Syntax & Grammars

Instructor: Wei Xu
Ohio State University

Some slides adapted from Ray Mooney, Marine Carpuat, Nathan Schneider, Michael Collins

What’s next in the class?

- From sequences to **trees**

- Syntax
  - Constituent, Grammatical relations, Dependency relations

- Formal Grammars
  - Context-free grammar
  - Dependency grammar

sýntaxis (setting out or arranging)

- The ordering of words and how they group into phrases
  - [[students][[cook and serve][grandparents]]]
  - [[students][[cook][and][serve grandparents]]]

Students Cook & Serve Grandparents

On Thursday, September 9, Gorman School hosted the first annual Grandparent's Day.

All Grandparents were invited to a school wide pancake breakfast. Upper grade students served as excellent chefs, as well as taking responsibility for serving the food and the clean up after-

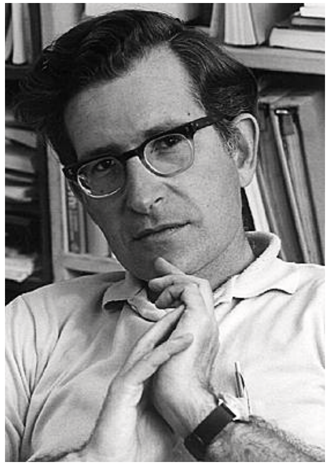
Syntax and Grammar

- Goal of syntactic theory
  - “explain how people combine words to form sentences and how children attain knowledge of sentence structure”

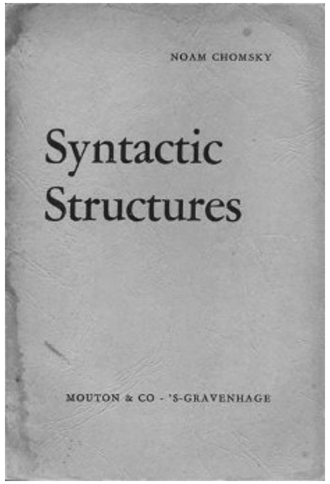
- Grammar
  - implicit knowledge of a native speaker
  - acquired without explicit instruction
  - minimally able to generate all and only the possible sentences of the language

Colin Phillips, “Syntax”, 2003

Syntax vs. Semantics



“Colorless green ideas sleep furiously.”
— Noam Chomsky (1957)

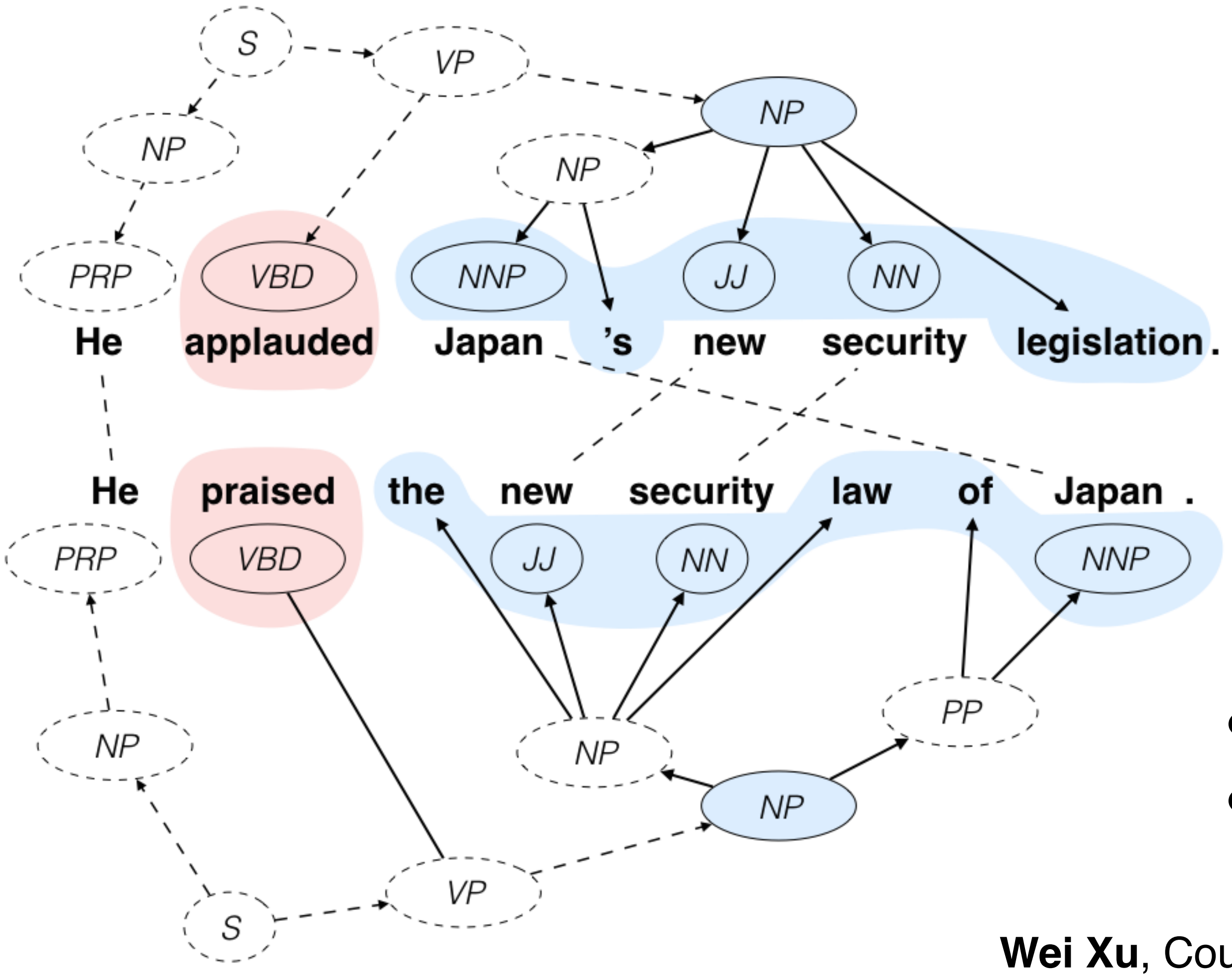


Contrast with: “sleep green furiously ideas colorless”

Syntax in NLP Applications

- Syntactic analysis is often a key component in applications
  - Grammar Checkers
  - Natural Language Generation:
    e.g. Sentence Compression, Fusion, Simplification, …
  - Information Extraction
  - Machine Translation
  - Question Answering
  - …

An Example: Sentence Simplification



- current state-of-the-art system
- syntactic machine translation techniques

Wei Xu, Courtney Napoles, Ellie Pavlick, Quanze Chen, Chris Callison-Burch.
"Optimizing Statistical Machine Translation for Simplification" in TACL (2016)

Another Example: 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*

Two Views of Syntactic Structure

- Constituency (phrase structure)
  - Phrase structure organizes words in nested constituents

- Dependency structure
  - Shows which words depend on (modify or are arguments of) which on other words

Syntax

Constituency Grammars

Constituency

- Basic idea: groups of words act as a single unit

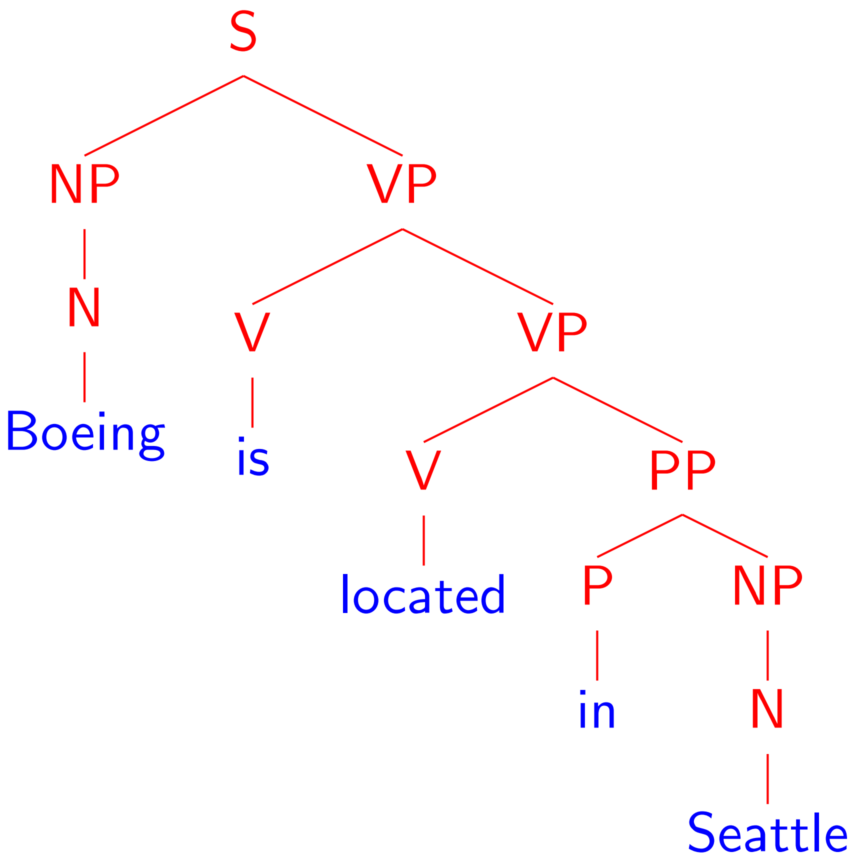
- Constituents form coherent classes that behave similarly
  - with respect to their internal structure:
    e.g. at the core of a noun phrase is a a noun
  - with respect to other constituents:
    e.g. noun phrases generally occur before verbs

Parsing (Syntactic Structure)

INPUT:

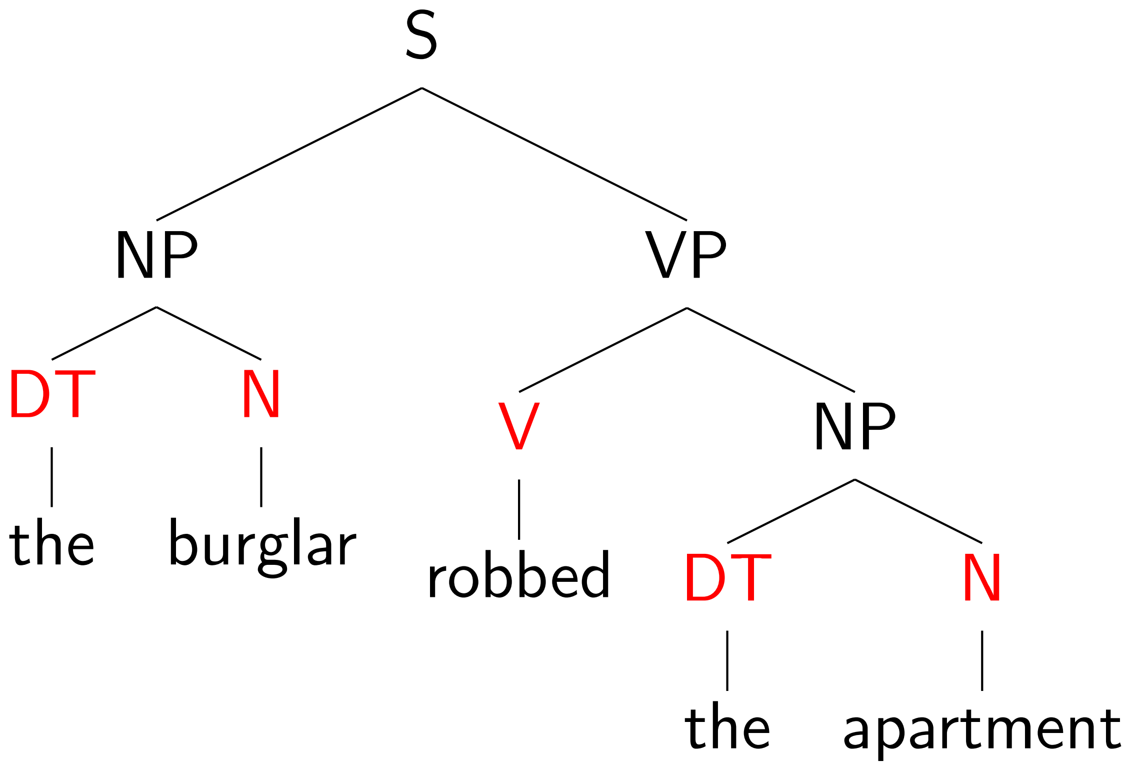
Boeing is located in Seattle.

OUTPUT:



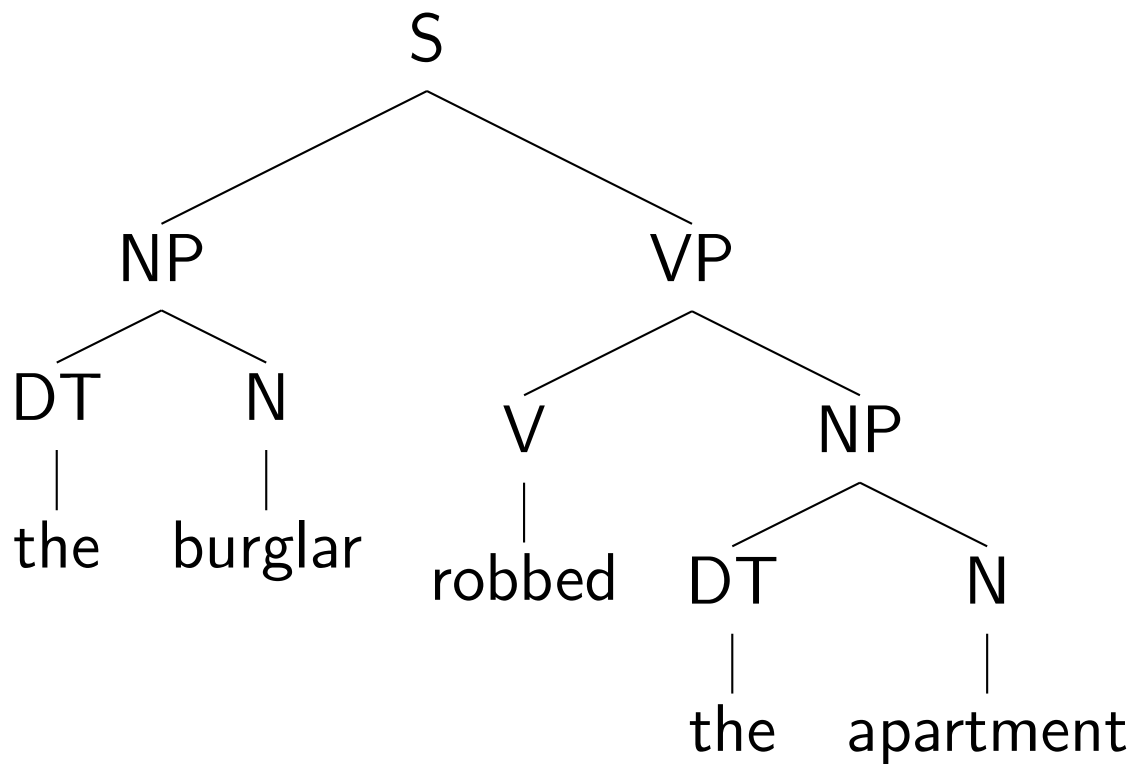
The Information Conveyed by Parse Trees

(1) Part of speech for each word
(N = noun, V = verb, DT = determiner)



The Information Conveyed by Parse Trees (continued)

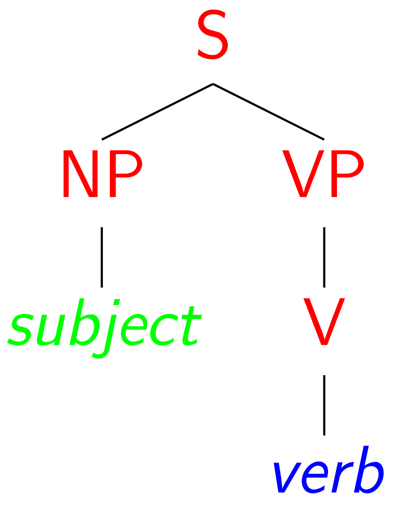
(2) Phrases

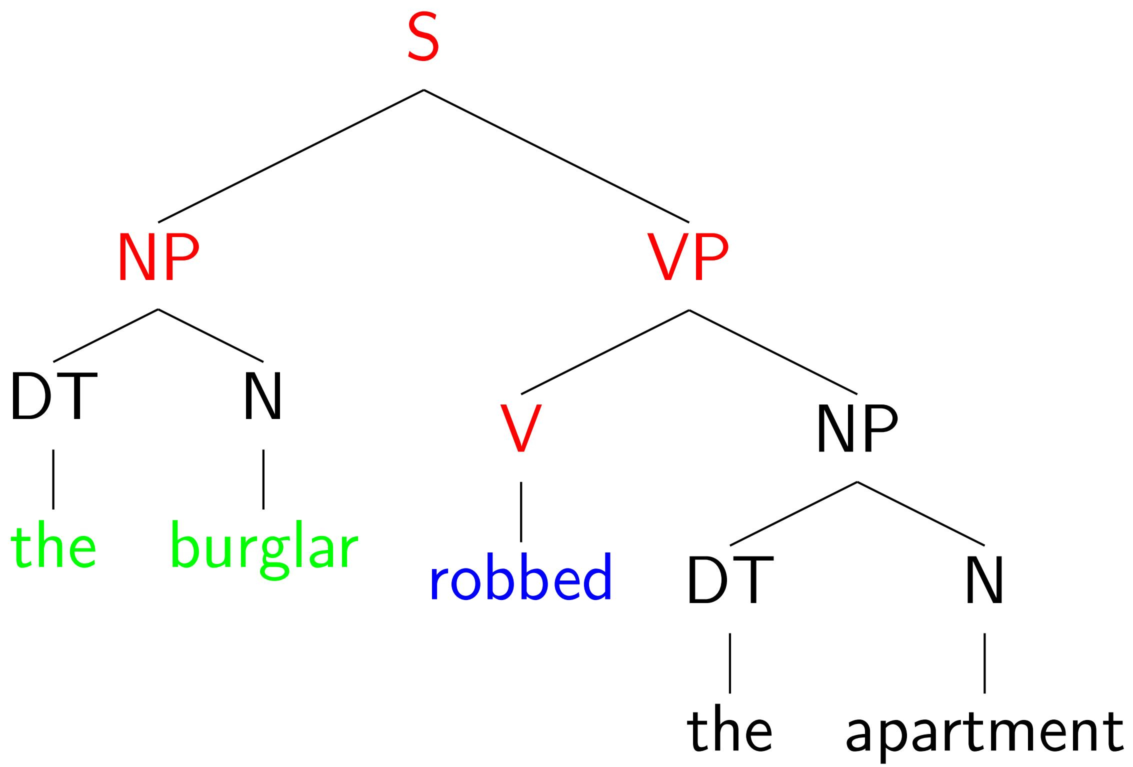


Noun Phrases (NP): "the burglar", "the apartment"
Verb Phrases (VP): "robbed the apartment"
Sentences (S): "the burglar robbed the apartment"

The Information Conveyed by Parse Trees (continued)

(3) Useful Relationships





⇒ “the burglar” is the subject of “robbed”

Grammars and Constituency

- For a particular language:
  - What are the “right” set of constituents?
  - What rules govern how they combine?

- Answer: not obvious and difficult
  - That’s why there are many different theories of grammar and competing analyses of the same data!

Syntactic Formalisms

- Work in formal syntax goes back to Chomsky's PhD thesis in the 1950s

The idea of basing a grammar on constituent structure dates back to Wilhem Wundt (1890).

- Examples of current formalisms: minimalism, lexical functional grammar (LFG), head-driven phrase-structure grammar (HPSG), tree adjoining grammars (TAG), categorial grammars

Regular Grammar

- You've already seen one class of grammars: **regular expressions**
    - A pattern like ^[a-z][0-9]$ corresponds to a grammar which accepts (matches) some strings but not others.

- **Q:** Can regular languages define infinite languages?

- **Q:** Can regular languages define arbitrarily complex languages?

Regular Grammar

- You've already seen one class of grammars: **regular expressions**
  - A pattern like ^[a-z][0-9]$ corresponds to a grammar which accepts (matches) some strings but not others.

- **Q:** Can regular languages define infinite languages?

  Yes, e.g. a*

- **Q:** Can regular languages define arbitrarily complex languages?

  No. Cannot match all strings with matched parentheses or in $a^n b^n$ forms in general (recursion/arbitrary nesting).

https://en.wikipedia.org/wiki/Pumping_lemma_for_regular_languages

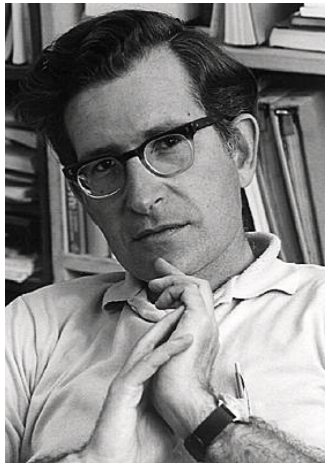
English is not a regular language

- There are certain types of sentences in English that look like $a^n b^n$
  - For example, “The dog that the man that the cat saw kicked barked” could be extended indefinitely.

- If syntax were regular, we should be able to reach a length after which we can just insert nouns, without adding the corresponding verb (by the Pumping Lemma).
  - For example, “The dog that the man that the cat that the rat that the mouse _____ feared saw kicked barked”

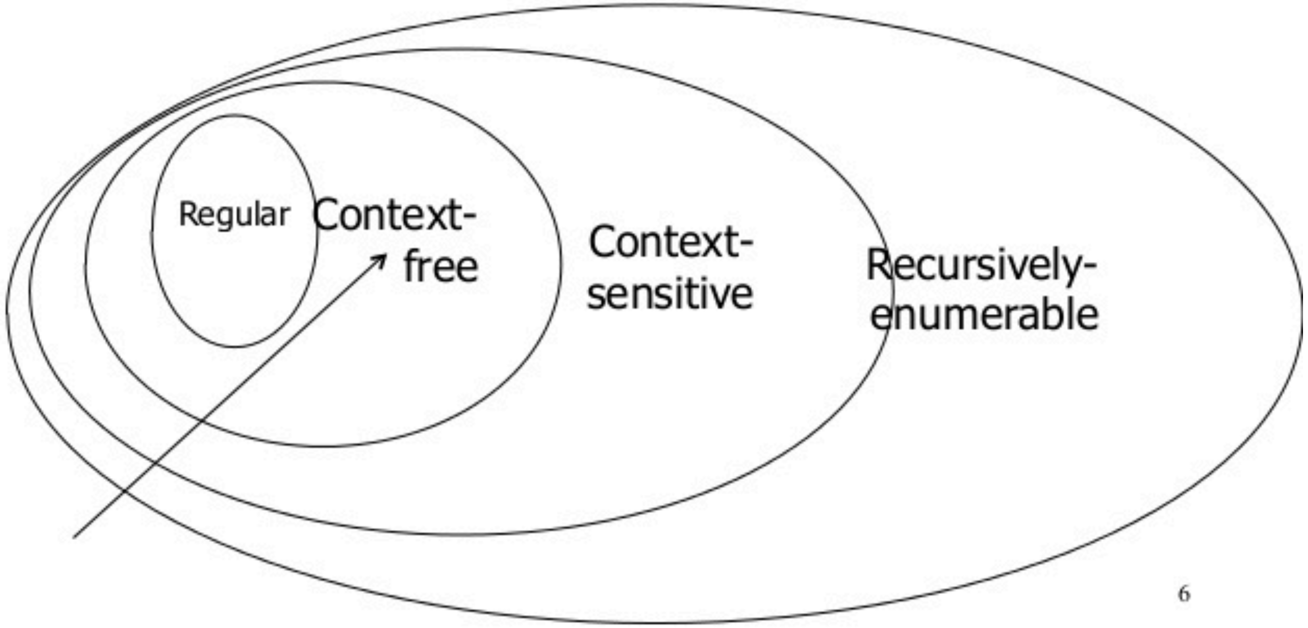
Noah Chomsky. 1956. The range of adequacy of various types of grammars.

The Chomsky Hierarchy



- Hierarchy of classes of formal languages

One language is of greater generative power or complexity than another if it can define a language that other cannot define. Context-free grammars are more powerful than regular grammars.



Context-Free Grammars

a.k.a phrase structure grammars,
Backus-Naur form (BNF)

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 $X \to Y_1 Y_2 \ldots Y_n$ 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
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

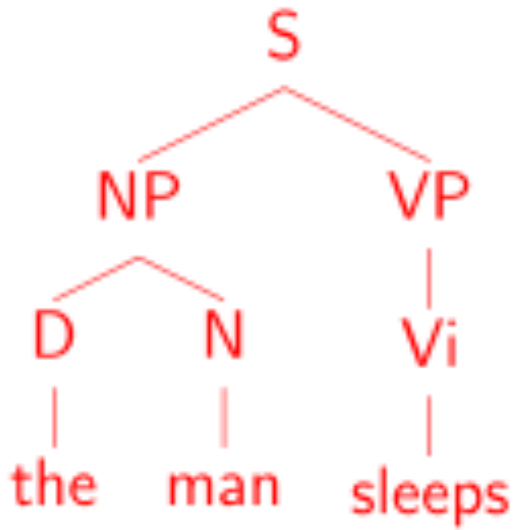
Left-Most Derivations

A left-most derivation is a sequence of strings $s_1 \ldots s_n$, 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 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:



An Example

DERIVATION	RULES USED
S	

An Example

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

An Example

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

An Example

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

An Example

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	

An Example

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	

An Example

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

An Example

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

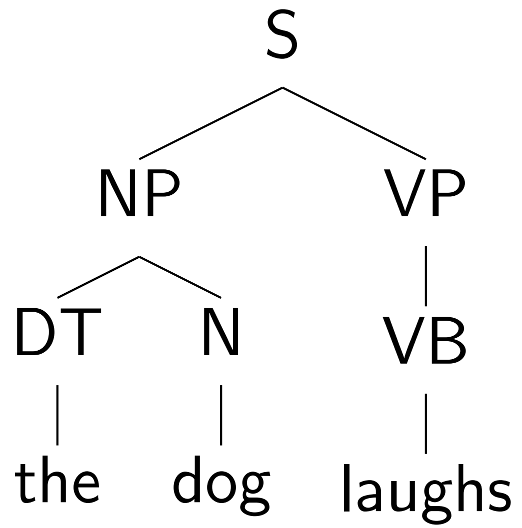
NP $\rightarrow$ DT N

DT $\rightarrow$ the

N $\rightarrow$ dog

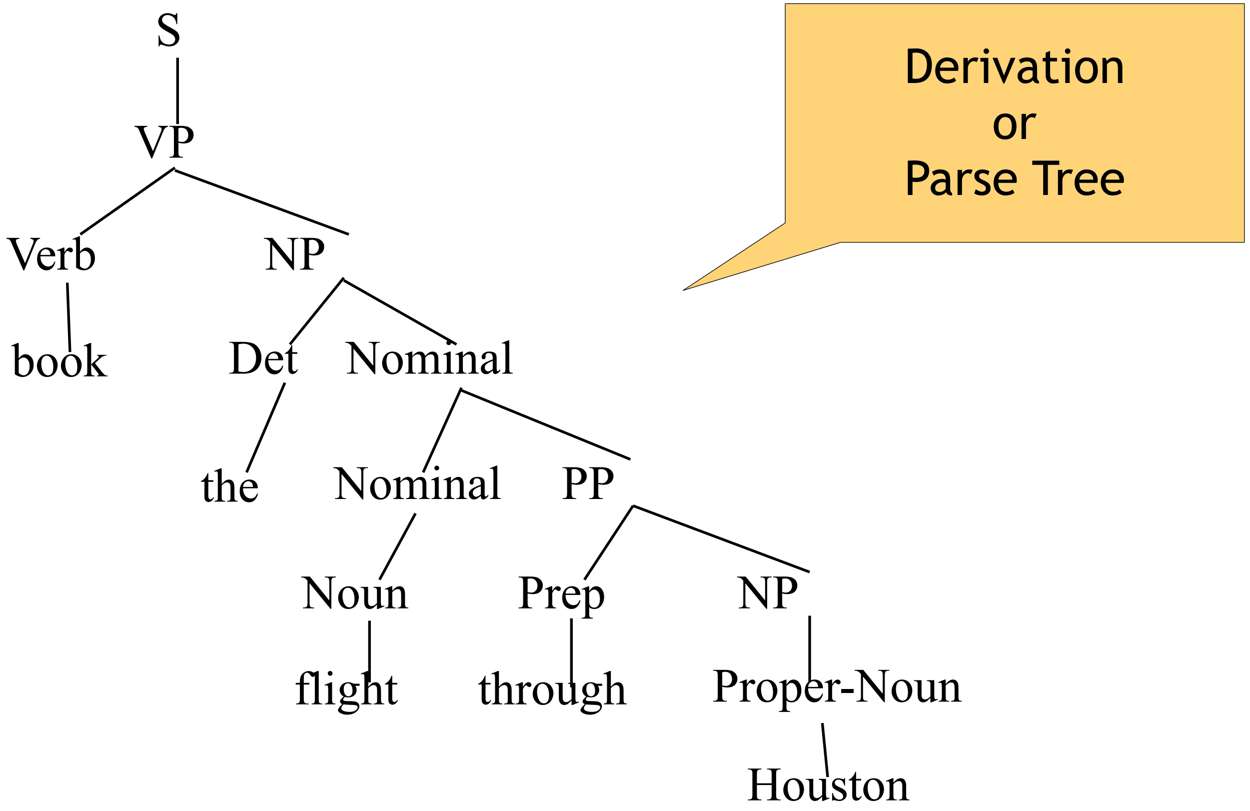
VP $\rightarrow$ VB

VB $\rightarrow$ laughs



Sentence Generation

- Sentences are generated by recursively rewriting the start symbol using the production rules in a CFG until only terminal symbols remain.



Parsing

- Given a string of terminals and a CFG, determine if the string can be generated by the CFG:
  - also return a parse tree for the string
  - also return all possible parse trees for the string

Properties of CFGs

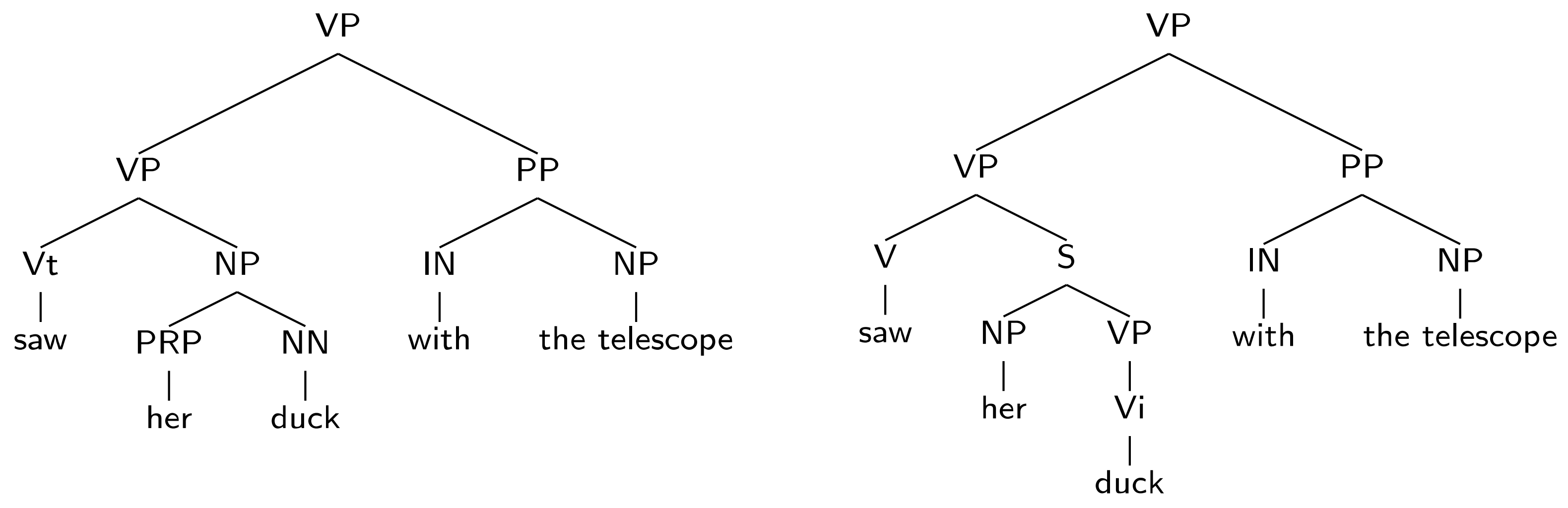
- A CFG defines a set of possible derivations
- A string $s \in \Sigma^*$ is in the *language* defined by the CFG if there is at least one derivation that yields s
- Each string in the language generated by the CFG may have more than one derivation ("ambiguity")

Sources of Ambiguity

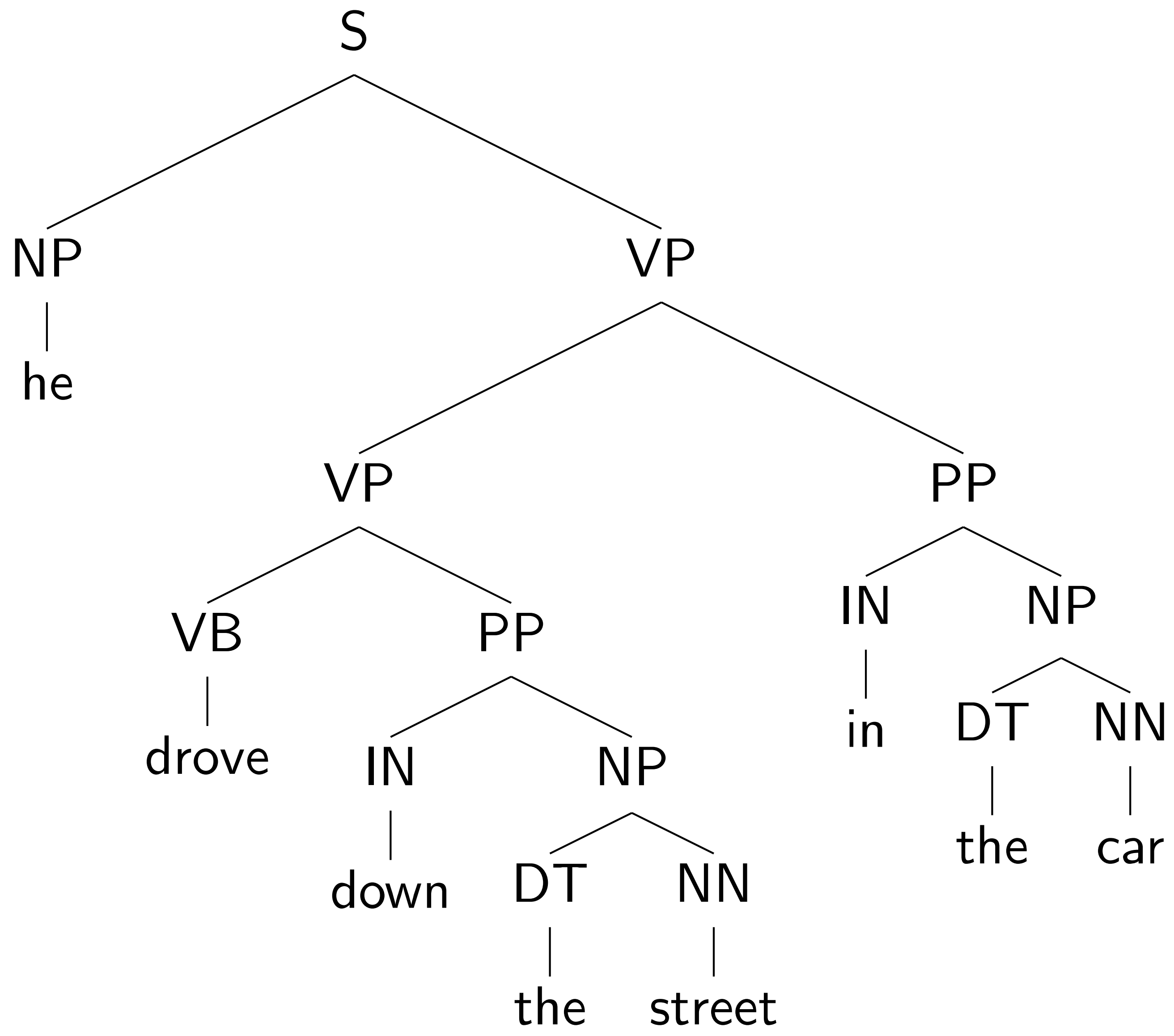
- Part-of-Speech ambiguity

  NN $\rightarrow$ duck

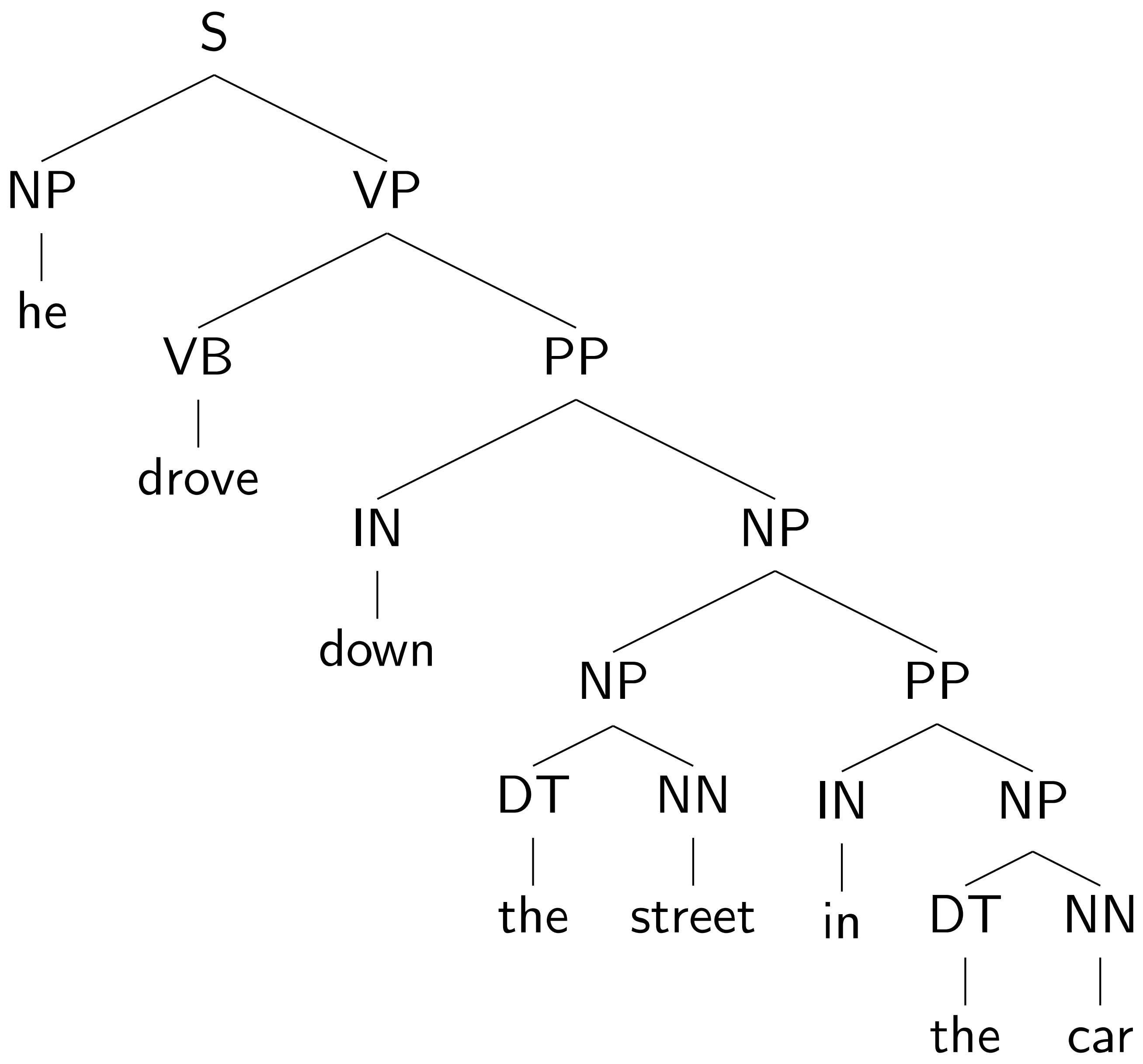
  Vi $\rightarrow$ duck



An Example of Ambiguity

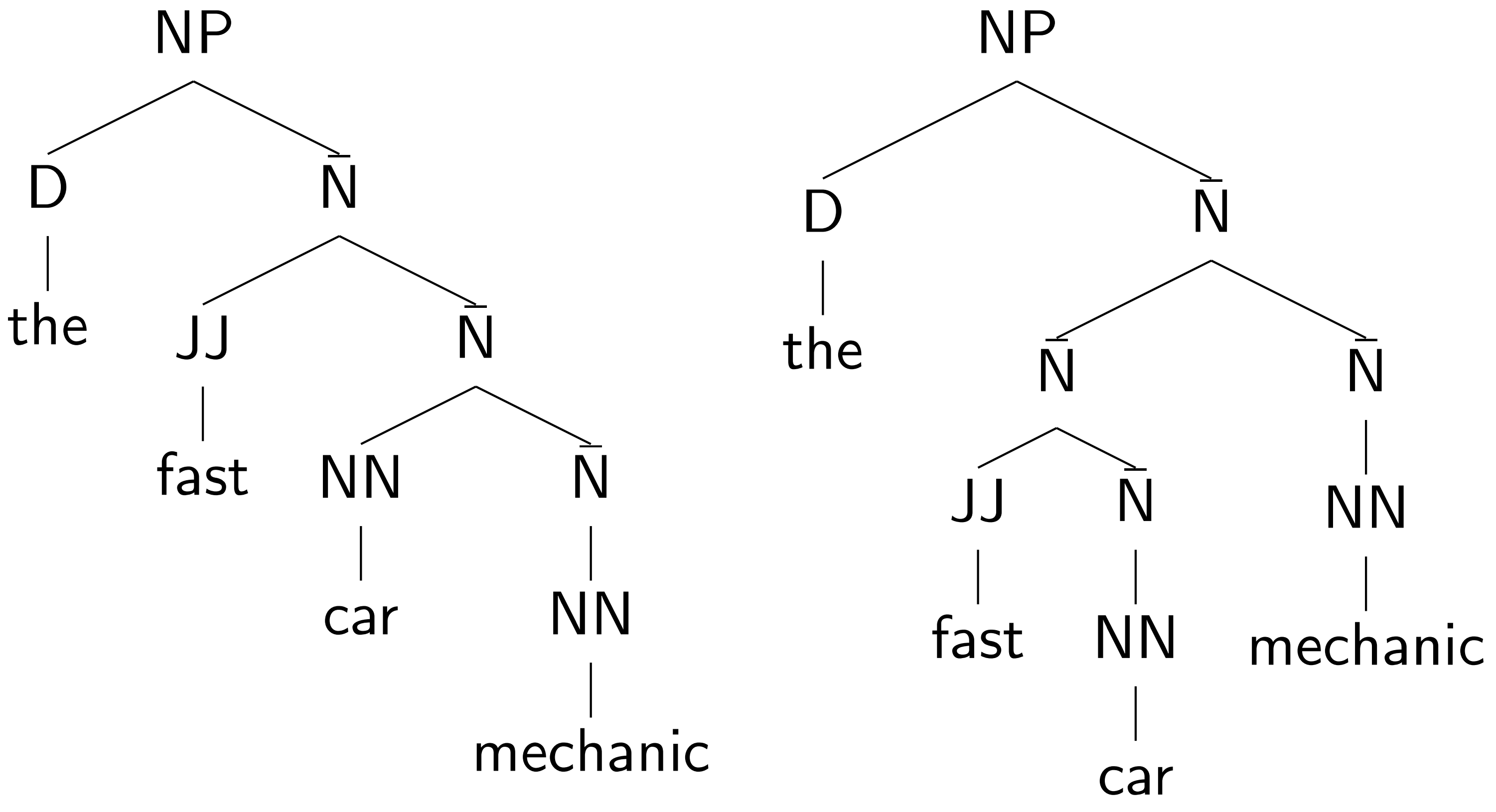


An Example of Ambiguity (continued)



Sources of Ambiguity: Noun Premodifiers

- Noun premodifiers:



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

Issues with CFGs

- Ambiguity
- addressing some grammatical constraints requires complex CFGs that do not compactly encode.
- some aspects of natural language syntax may not be captured by CFGs and require context-sensitivity

Swiss-German:

...de Karl d'Maria em Peter de Hans laat hälfe lärne schwüme

English:

...Charles lets Mary help Peter to teach John to Swim

- Regardless, good enough for most applications!
  (and many other alternative grammars exist)

Syntax

Dependency Grammars

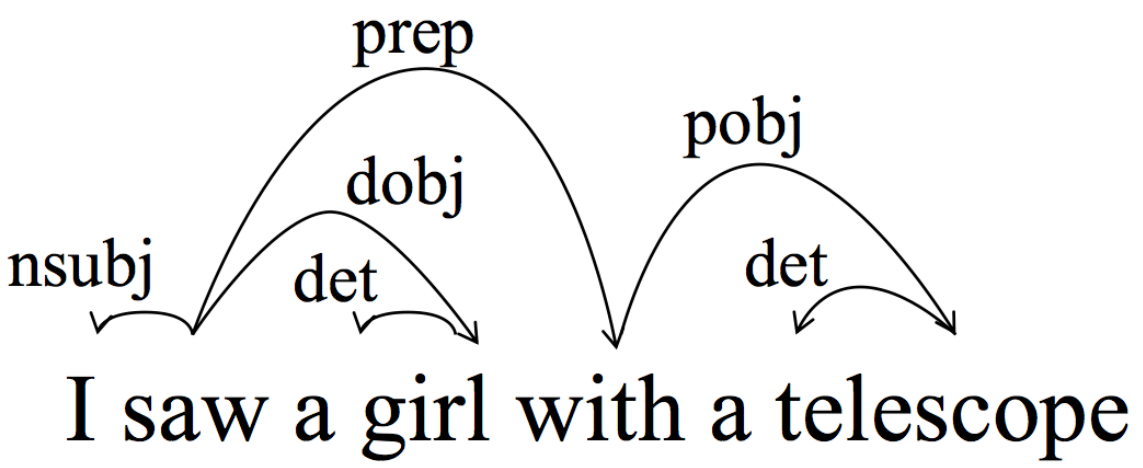
Dependency Grammars

- CFGs focus on constituents
  - Non-terminals don’t actually appear in the sentence

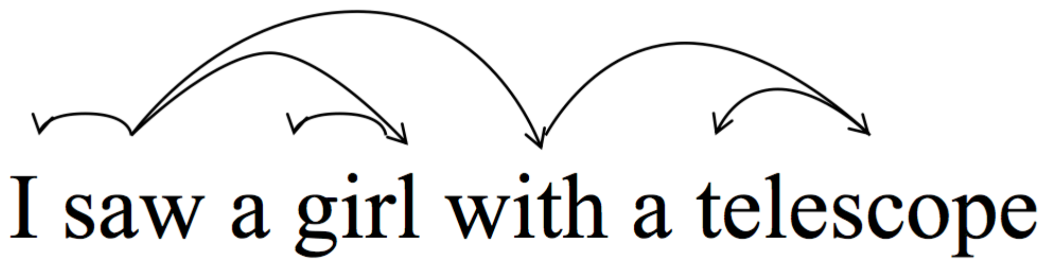
- In dependency grammar, a parse is a graph (usually a tree) where:
  - Nodes represent words
  - Edges represent dependency relations between words

Dependencies

- Typed: Label indicating relationship between words

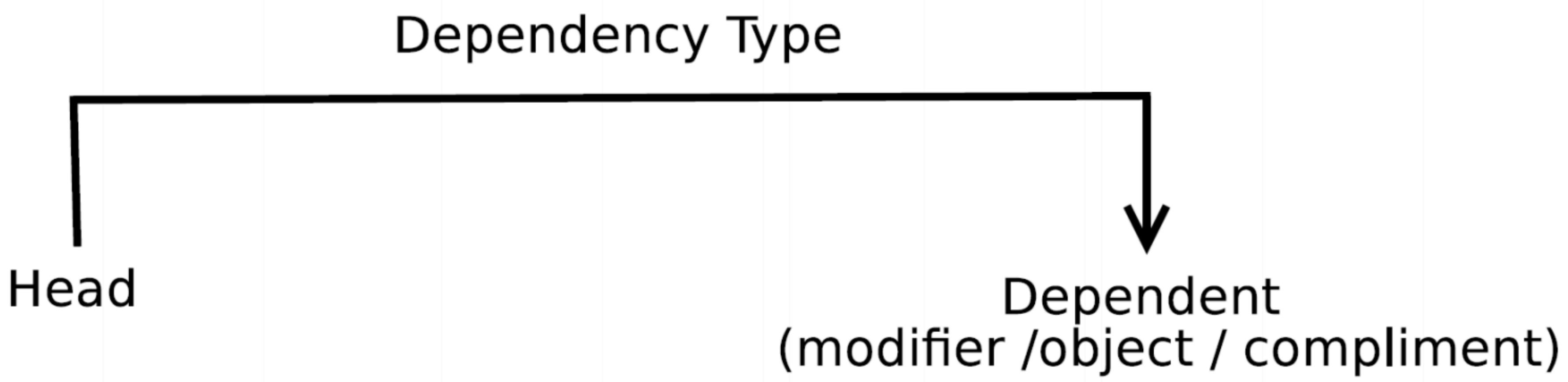


- Untyped: Only which words depend



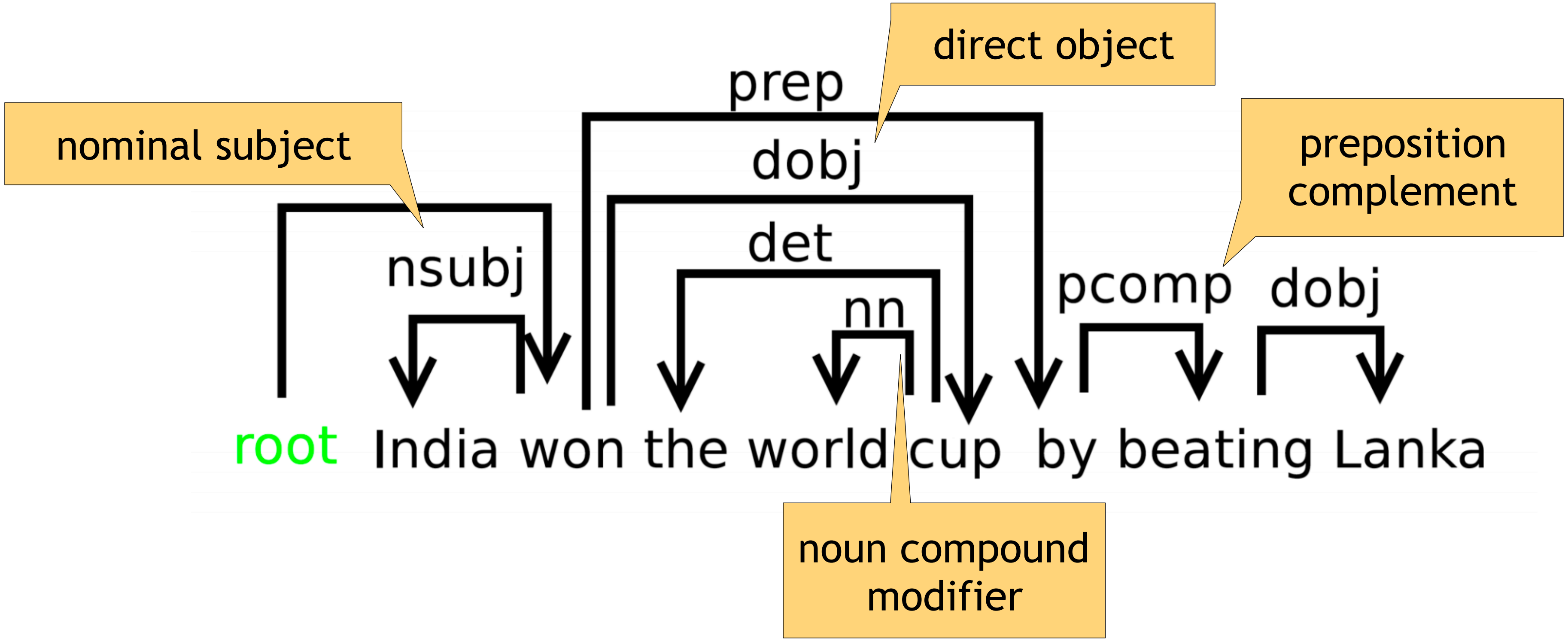
Dependency Grammars

- Syntactic Structure = Lexical items linked by binary asymmetrical relations called dependencies



Example Dependency Grammars

- Syntactic Structure = Lexical items linked by binary asymmetrical relations called dependencies



Syntax

English Grammar in a Nutshell

Product Details (from Amazon)
Hardcover: 1779 pages
Publisher: Longman; 2nd Revised edition
Language: English
ISBN-10: 0582517346
ISBN-13: 978-0582517349
Product Dimensions: 8.4 x 2.4 x 10 inches
Shipping Weight: 4.6 pounds

An English Grammar Fragment

- Sentences
- Noun phrases
  - Issue: agreement
- Verb phrases
  - Issue: subcategorization

Sentence Types

- Declaratives:

  S → NP VP   A plane left.

- Imperatives:

  S → VP   Leave!

- Yes-No Questions:

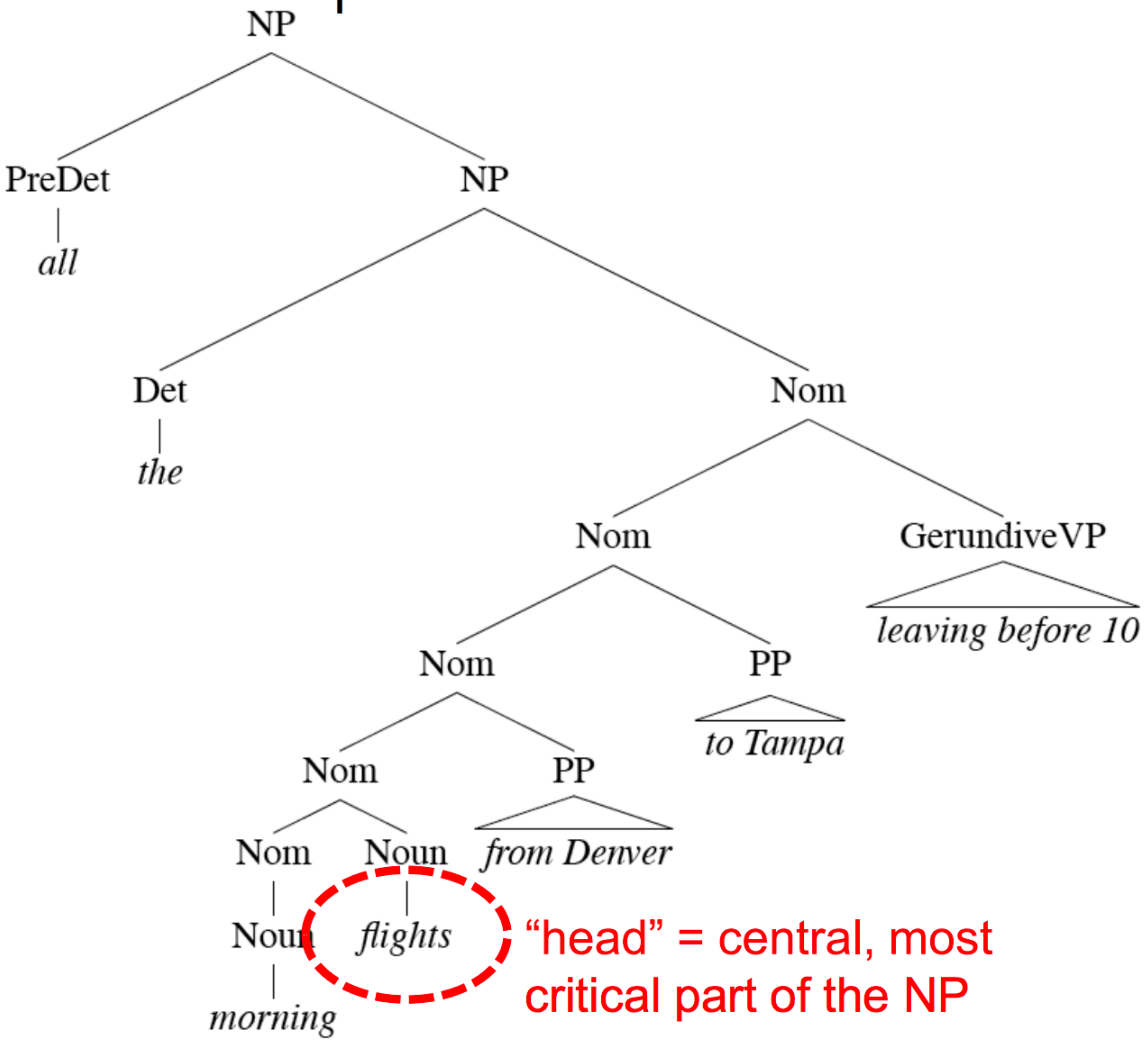
  S → Aux NP VP   Did the plane leave?

- WH Questions:

  S → WH–NP Aux NP VP   When did the plane leave?

Noun Phrases

- can be complicated
  - Determiners
  - Pre-modifiers
  - Post-modifiers



Determiners

- Noun phrases can start with determiners...
- Determiners can be
  - simple lexical items: the, this, a, an, etc. a car
  - simple possessives John's car
  - complex recursive versions John's sister's husband's son's car

Pre-modifiers

- Come before the head
- Examples:
  - Cardinals, ordinals, etc. three cars
  - Adjectives large car
- Ordering constraints:

three large cars vs. large three cars

Post-modifiers

- Come after the head
- Three kinds:
  - Prepositional phrases from Seattle
  - Non-finite clauses arriving before noon
  - Relative clauses that serve breakfast
- Similar recursive rules to handle these:
  - Nominal → Nominal PP
  - Nominal → Nominal GerundVP
  - Nominal → Nominal RelClause

Agreement Issues

- Agreement: constraints that hold among various constituents
- For example, subjects must agree with their verbs on person and number:

  I am cold. You are cold. He is cold.
  * I are cold * You is cold. *He am cold.

- Requires separate productions for each combination in CFG:

  S → NP1stPersonSing VP1stPersonSing
  S → NP2ndPersonSing VP2ndPersonSing
  NP1stPersonSing → …
  VP1stPersonSing → …
  NP2ndPersonSing → …
  VP2ndPersonSing → …

Other Agreement Issues

- Pronouns have case (e.g. nominative, accusative) that must agree with their syntactic position.

I gave him the book.	* I gave he the book.
He gave me the book.	* Him gave me the book.

- Many languages have gender agreement.

Los Angeles	* Las Angeles
Las Vegas	* Los Vegas

Verb Phrases

- English verb phrases consists of
  - Head verb
  - Zero or more following constituents (called arguments)
- Sample rules:

  VP → Verb   disappear

  VP → Verb NP   prefer a morning flight

  VP → Verb NP PP   leave Boston in the morning

  VP → Verb PP   leaving on Thursday

Subcategorization Issues

- Specific verbs take some types of arguments but not others.
  - Transitive verb: “found” requires a direct object

    John found the ring. * John found.
  - Intransitive verb: “disappeared” cannot take one

    John disappeared. * John disappeared the ring.
  - “gave” takes both a direct and indirect object

    John gave Mary the ring. * John gave Mary. * John gave the ring.
  - “want” takes an NP, or non-finite VP or S

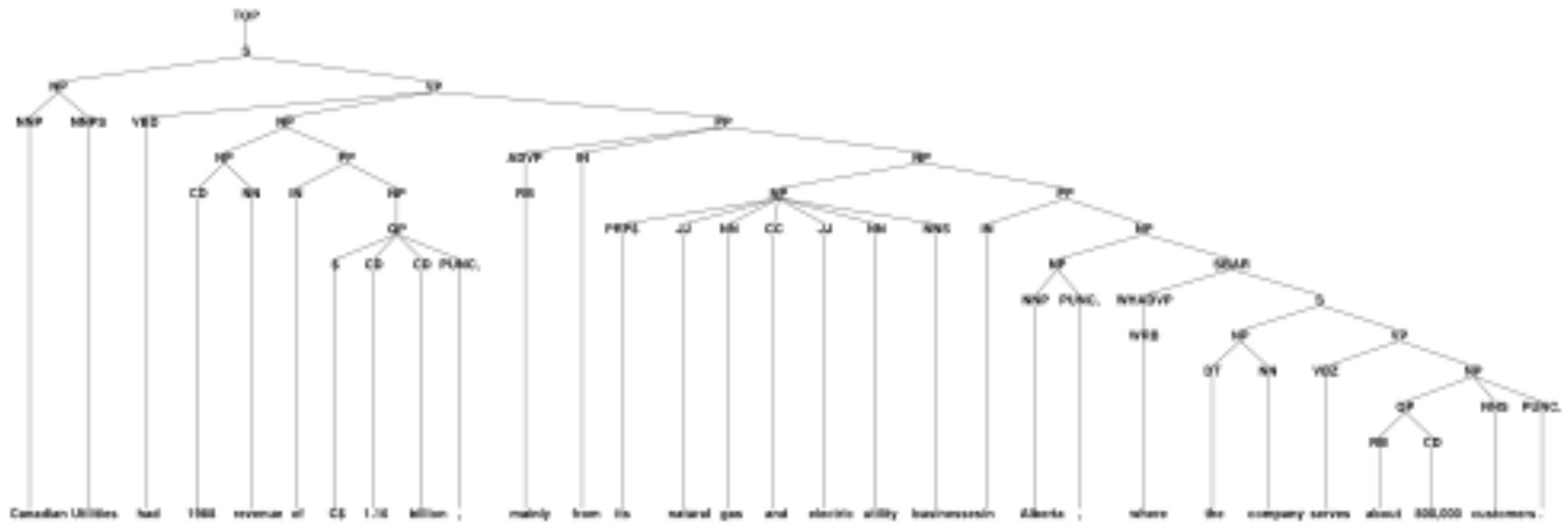
    John wants a car. John wants to buy a car.
    John wants Mary to take the ring. * John wants.

- Subcategorization frames specify the range of argument types that a given verb can take.

Data: Penn Treebank

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

An example tree:



Data: Penn Treebank

- Treebanks implicitly define a grammar for the language
- Penn Treebank has 4500 different rules for VPs, including…
    - VP → BD PP
    - VP → VBD PP PP
    - VP → VBD PP PP PP
    - VP → VBD PP PP PP PP

Summary

- Two views of syntactic structures
  - Constituency grammars (in particular, Context Free Grammars)
  - Dependency grammars

- Can be used to capture various facts about the structure of language (but not all!)

Syntax

Parsing

Parsing

- Given a string of terminals and a CFG, determine if the string can be generated by the CFG:
  - also return a parse tree for the string
  - also return all possible parse trees for the string

- Must search space of derivations for one that derives the given string.
  - Top-Down Parsing
  - Bottom-Up Parsing

Simple CFG for ATIS English

Grammar

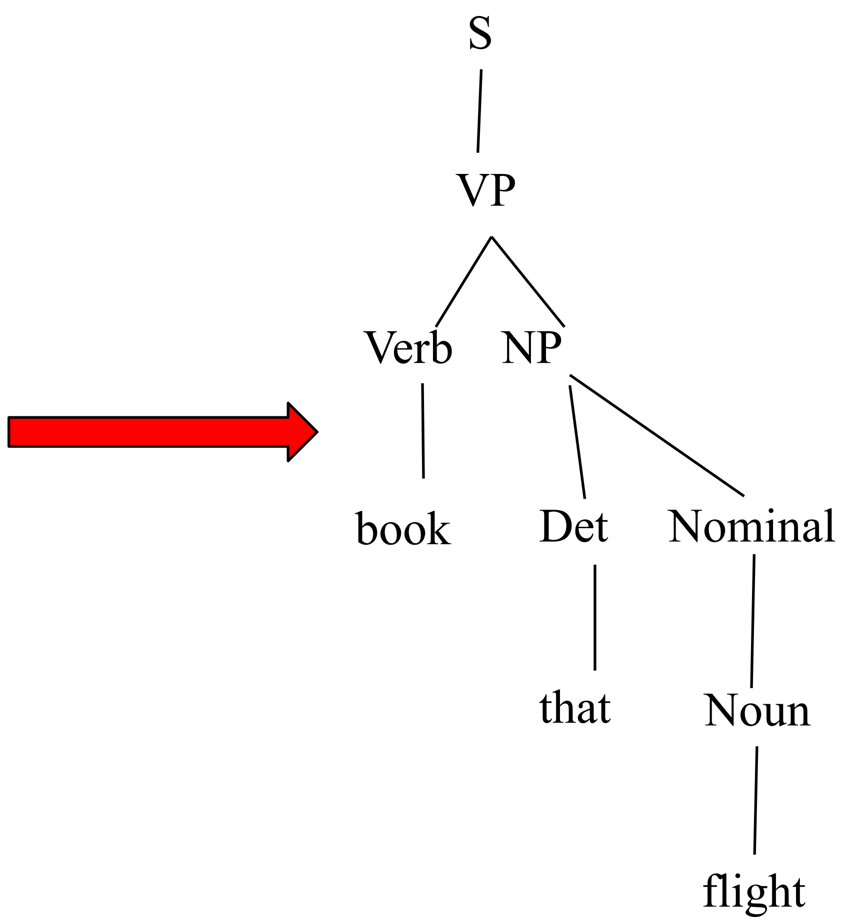
$S \rightarrow NP\ VP$
$S \rightarrow Aux\ NP\ VP$
$S \rightarrow VP$
$NP \rightarrow Pronoun$
$NP \rightarrow Proper\text{-}Noun$
$NP \rightarrow Det\ Nominal$
$Nominal \rightarrow Noun$
$Nominal \rightarrow Nominal\ Noun$
$Nominal \rightarrow Nominal\ PP$
$VP \rightarrow Verb$
$VP \rightarrow Verb\ NP$
$VP \rightarrow VP\ PP$
$PP \rightarrow Prep\ NP$

Lexicon

Det $\rightarrow$ the | a | that | this
Noun $\rightarrow$ book | flight | meal | money
Verb $\rightarrow$ book | include | prefer
Pronoun $\rightarrow$ I | he | she | me
Proper-Noun $\rightarrow$ Houston | NWA
Aux $\rightarrow$ does
Prep $\rightarrow$ from | to | on | near | through

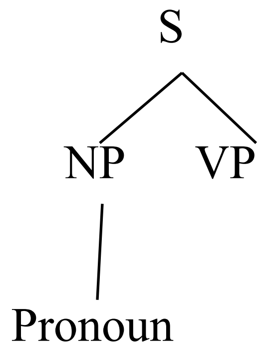
Parsing Example

book that flight

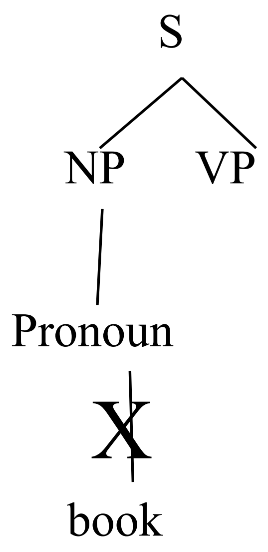


Top Down Parsing

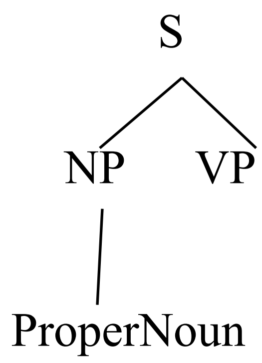
- Start searching space of derivations for the start symbol.



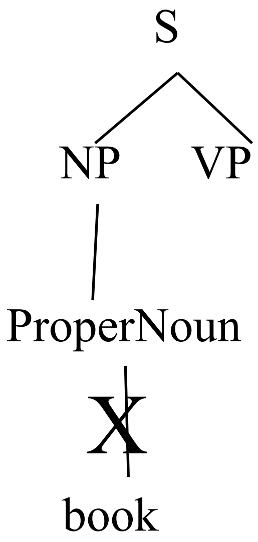
Top Down Parsing



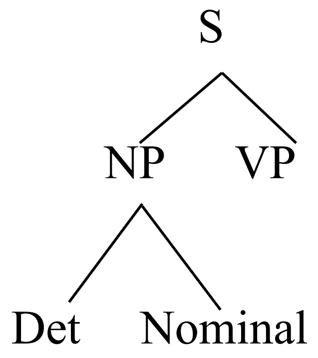
Top Down Parsing



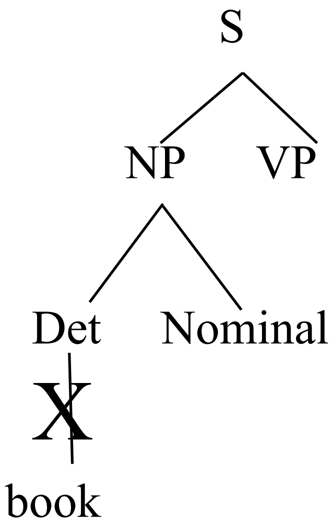
Top Down Parsing



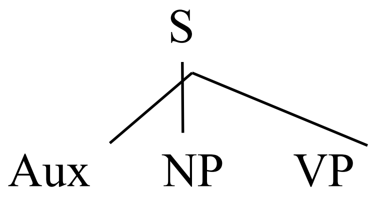
Top Down Parsing



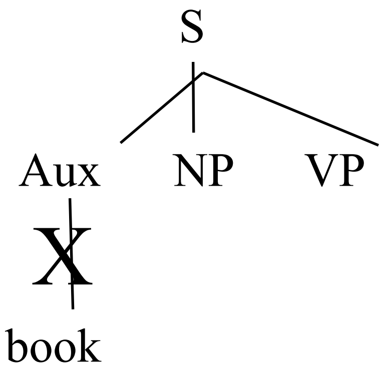
Top Down Parsing



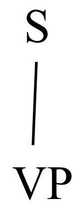
Top Down Parsing



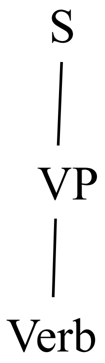
Top Down Parsing



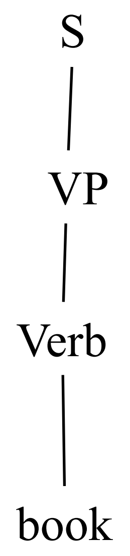
Top Down Parsing



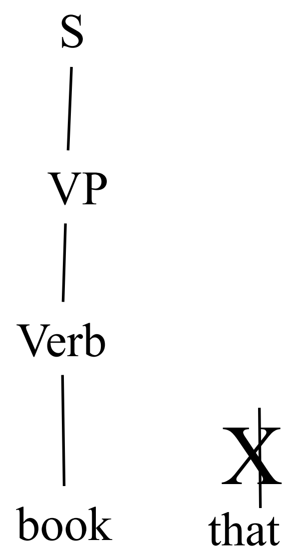
Top Down Parsing



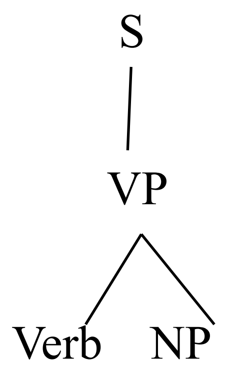
Top Down Parsing



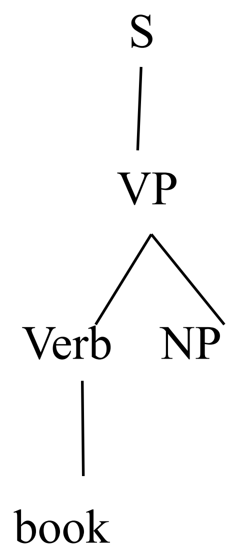
Top Down Parsing



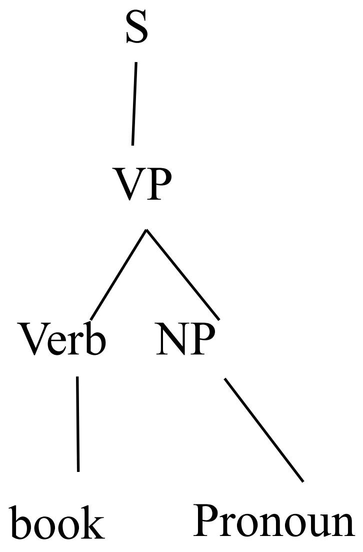
Top Down Parsing



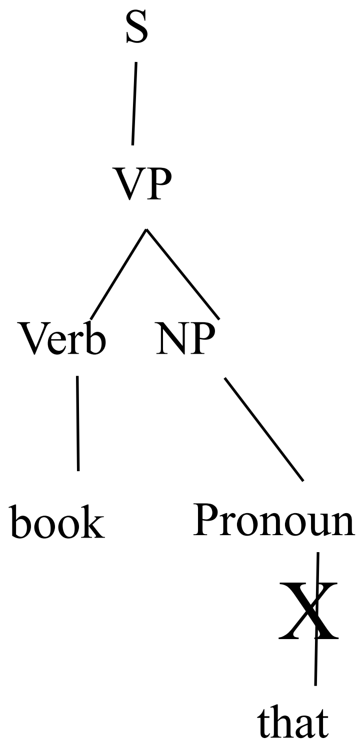
Top Down Parsing



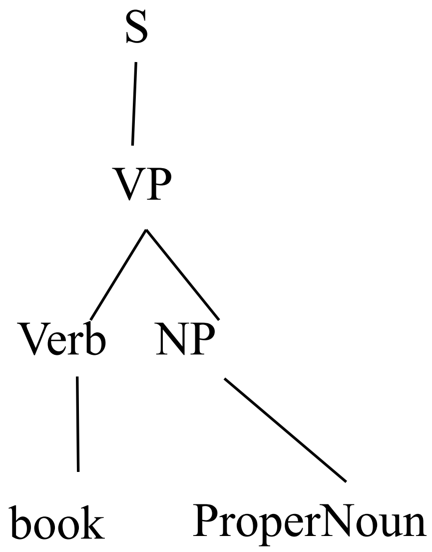
Top Down Parsing



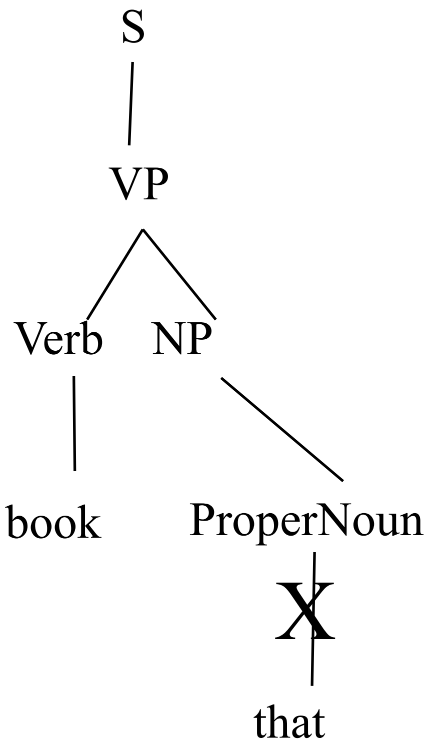
Top Down Parsing



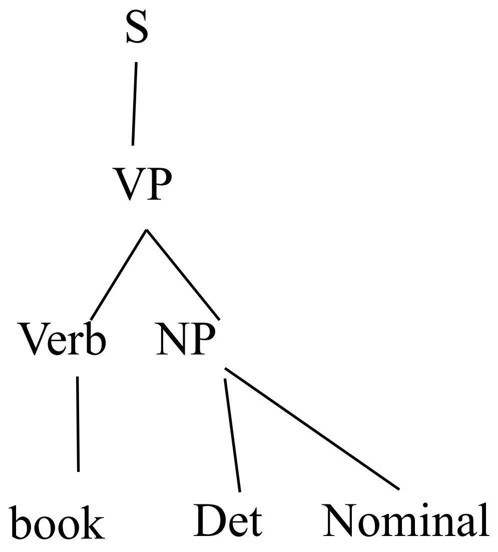
Top Down Parsing



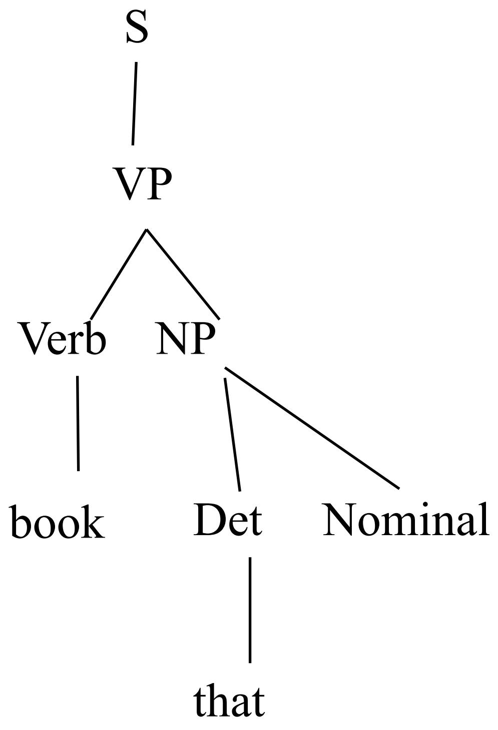
Top Down Parsing



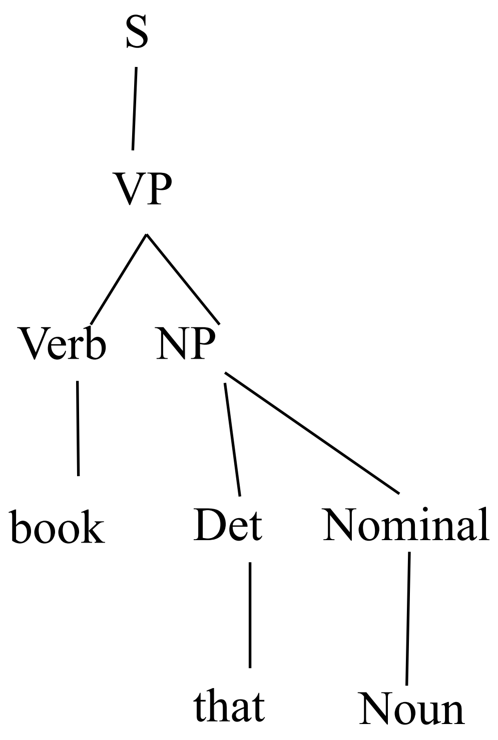
Top Down Parsing



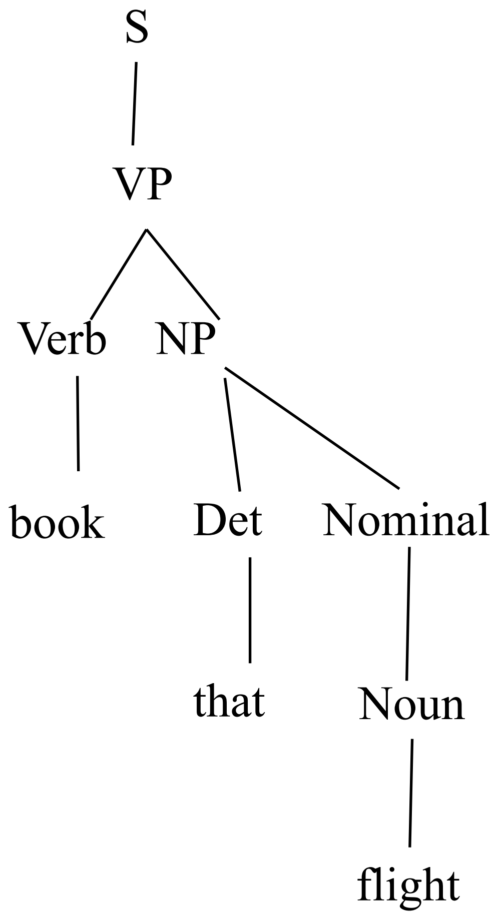
Top Down Parsing



Top Down Parsing



Top Down Parsing



Bottom Up Parsing

- Start searching space of reverse derivations from the terminal symbols in the string.

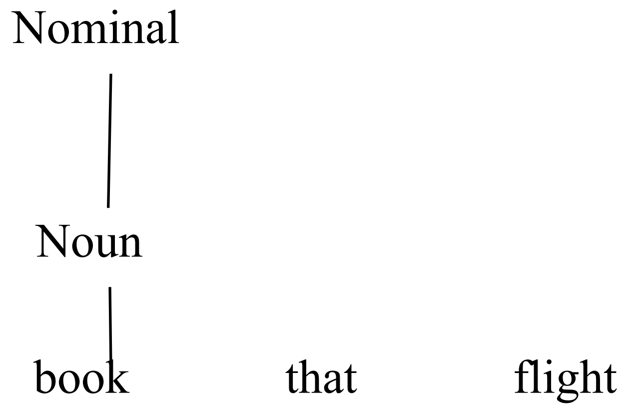
book that flight

Bottom Up Parsing

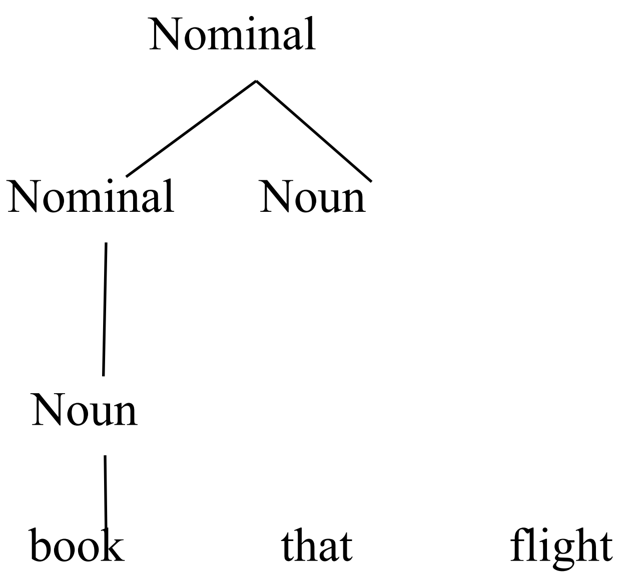
Noun

book that flight

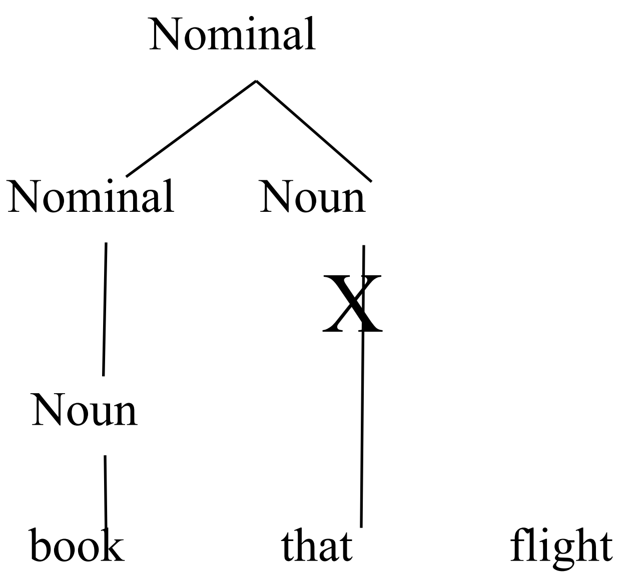
Bottom Up Parsing



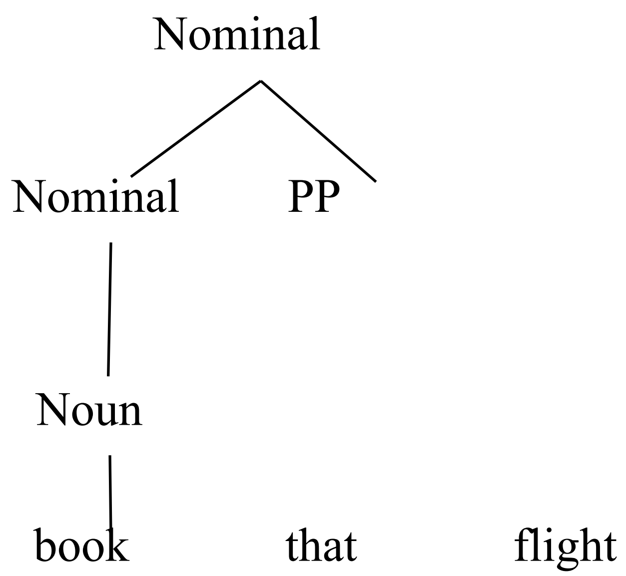
Bottom Up Parsing



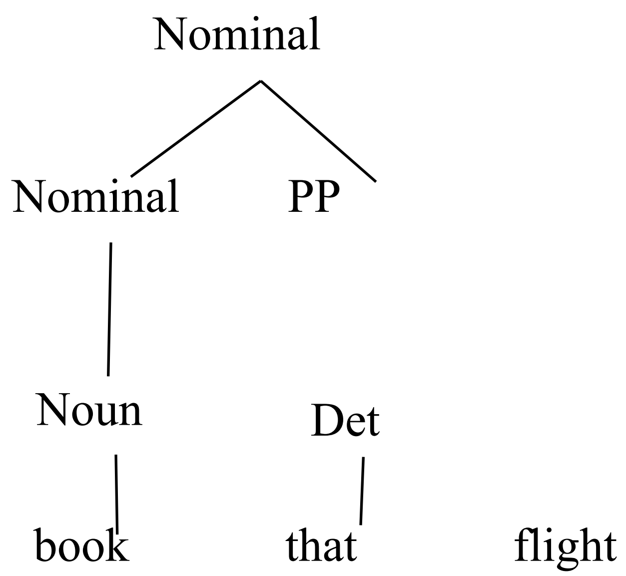
Bottom Up Parsing



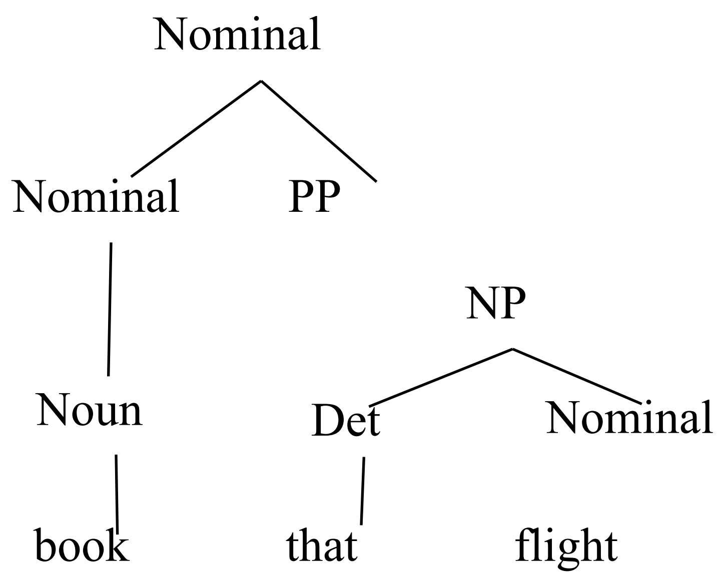
Bottom Up Parsing



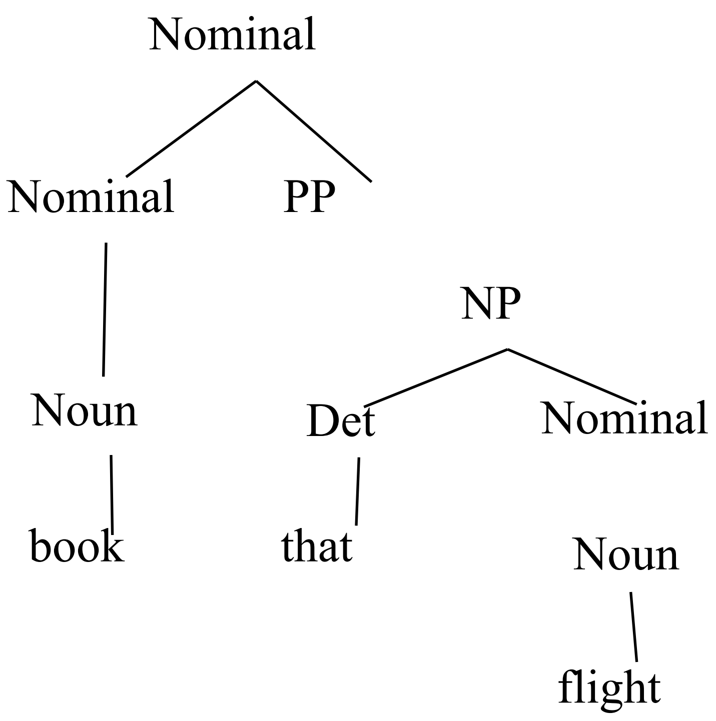
Bottom Up Parsing



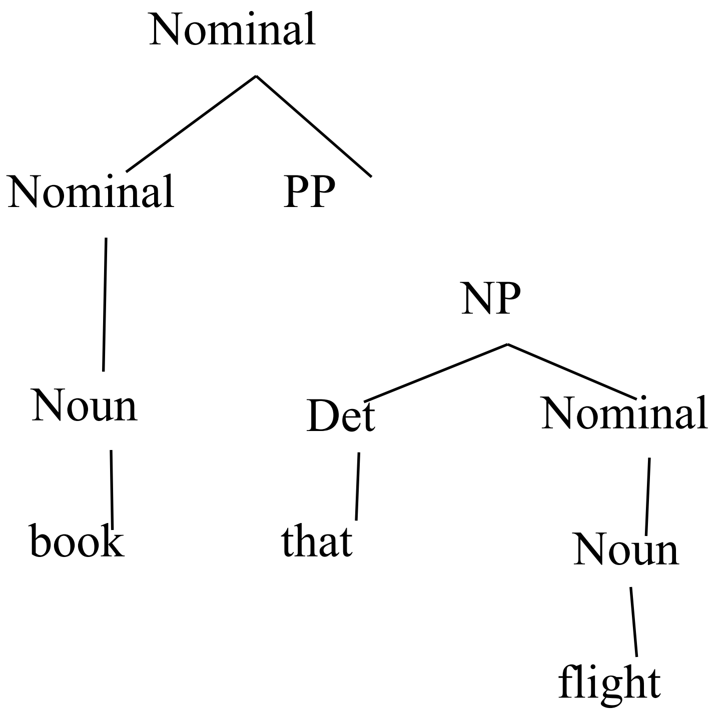
Bottom Up Parsing



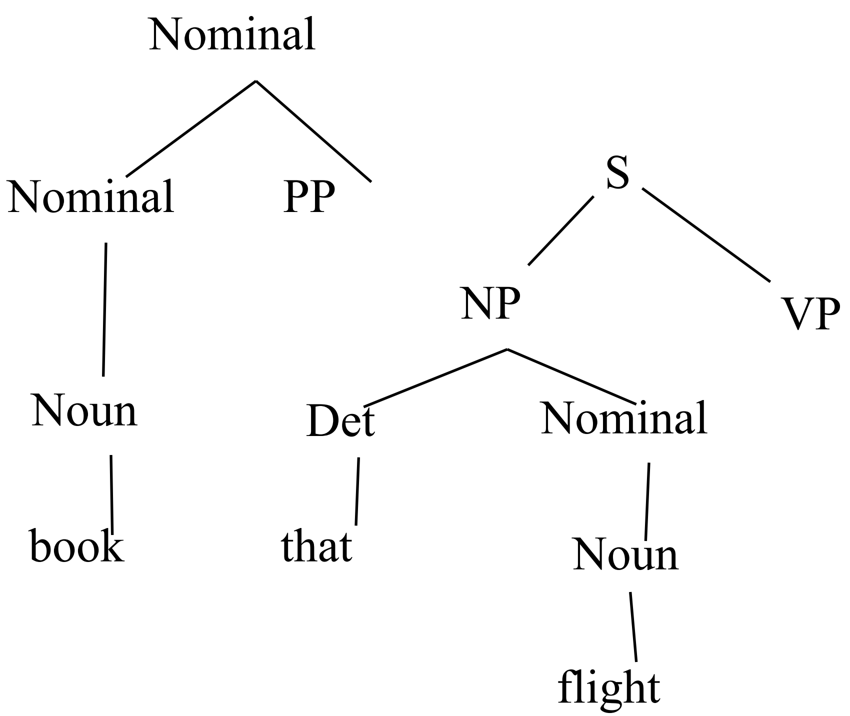
Bottom Up Parsing



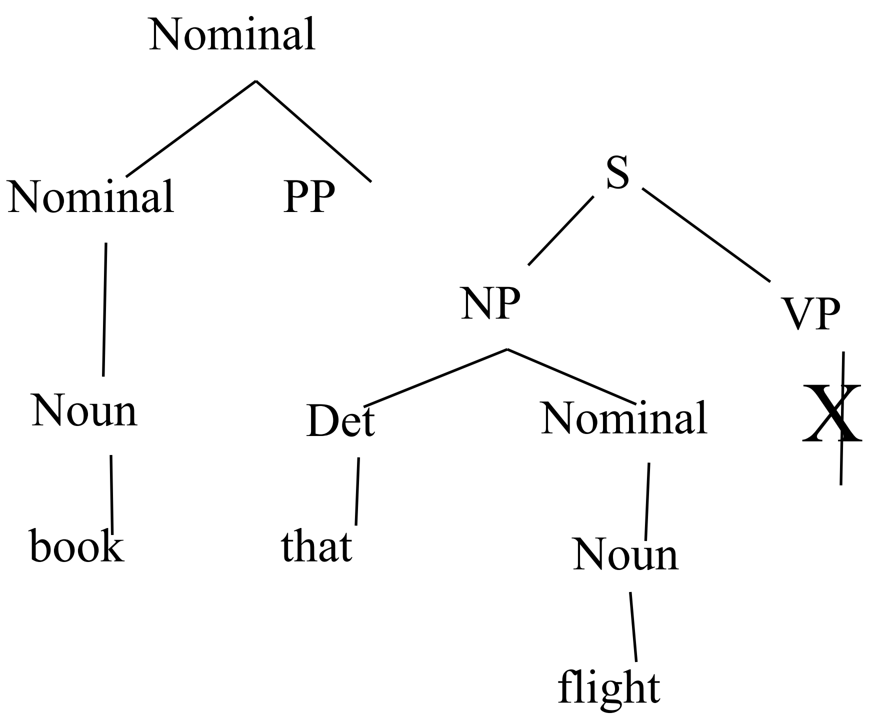
Bottom Up Parsing



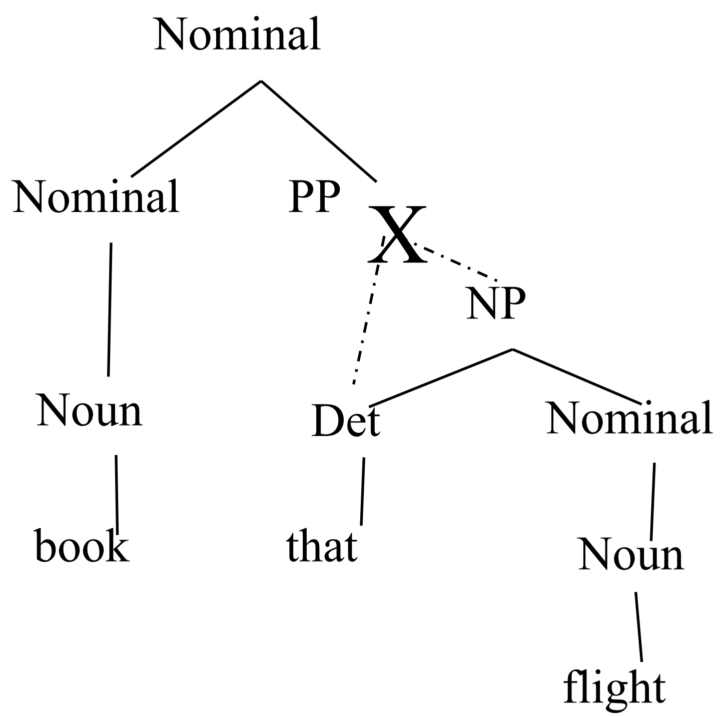
Bottom Up Parsing



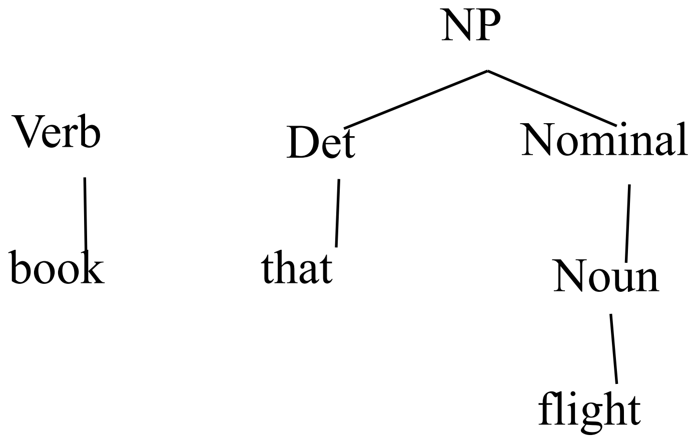
Bottom Up Parsing



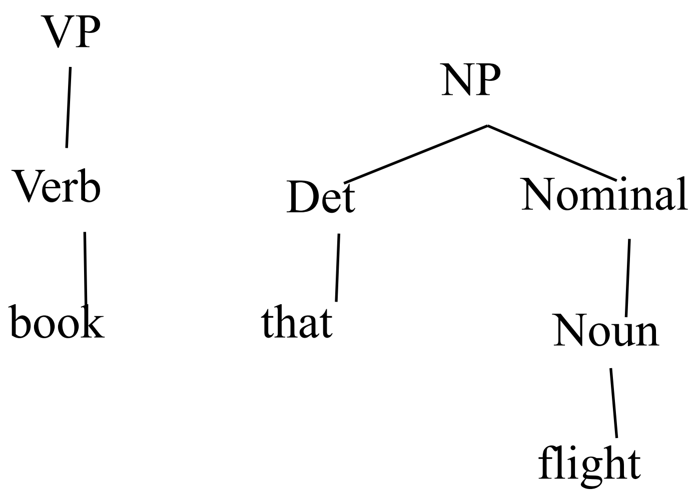
Bottom Up Parsing



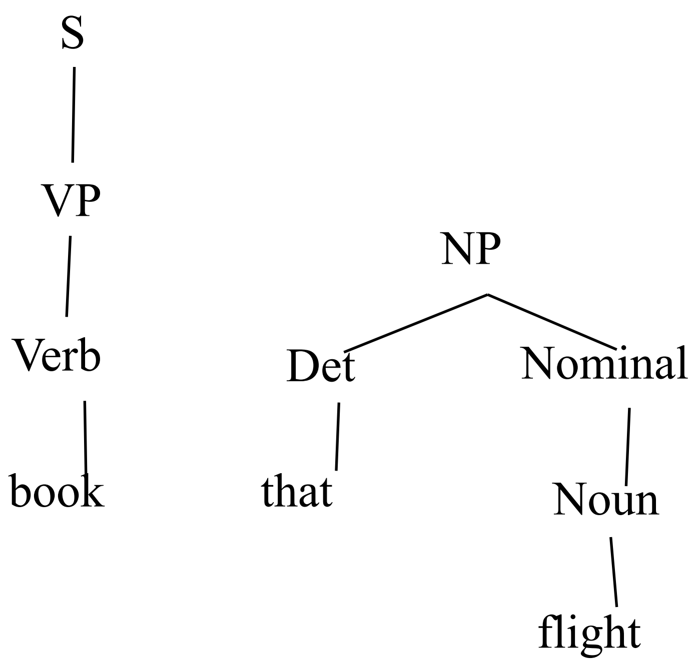
Bottom Up Parsing



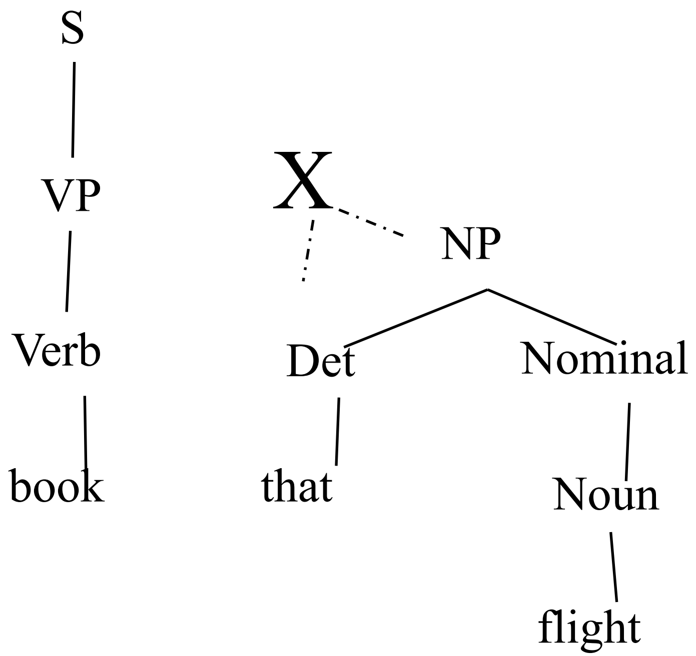
Bottom Up Parsing



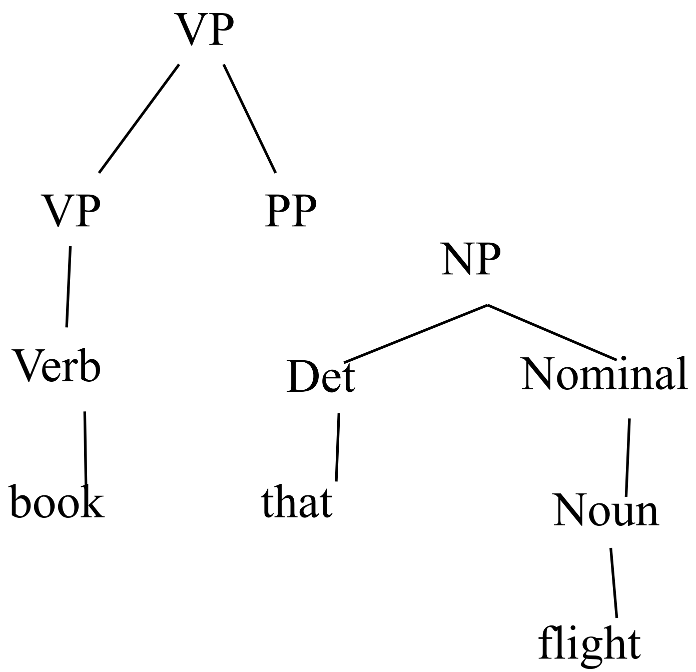
Bottom Up Parsing



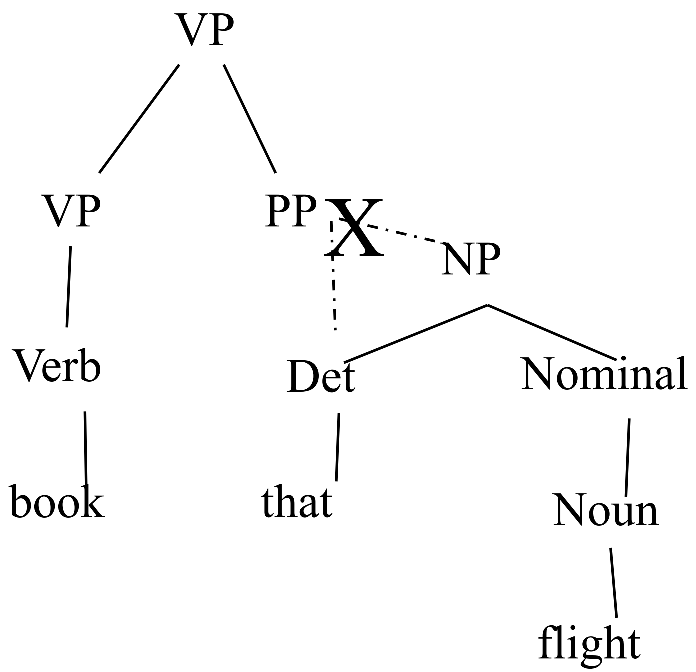
Bottom Up Parsing



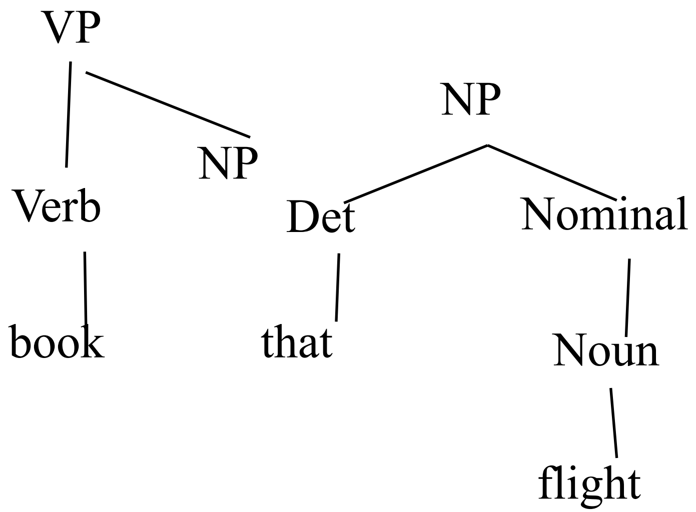
Bottom Up Parsing



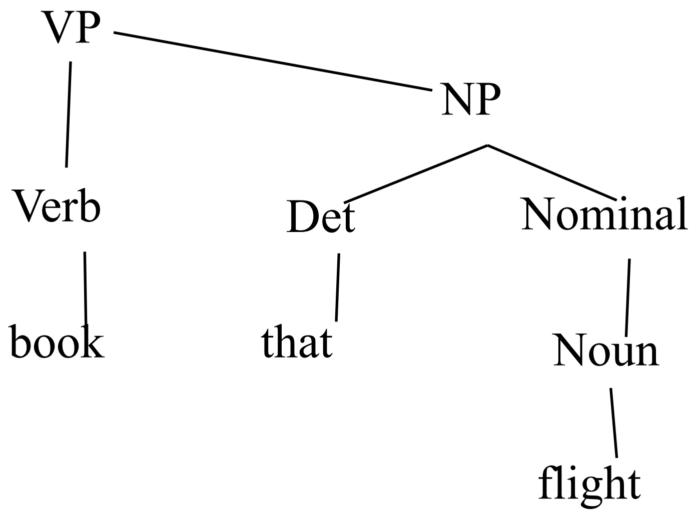
Bottom Up Parsing



Bottom Up Parsing



Bottom Up Parsing



Bottom Up Parsing