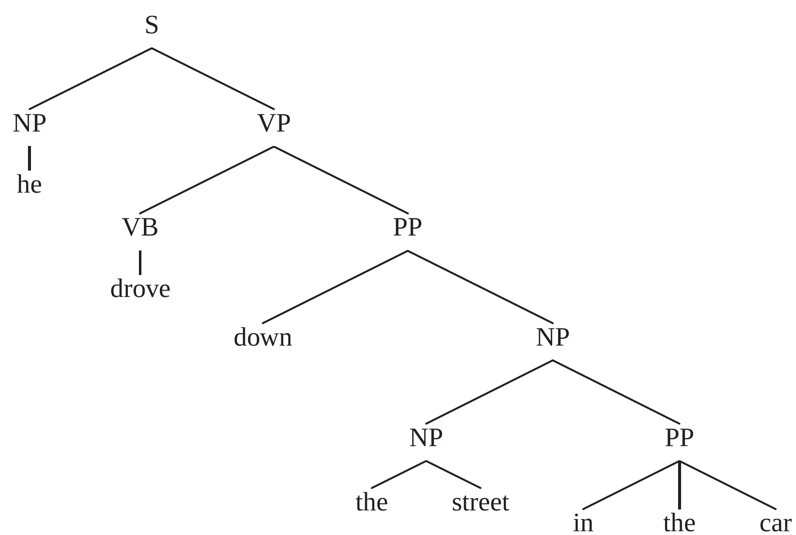
RULES USED

$S \rightarrow NP\ VP$

$NP \rightarrow he$

$VP \rightarrow VB\ PP$

$VB \rightarrow drove$



DERIVATION

S

NP VP

he VP

he VB PP

he drove PP

he drove down NP

RULES USED

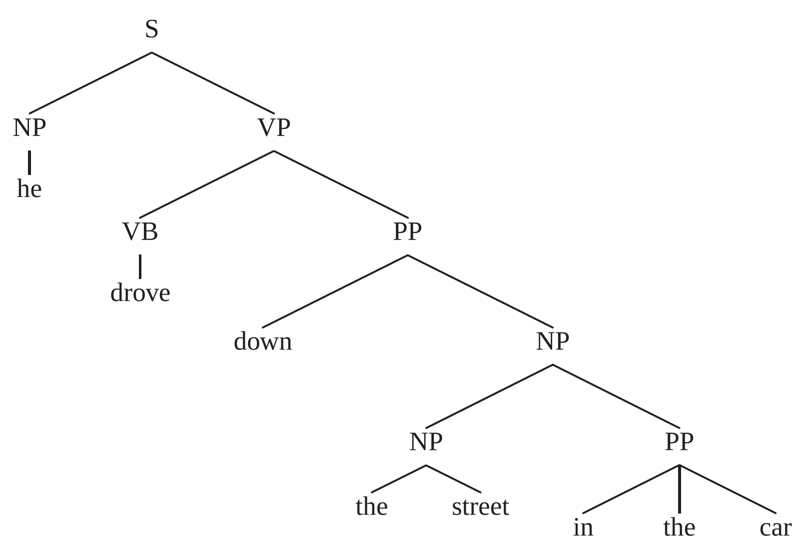
S $\rightarrow$ NP VP

NP $\rightarrow$ he

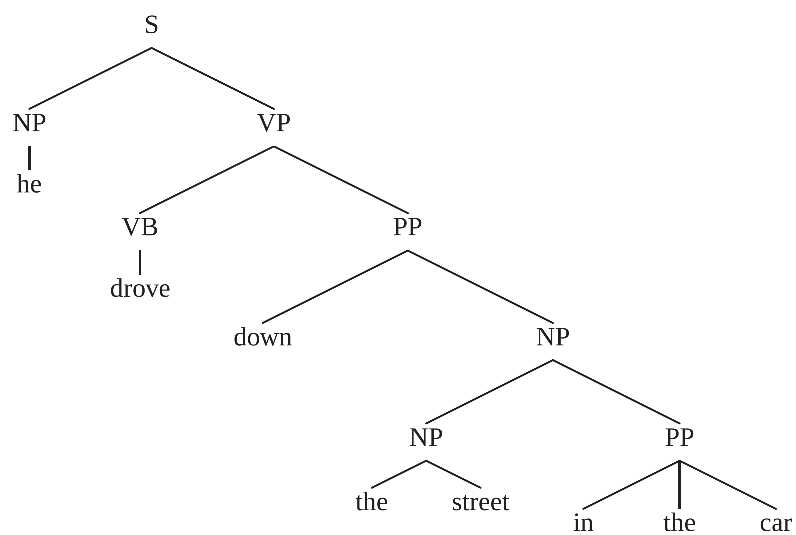
VP $\rightarrow$ VB PP

VB $\rightarrow$ drove

PP $\rightarrow$ down NP



DERIVATION	RULES USED
S	S $\rightarrow$ NP VP
NP VP	NP $\rightarrow$ he
he VP	VP $\rightarrow$ VB PP
he VB PP	VB $\rightarrow$ drove
he drove PP	PP $\rightarrow$ down NP
he drove down NP	NP $\rightarrow$ NP PP
he drove down NP PP	



DERIVATION

S

NP VP

he VP

he VB PP

he drove PP

he drove down NP

he drove down NP PP

he drove down the street PP

RULES USED

S $\rightarrow$ NP VP

NP $\rightarrow$ he

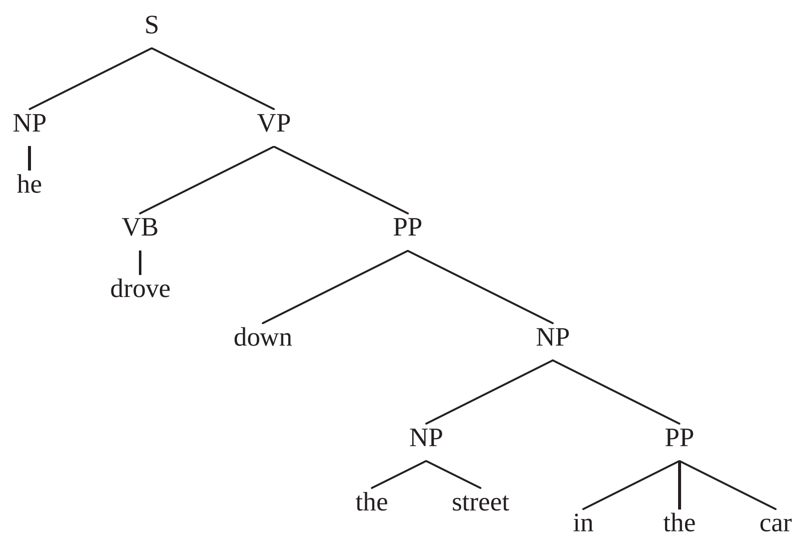
VP $\rightarrow$ VB PP

VB $\rightarrow$ drove

PP $\rightarrow$ down NP

NP $\rightarrow$ NP PP

NP $\rightarrow$ the street



DERIVATION

S

NP VP

he VP

he VB PP

he drove PP

he drove down NP

he drove down NP PP

he drove down the street PP

he drove down the street in the car

RULES USED

S $\rightarrow$ NP VP

NP $\rightarrow$ he

VP $\rightarrow$ VB PP

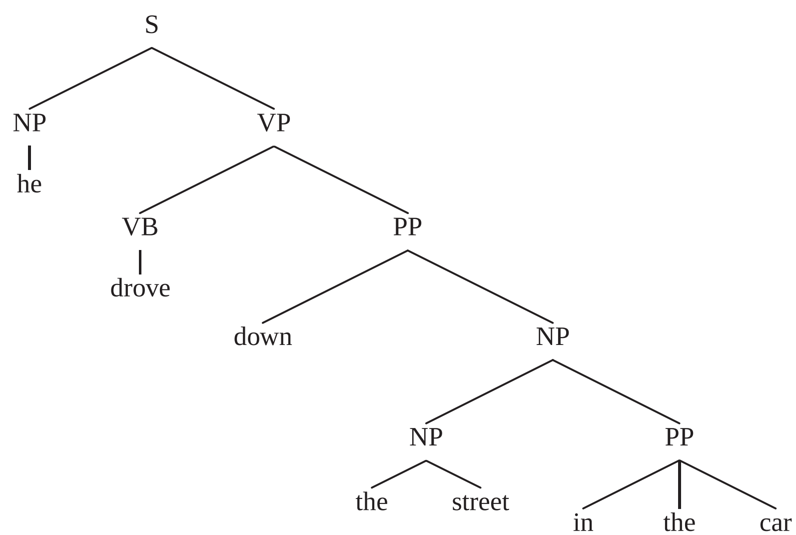
VB $\rightarrow$ drove

PP $\rightarrow$ down NP

NP $\rightarrow$ NP PP

NP $\rightarrow$ the street

PP $\rightarrow$ in the car



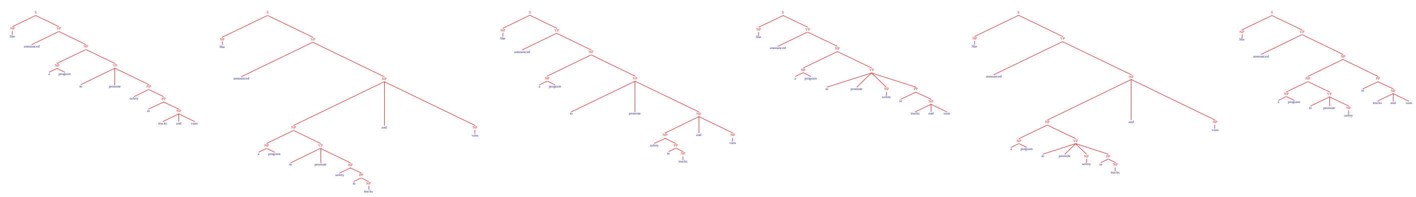
The Problem with Parsing: Ambiguity

INPUT:

She announced a program to promote safety in trucks and vans

$$\Downarrow$$

POSSIBLE OUTPUTS:



And there are more...

A Brief Overview of English Syntax

Parts of Speech:

- Nouns
  (Tags from the *Brown corpus*)
  NN = singular noun e.g., man, dog, park
  NNS = plural noun e.g., telescopes, houses, buildings
  NNP = proper noun e.g., Smith, Gates, IBM

- Determiners
  DT = determiner e.g., the, a, some, every

- Adjectives
  JJ = adjective e.g., red, green, large, idealistic

A Fragment of a Noun Phrase Grammar

$\bar{N}$	$\Rightarrow$	NN	
$\bar{N}$	$\Rightarrow$	NN	$\bar{N}$
$\bar{N}$	$\Rightarrow$	JJ	$\bar{N}$
$\bar{N}$	$\Rightarrow$	$\bar{N}$	$\bar{N}$
NP	$\Rightarrow$	DT	$\bar{N}$

NN	$\Rightarrow$	box
NN	$\Rightarrow$	car
NN	$\Rightarrow$	mechanic
NN	$\Rightarrow$	pigeon
DT	$\Rightarrow$	the
DT	$\Rightarrow$	a
JJ	$\Rightarrow$	fast
JJ	$\Rightarrow$	metal
JJ	$\Rightarrow$	idealistic
JJ	$\Rightarrow$	clay

Generates:

a box, the box, the metal box, the fast car mechanic, . . .

Prepositions, and Prepositional Phrases

- Prepositions

  IN = preposition   e.g., of, in, out, beside, as

An Extended Grammar

$\bar{N}$	$\Rightarrow$	NN	
$\bar{N}$	$\Rightarrow$	NN	$\bar{N}$
$\bar{N}$	$\Rightarrow$	JJ	$\bar{N}$
$\bar{N}$	$\Rightarrow$	$\bar{N}$	$\bar{N}$
NP	$\Rightarrow$	DT	$\bar{N}$
PP	$\Rightarrow$	IN	NP
$\bar{N}$	$\Rightarrow$	$\bar{N}$	PP

NN	$\Rightarrow$	box
NN	$\Rightarrow$	car
NN	$\Rightarrow$	mechanic
NN	$\Rightarrow$	pigeon
DT	$\Rightarrow$	the
DT	$\Rightarrow$	a

JJ	$\Rightarrow$	fast
JJ	$\Rightarrow$	metal
JJ	$\Rightarrow$	idealistic
JJ	$\Rightarrow$	clay
IN	$\Rightarrow$	in
IN	$\Rightarrow$	under
IN	$\Rightarrow$	of
IN	$\Rightarrow$	on
IN	$\Rightarrow$	with
IN	$\Rightarrow$	as

Generates:

in a box, under the box, the fast car mechanic under the pigeon in the box, . . .

Verbs, Verb Phrases, and Sentences

- Basic Verb Types

  Vi = Intransitive verb e.g., sleeps, walks, laughs

  Vt = Transitive verb e.g., sees, saw, likes

  Vd = Ditransitive verb e.g., gave

- Basic VP Rules

  $\text{VP} \rightarrow \text{Vi}$

  $\text{VP} \rightarrow \text{Vt} \quad \text{NP}$

  $\text{VP} \rightarrow \text{Vd} \quad \text{NP} \quad \text{NP}$

- Basic S Rule

  $\text{S} \rightarrow \text{NP} \quad \text{VP}$

Examples of VP:

sleeps, walks, likes the mechanic, gave the mechanic the fast car, gave the fast car mechanic the pigeon in the box, . . .

Examples of S:
the man sleeps, the dog walks, the dog likes the mechanic, the dog in the box gave the mechanic the fast car,. . .

PPs Modifying Verb Phrases

A new rule:

VP $\rightarrow$ VP PP

New examples of VP:

sleeps in the car, walks like the mechanic, gave the mechanic the fast car on Tuesday, . . .

Complementizers, and SBARs

- Complementizers
  COMP = complementizer   e.g., that

- SBAR
  SBAR $\rightarrow$ COMP S

Examples:
that the man sleeps, that the mechanic saw the dog . . .

More Verbs

- New Verb Types

  V[5] e.g., said, reported

  V[6] e.g., told, informed

  V[7] e.g., bet

- New VP Rules

  VP $\rightarrow$ V[5] SBAR

  VP $\rightarrow$ V[6] NP SBAR

  VP $\rightarrow$ V[7] NP NP SBAR

Examples of New VPs:

said that the man sleeps

told the dog that the mechanic likes the pigeon

bet the pigeon $50 that the mechanic owns a fast car

Coordination

- A New Part-of-Speech:
  CC = Coordinator e.g., and, or, but

- New Rules

NP	$\rightarrow$	NP	CC	NP
$\bar{N}$	$\rightarrow$	$\bar{N}$	CC	$\bar{N}$
VP	$\rightarrow$	VP	CC	VP
S	$\rightarrow$	S	CC	S
SBAR	$\rightarrow$	SBAR	CC	SBAR

Sources of Ambiguity

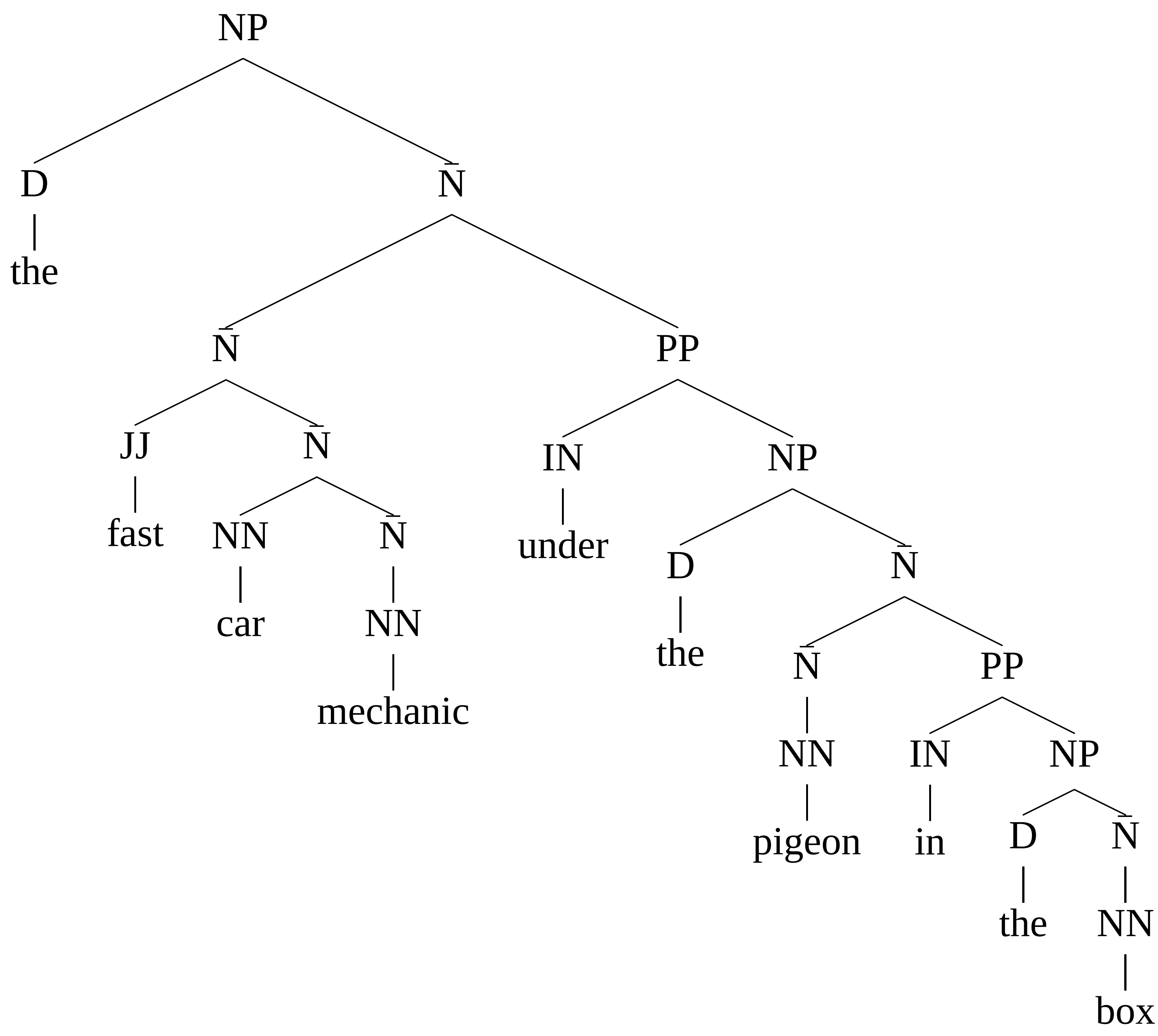
- Part-of-Speech ambiguity

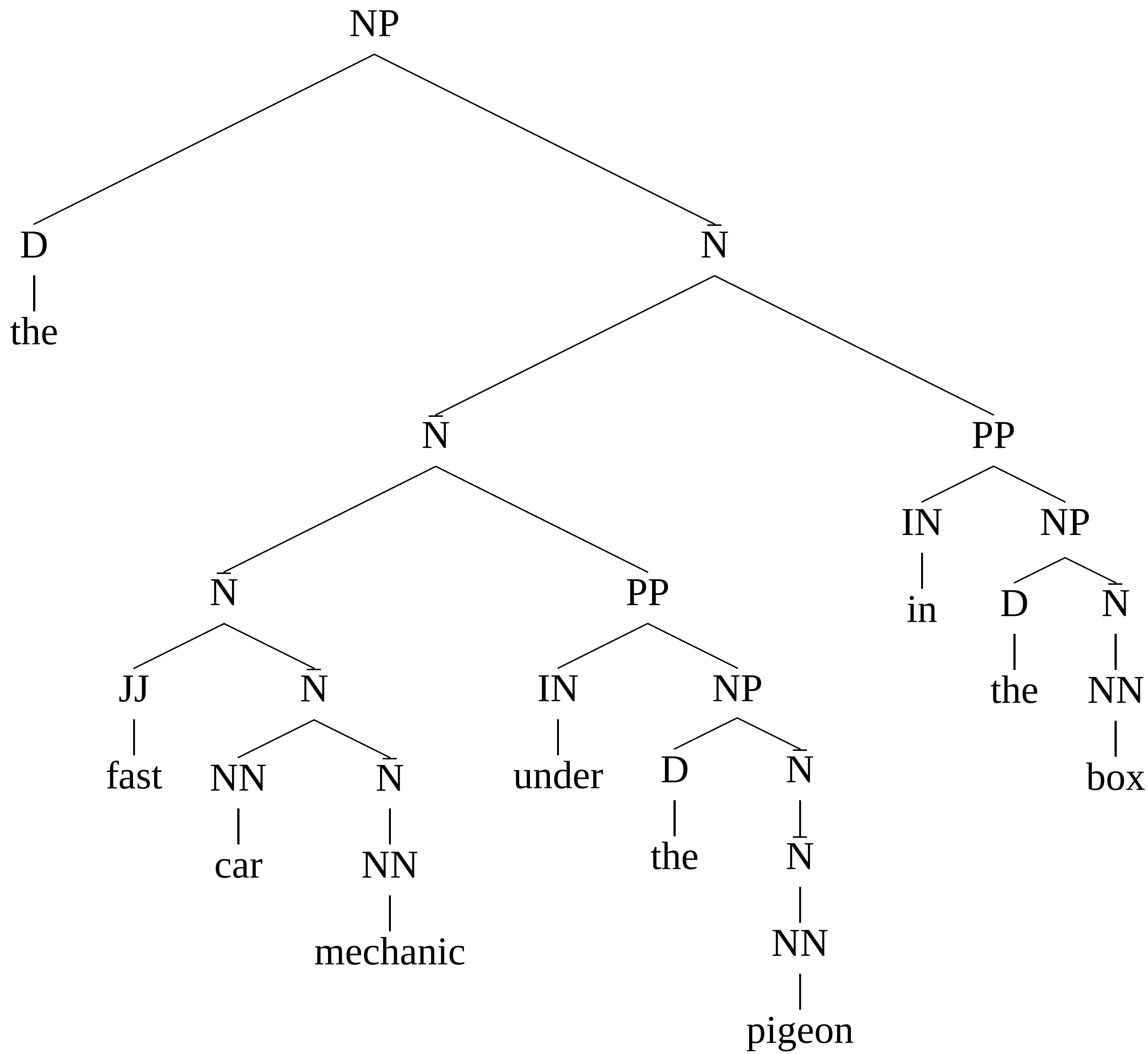
  NNS $\rightarrow$ walks

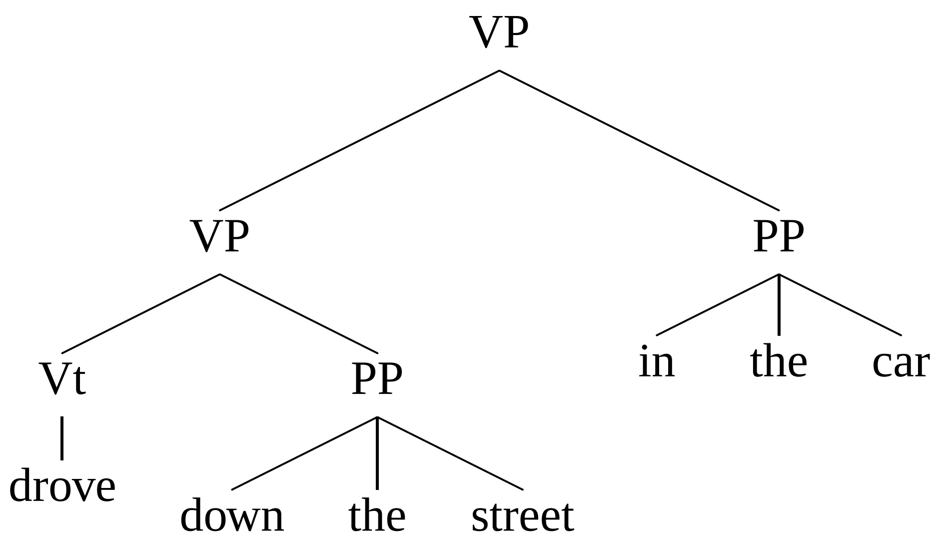
  Vi $\rightarrow$ walks

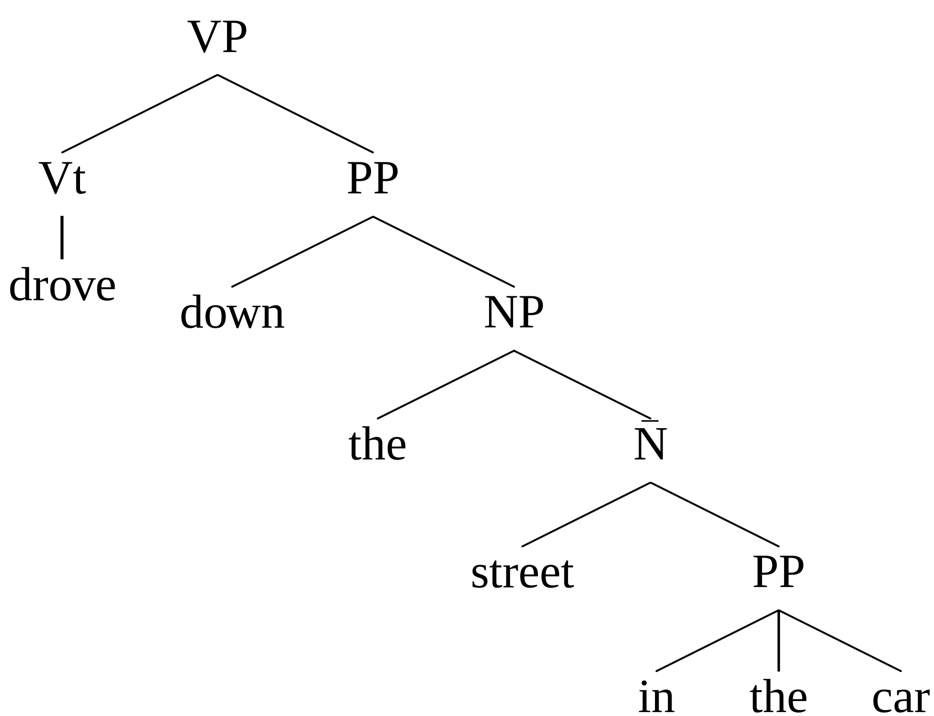
- Prepositional Phrase Attachment

  the fast car mechanic under the pigeon in the box


NP
D
N̄
the
N̄
PP
JJ
N̄
IN
NP
fast
NN
N̄
under
D
N̄
car
NN
the
N̄
PP
mechanic
NN
IN
NP
pigeon
in
D
N̄
the
NN
box


NP
D
the
N̄
N̄
PP
N̄
PP
IN
NP
JJ
N̄
IN
NP
in
D
N̄
fast
NN
N̄
under
D
N̄
the
NN
car
NN
the
N̄
box
mechanic
NN
pigeon

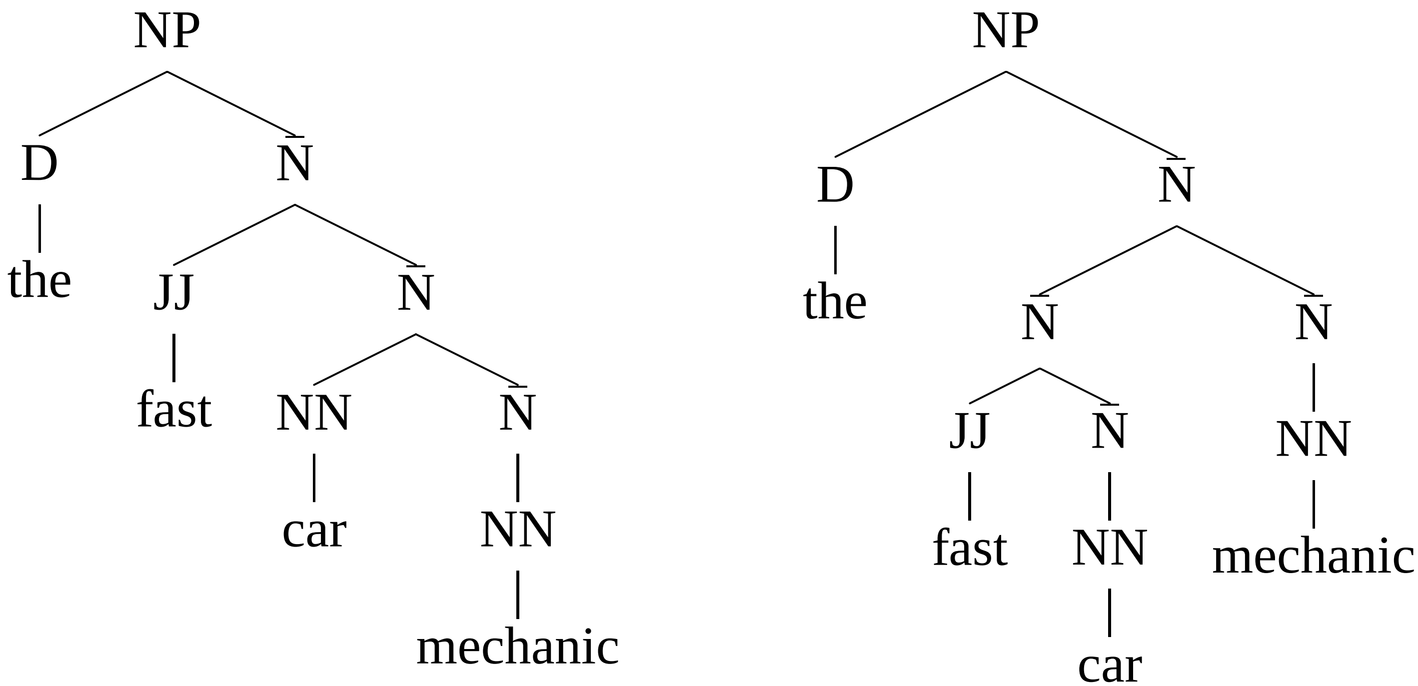

VP
VP
PP
Vt
PP
in
the
car
drove
down
the
street


VP
Vt
PP
drove
down
NP
the
N̄
street
PP
in
the
car

Two analyses for: John was believed to have been shot by Bill

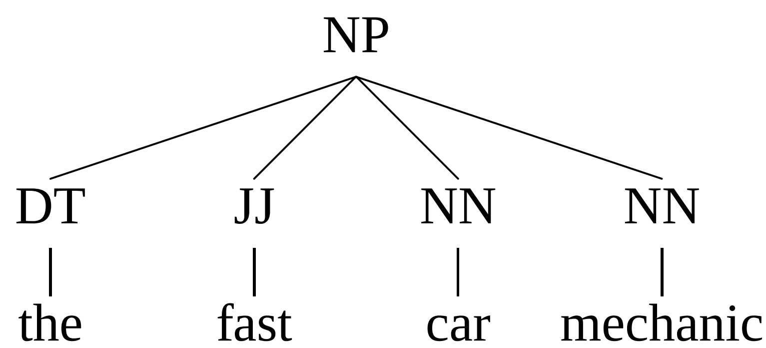
Sources of Ambiguity: Noun Premodifiers

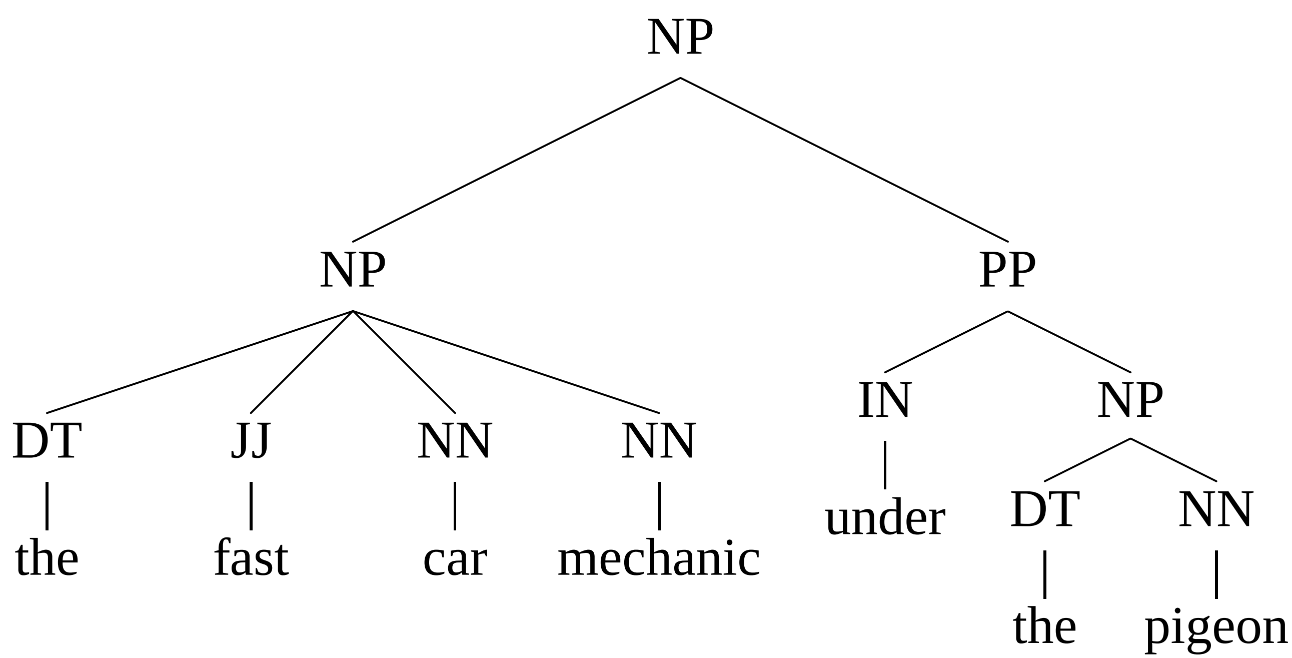
- Noun premodifiers:



A Funny Thing about the Penn Treebank

Leaves NP premodifier structure flat, or underspecified:





A Probabilistic Context-Free Grammar

S	$\Rightarrow$	NP	VP	1.0
VP	$\Rightarrow$	Vi		0.4
VP	$\Rightarrow$	Vt	NP	0.4
VP	$\Rightarrow$	VP	PP	0.2
NP	$\Rightarrow$	DT	NN	0.3
NP	$\Rightarrow$	NP	PP	0.7
PP	$\Rightarrow$	P	NP	1.0

Vi	$\Rightarrow$	sleeps	1.0
Vt	$\Rightarrow$	saw	1.0
NN	$\Rightarrow$	man	0.7
NN	$\Rightarrow$	woman	0.2
NN	$\Rightarrow$	telescope	0.1
DT	$\Rightarrow$	the	1.0
IN	$\Rightarrow$	with	0.5
IN	$\Rightarrow$	in	0.5

- Probability of a tree with rules $\alpha_i \rightarrow \beta_i$ is $\prod_i P(\alpha_i \rightarrow \beta_i | \alpha_i)$

DERIVATION	RULES USED	PROBABILITY
S		

DERIVATION	RULES USED	PROBABILITY
S	S $\rightarrow$ NP VP	1.0
NP VP		

DERIVATION	RULES USED	PROBABILITY
S	S $\rightarrow$ NP VP	1.0
NP VP	NP $\rightarrow$ DT N	0.3
DT N VP		

DERIVATION	RULES USED	PROBABILITY
S	$S \rightarrow NP\ VP$	1.0
NP VP	$NP \rightarrow DT\ N$	0.3
DT N VP	$DT \rightarrow$ the	1.0
the N VP		

DERIVATION	RULES USED	PROBABILITY
S	S $\rightarrow$ NP VP	1.0
NP VP	NP $\rightarrow$ DT N	0.3
DT N VP	DT $\rightarrow$ the	1.0
the N VP	N $\rightarrow$ dog	0.1
the dog VP		

DERIVATION	RULES USED	PROBABILITY
S	S $\rightarrow$ NP VP	1.0
NP VP	NP $\rightarrow$ DT N	0.3
DT N VP	DT $\rightarrow$ the	1.0
the N VP	N $\rightarrow$ dog	0.1
the dog VP	VP $\rightarrow$ VB	0.4
the dog VB		

DERIVATION	RULES USED	PROBABILITY
S	S $\rightarrow$ NP VP	1.0
NP VP	NP $\rightarrow$ DT N	0.3
DT N VP	DT $\rightarrow$ the	1.0
the N VP	N $\rightarrow$ dog	0.1
the dog VP	VP $\rightarrow$ VB	0.4
the dog VB	VB $\rightarrow$ laughs	0.5
the dog laughs		

TOTAL PROBABILITY = $1.0 \times 0.3 \times 1.0 \times 0.1 \times 0.4 \times 0.5$

Properties of PCFGs

- Assigns a probability to each *left-most derivation*, or parse-tree, allowed by the underlying CFG

- Say we have a sentence S, set of derivations for that sentence is $\mathcal{T}(S)$. Then a PCFG assigns a probability to each member of $\mathcal{T}(S)$. i.e., *we now have a ranking in order of probability*.

- The probability of a string S is

$$\sum_{T \in \mathcal{T}(S)} P(T, S)$$

Deriving a PCFG from a Corpus

- Given a set of example trees, the underlying CFG can simply be **all rules seen in the corpus**

- Maximum Likelihood estimates:

$$P_{ML}(\alpha \rightarrow \beta \mid \alpha) = \frac{\text{Count}(\alpha \rightarrow \beta)}{\text{Count}(\alpha)}$$

  where the counts are taken from a training set of example trees.

- **If the training data is generated by a PCFG**, then as the training data size goes to infinity, the maximum-likelihood PCFG will converge to the same distribution as the "true" PCFG.

PCFGs

[Booth and Thompson 73] showed that a CFG with rule probabilities correctly defines a distribution over the set of derivations provided that:

1. The rule probabilities define conditional distributions over the different ways of rewriting each non-terminal.

2. A technical condition on the rule probabilities ensuring that the probability of the derivation terminating in a finite number of steps is 1. (This condition is not really a practical concern.)

Algorithms for PCFGs

- Given a PCFG and a sentence S, define $\mathcal{T}(S)$ to be the set of trees with S as the yield.
- Given a PCFG and a sentence S, how do we find

$$\arg\max_{T \in \mathcal{T}(S)} P(T, S)$$

- Given a PCFG and a sentence S, how do we find

$$P(S) = \sum_{T \in \mathcal{T}(S)} P(T, S)$$

Chomsky Normal Form

A context free grammar $G = (N, \Sigma, R, S)$ in Chomsky Normal Form is as follows

- N is a set of non-terminal symbols
- Σ is a set of terminal symbols
- R is a set of rules which take one of two forms:
  - $X \rightarrow Y_1 Y_2$ for $X \in N$, and $Y_1, Y_2 \in N$
  - $X \rightarrow Y$ for $X \in N$, and $Y \in \Sigma$
- $S \in N$ is a distinguished start symbol

A Dynamic Programming Algorithm

- Given a PCFG and a sentence S, how do we find

$$\max_{T \in \mathcal{T}(S)} P(T, S)$$

- Notation:

$$\begin{array}{ll} & n = \text{number of words in the sentence} \\ & N_k \text{ for } k = 1 \ldots K \text{ is } k\text{'th non-terminal} \\ \text{w.l.g.,} & N_1 = S \text{ (the start symbol)} \end{array}$$

- Define a dynamic programming table

$$\begin{array}{rcl} \pi[i, j, k] & = & \text{maximum probability of a constituent with non-terminal } N_k \\ & & \text{spanning words } i \ldots j \text{ inclusive} \end{array}$$

- Our goal is to calculate $\max_{T \in \mathcal{T}(S)} P(T, S) = \pi[1, n, 1]$

A Dynamic Programming Algorithm

- Base case definition: for all $i = 1 \ldots n$, for $k = 1 \ldots K$

$$\pi[i, i, k] = P(N_k \rightarrow w_i \mid N_k)$$

- Recursive definition: for all $i = 1 \ldots n$, $j = (i+1) \ldots n$, $k = 1 \ldots K$,

$$\pi[i, j, k] = \max_{\substack{i \leq s < j \\ 1 \leq l \leq K \\ 1 \leq m \leq K}} \{P(N_k \rightarrow N_l N_m \mid N_k) \times \pi[i, s, l] \times \pi[s+1, j, m]\}$$

Initialization:

For i = 1 ... n, k = 1 ... K

$\pi[i, i, k] = P(N_k \rightarrow w_i | N_k)$

Main Loop:

For $length = 1 \ldots (n-1)$, $i = 1 \ldots (n - 1ength)$, $k = 1 \ldots K$

$j \leftarrow i + length$

$max \leftarrow 0$

For $s = i \ldots (j-1)$,

For $l = 1 \ldots K$,

For $m = 1 \ldots K$,

$prob \leftarrow P(N_k \rightarrow N_l N_m) \times \pi[i, s, l] \times \pi[s+1, j, m]$

If $prob > max$

$max \leftarrow prob$

//Store backpointers which imply the best parse

$Split(i, j, k) = \{s, l, m\}$

$\pi[i, j, k] = max$

A Dynamic Programming Algorithm for the Sum

- Given a PCFG and a sentence S, how do we find

$$\sum_{T \in \mathcal{T}(S)} P(T, S)$$

- Notation:

$$\begin{aligned} & n = \text{number of words in the sentence} \\ & N_k \text{ for } k = 1 \ldots K \text{ is } k\text{'th non-terminal} \\ \text{w.l.g.,} \quad & N_1 = S \text{ (the start symbol)} \end{aligned}$$

- Define a dynamic programming table

$$\begin{aligned} \pi[i, j, k] \quad = \quad & \text{sum of probability of parses with root label } N_k \\ & \text{spanning words } i \ldots j \text{ inclusive} \end{aligned}$$

- Our goal is to calculate $\sum_{T \in \mathcal{T}(S)} P(T, S) = \pi[1, n, 1]$

A Dynamic Programming Algorithm for the Sum

- Base case definition: for all $i = 1 \ldots n$, for $k = 1 \ldots K$

$$\pi[i, i, k] = P(N_k \rightarrow w_i \mid N_k)$$

- Recursive definition: for all $i = 1 \ldots n$, $j = (i + 1) \ldots n$, $k = 1 \ldots K$,

$$\pi[i, j, k] = \sum_{\substack{i \leq s < j \\ 1 \leq l \leq K \\ 1 \leq m \leq K}} \{P(N_k \rightarrow N_l N_m \mid N_k) \times \pi[i, s, l] \times \pi[s + 1, j, m]\}$$

Initialization:

For i = 1 ... n, k = 1 ... K

$\pi[i, i, k] = P(N_k \rightarrow w_i | N_k)$

Main Loop:

For $length = 1 \ldots (n-1)$, $i = 1 \ldots (n - 1ength)$, $k = 1 \ldots K$

$j \leftarrow i + length$

$sum \leftarrow 0$

For $s = i \ldots (j-1)$,

For $l = 1 \ldots K$,

For $m = 1 \ldots K$,

$prob \leftarrow P(N_k \rightarrow N_l N_m) \times \pi[i, s, l] \times \pi[s+1, j, m]$

$sum \leftarrow sum + prob$

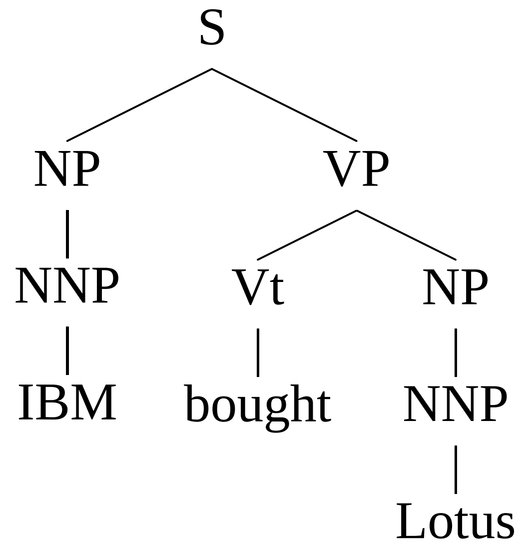
Overview

- An introduction to the parsing problem
- Context free grammars
- A brief(!) sketch of the syntax of English
- Examples of ambiguous structures
- PCFGs, their formal properties, and useful algorithms
- Weaknesses of PCFGs

Weaknesses of PCFGs

- Lack of sensitivity to lexical information

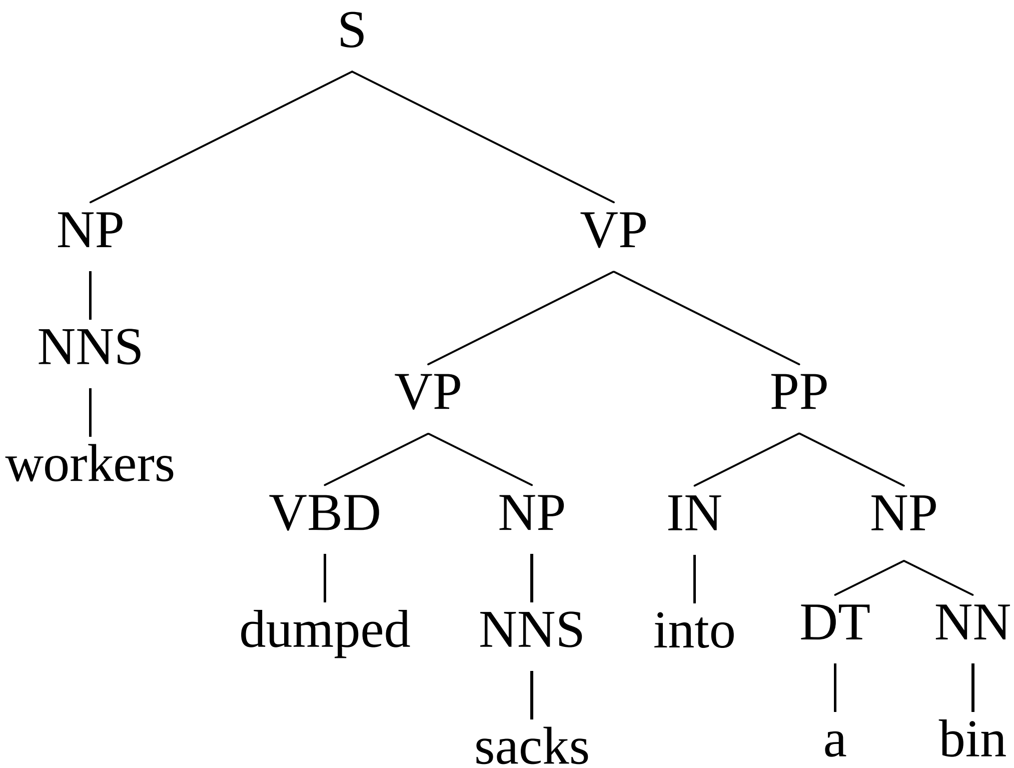
- Lack of sensitivity to structural frequencies



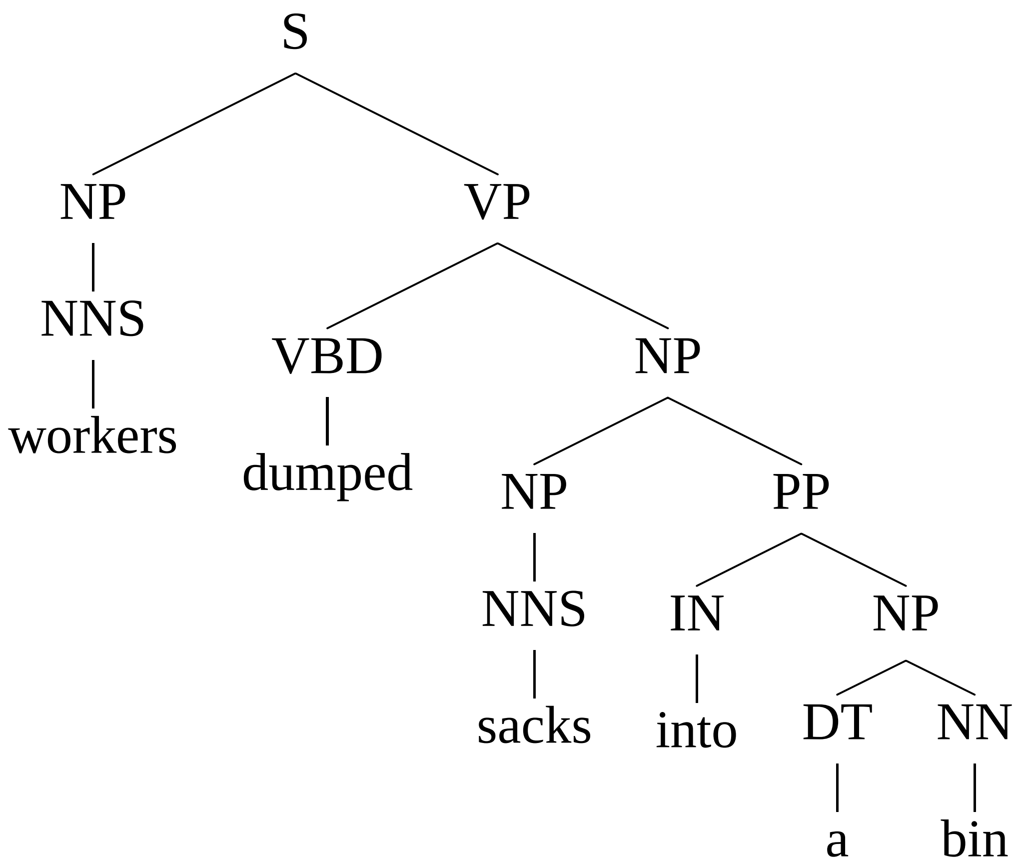
$$
\begin{aligned}
\text{PROB} = \quad & P(\text{S} \rightarrow \text{NP VP} \mid \text{S}) && \times P(\text{NNP} \rightarrow IBM \mid \text{NNP}) \\
& \times P(\text{VP} \rightarrow \text{V NP} \mid \text{VP}) && \times P(\text{Vt} \rightarrow bought \mid \text{Vt}) \\
& \times P(\text{NP} \rightarrow \text{NNP} \mid \text{NP}) && \times P(\text{NNP} \rightarrow Lotus \mid \text{NNP}) \\
& \times P(\text{NP} \rightarrow \text{NNP} \mid \text{NP}) &&
\end{aligned}
$$

Another Case of PP Attachment Ambiguity

(a)



(b)



S
NP
VP
NNS
VBD
NP
workers
dumped
NP
PP
NNS
IN
NP
sacks
into
DT
NN
a
bin

(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)

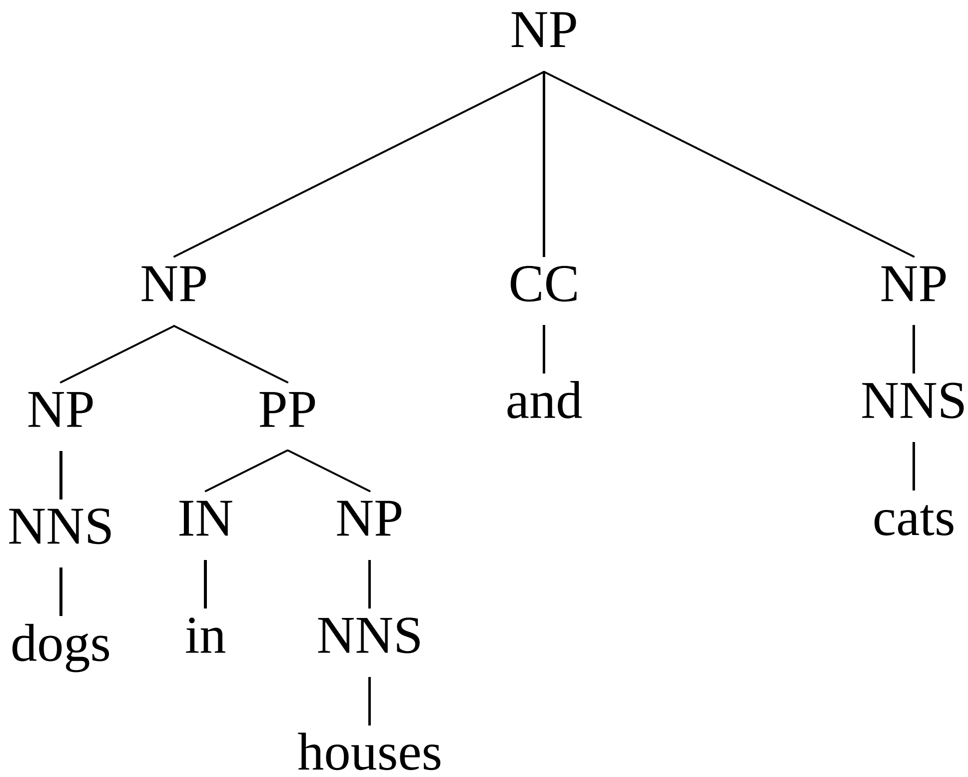
Rules
S $\rightarrow$ NP VP
NP $\rightarrow$ NNS
NP $\rightarrow$ **NP PP**
VP $\rightarrow$ VBD NP
NP $\rightarrow$ NNS
PP $\rightarrow$ IN NP
NP $\rightarrow$ DT NN
NNS $\rightarrow$ workers
VBD $\rightarrow$ dumped
NNS $\rightarrow$ sacks
IN $\rightarrow$ into
DT $\rightarrow$ a
NN $\rightarrow$ bin

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

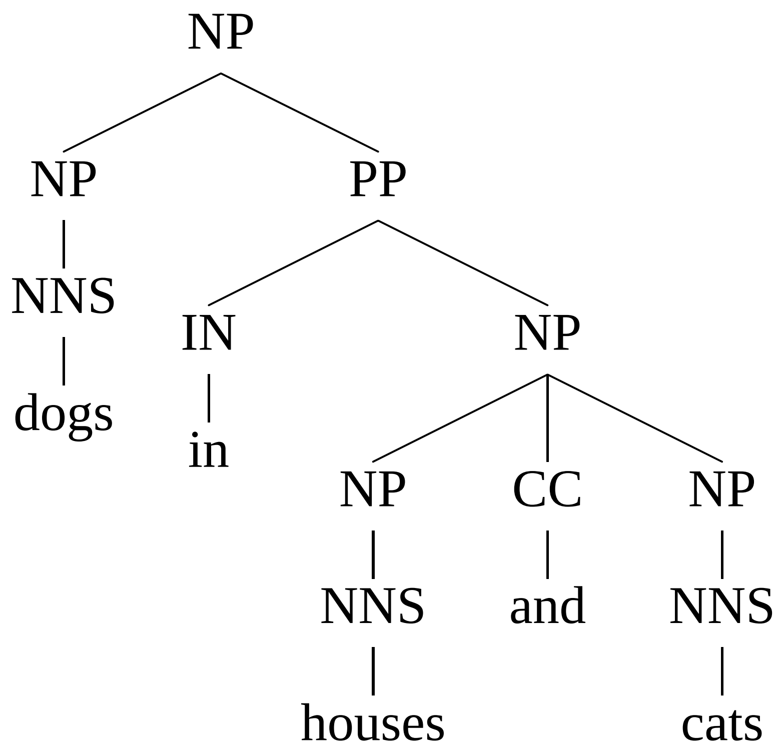
Attachment decision is completely independent of the words

A Case of Coordination Ambiguity

(a)



(b)



(a)

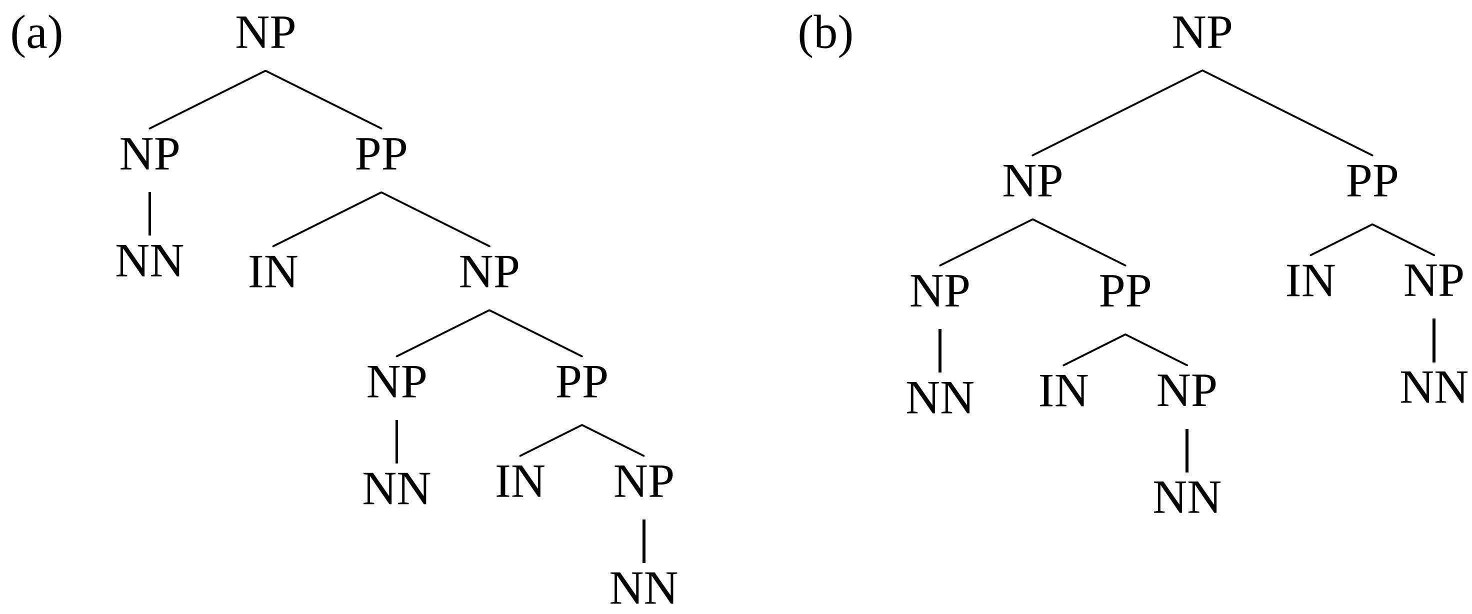
Rules
NP $\rightarrow$ NP CC NP
NP $\rightarrow$ NP PP
NP $\rightarrow$ NNS
PP $\rightarrow$ IN NP
NP $\rightarrow$ NNS
NP $\rightarrow$ NNS
NNS $\rightarrow$ dogs
IN $\rightarrow$ in
NNS $\rightarrow$ houses
CC $\rightarrow$ and
NNS $\rightarrow$ cats

(b)

Rules
NP $\rightarrow$ NP CC NP
NP $\rightarrow$ NP PP
NP $\rightarrow$ NNS
PP $\rightarrow$ IN NP
NP $\rightarrow$ NNS
NP $\rightarrow$ NNS
NNS $\rightarrow$ dogs
IN $\rightarrow$ in
NNS $\rightarrow$ houses
CC $\rightarrow$ and
NNS $\rightarrow$ cats

Here the two parses have identical rules, and therefore have identical probability under any assignment of PCFG rule probabilities

Structural Preferences: Close Attachment



- Example: president of a company in Africa
- Both parses have the same rules, therefore receive same probability under a PCFG
- “Close attachment” (structure (a)) is twice as likely in Wall Street Journal text.

Structural Preferences: Close Attachment

Previous example: John was believed to have been shot by Bill

Here the low attachment analysis (Bill does the *shooting*) contains same rules as the high attachment analysis (Bill does the *believing*), so the two analyses receive same probability.

References

[Altun, Tsochantaridis, and Hofmann, 2003] Altun, Y., I. Tsochantaridis, and T. Hofmann. 2003. Hidden Markov Support Vector Machines. In *Proceedings of ICML 2003*.

[Bartlett 1998] P. L. Bartlett. 1998. The sample complexity of pattern classification with neural networks: the size of the weights is more important than the size of the network, IEEE Transactions on Information Theory, 44(2): 525-536, 1998.

[Bod 98] Bod, R. (1998). *Beyond Grammar: An Experience-Based Theory of Language*. CSLI Publications/Cambridge University Press.

[Booth and Thompson 73] Booth, T., and Thompson, R. 1973. Applying probability measures to abstract languages. *IEEE Transactions on Computers*, C-22(5), pages 442–450.

[Borthwick et. al 98] Borthwick, A., Sterling, J., Agichtein, E., and Grishman, R. (1998). Exploiting Diverse Knowledge Sources via Maximum Entropy in Named Entity Recognition. *Proc. of the Sixth Workshop on Very Large Corpora*.

[Collins and Duffy 2001] Collins, M. and Duffy, N. (2001). Convolution Kernels for Natural Language. In *Proceedings of NIPS 14*.

[Collins and Duffy 2002] Collins, M. and Duffy, N. (2002). New Ranking Algorithms for Parsing and Tagging: Kernels over Discrete Structures, and the Voted Perceptron. In *Proceedings of ACL 2002*.

[Collins 2002a] Collins, M. (2002a). Discriminative Training Methods for Hidden Markov Models: Theory and Experiments with the Perceptron Algorithm. In *Proceedings of EMNLP 2002*.

[Collins 2002b] Collins, M. (2002b). Parameter Estimation for Statistical Parsing Models: Theory and Practice of Distribution-Free Methods. To appear as a book chapter.

[Crammer and Singer 2001a] Crammer, K., and Singer, Y. 2001a. On the Algorithmic Implementation of Multiclass Kernel-based Vector Machines. In *Journal of Machine Learning Research*, 2(Dec):265-292.

[Crammer and Singer 2001b] Koby Crammer and Yoram Singer. 2001b. Ultraconservative Online Algorithms for Multiclass Problems In *Proceedings of COLT 2001*.

[Freund and Schapire 99] Freund, Y. and Schapire, R. (1999). Large Margin Classification using the Perceptron Algorithm. In *Machine Learning*, 37(3):277–296.

[Helmbold and Warmuth 95] Helmbold, D., and Warmuth, M. On Weak Learning. *Journal of Computer and System Sciences*, 50(3):551-573, June 1995.

[Hopcroft and Ullman 1979] Hopcroft, J. E., and Ullman, J. D. 1979. *Introduction to automata theory, languages, and computation*. Reading, Mass.: Addison–Wesley.

[Johnson et. al 1999] Johnson, M., Geman, S., Canon, S., Chi, S., & Riezler, S. (1999). Estimators for stochastic 'unification-based" grammars. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*. San Francisco: Morgan Kaufmann.

[Lafferty et al. 2001] John Lafferty, Andrew McCallum, and Fernando Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In Proceedings of ICML-01, pages 282-289, 2001.

[Littlestone and Warmuth, 1986] Littlestone, N., and Warmuth, M. 1986. Relating data compression and learnability. *Technical report, University of California, Santa Cruz*.

[MSM93] Marcus, M., Santorini, B., & Marcinkiewicz, M. (1993). Building a large annotated corpus of english: The Penn treebank. *Computational Linguistics*, *19*, 313-330.

[McCallum et al. 2000] McCallum, A., Freitag, D., and Pereira, F. (2000) Maximum entropy markov models for information extraction and segmentation. In *Proceedings of ICML 2000*.

[Miller et. al 2000] Miller, S., Fox, H., Ramshaw, L., and Weischedel, R. 2000. A Novel Use of Statistical Parsing to Extract Information from Text. In *Proceedings of ANLP 2000*.

[Ramshaw and Marcus 95] Ramshaw, L., and Marcus, M. P. (1995). Text Chunking Using Transformation-Based Learning. In *Proceedings of the Third ACL Workshop on Very Large Corpora*, Association for Computational Linguistics, 1995.

[Ratnaparkhi 96] A maximum entropy part-of-speech tagger. In *Proceedings of the empirical methods in natural language processing conference*.

[Schapire et al., 1998] Schapire R., Freund Y., Bartlett P. and Lee W. S. 1998. Boosting the margin: A new explanation for the effectiveness of voting methods. *The Annals of Statistics*, 26(5):1651-1686.

[Zhang, 2002] Zhang, T. 2002. Covering Number Bounds of Certain Regularized Linear Function Classes. In *Journal of Machine Learning Research*, 2(Mar):527-550, 2002.