Syntax:
Context-Free Grammars

LING 571 — Deep Processing Techniques for NLP

Oct 5, 2020

Shane Steinert-Threlkeld

Announcements

- Please use Canvas for discussions! There really are no stupid questions :)
- Output format:
  - Try to copy *exactly*; your hw1 script run with the toy data should produce output that exactly matches toy_output.txt
  - Single space after the colon
  - Truncate decimals to 3 places
- Python versions: use full paths to binaries; see `ls /opt | grep python`
- File paths will be given as full paths, so your script should accept those
- Condor: we will use for grading, so if you want to test, that's a good idea (and will be necessary in the future)

Roadmap

- **Constituency**
- Context-free grammars (CFGs)
- English Grammar Rules
- Grammars — Revisiting our Motivation
- Treebanks
- Speech and Text
- Parsing

Constituency

- Some examples of noun phrases (NPs):

Harry the Horse	a high-class spot such as Mindy's
the Broadway coppers	the reason he comes into the Hot Box
they	three parties from Brooklyn

- How do we know that these are constituents?
  - We can perform constituent tests

Constituent Tests

- Many types of tests for constituency *(see Sag, Wasow, Bender (2003), pp. 29-33)*
- One type (for English) is **clefting**
  - It is ______ that ______
  - Is the resulting sentence valid English?

It is **the Supreme Court** that made the ruling	✔
It is **the Supreme Court of the United States** that made the ruling	✔
It is **they** that made the ruling	✔
It is **the Supreme Court of** that made the ruling	✗

Constituent Tests

- Another popular one: **coordination.**
  - Only constituents *of the same type* can be coordinated.
  - … ______ CONJ ______ …

Shane and all of the students	✔
three players and the coach’s brother	✔
The friends drank wine and laughed at the show together.	✔
The friends drank wine and all of the students together.	✗

Roadmap

- Constituency
- **Context-free grammars (CFGs)**
- English Grammar Rules
- Grammars — Revisiting our Motivation
- Treebanks
- Speech and Text
- Parsing

Representation: Context-free Grammars

- CFGs: 4-tuple
  - A set of **terminal** symbols: Σ
    - (think: words)
  - A set of **nonterminal** symbols: N
    - *(Think: phrase categories)*
  - A set of **productions** P:
    - of the form $\boldsymbol{A} \rightarrow \alpha$
    - Where A is a non-terminal and $\alpha \in (\Sigma \cup N)^*$
  - A **start** symbol $S \in N$

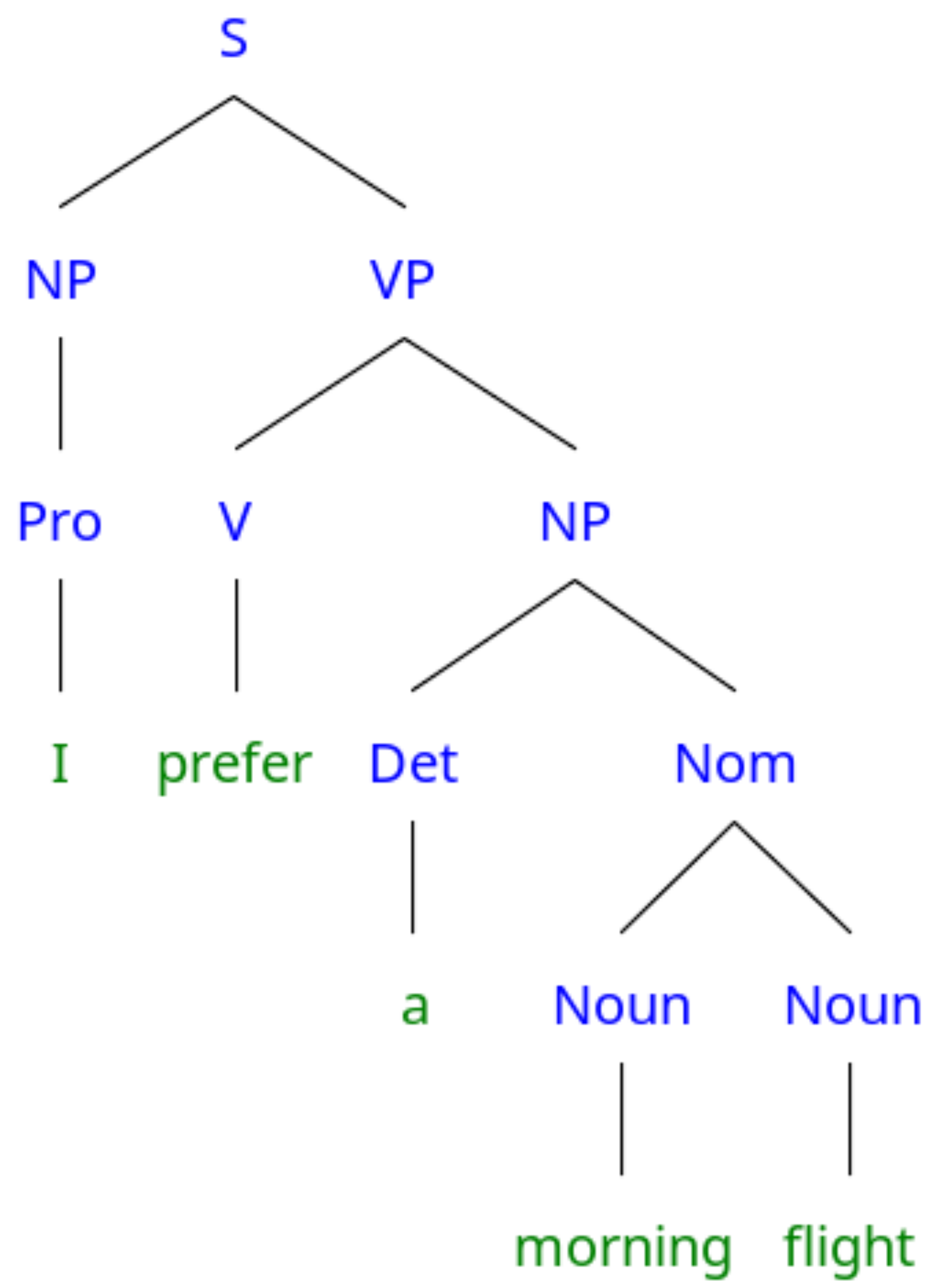
CFG Components

- Productions:
  - One non-terminal on LHS and any seq. of terminals and non-terminals on RHS
    - *S* → *NP VP*
    - *VP* → *V NP PP* | *V NP*
    - *Nominal* → *Noun* | *Nominal Noun*
    - *Noun* → ‘dog’ | ‘cat’ | ‘rat’
    - *Det* → ‘the’

Grammar Rules			Examples
S	⟶	*NP VP*	I + want a morning flight
NP	⟶	*Pronoun*	I
	|	*Proper-Noun*	Los Angeles
	|	*Det Nominal*	a + flight
Nominal	⟶	*Nominal Noun*	morning + flight
	|	*Noun*	flights
VP	⟶	*Verb*	do
	|	*Verb NP*	want + a flight
	|	*Verb NP PP*	leave + Boston + in the morning
	|	*Verb PP*	leaving + on Thursday
PP	⟶	*Preposition NP*	from + Los Angeles

Jurafsky & Martin, Speech and Language Processing, p.390

Parse Tree

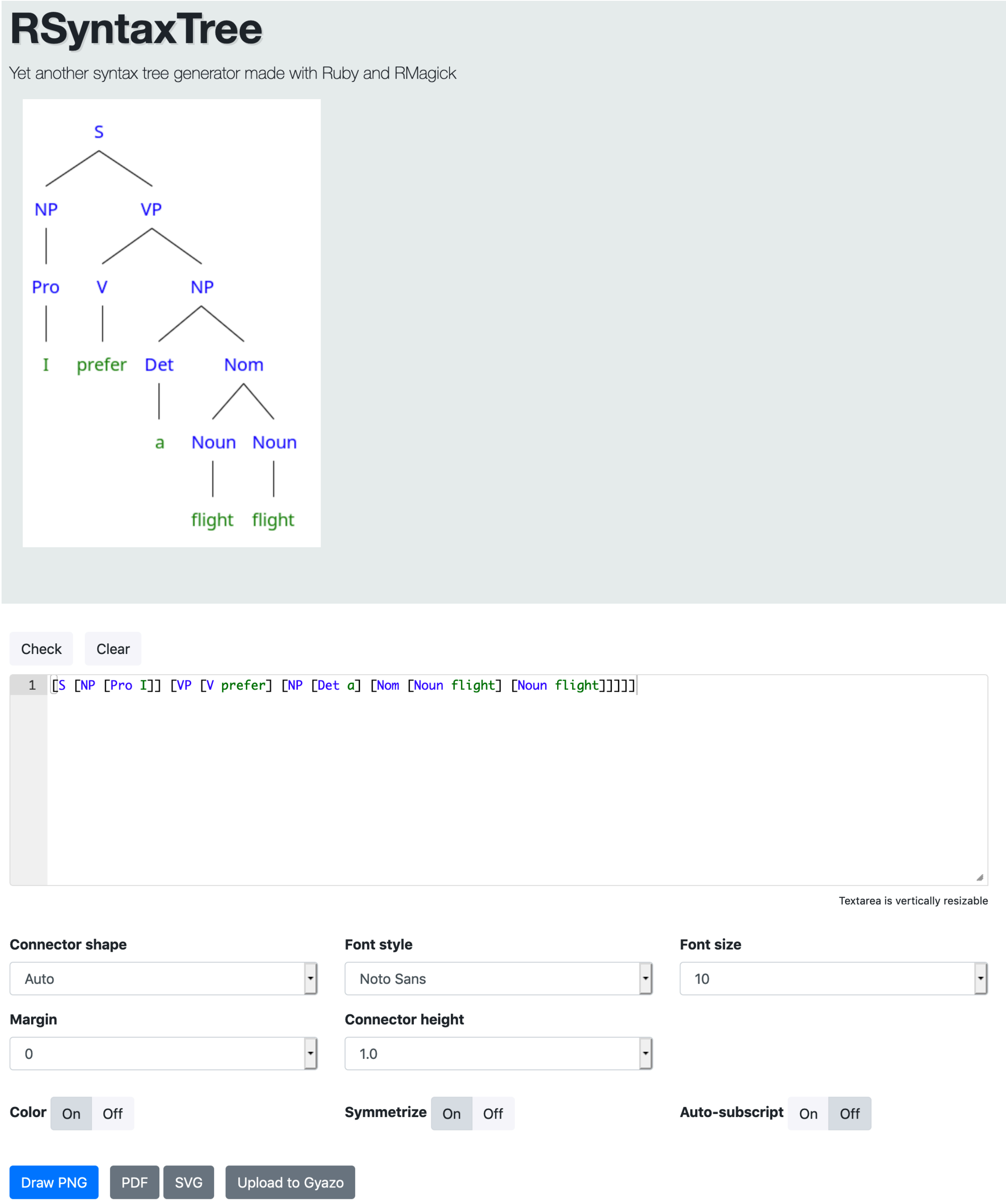


Some English Grammar

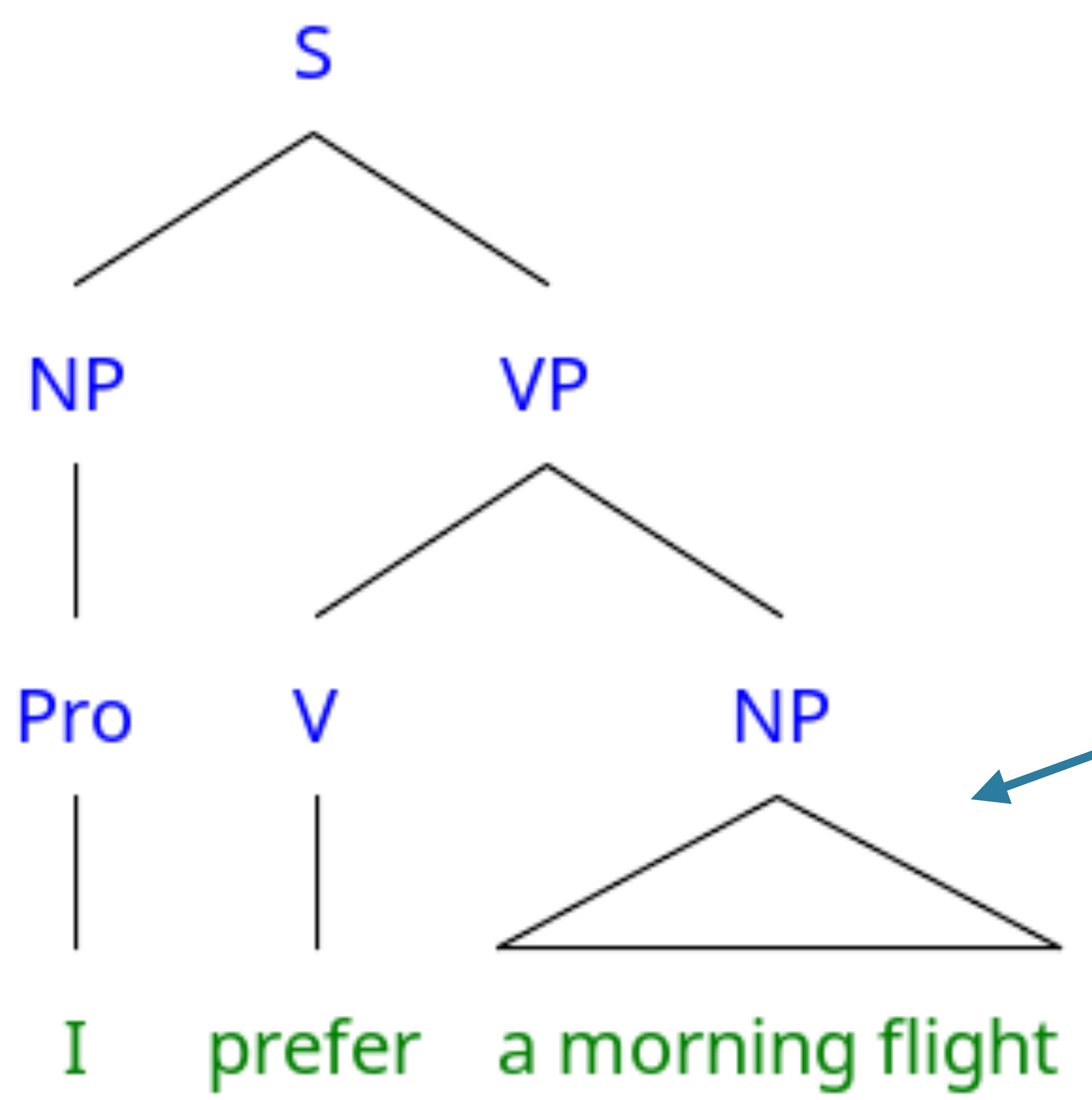
- Sentences: Full sentence or clause; a complete thought
- **Declarative**: ***S*** → ***NP VP***
  - (S (NP I) (VP want a flight from SeaTac to Amsterdam))
- **Imperative**: ***S*** → ***VP***
  - (VP Show me the cheapest flight from New York to Los Angeles.)
- **Yes-no Question**: ***S*** → ***Aux NP VP***
  - (Aux Can) (NP you) (NP give me the nonstop flights to Boston?)
- **Wh-subject question**: ***S*** → ***Wh-NP VP***
  - (Wh-NP Which flights) (VP arrive in Pittsburgh before 10pm?)
- **Wh-non-subject question**: ***S*** → ***Wh-NP Aux NP VP***
  - (Wh-NP What flights) (Aux do) (NP you) (VP have from Seattle to Orlando?)

Visualizing Parse Trees

- ```
  >>> tree = nltk.tree.Tree.fromstring("(S (NP (Pro I)) (VP (V prefer) (NP (Det a) (Nom (Noun flight) (Noun flight)))))")
  >>> tree.draw()
  ```
- Web apps: https://yohasebe.com/rsyntaxtree/
- LaTeX: qtree (/ tikz-qtree) package


  ```

Partial Parses



When internal structure
doesn't matter
for whatever reason

The Noun Phrase

- Noun phrase constituents can take a range of different forms:

Harry the Horse	a magazine
water	twenty-three alligators
Ram’s homework	the last page of Ram’s homework’s

- We’ll examine a few ways these differ

The Determiner

- Determiners provide referential information about an NP
- Often position the NP within the current discourse

a stop	the flights	this flight
those flights	any flights	some flights

- Can more explicitly introduce an entity as part of the specifier

United's flight
United's pilot's union
Denver's mayor's mother's canceled flight

The Determiner

- *Det → DT*
  - *'the', 'this', 'a', 'those'*
- *Det → **NP*** **'s**
  - **"United's flight": (Det (NP United) 's)**
  - **"the professor's favorite brewery": (Det (NP (Det the) (NP professor)) 's)**

The Nominal

- Nominals contain pre- and post-head noun modifiers
  - Occurs after the determiner (in English)
- Can exist as just a bare noun:
  - *Nominal* → *Noun*
    - PTB POS: `NN, NNS, NNP, NNPS`
    - *'flight', 'dinners', 'Chicago Midway', 'UW Libraries'*

Pre-nominal modifiers *("Postdeterminers")*

- Occur before the head noun in a nominal
- Can be any combination of:
  - Cardinal numbers (e.g. *one*, *fifteen*)
  - Ordinal numbers (e.g. *first*, *thirty-second*)
  - Quantifiers (e.g. *some*, *a few*)
  - Adjective phrases (e.g. *longest*, *non-stop*)

Postmodifiers

- Occur after the head noun
- In English, most common are: (*a flight…*)
  - Prepositional phrase (e.g. … *from Cleveland*)
  - non-finite clause (e.g. … *arriving after eleven a.m.*)
  - relative clause (e.g. … *that serves breakfast*)

Combining Everything

- *NP → (Det) Nom*
- *Nom → (Card) (Ord) (Quant) (AP) Nom*
- *Nom → Nom PP*

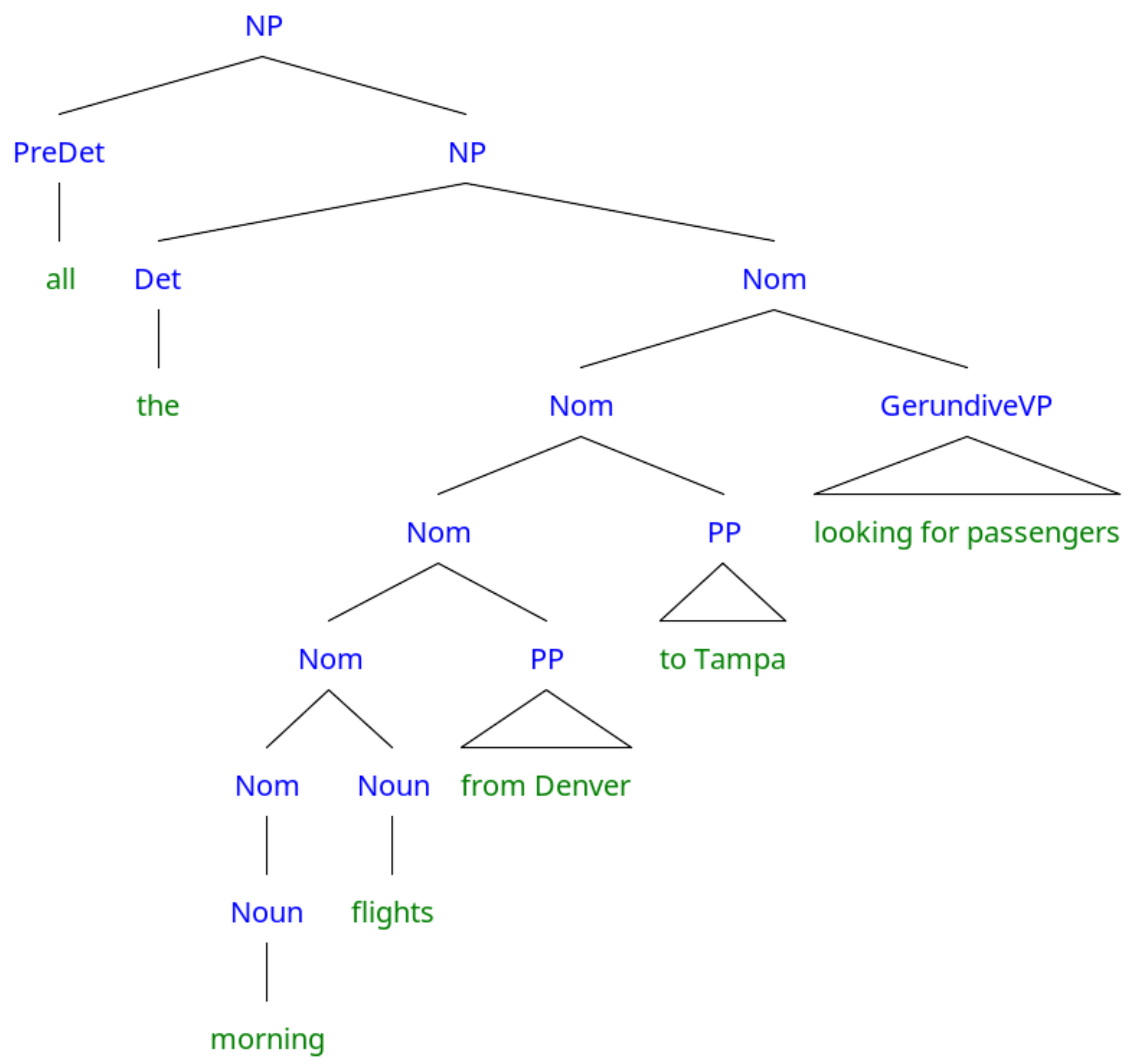
- The least expensive fare
- one flight
- the first route
- the last flight from Chicago

Before the Noun Phrase

- “Predeterminers” can “scope” noun phrases
  - e.g. *‘all,’*
  - *“all the morning flights from Denver to Tampa”*

A Complex Example

- *“all the morning flights from Denver to Tampa looking for passengers”*



Verb Phrases and Subcategorization

- With this grammar:

VP	$\longrightarrow$	*Verb*
	|	*Verb NP*
	|	*Verb NP NP*

- This grammar licenses the following ***correctly***:
  - *The teacher handed the student a book*
- And the following ***incorrectly*** (i.e. the grammar "overgenerates"):
  - **The teacher handed the student*
  - **The teacher handed a book*
  - **The teacher handed*

Verb Phrases and Subcategorization

- With this grammar:

VP	$\longrightarrow$	*Verb*
	|	*Verb NP*
	|	*Verb NP NP*

- It also licenses
  - **The teacher handed a book the student*
- This is problematic for semantic reasons, which we'll cover later.

Verb Phrase and Subcategorization

- Verb phrases include a verb and ***optionally other constituents***
- Subcategorization frame
  - what constituent arguments the verb requires

VP → *Verb Ø*	disappear
VP → *Verb* ***NP***	book a flight
VP → *Verb* ***PP PP***	fly from Chicago to Seattle
VP → *Verb* ***S***	think I want that flight
VP → *Verb* ***VP***	want to arrange three flights

CFGs and Subcategorization

- Issues?
  - “I prefer United has a flight.” (→ ***S***)
  - “I prefer a window seat.” (→ ***NP***)
- How can we solve this problem?
  - Create explicit subclasses of verb
    - *Verb-with-NP* → …
    - *Verb-with-S-complement* → …
  - Is this a good solution?
    - No, explosive increase in number of rules
    - Similar problem with agreement (NN↔ADJ↔PRON↔VB)

CFGs and Subcategorization

- Better solution:
  - *Feature structures:*
    - Further nested information
    - a.k.a → *Deeper* analysis!
  - Will get to this toward end of the month

Roadmap

- Constituency
- Context-free grammars (CFGs)
- English Grammar Rules
- **Grammars — Revisiting our Motivation**
- Treebanks
- Speech and Text
- Parsing

Grammars… So What?

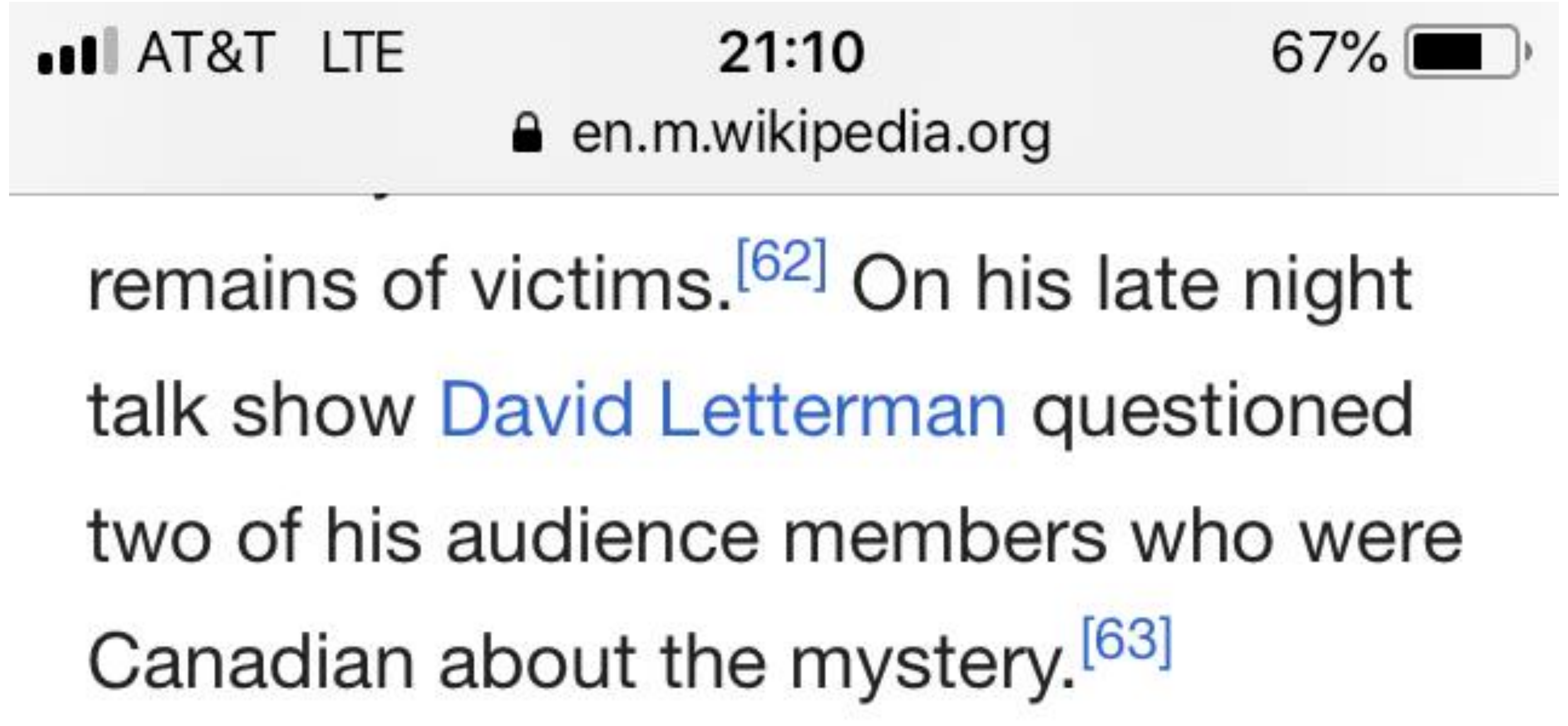
- Grammars propose a formal way to make distinctions in syntax
- Distinctions in syntax can help us get a hold on distinctions in meaning

Syntax to the Rescue!

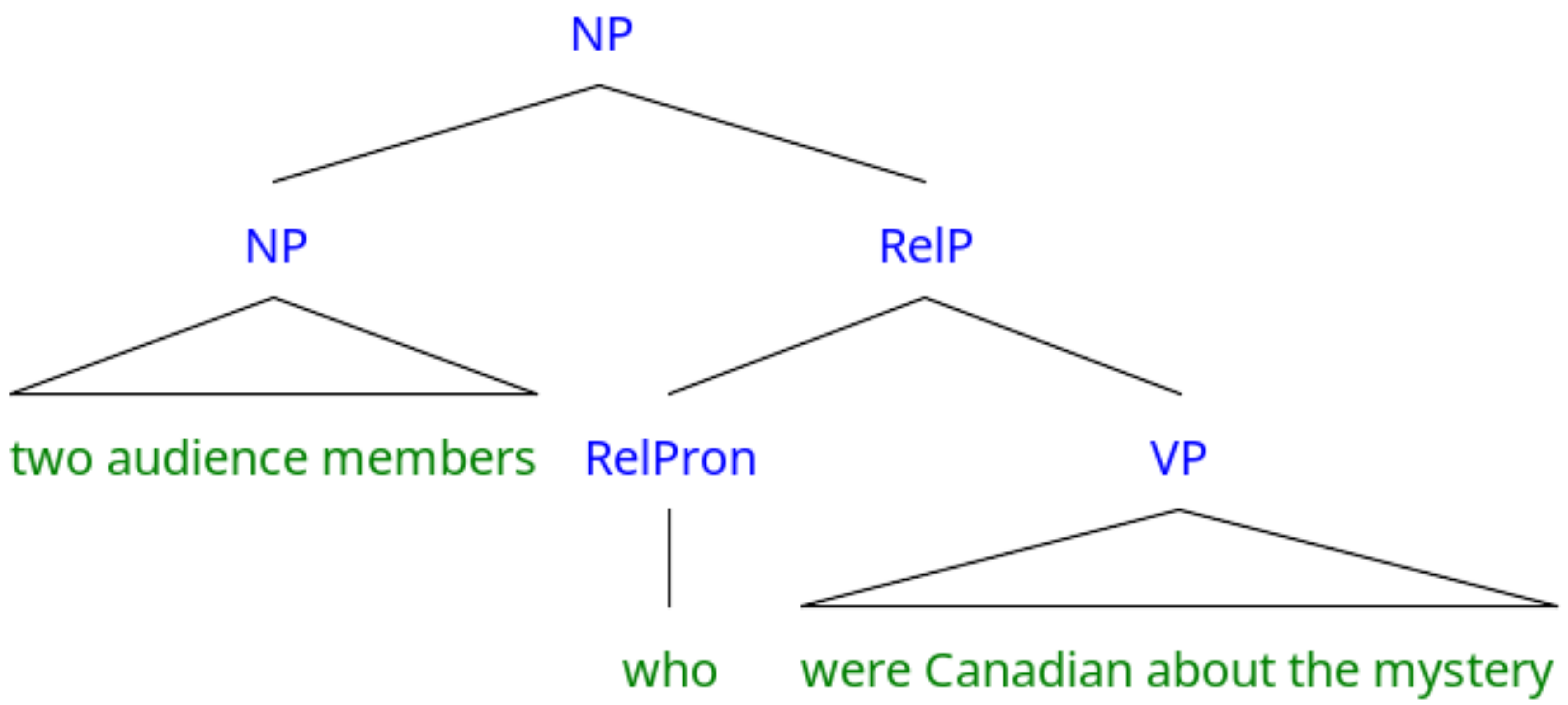
- Possible Interpretations:

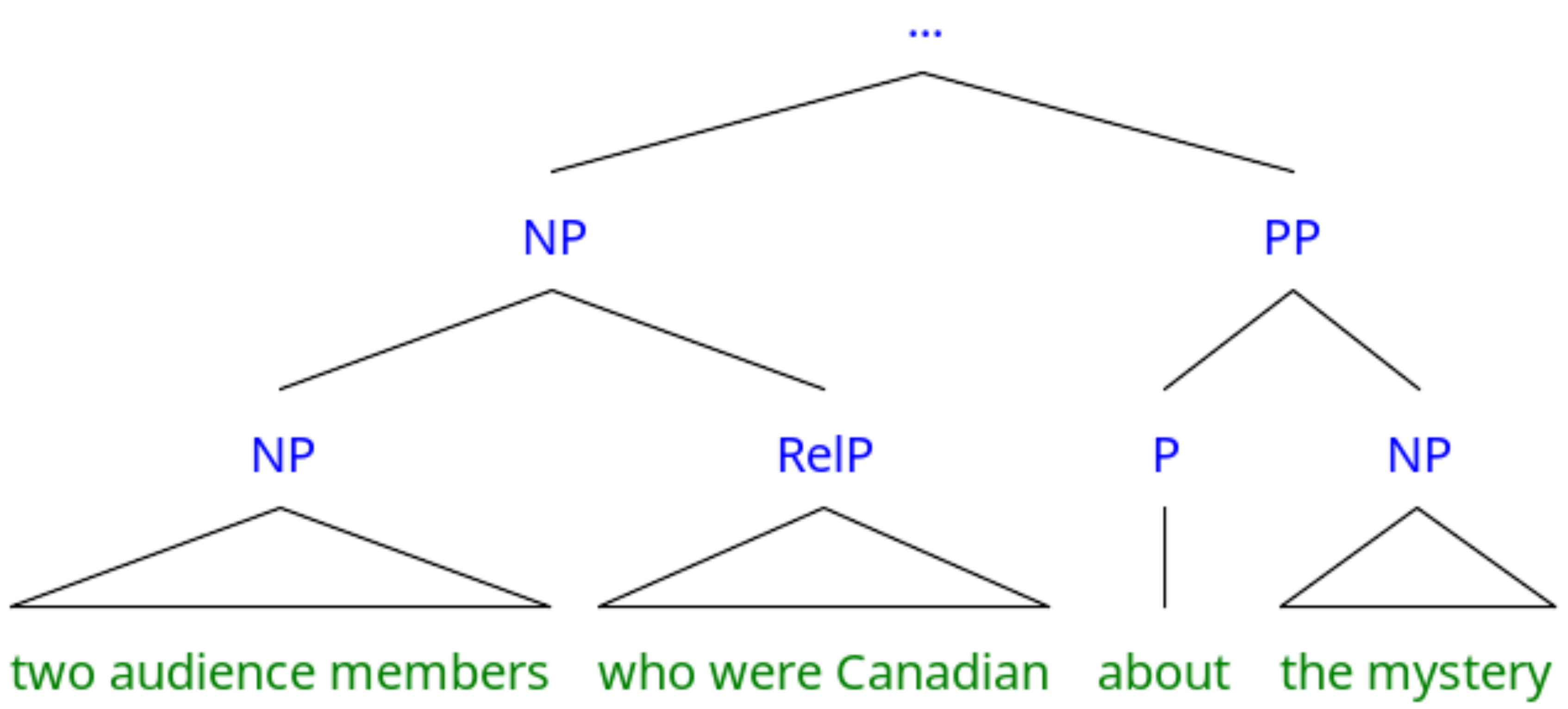
  Two audience members, when questioned, behaved Canadian-ly

  Two audience members, who happened to be Canadian Citizens, were questioned



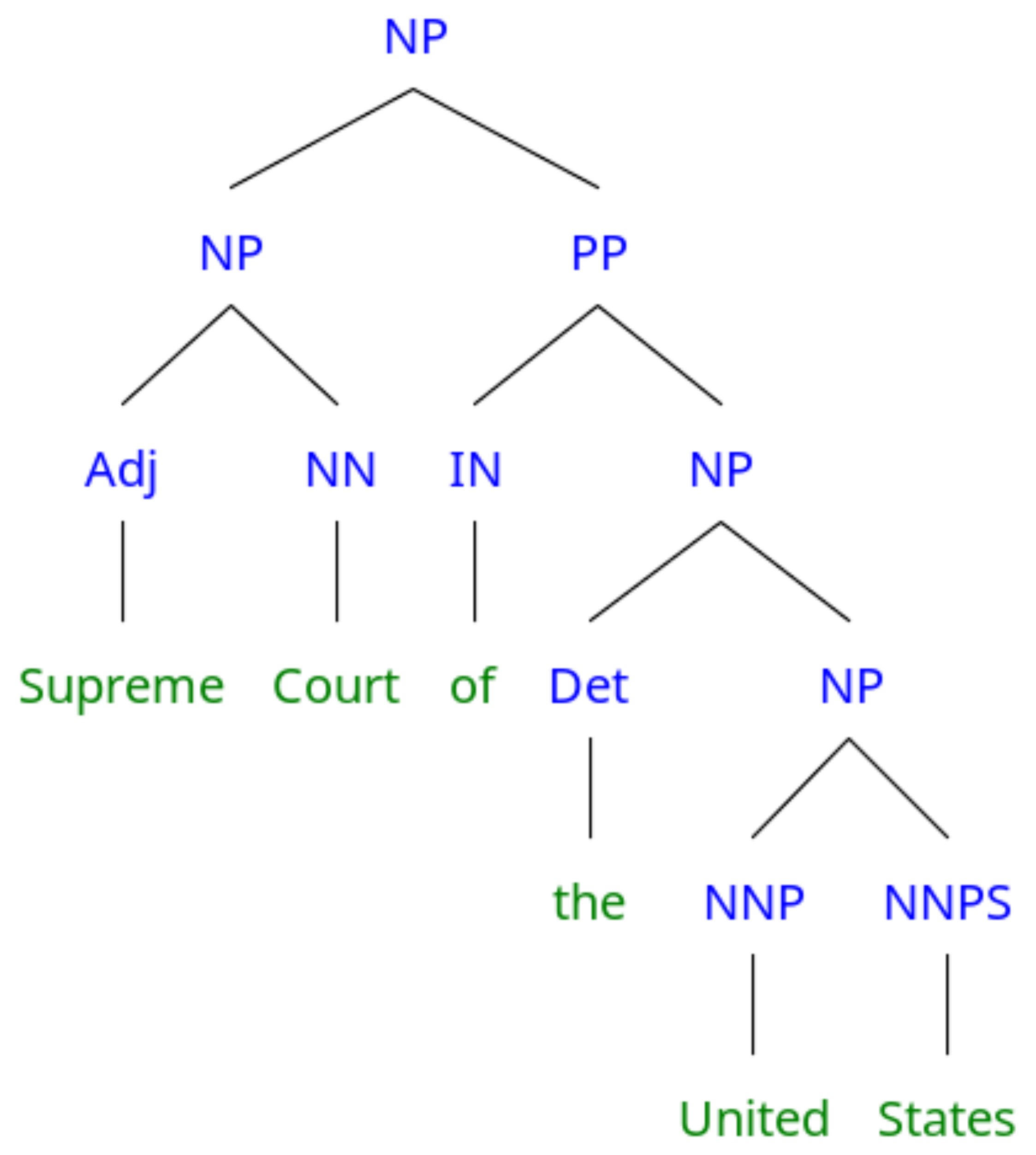
h/t to Amandalynne Paullada


NP
NP
RelP
two audience members
RelPron
VP
who
were Canadian about the mystery


...
NP
PP
NP
RelP
P
NP
two audience members
who were Canadian
about
the mystery

Grammars Promote Deeper Analysis

- Shallow techniques useful, but limited
  - “Supreme Court of the United States”
  - `ADJ NN IN DET NNP NNPS`
  - What does this tell us about the fragment?
- vs.



Grammars Promote Deeper Analysis

- Meaning implicit in this analysis tree:
  - "*The United States*" is an entity
  - The court is specific to the US
- Inferable from this tree:
  - "*The United States*" is an entity that can possess (grammatically) other institutions

Roadmap

- Constituency
- Context-free grammars (CFGs)
- English Grammar Rules
- Grammars — Revisiting our Motivation
- **Treebanks**
- Speech and Text
- Parsing

Treebanks

- Instead of writing out grammars by hand, could we learn them from data?
- Large corpus of sentences
- All sentences annotated syntactically with a parse
- Built semi-automatically
  - Automatically parsed, manually corrected

Penn Treebank

- A well-established and large treebank
- English:
  - Brown Univ. Standard Corp. of Present-Day Am. Eng.
  - Switchboard (conversational speech)
  - ATIS (human-computer dialog, Airline bookings)
  - Wall Street Journal
- Chinese:
  - Xinhua, Sinoarma (newswire)
- Arabic
  - Newswire, Broadcast News + Conversation, Web Text…

Other Treebanks

- DeepBank (HPSG)
- Prague Dependency Treebank (Czech: Morphologically rich)
- Universal Dependency Treebank (60 languages, reduced POS tags)
- CCGBank (Penn, but with CCG annotations)

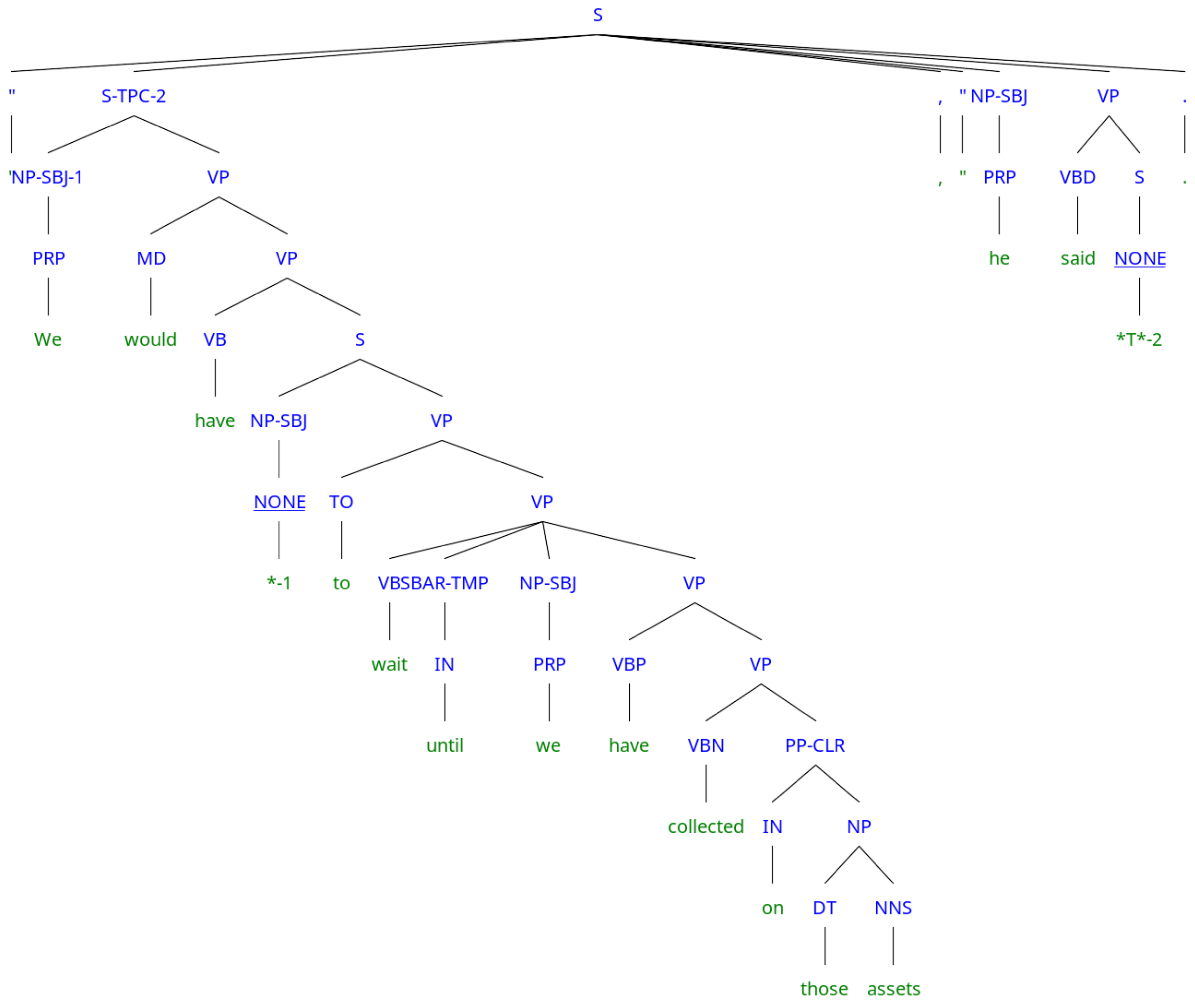
Treebanks

- Include wealth of language information
  - Traces (for movement analyses)
  - Grammatical function (subject, topic, etc)
  - Semantic function (temporal, location)
- Implicitly constitute grammar of language
  - Can read off rewrite rules from bracketing
  - Not only presence of rules, but frequency counts
  - Will be crucial in building statistical parsers

Treebank WSJ Example

```
(S ('' '')
   (S-TPC-2
   (NP-SBJ-1 (PRP We))
   (VP (MD would)
     (VP (VB have)
         (S
           (NP-SBJ (-NONE- *-1))
           (VP (TO to)
                (VP (VB wait)
                     (SBAR-TMP (IN until))
                     (NP-SBJ (PRP we))
                     (VP (VBP have)
                       (VP (VBN collected)
                         (PP-CLR (IN on)
                            (NP (DT those) (NNS assets)))))))))))
   (, ,) ('' '')
   (NP-SBJ (PRP he))
   (VP (VBD said)
     (S (-NONE- *T*-2) ))
   (. .)
)
```

Treebank WSJ Example



Treebanks & Corpora on Patas

```
patas$ ls /corpora
birkbeck                    grammars          opt
coconut                     HathiTrust        private
Communicator2000_Emotion    ICAME             proj-gutenberg
ComParE                     ICSI              reuters
Conll                       JRC-Acquis.3.0    scope
delph-in                    LDC               tc-wikipedia
DUC                         LEAP              TREC
ELRA                        lemur             treebanks
enron_email_dataset         levow             UIC
europarl                    mdsd-2.0          UWCL
europarl-old                med-data          UWCSE
framenet                    nltk
freebase                    OANC
```

Treebanks & Corpora on Patas

- Many large corpora from LDC, such as the Penn Treebank v3:
  - `/corpora/LDC/LDC99T42/`
  - Find the full LDC corpora catalog online: catalog.ldc.upenn.edu
- Web search interface: https://cldb.ling.washington.edu/livesearch-corpus-form.php
- Many corpus samples in NLTK
  - `/corpora/nltk/nltk-data`
- **NOTE:** do not move corpora, either *within* or *off of* patas!!

Treebank Issues

- Large, expensive to produce
- Complex
  - Agreement among annotators can be an issue
- Labeling implicitly captures bias in theory
  - Penn Treebank is “bushy,” long productions
- Enormous numbers of rules
  - **4,500** rules in PTB for VP alone
  - 1M rule tokens; 17,500 distinct types — and counting!

Roadmap

- Constituency
- Context-free grammars (CFGs)
- English Grammar Rules
- Grammars — Revisiting our Motivation
- Treebanks
- **Speech and Text**
- Parsing

Spoken vs. Written

- Can we just use models for written language directly?
- **NO!**
- Challenges of spoken language:
  - Disfluency
    - *Can I um uh can I g– get a flight to Boston on the fifteenth?*
  - Short, fragmentary
    - *Uh one way*
    - Only 37% of Switchboard utterances > 2 words
  - More pronouns, ellipsis
    - *That one*

Roadmap

- Constituency
- Context-free grammars (CFGs)
- English Grammar Rules
- Grammars — Revisiting our Motivation
- Treebanks
- Speech and Text
- **Parsing**

Computational Parsing

- Given a grammar, how can we derive the analysis of an input sentence?
  - Parsing as search
  - CKY parsing
- Given a body of (annotated) text, how can we derive the grammar rules of a language, and employ them in automatic parsing?
  - Treebanks & PCFGs

What is Parsing?

- CFG parsing is the task of assigning trees to input strings
  - For any input ***A*** and grammar ***G***
    - …assign ≥0 parse trees ***T*** that represent its syntactic structure, and…
    - Cover all and only the elements of ***A***
    - Have, as root, the start symbol ***S*** of ***G***
    - …do not necessarily pick one single (or correct) analysis
- Subtask: Recognition
  - Given input ***A***, ***G*** – is ***A*** in language defined by ***G*** or not?

Motivation

- Is this sentence in the language — i.e. is it “grammatical?”
  - ** I prefer United has the earliest flight.*
  - FSAs accept regular languages defined by finite-state automata.
  - Parsers accept languages defined by CFG (equiv. pushdown automata).
- What is the syntactic structure of this sentence?
  - *What airline has the cheapest flight?*
  - *What airport does Southwest fly from near Boston?*
  - Syntactic parse provides framework for semantic analysis
    - What is the subject? Direct object?

Parsing as Search

- Syntactic parsing searches through possible trees to find one or more trees that derive input
- Formally, search problems are defined by:
  - Start state ***S***
  - Goal state ***G*** (with a test)
  - Set of actions that transition from one state to another
    - "Successor function"
  - A path cost function

Parsing as Search: One Model

- Start State ***S***: Start Symbol
- Goal test:
  - Does the parse tree cover all of, and only, the input?
- Successor function:
  - Expand a nonterminal using a production where nonterminal is the LHS of the production
- Path cost:
  - …ignored for now.

Parsing as Search: One Model

- Node:
  - Partial solution to search problem (partial parse)
- Search start node (initial state):
  - Input string
  - Start symbol of CFG
- Goal node:
  - Full parse tree: covering all of, and only the input, rooted at ***S***

Search Algorithms

- Depth First
  - Keep expanding nonterminals until they reach words
  - If no more expansions available, back up
- Breadth First
  - Consider all parses that expand a single nonterminal…
  - …then all with two expanded, etc…
- Other alternatives, if have associated path costs.

Parse Search Strategies

- Two constraints on parsing:
  - Must start with the start symbol
  - Must cover exactly the input string
- Correspond to main parsing search strategies
  - Top-down search (Goal-directed)
  - Bottom-up search (Data-driven search)

A Grammar

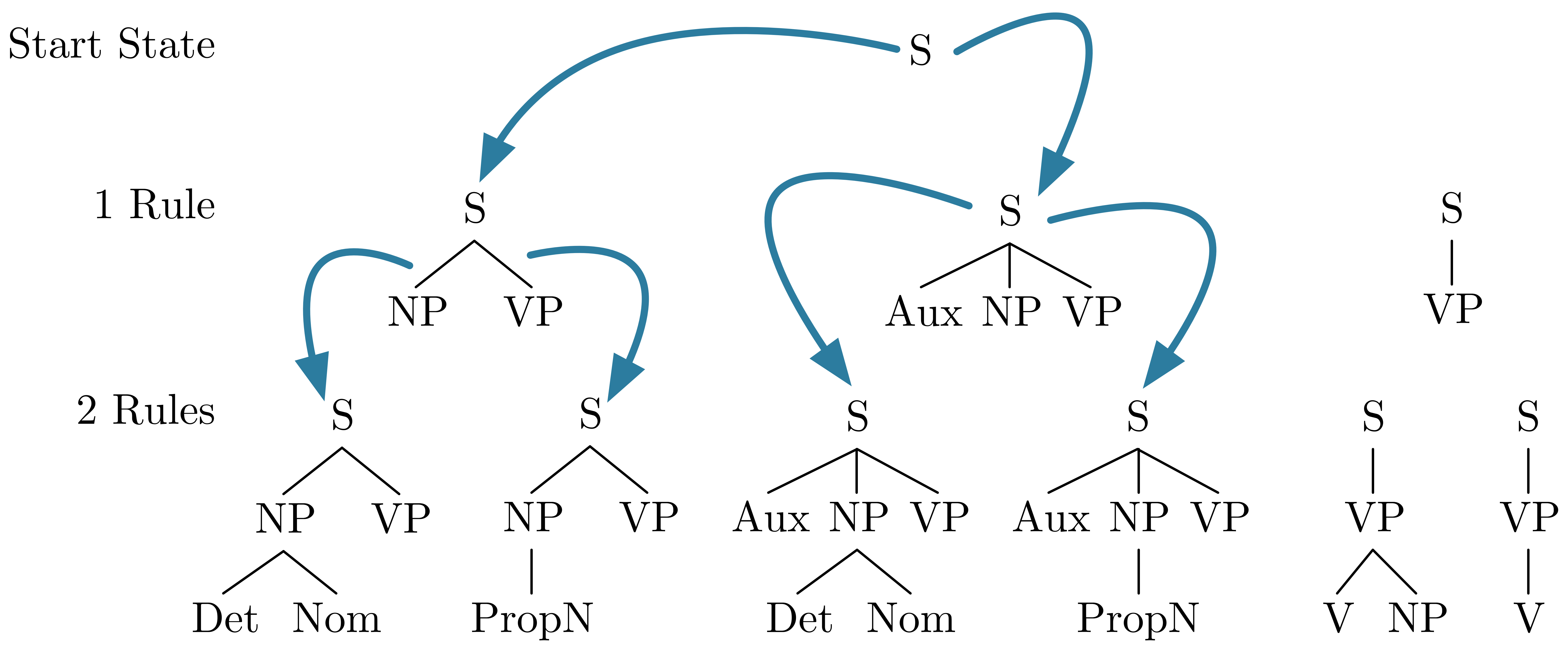
Grammar	Lexicon
S → NP VP	*Det → that ǀ this ǀ a*
S → Aux NP VP	*Noun → book ǀ flight ǀ meal ǀ money*
S → VP	*Verb → book ǀ include ǀ prefer*
NP → Pronoun	*Pronoun → I ǀ she ǀ me*
NP → Proper-Noun	*Proper-Noun → Houston ǀ NWA*
NP → Det Nominal	*Aux → does*
Nominal → Noun	*Preposition → from ǀ to ǀ on ǀ near ǀ through*
Nominal → Nominal Noun	
Nominal → Nominal PP	
VP → Verb	
VP → Verb NP	
VP → Verb NP PP	
VP → Verb PP	
VP → VP PP	
PP → Preposition NP	

Jurafsky & Martin, Speech and Language Processing, p.390

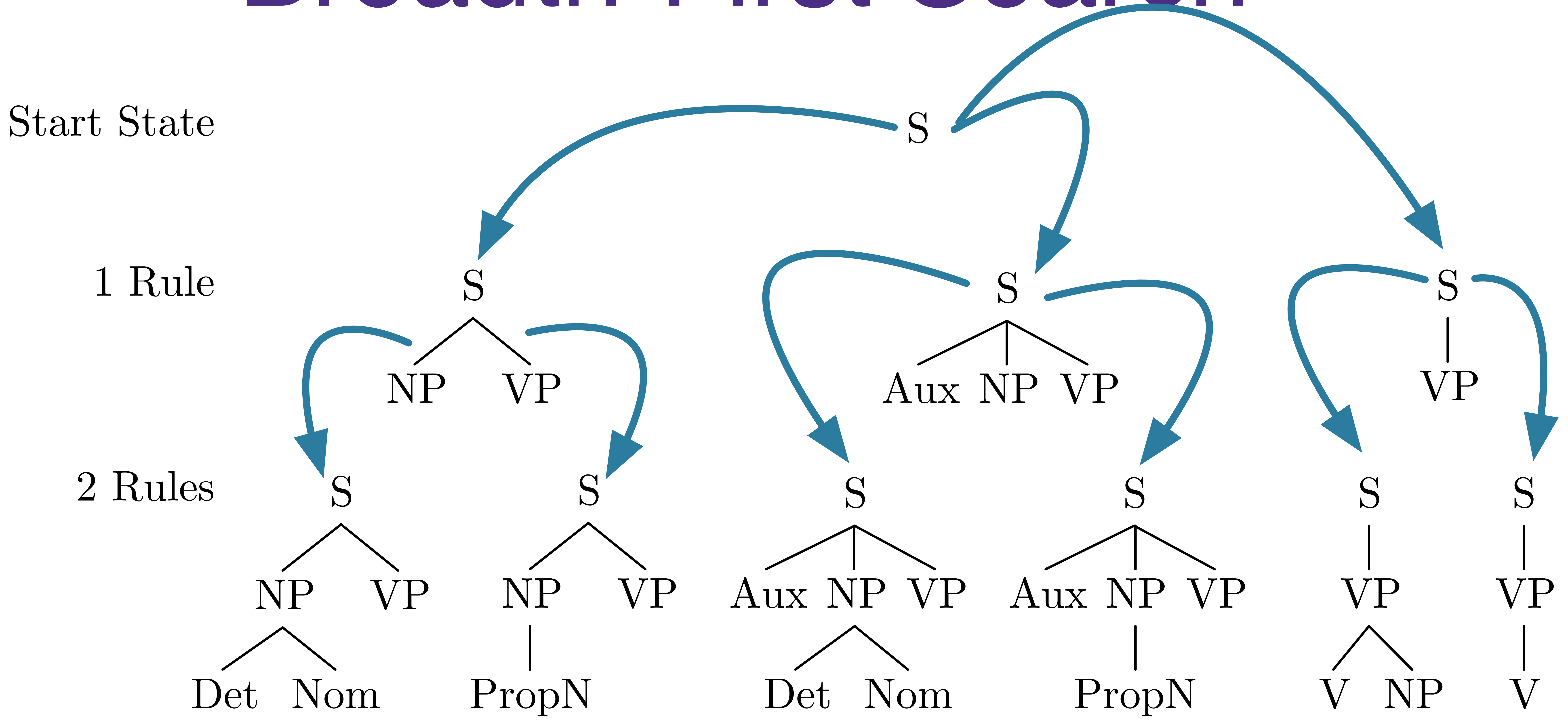
Top-down Search

- All valid parse trees must be rooted with start symbol
- Begin search with productions where S is on LHS
  - e.g. $S \rightarrow NP\ VP$
- Successively expand nonterminals
  - e.g. $NP \rightarrow Det\ Nominal$; $VP \rightarrow V\ NP$
- Terminate when all leaves are terminals

Depth-First Search

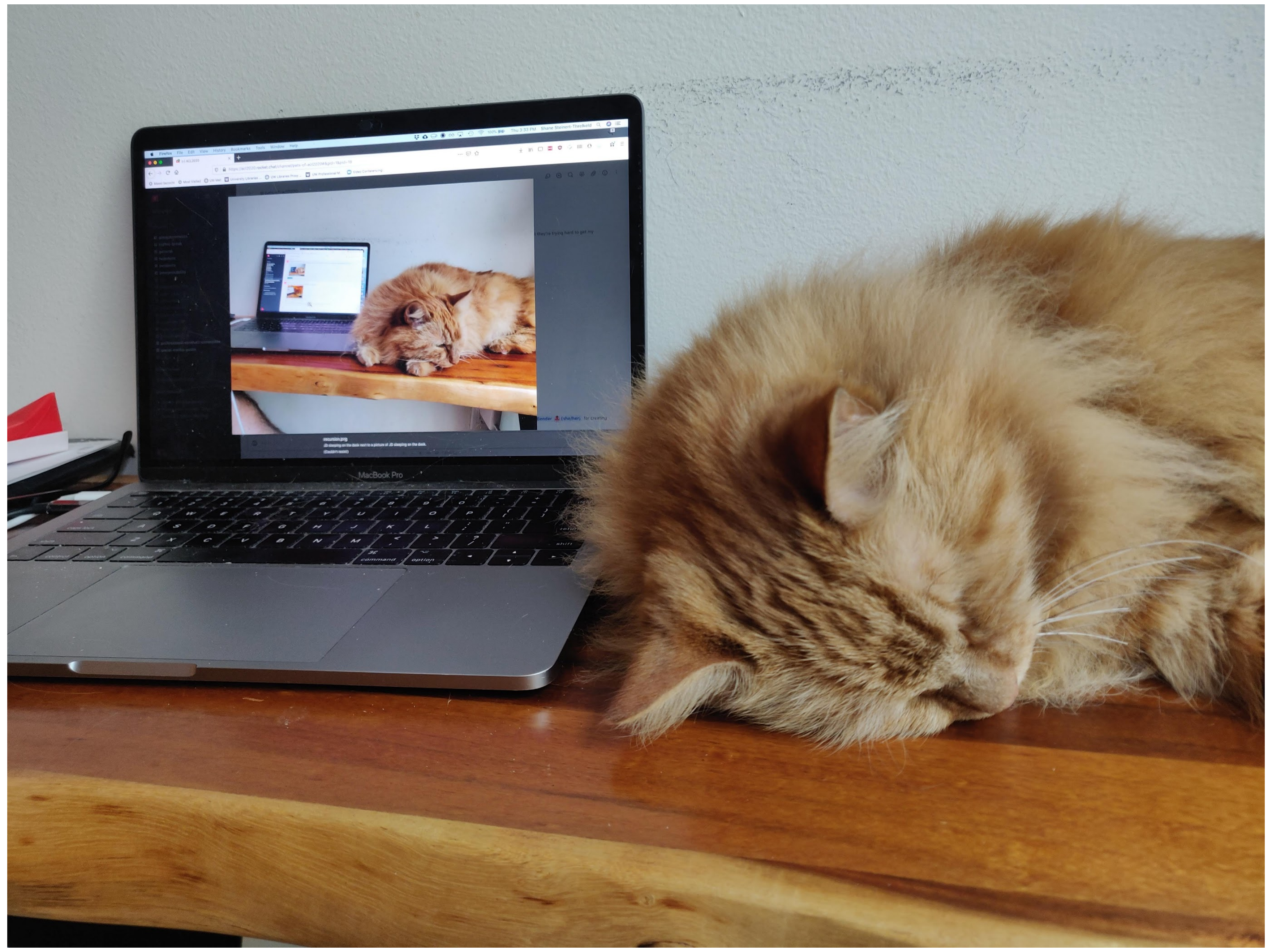


Breadth-First Search



Pros and Cons of Top-down Parsing

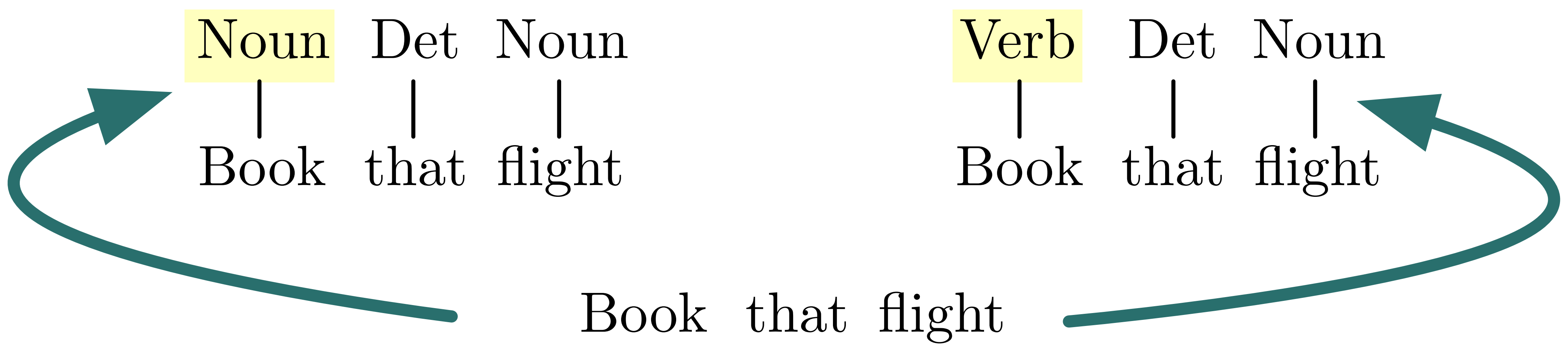
- Pros:
  - Doesn't explore trees not rooted at S
  - Doesn't explore subtrees that don't fit valid trees
- Cons:
  - Produces trees that may not match input
  - May not terminate in presence of recursive rules
  - May re-derive subtrees as part of search

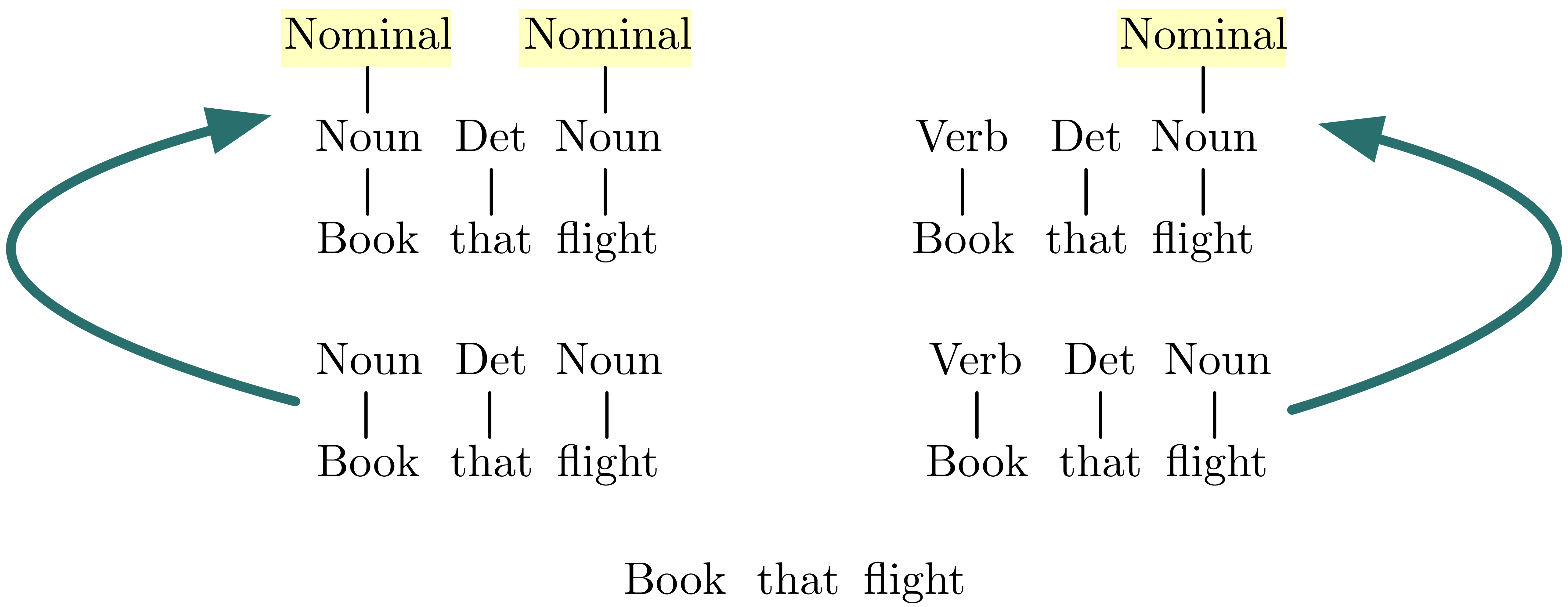


Bottom-Up Parsing

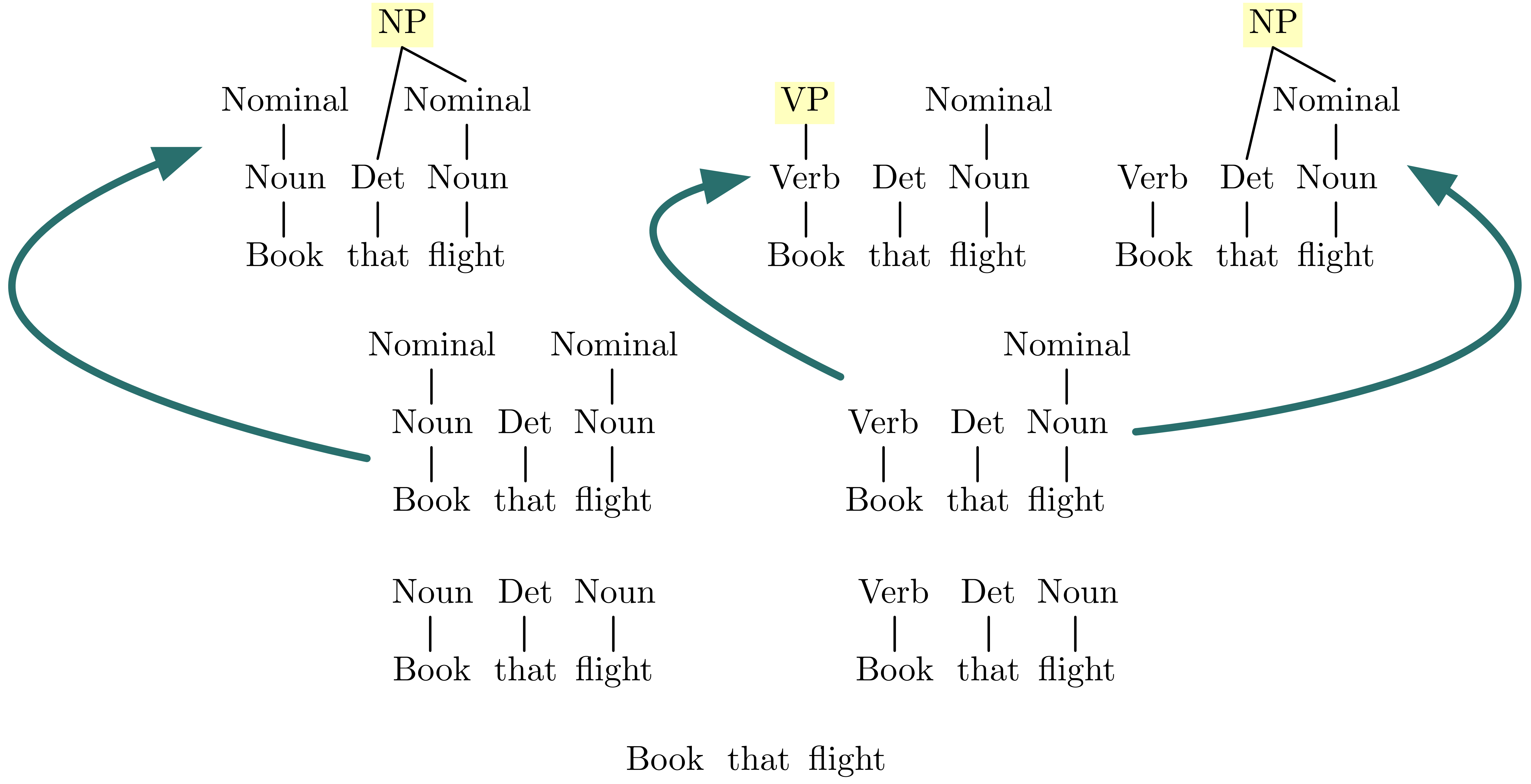
- Try to find all trees that span the input
  - Start with input string
    - Book that flight
- Use all productions with current subtree(s) on RHS
  - e.g. ***N*** → Book; ***V*** → Book
- Stop when spanned by S, or no more rules apply

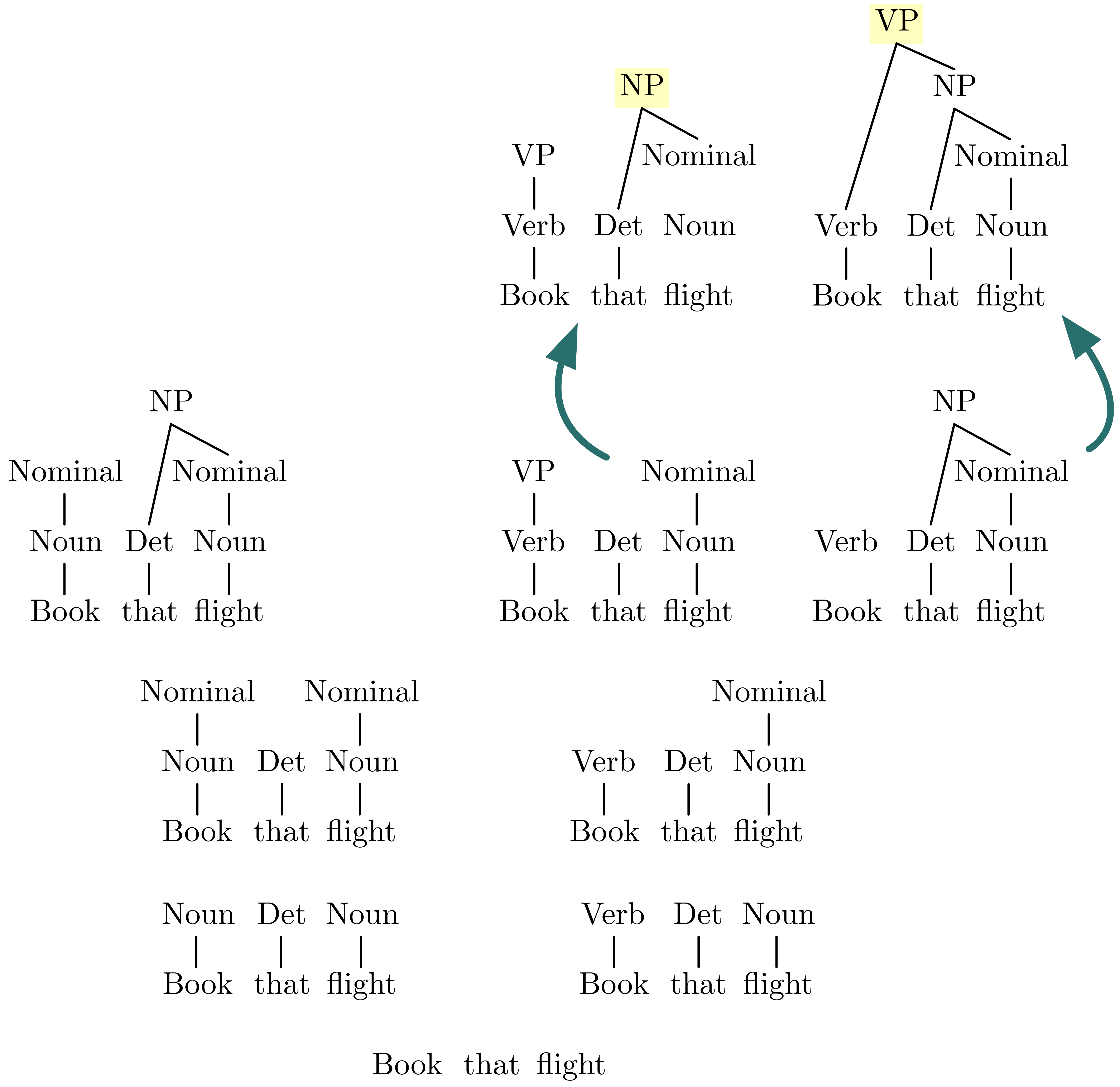
Book that flight


Noun Det Noun
Book that flight
Verb Det Noun
Book that flight
Book that flight



Nominal
Nominal
Noun
Det
Noun
Book
that
flight
Nominal
Verb
Det
Noun
Book
that
flight
Noun
Det
Noun
Book
that
flight
Verb
Det
Noun
Book
that
flight
Book that flight


NP
Nominal Nominal
Noun Det Noun
Book that flight
VP Nominal
Verb Det Noun
Book that flight
NP
Nominal
Verb Det Noun
Book that flight
Nominal Nominal
Noun Det Noun
Book that flight
Nominal
Verb Det Noun
Book that flight
Noun Det Noun
Book that flight
Verb Det Noun
Book that flight
Book that flight


VP
NP
NP
VP
Nominal
Nominal
Verb
Det
Noun
Verb
Det
Noun
Book
that
flight
Book
that
flight
NP
NP
Nominal
Nominal
VP
Nominal
Nominal
Noun
Det
Noun
Verb
Det
Noun
Verb
Det
Noun
Book
that
flight
Book
that
flight
Book
that
flight
Nominal
Nominal
Nominal
Noun
Det
Noun
Verb
Det
Noun
Book
that
flight
Book
that
flight
Noun
Det
Noun
Verb
Det
Noun
Book
that
flight
Book
that
flight
Book
that
flight

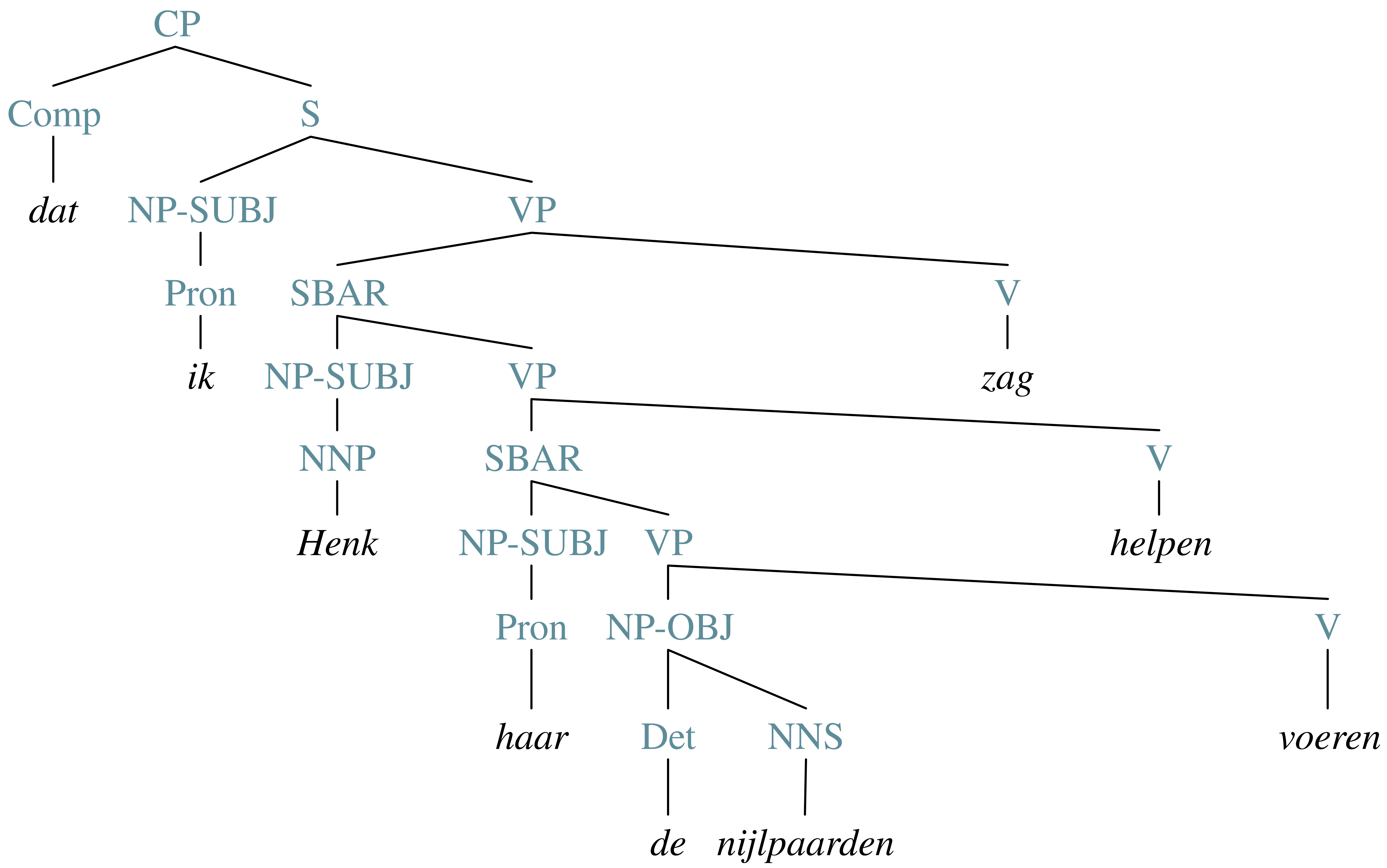
Pros and Cons of Bottom-Up Search

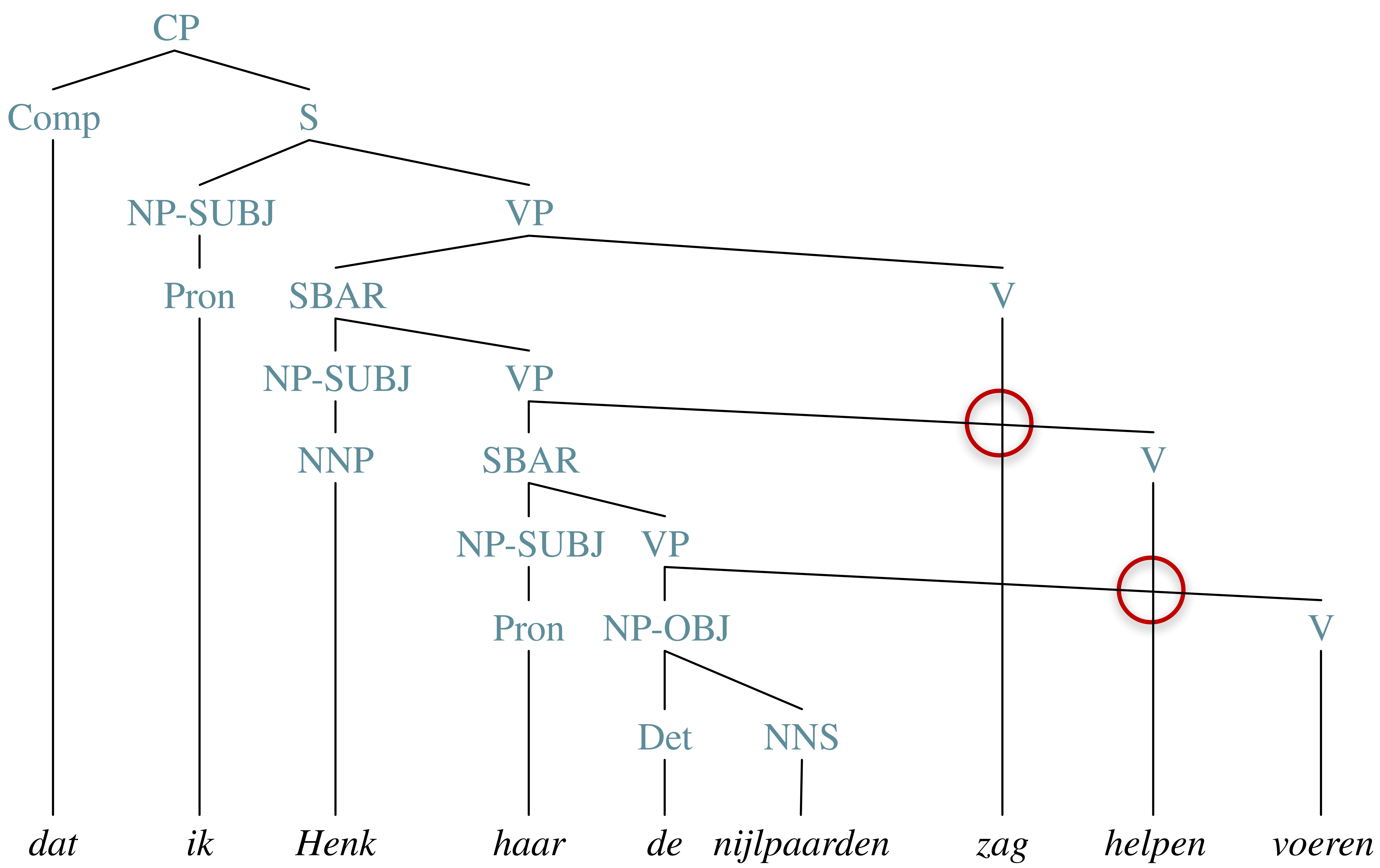
- Pros:
  - Will not explore trees that don’t match input
  - Recursive rules less problematic
  - Useful for incremental/fragment parsing
- Cons:
  - Explore subtrees that will not fit full input

Cross-Serial Dependencies, Revisited

$L' = a^m b^n c^m d^n$

ik_1	$Henk_2$	$haar_3$	$nijlpaarden_3$	zag_1	$helpen_2$	$voeren_3$
I_1	$Henk_2$	her_3	hippos	saw_1	$help_2$	$feed_3$


CP
Comp
S
dat
NP-SUBJ
VP
Pron
SBAR
V
ik
NP-SUBJ
VP
zag
NNP
SBAR
V
Henk
NP-SUBJ
VP
helpen
Pron
NP-OBJ
V
haar
Det
NNS
voeren
de
nijlpaarden


CP
Comp
S
NP-SUBJ
VP
Pron
SBAR
V
NP-SUBJ
VP
NNP
SBAR
V
NP-SUBJ
VP
Pron
NP-OBJ
V
Det
NNS
dat
ik
Henk
haar
de
nijlpaarden
zag
helpen
voeren

Next Time

- Beginning to implement CFG parsing algorithms
- Conversion to Chomsky Normal Form
  - Required for CKY algorithm
- HW2 out