# Introduction to Syntactic Parsing

Roxana Girju

November 18, 2004

Some slides were provided by Michael Collins (MIT) and Dan Moldovan (UT Dallas)

# 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
- Weaknesses of PCFGs
- Heads in CFGs
- Chart parsing – algorithm and an example

# Syntactic Parsing

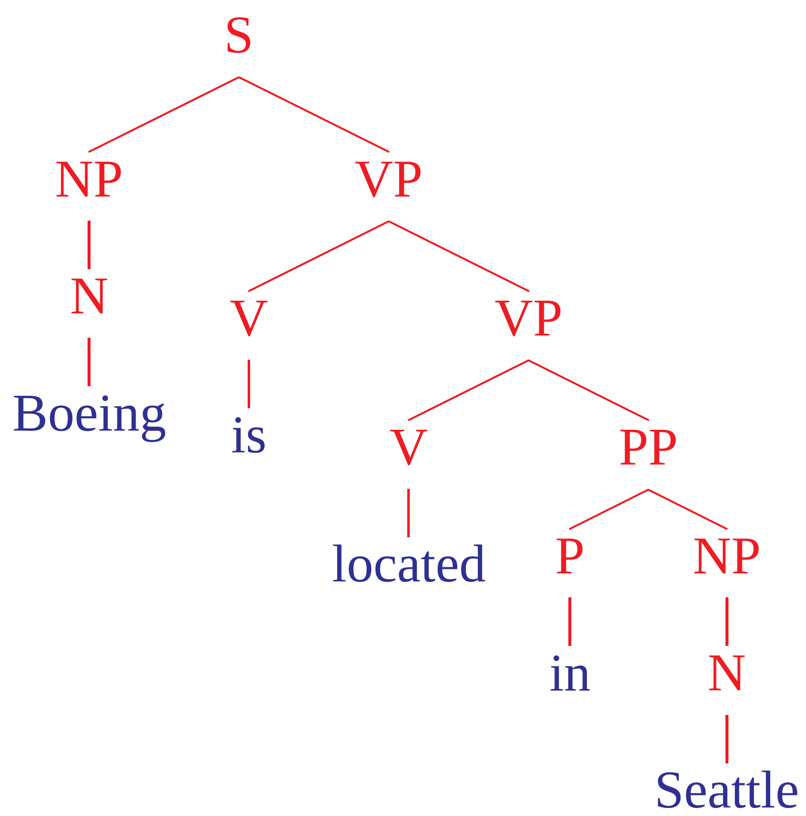
- Syntax: provides rules to put together words to form components of sentence and to put together these components to form sentences.
- Knowledge of syntax is useful for:
  - Parsing
  - QA
  - IE
  - Generation
  - Translation, etc.
- Grammar is the formal specification of rules of a language.
- Parsing is a method to perform syntactic analysis of a sentence.

# Parsing (Syntactic Structure)

INPUT:

Boeing is located in Seattle.

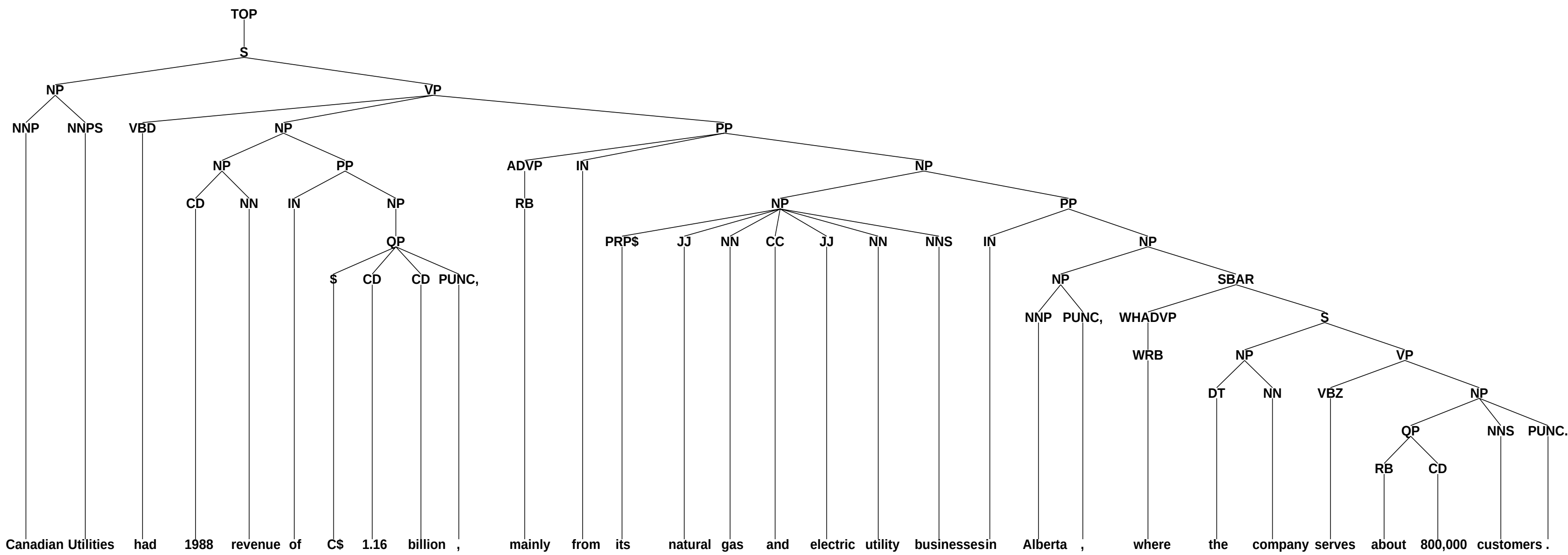
OUTPUT:



# 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:**

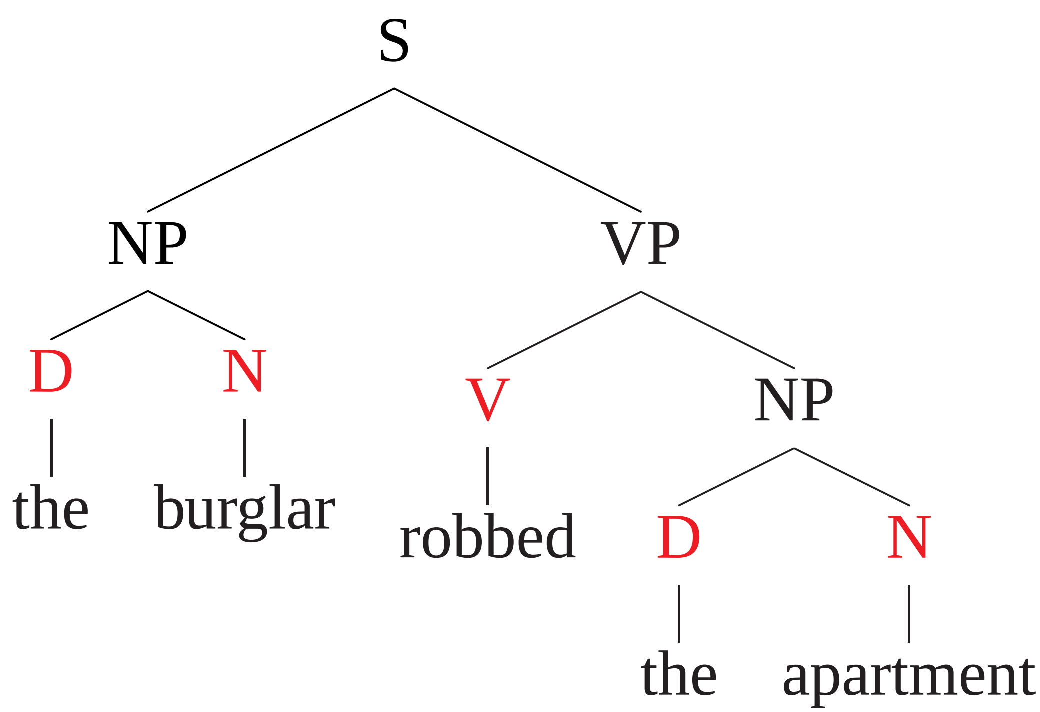



Canadian Utilities had 1988 revenue of C$ 1.16 billion , mainly from its natural gas and electric utility businesses in Alberta , where the company serves about 800,000 customers .

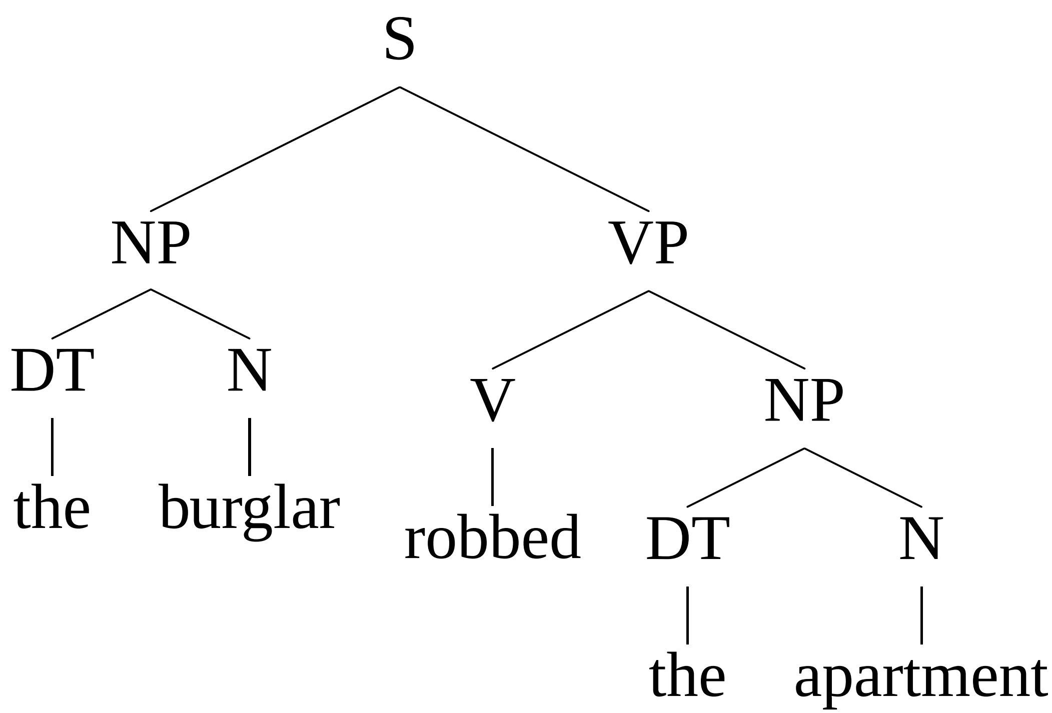
# The Information Conveyed by Parse Trees

**1)** Part of speech for each word

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



**2)** Phrases

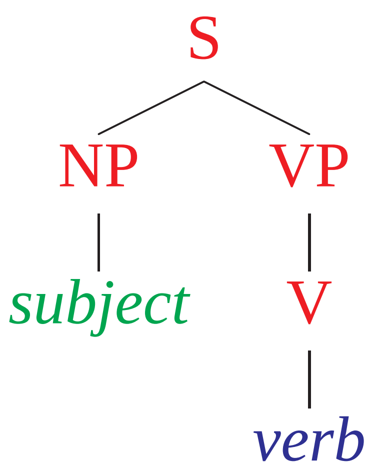


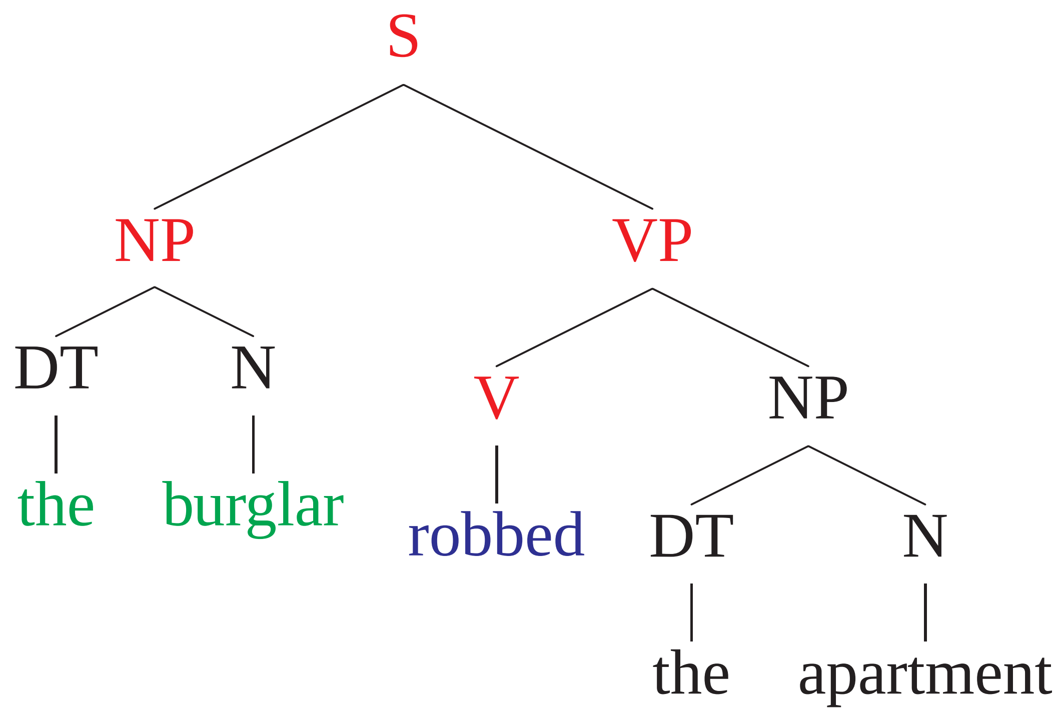

Noun Phrases (NP): “the burglar”, “the apartment”

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
- $\Sigma$ is a set of terminal symbols
- $R$ is a set of rules of the form $X \rightarrow Y_1 Y_2 \ldots Y_n$ 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

$\Sigma$ = {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

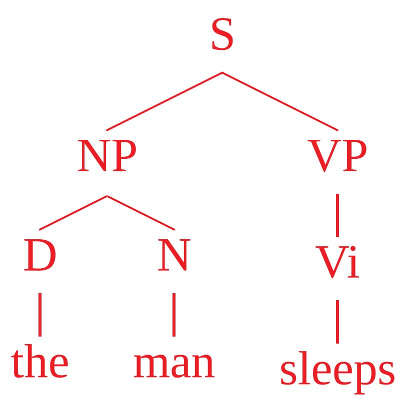
# Left-Most Derivations

A left-most derivation is a sequence of strings $s_1 \ldots s_n$, 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 $i = 2 \ldots n$ is derived from $s_{i-1}$ by picking the left-most non-terminal $X$ in $s_{i-1}$ and replacing it by some $\beta$ 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 | $S \to NP\ VP$ |
| NP VP | $NP \to DT\ N$ |
| DT N VP | $DT \to the$ |
| 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

S

NP VP

DT N VP

the N VP

the dog VP

the dog VB

the dog laughs

## RULES USED

S $\rightarrow$ NP VP

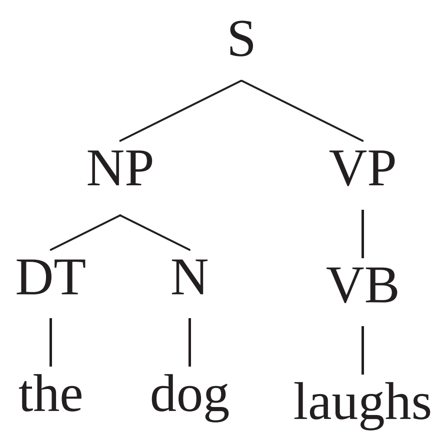
NP $\rightarrow$ DT N

DT $\rightarrow$ the

N $\rightarrow$ dog

VP $\rightarrow$ VB

VB $\rightarrow$ laughs



# Properties of CFGs

- A CFG defines a set of possible derivations

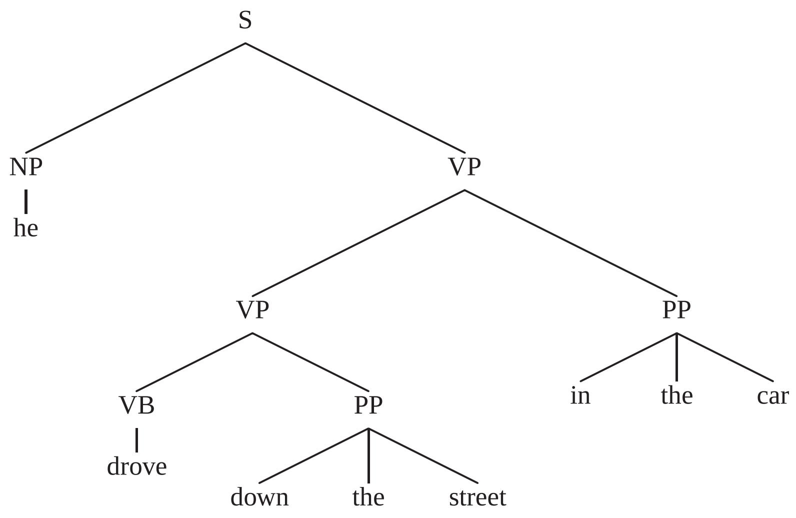
- A string $s \in \Sigma^*$ is in the *language* defined by the CFG if there is at least one derivation which yields $s$

- Each string in the language generated by the CFG may have more than one derivation (“ambiguity”)

DERIVATION

S

RULES USED



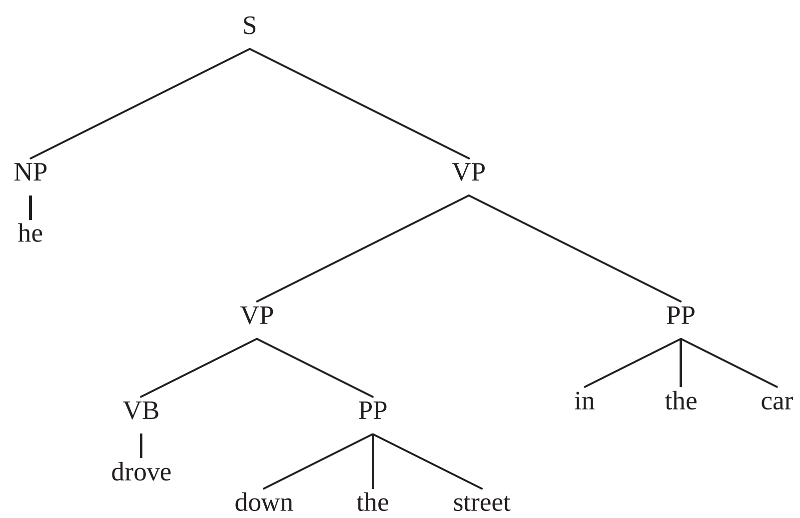
DERIVATION

S

NP VP

RULES USED

S $\rightarrow$ NP VP



DERIVATION

S

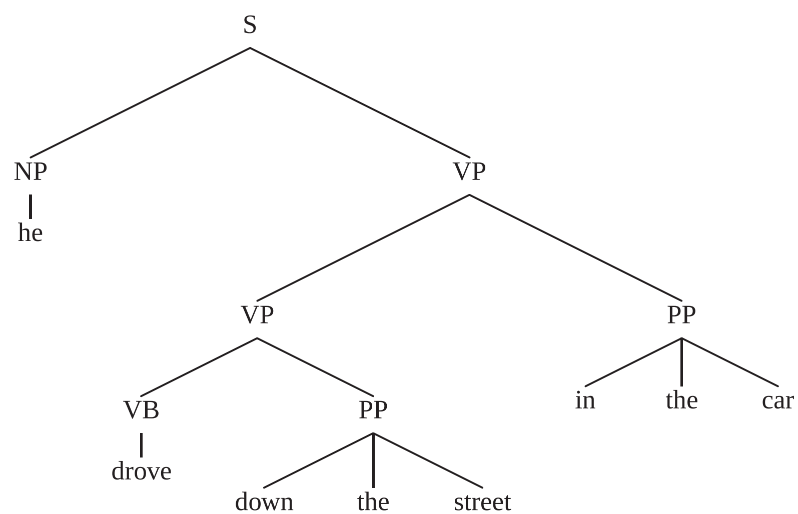
NP VP

he VP

RULES USED

S $\rightarrow$ NP VP

NP $\rightarrow$ he



DERIVATION

S

NP VP

he VP

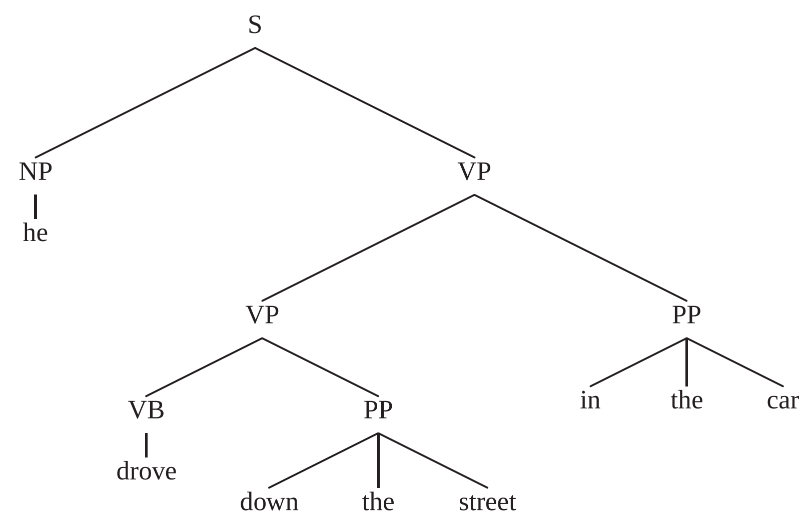
he VP PP

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

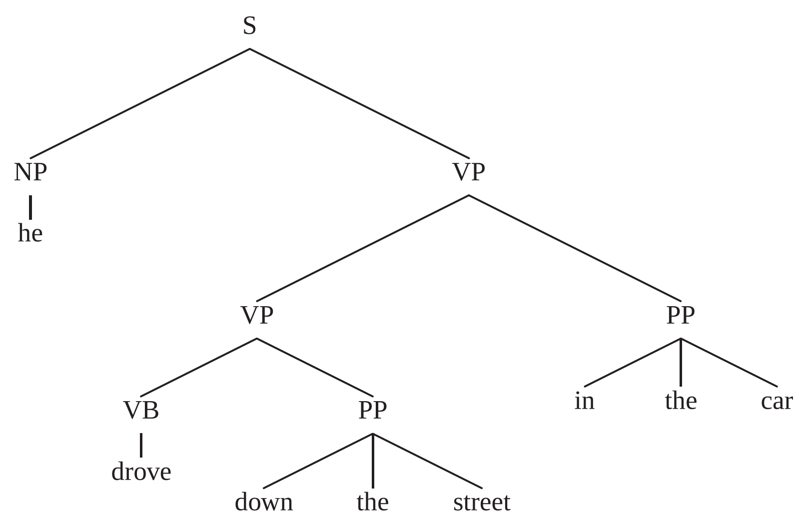
RULES USED

S $\rightarrow$ NP VP

NP $\rightarrow$ he

VP $\rightarrow$ VP PP

VP $\rightarrow$ VB PP



DERIVATION

S

NP VP

he VP

he VP PP

he VB PP PP

he drove PP PP

RULES USED

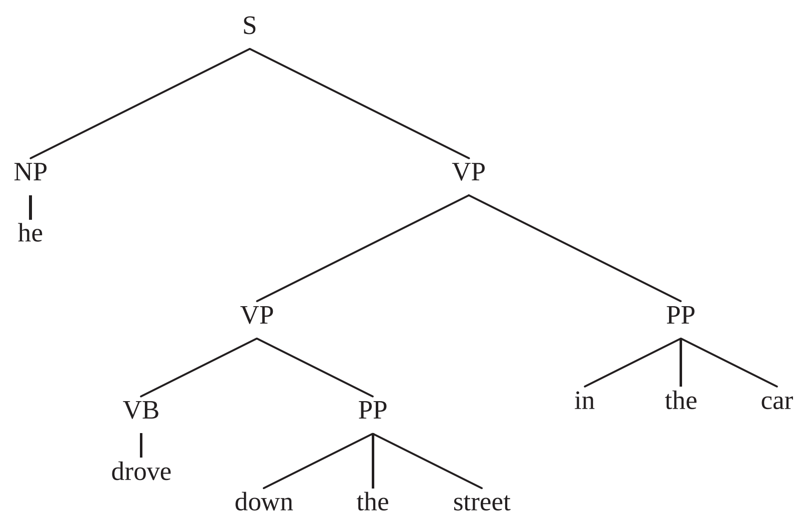
$S \rightarrow NP\ VP$

$NP \rightarrow he$

$VP \rightarrow VP\ PP$

$VP \rightarrow VB\ PP$

$VB \rightarrow drove$



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

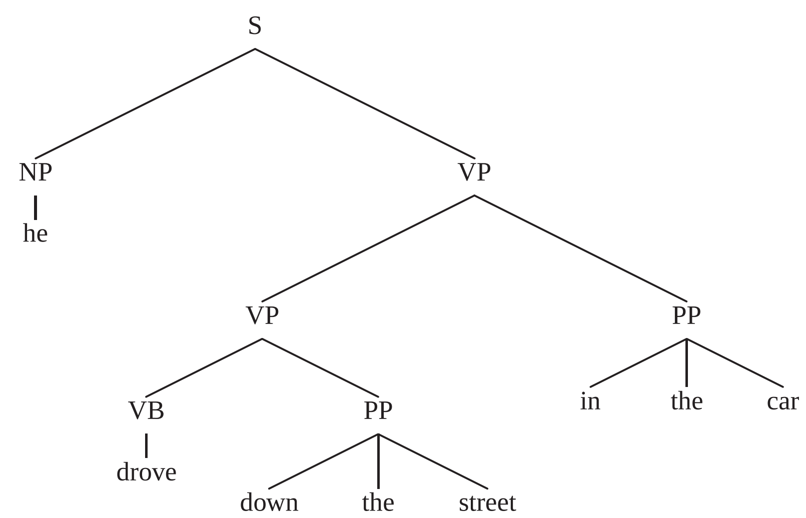
NP $\rightarrow$ he

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

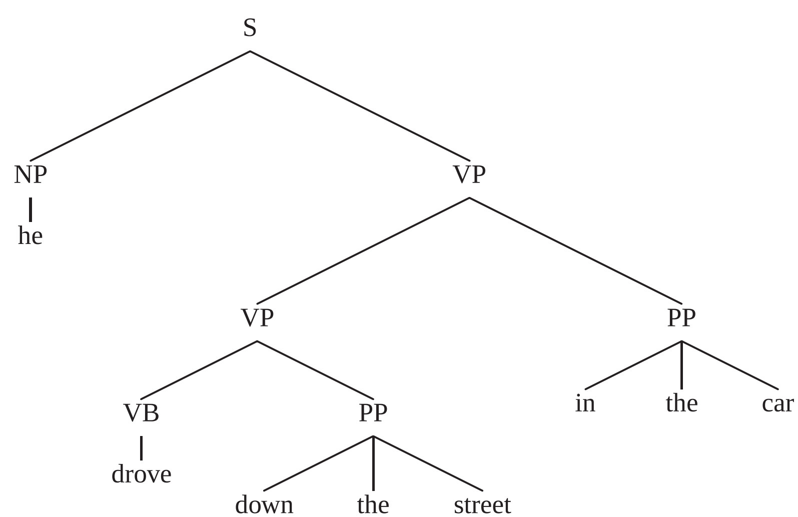
VP $\rightarrow$ VP PP

VP $\rightarrow$ VB PP

VB$\rightarrow$ drove

PP$\rightarrow$ down the street

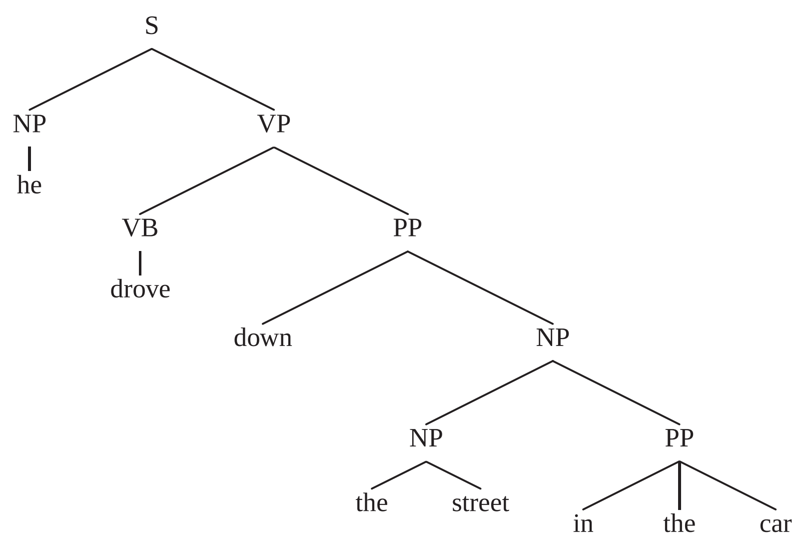
PP$\rightarrow$ in the car



DERIVATION

RULES USED

S



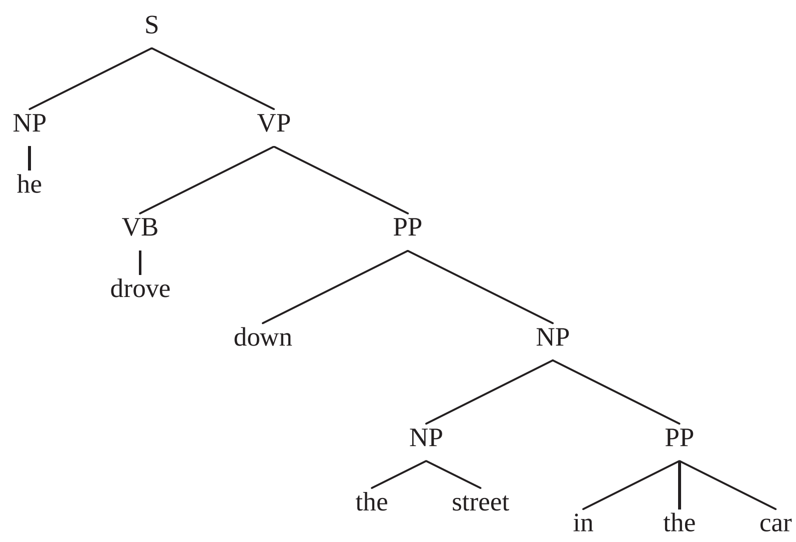
DERIVATION

S

NP VP

RULES USED

$S \rightarrow NP\ VP$



DERIVATION

S

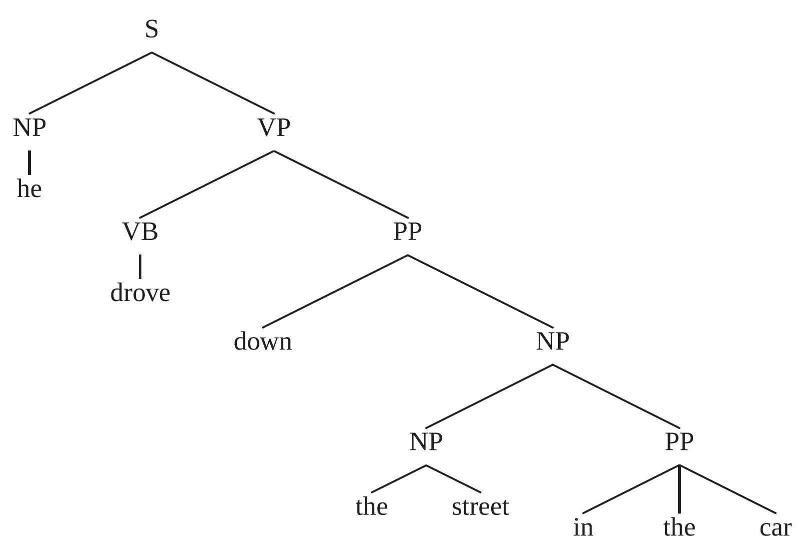
NP VP

he VP

RULES USED

$S \rightarrow NP\ VP$

$NP \rightarrow he$



DERIVATION

S

NP VP

he VP

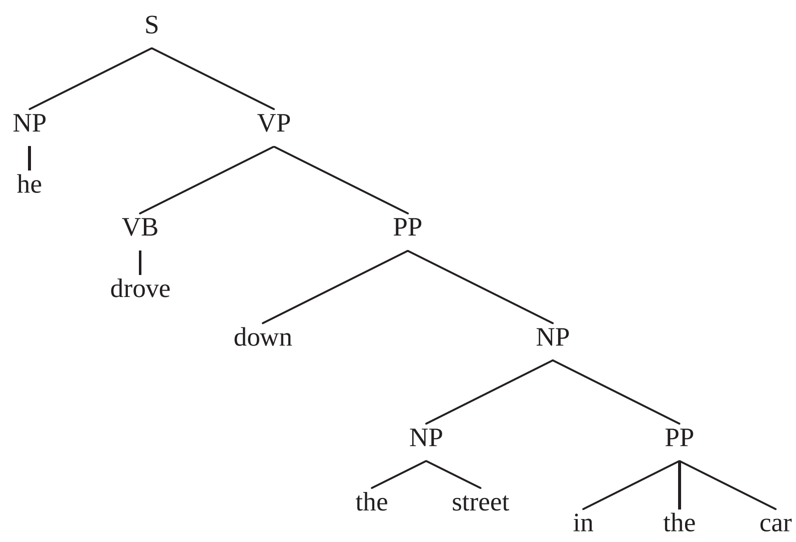
he VB PP

RULES USED

S $\rightarrow$ NP VP

NP $\rightarrow$ he

VP $\rightarrow$ VB PP



DERIVATION

S

NP VP

he VP

he VB PP

he drove PP

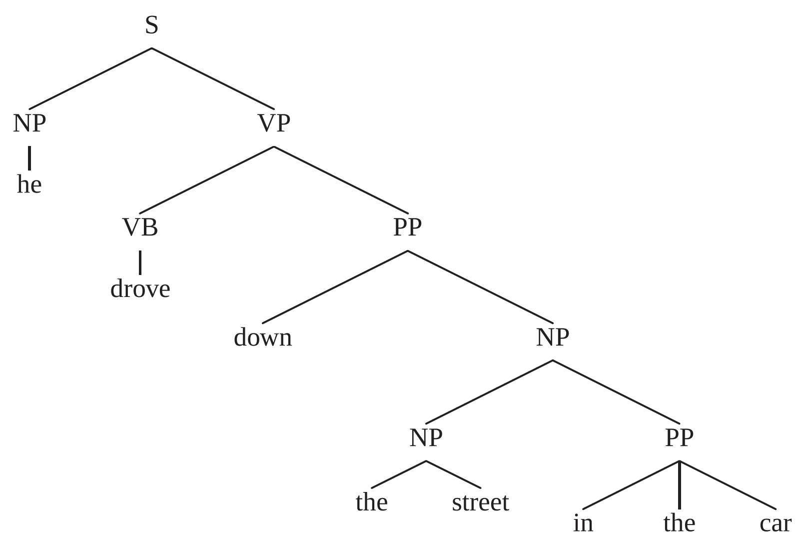
RULES USED

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

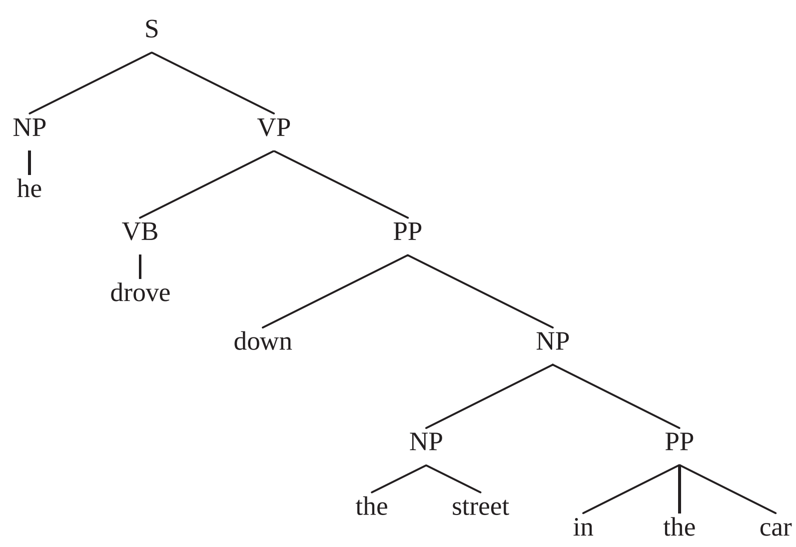
$S \rightarrow NP\ VP$

$NP \rightarrow he$

$VP \rightarrow VB\ PP$

$VB \rightarrow drove$

$PP \rightarrow down\ NP$



DERIVATION

S

NP VP

he VP

he VB PP

he drove PP

he drove down NP

he drove down NP PP

RULES USED

S $\rightarrow$ NP VP

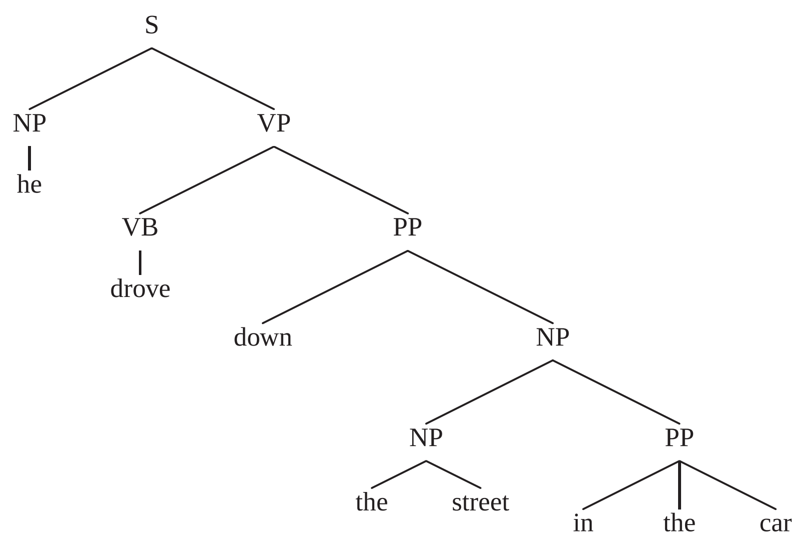
NP $\rightarrow$ he

VP $\rightarrow$ VB PP

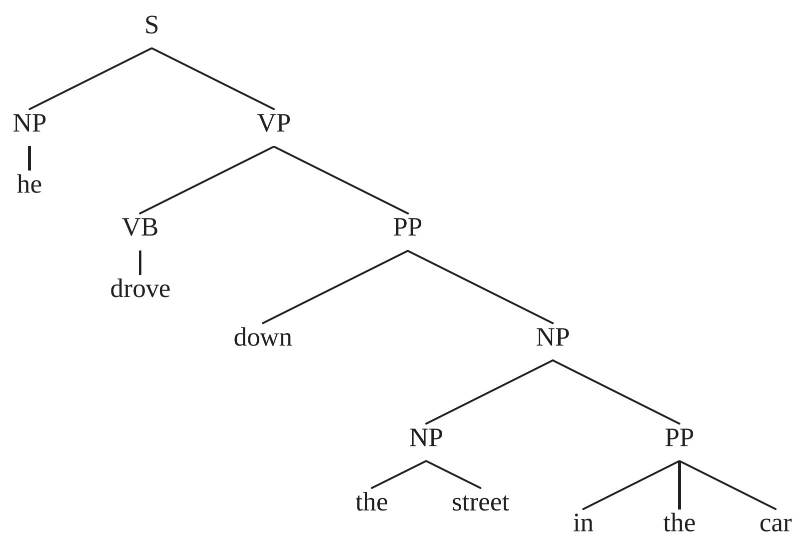
VB $\rightarrow$ drove

PP $\rightarrow$ down NP

NP $\rightarrow$ NP PP



| 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 | NP $\rightarrow$ the street |
| he drove down the street 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

he drove down the street in the car

RULES USED

S $\rightarrow$ NP VP

NP $\rightarrow$ he

VP $\rightarrow$ VB PP

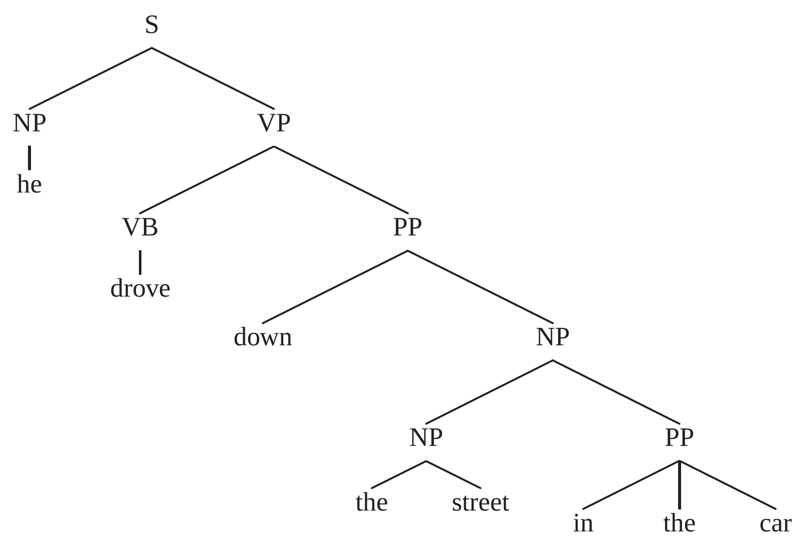
VB $\rightarrow$ drove

PP $\rightarrow$ down NP

NP $\rightarrow$ NP PP

NP $\rightarrow$ the street

PP $\rightarrow$ in the car



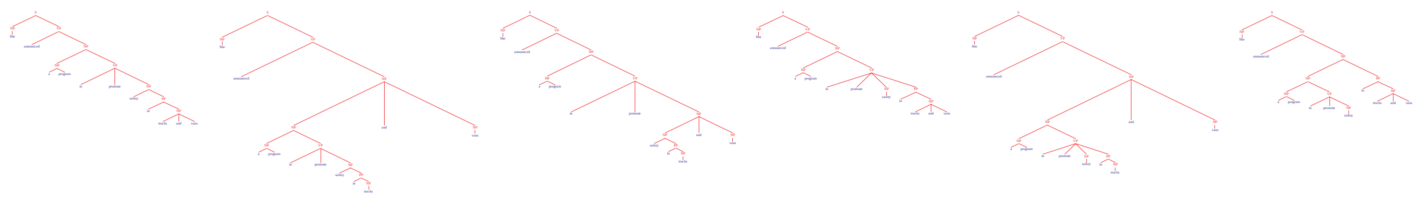
# The Problem with Parsing: Ambiguity

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

  $$\begin{array}{llll} \mathrm{VP} & \rightarrow & \mathrm{Vi} & \\ \mathrm{VP} & \rightarrow & \mathrm{Vt} & \mathrm{NP} \\ \mathrm{VP} & \rightarrow & \mathrm{Vd} & \mathrm{NP} \quad \mathrm{NP} \end{array}$$

- Basic S Rule

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

**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

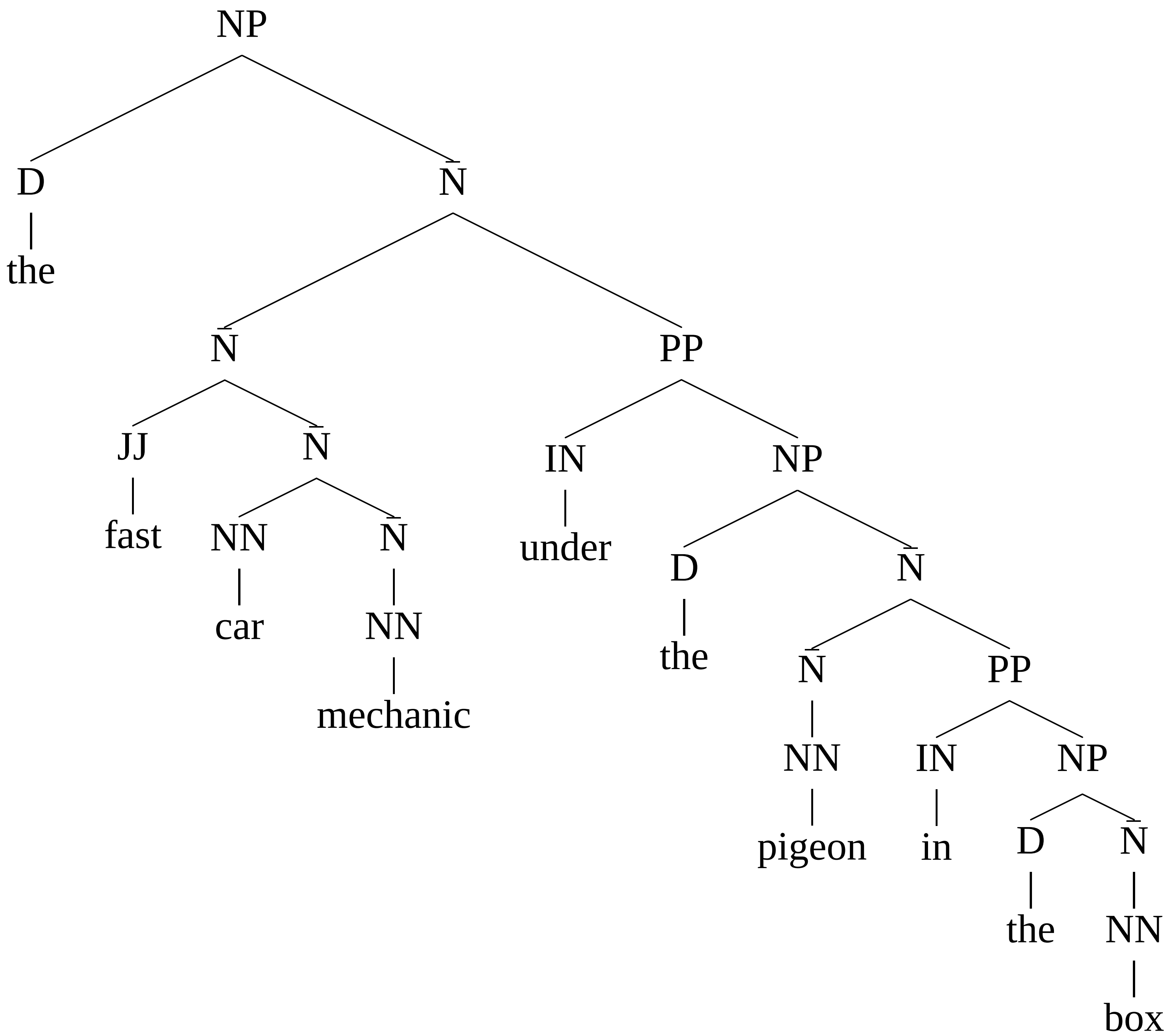
- New Rules

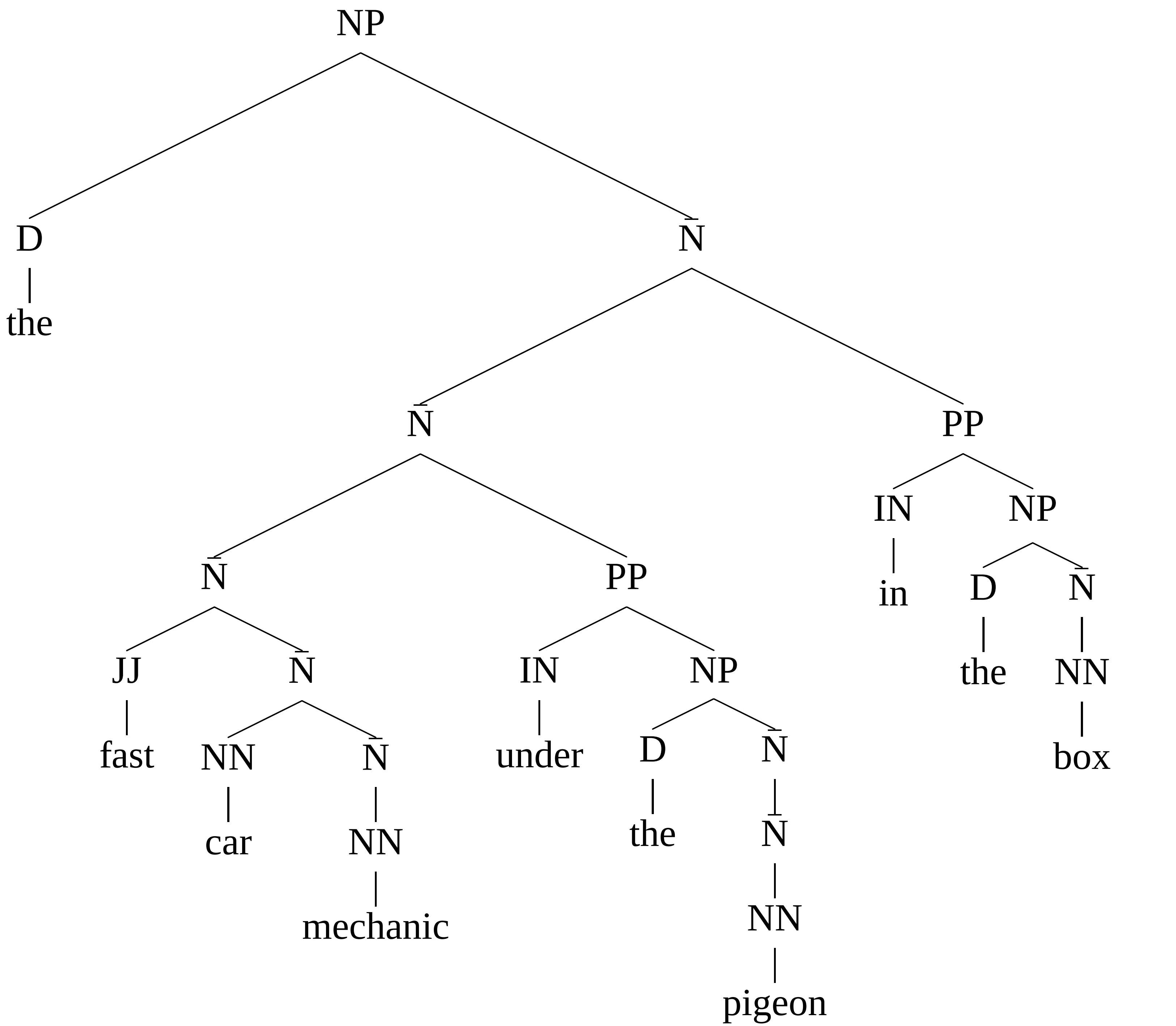
| | | | | |
|---|---|---|---|---|
| NP | $\rightarrow$ | NP | CC | NP |
| $\bar{N}$ | $\rightarrow$ | $\bar{N}$ | CC | $\bar{N}$ |
| VP | $\rightarrow$ | VP | CC | VP |
| S | $\rightarrow$ | S | CC | S |
| SBAR | $\rightarrow$ | SBAR | CC | SBAR |

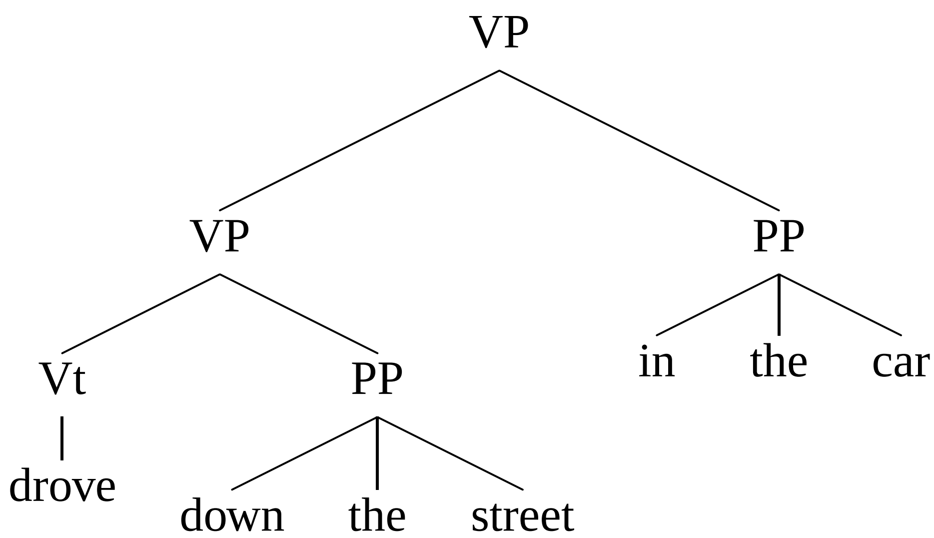
# Sources of Ambiguity

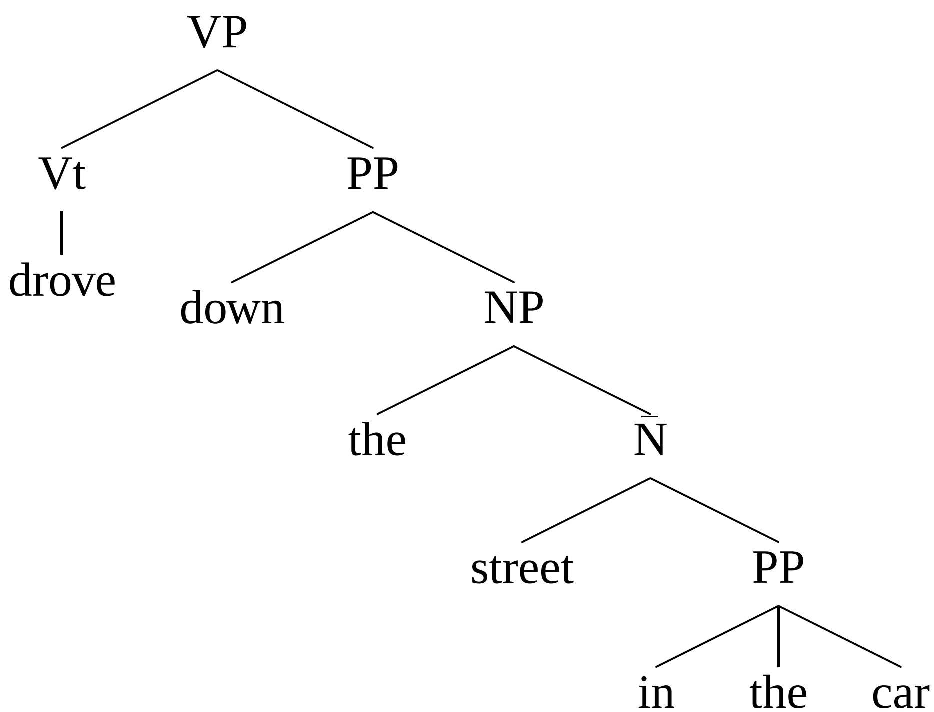
- Part-of-Speech ambiguity
  NNS $\rightarrow$ walks
  Vi $\rightarrow$ walks

- Prepositional Phrase Attachment
  the fast car mechanic under the pigeon in the box


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

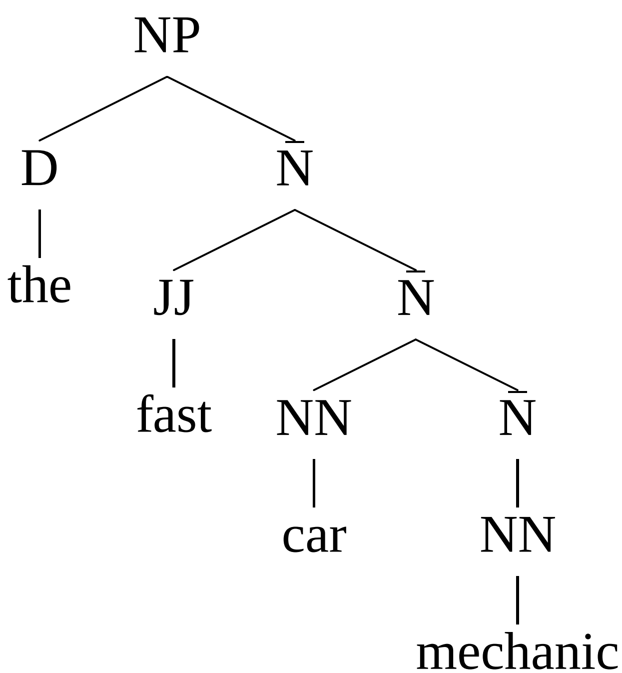

NP
D
the
N̄
N̄
PP
N̄
PP
IN
NP
in
D
N̄
JJ
N̄
IN
NP
the
NN
fast
NN
N̄
under
D
N̄
box
car
NN
the
N
mechanic
NN
pigeon

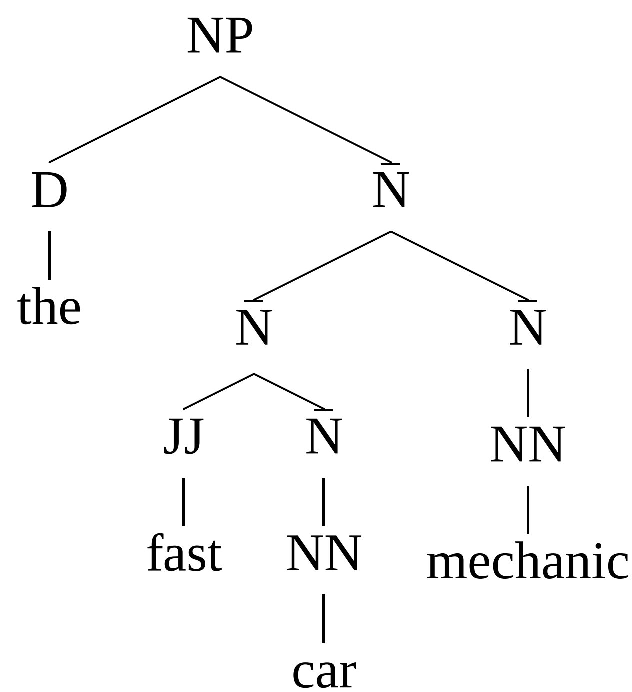

VP
VP
PP
Vt
PP
in the car
drove
down the street


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

# Sources of Ambiguity: Noun Premodifiers

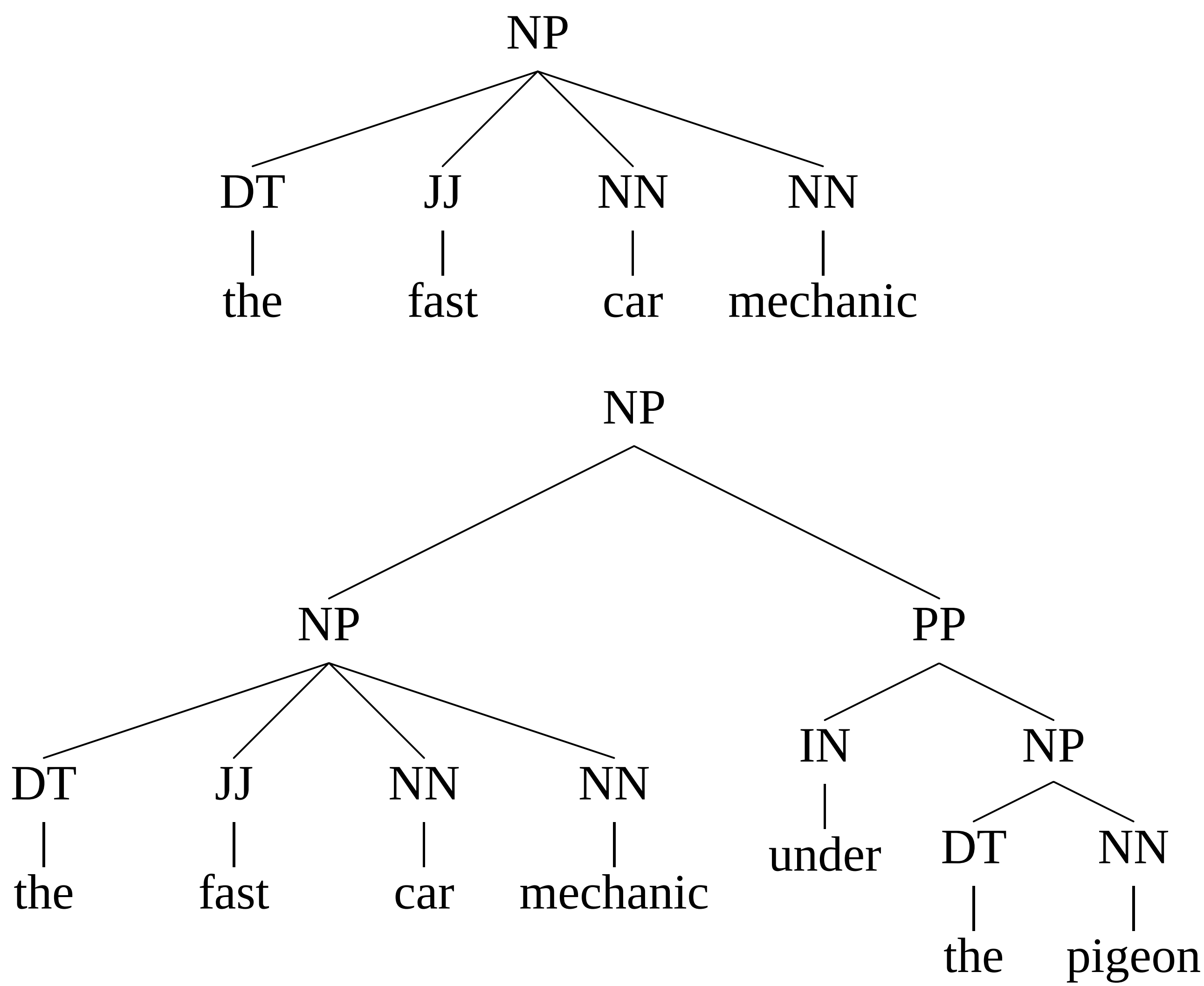
- Noun premodifiers:



# 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.

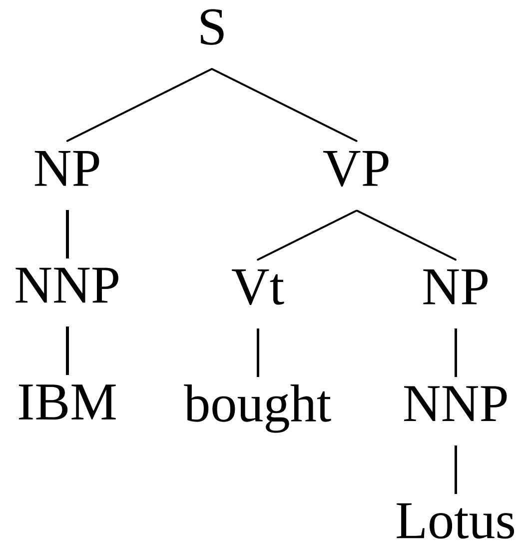
# Overview

- 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

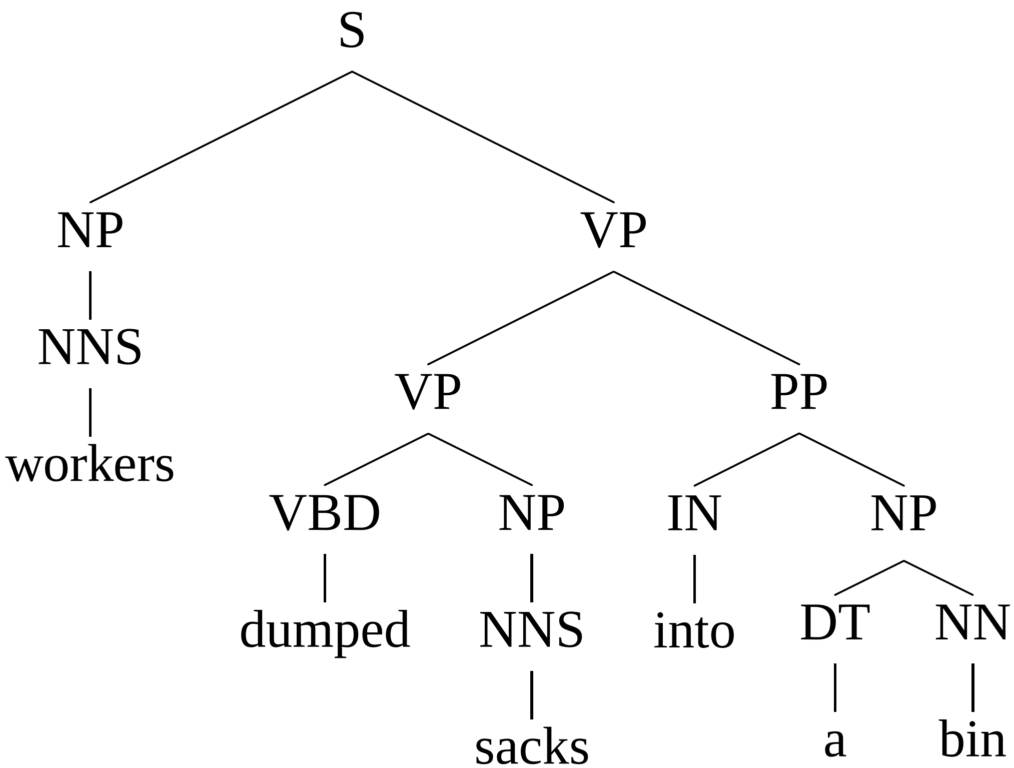
- Lack of sensitivity to structural frequencies



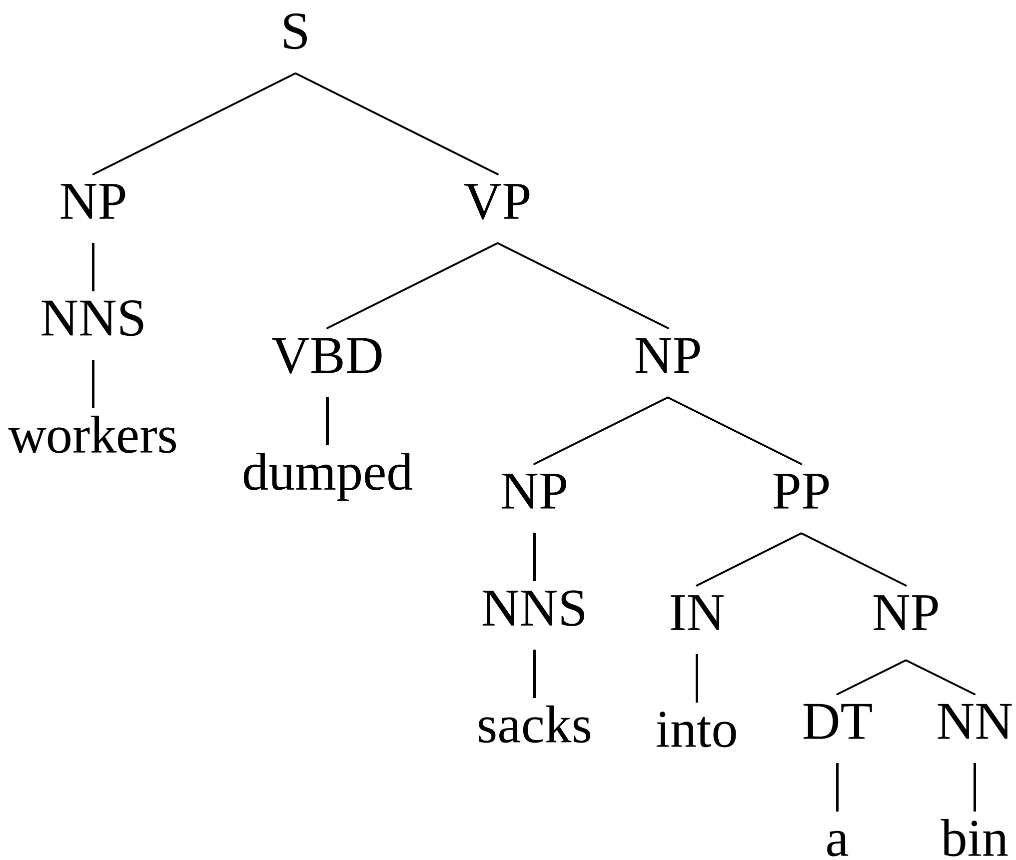

$$
\begin{aligned}
\text{PROB} = \quad & P(\text{S} \rightarrow \text{NP VP} \mid \text{S}) & \times P(\text{NNP} \rightarrow IBM \mid \text{NNP}) \\
& \times P(\text{VP} \rightarrow \text{V NP} \mid \text{VP}) & \times P(\text{Vt} \rightarrow bought \mid \text{Vt}) \\
& \times P(\text{NP} \rightarrow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

# 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)

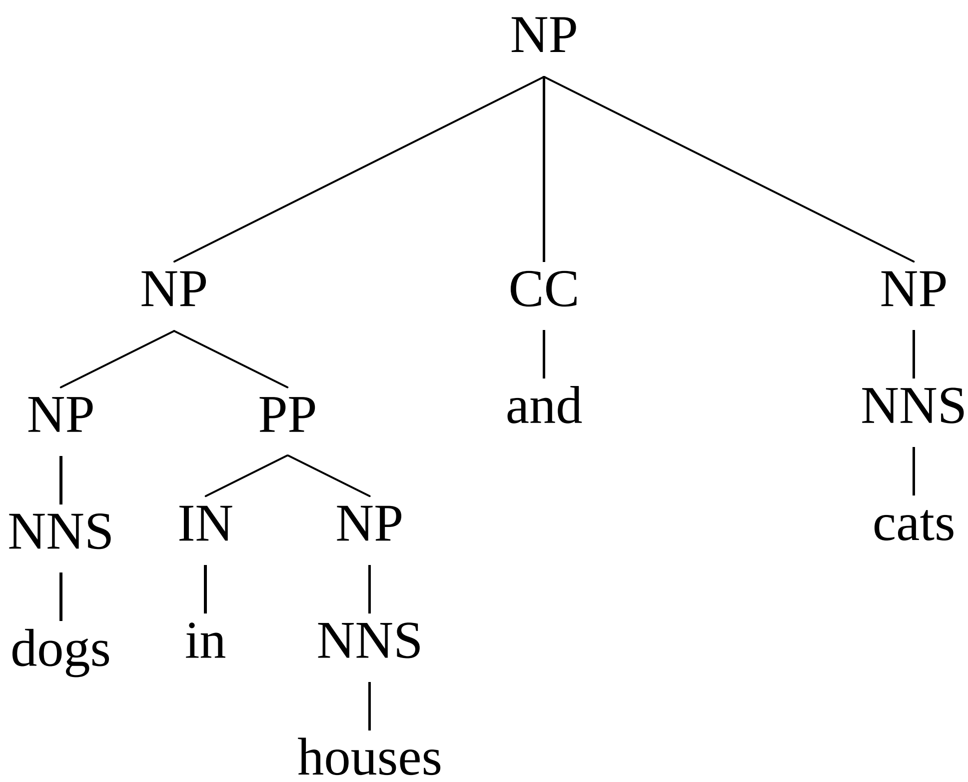
| Rules |
|---|
| S $\rightarrow$ NP VP |
| NP $\rightarrow$ NNS |
| **NP $\rightarrow$ NP PP** |
| VP $\rightarrow$ VBD NP |
| NP $\rightarrow$ NNS |
| PP $\rightarrow$ IN NP |
| NP $\rightarrow$ DT NN |
| NNS $\rightarrow$ workers |
| VBD $\rightarrow$ dumped |
| NNS $\rightarrow$ sacks |
| IN $\rightarrow$ 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.

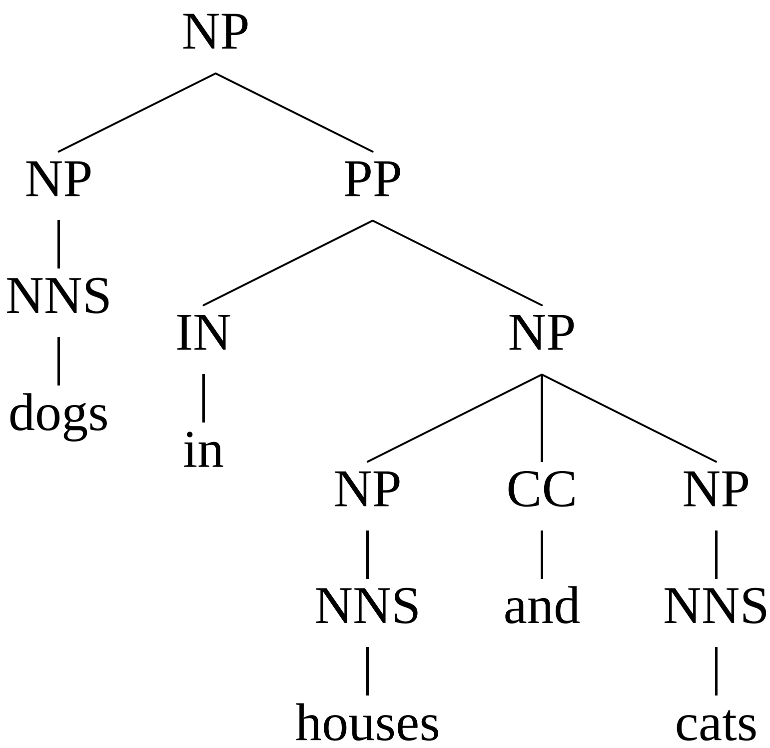
**Attachment decision is completely independent of the words**

# A Case of Coordination Ambiguity

(a)



(b)



(a)

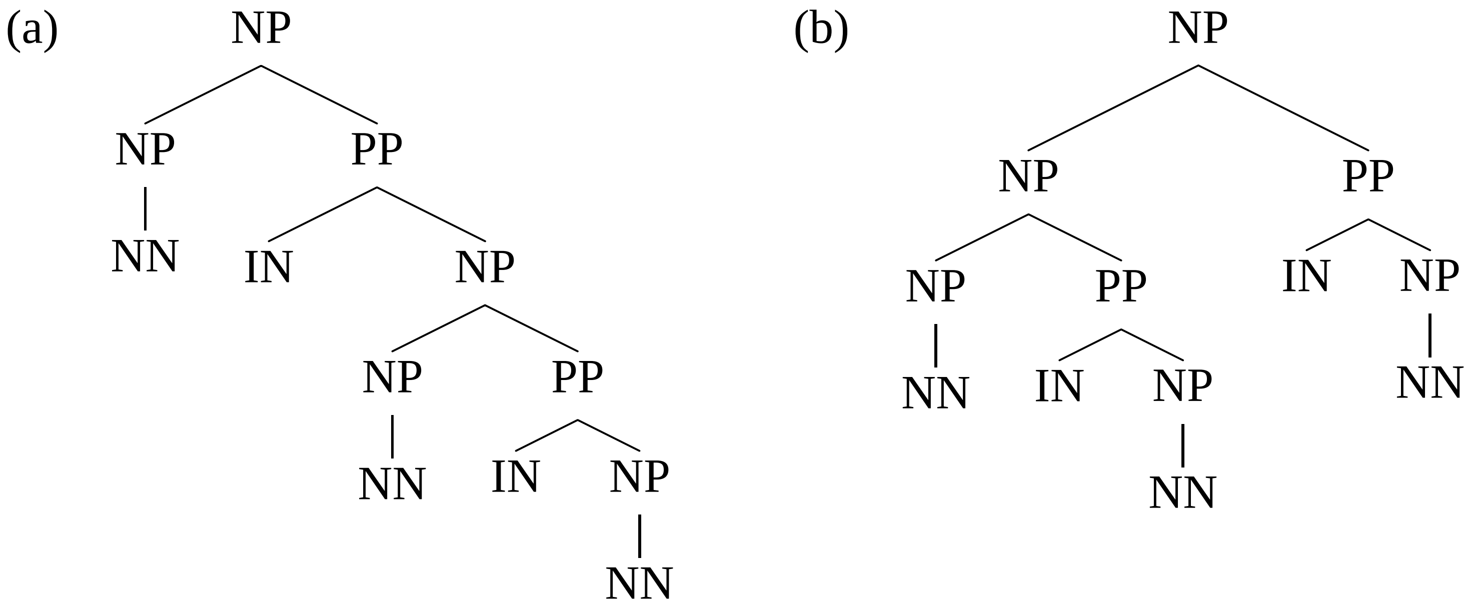
| Rules |
| --- |
| NP $\rightarrow$ NP CC NP |
| NP $\rightarrow$ NP PP |
| NP $\rightarrow$ NNS |
| PP $\rightarrow$ IN NP |
| NP $\rightarrow$ NNS |
| NP $\rightarrow$ NNS |
| NNS $\rightarrow$ dogs |
| IN $\rightarrow$ in |
| NNS $\rightarrow$ houses |
| CC $\rightarrow$ and |
| NNS $\rightarrow$ 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.

# Heads in Context-Free Rules

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

| | | | |
|---|---|---|---|
| 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

# More about Heads

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

| | | | | | |
|---|---|---|---|---|---|
| S | $\Rightarrow$ | NP | VP | | (VP is the head) |
| VP | $\Rightarrow$ | Vt | NP | | (Vt is the head) |
| NP | $\Rightarrow$ | DT | NN | NN | (NN is the head) |

- A core idea in linguistics
(X-bar Theory, Head-Driven Phrase Structure Grammar)

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

# Rules which Recover Heads: An Example of rules for NPs

**If** the rule contains NN, NNS, or NNP:
Choose the rightmost NN, NNS, or NNP

**Else If** the rule contains an NP: Choose the leftmost NP

**Else If** the rule contains a JJ: Choose the rightmost JJ

**Else If** the rule contains a CD: Choose the rightmost CD

**Else** Choose the rightmost child

e.g.,

| | | | | |
|---|---|---|---|---|
| NP | $\Rightarrow$ | DT | NNP | NN |
| NP | $\Rightarrow$ | DT | NN | NNP |
| NP | $\Rightarrow$ | NP | PP | |
| NP | $\Rightarrow$ | DT | JJ | |
| NP | $\Rightarrow$ | DT | | |

# Rules which Recover Heads: An Example of rules for VPs

**If** the rule contains Vi or Vt: Choose the leftmost Vi or Vt

**Else If** the rule contains an VP: Choose the leftmost VP

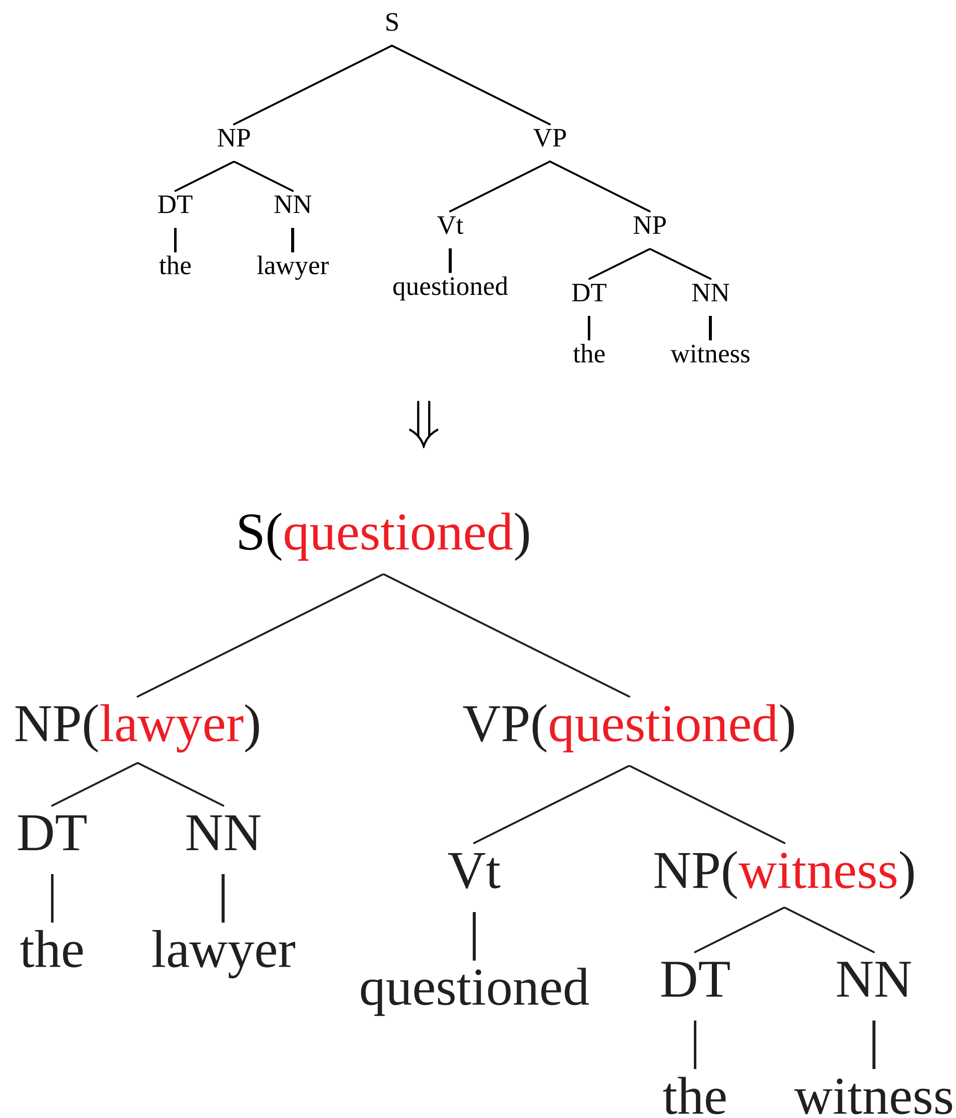
**Else** Choose the leftmost child

e.g.,

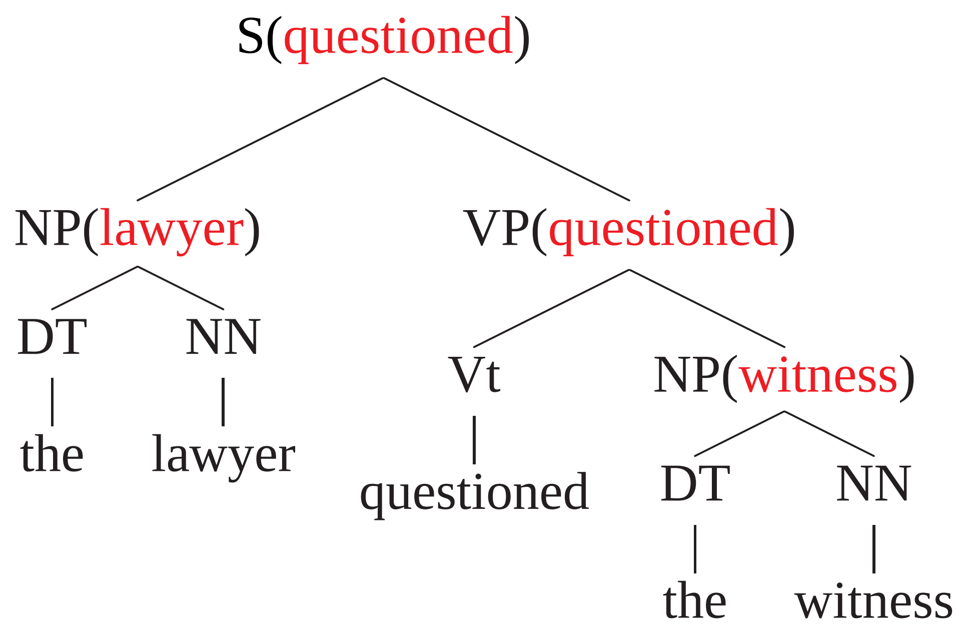
$$\text{VP} \Rightarrow \text{Vt} \quad \text{NP}$$

$$\text{VP} \Rightarrow \text{VP} \quad \text{PP}$$

# Adding Headwords to Trees



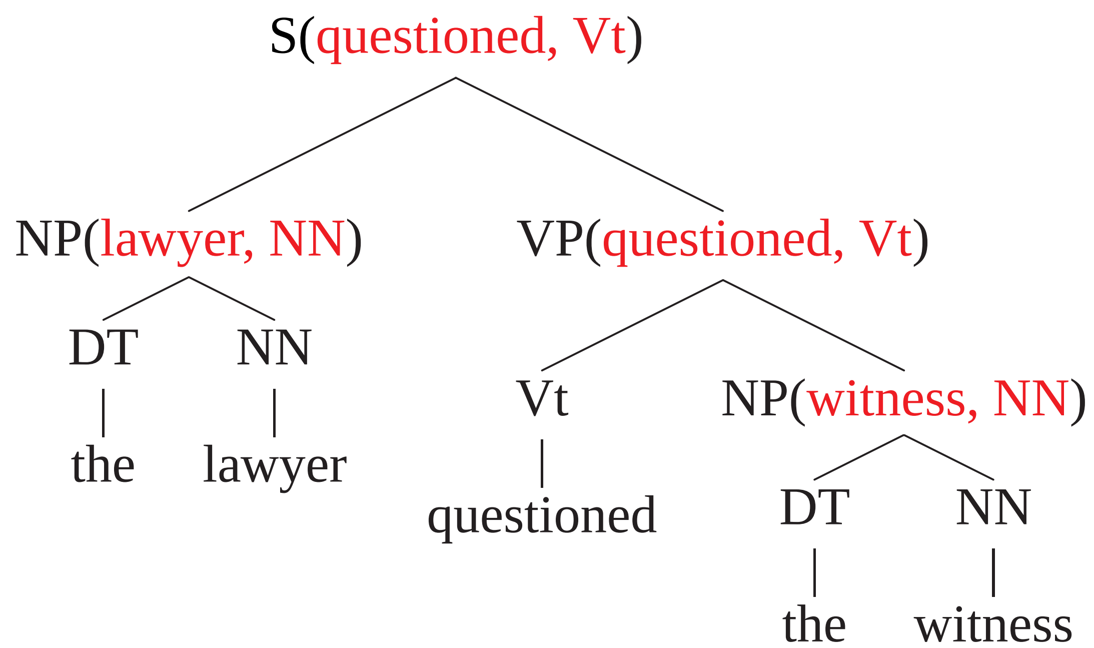
# Adding Headwords to Trees




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

| | | | | | |
|---|---|---|---|---|---|
| S | $\Rightarrow$ | NP | VP | | (S receives headword from VP) |
| VP | $\Rightarrow$ | Vt | NP | | (VP receives headword from Vt) |
| NP | $\Rightarrow$ | DT | | NN | (NP receives headword from NN) |

# Adding Headtags to Trees




- Also propogate **part-of-speech tags** up the trees
  (We'll see soon why this is useful!)

# A Bottom-Up Chart Parser

- The main difference between top-down and bottom-up parser is the way the grammar rules are used
- The basic operation in bottom-up parsing is to take a sequence of symbols and match it to the right-hand side of the rules
  - rewrite a word by its possible lexical categories
  - replace a sequence of symbols that matches the right-hand side of the grammar rule by its left-hand side symbol
  - use a chart structure to keep track of the partial results, so that the work need not be reduplicated

# A Bottom-Up Chart Parser (The Algorithm)

To add a constituent C from position $p_1$ to $p_2$:

1. Insert C into the chart from position $p_1$ to $p_2$.
2. For any active arc of the form $X \rightarrow X_1 \ldots \circ C \ldots X_n$ from position $p_0$ to $p_1$, add a new active arc $X \rightarrow X_1 \ldots C \circ \ldots X_n$ from position $p_0$ to $p_2$.
3. For any active arc of the form $X \rightarrow X_1 \ldots X_n \circ C$ from position $p_0$ to $p_1$, then add a new constituent of type X from $p_0$ to $p_2$ to the agenda.

**Figure 3.10** The arc extension algorithm

Do until there is no input left:

1. If the agenda is empty, look up the interpretations for the next word in the input and add them to the agenda.
2. Select a constituent from the agenda (let's call it constituent C from position $p_1$ to $p_2$).
3. For each rule in the grammar of form $X \rightarrow C\ X_1 \ldots X_n$, add an active arc of form $X \rightarrow \circ\ C\ X_1 \ldots X_n$ from position $p_1$ to $p_2$.
4. Add C to the chart using the arc extension algorithm above.

**Figure 3.11** A bottom-up chart parsing algorithm

# A Bottom-Up Chart Parser
# (An Example) 1/5

- Let's consider the sentence to be parsed:
  - $_1$ *The* $_2$ *large* $_3$ *can* $_4$ *can* $_5$ *hold* $_6$ *the* $_7$ *water* $_8$
- Lexicon:
  - the: ART
  - large: ADJ
  - can: N, AUX, V
  - hold: N, V
  - water: N,V
- Grammar:

1. S → NP VP
2. NP → ART ADJ N
3. NP → ART N
4. NP → ADJ N
5. VP → AUX VP
6. VP → V NP

# A Bottom-Up Chart Parser
(An Example) 2/5

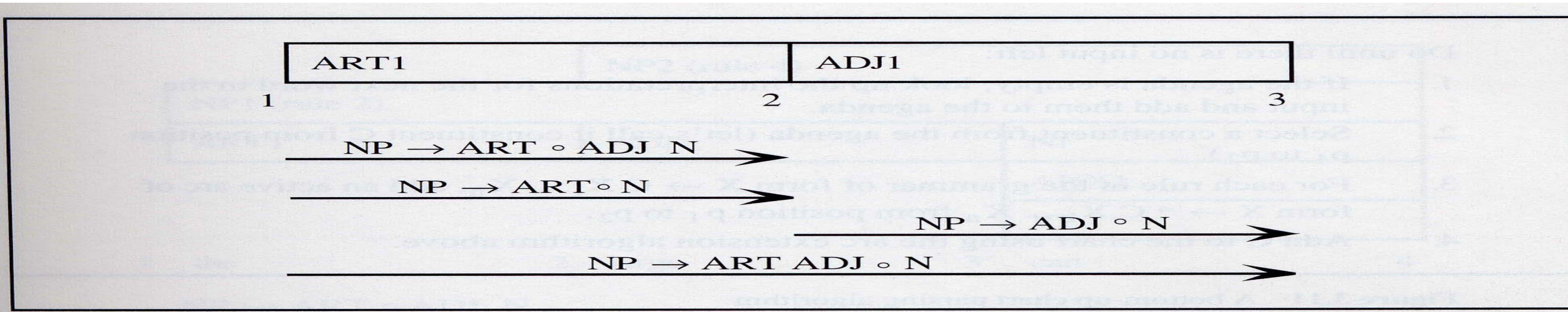



**Figure 3.9** The chart after seeing an ADJ in position 2

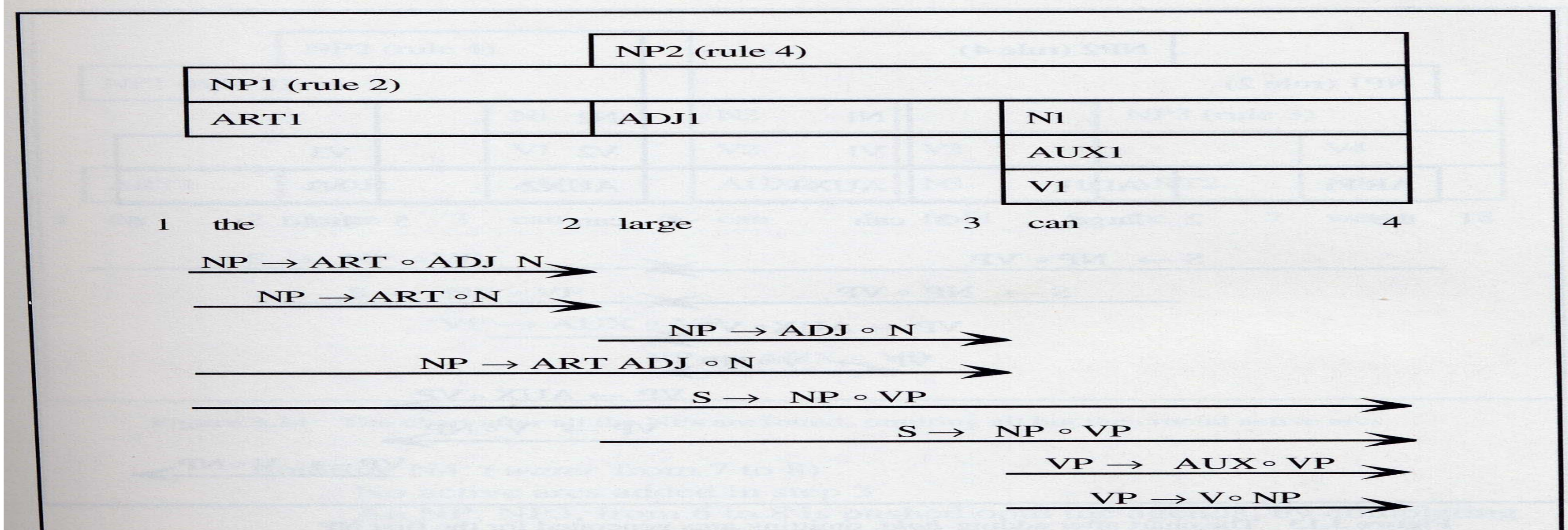



**Figure 3.12** After parsing *the large can*

# A Bottom-Up Chart Parser
(An Example) 3/5

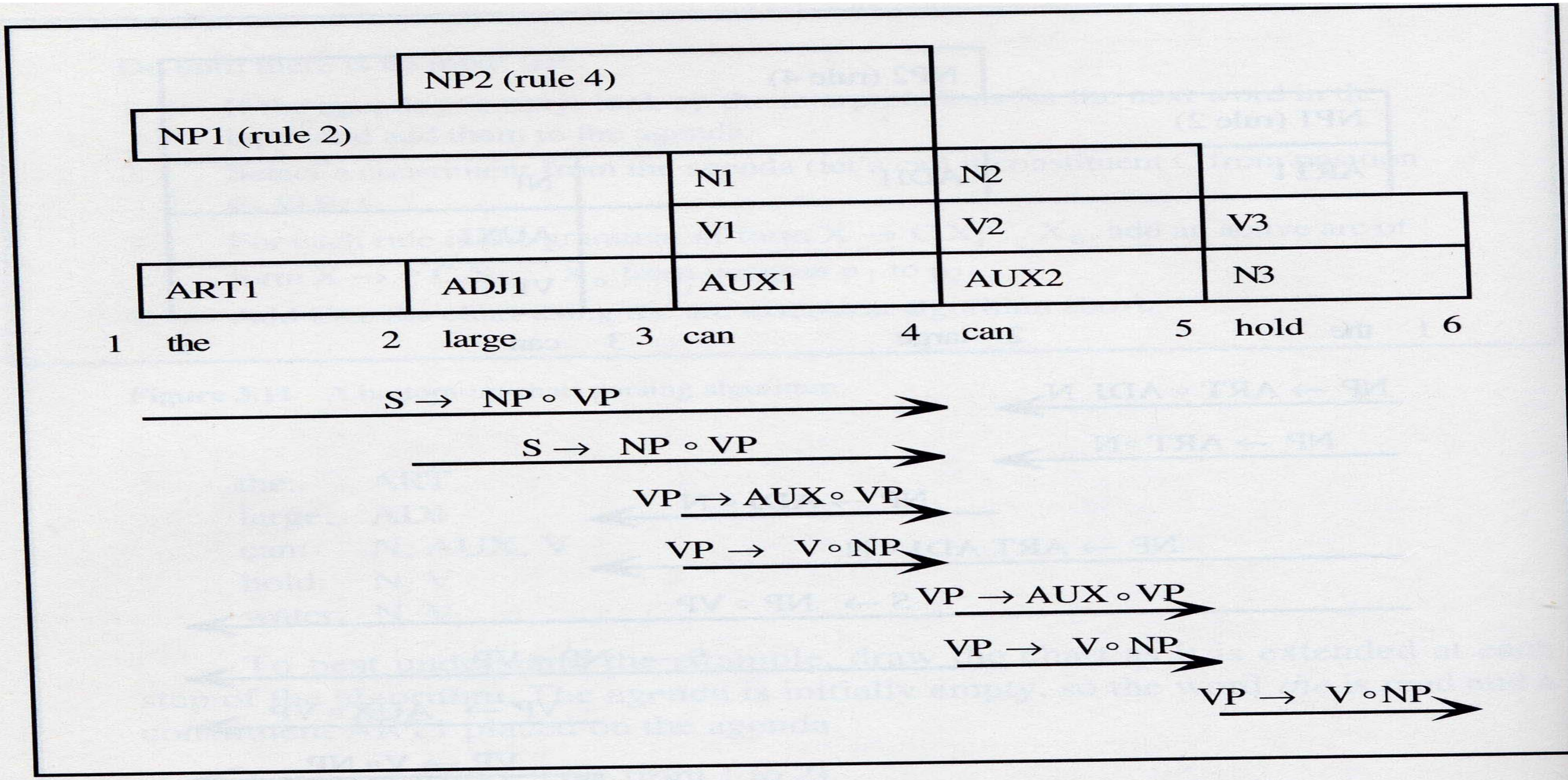



**Figure 3.13** The chart after adding *hold*, omitting arcs generated for the first NP

# A Bottom-Up Chart Parser
# (An Example) 4/5

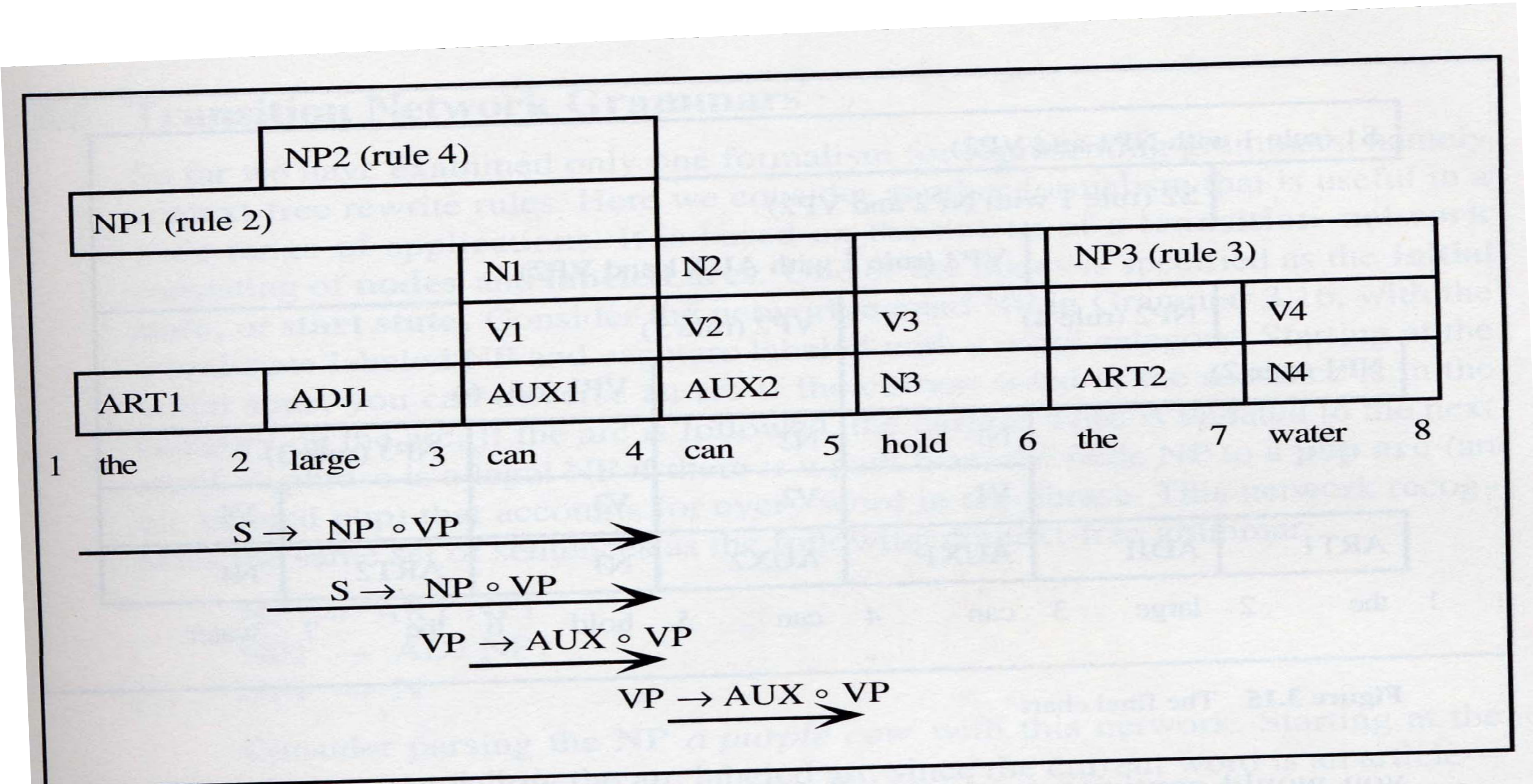



**Figure 3.14** The chart after all the NPs are found, omitting all but the crucial active arcs

# A Bottom-Up Chart Parser
(An Example) 5/5

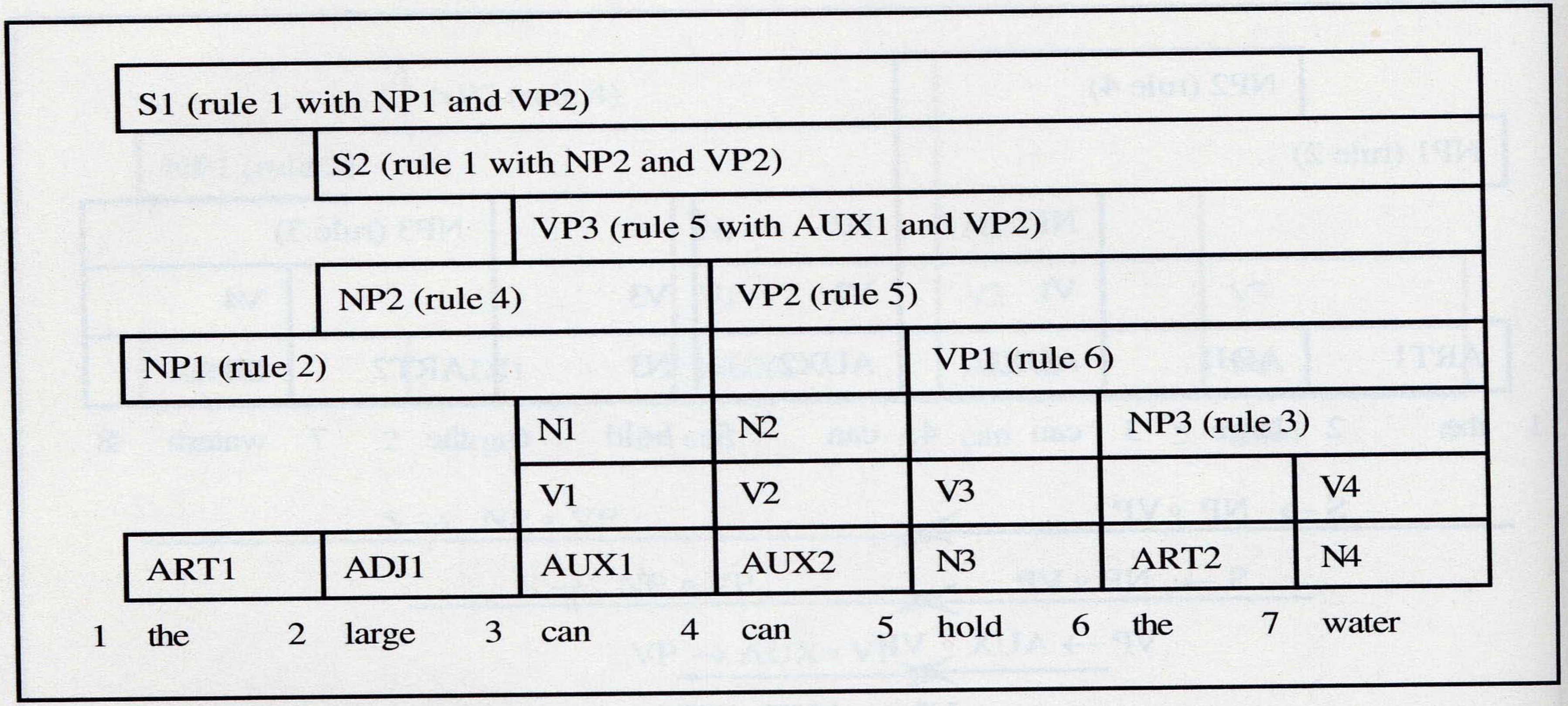



**Figure 3.15** The final chart

# 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.