Natural Language Processing

Lecture 13—2/26/2015

Martha Palmer

Today

- Start on Parsing
  - Top-down vs. Bottom-up

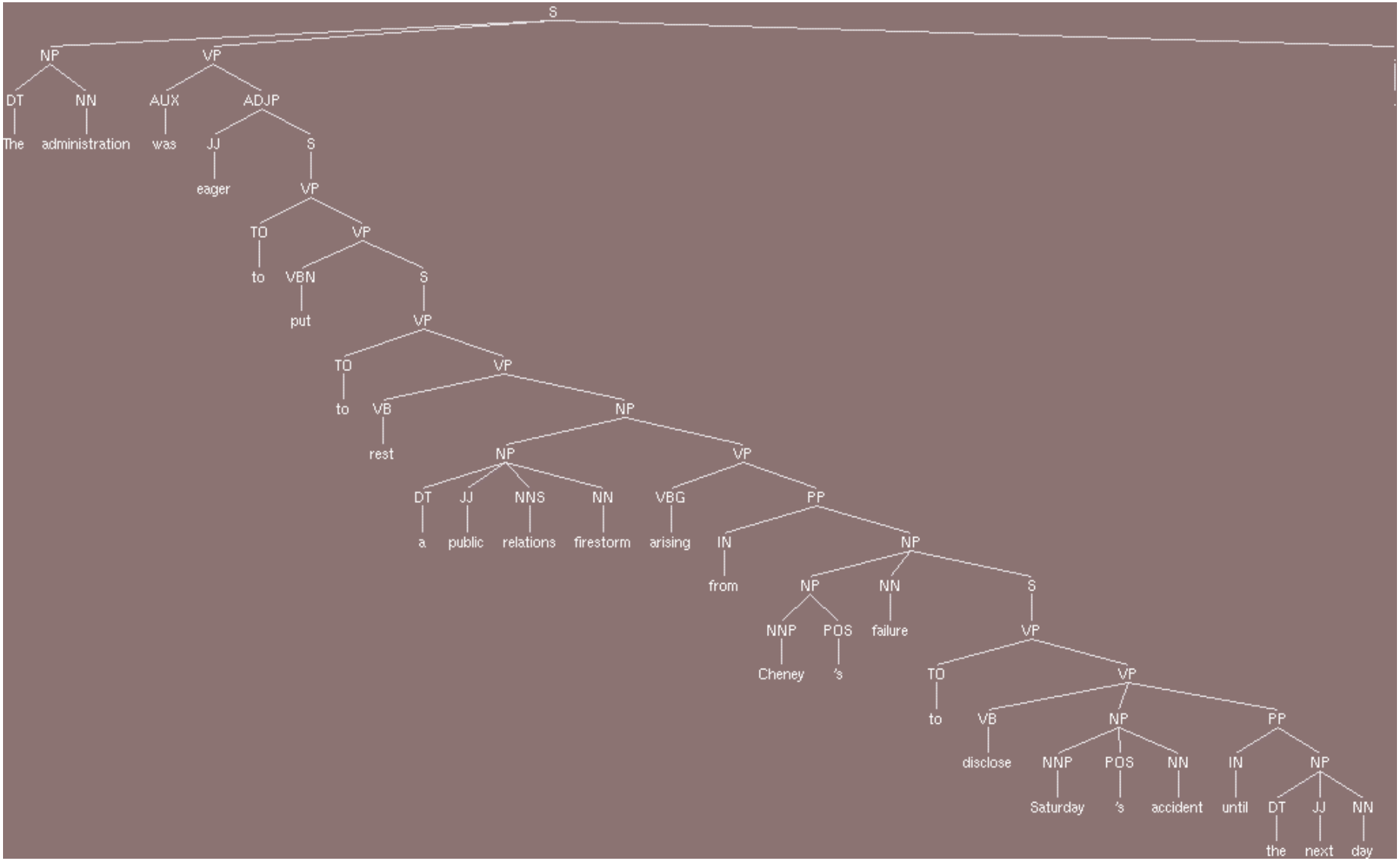
Summary

- Context-free grammars can be used to model various facts about the syntax of a language.
- When paired with parsers, such grammars consititute a critical component in many applications.
- Constituency is a key phenomena easily captured with CFG rules.
  - But agreement and subcategorization do pose significant problems
- Treebanks pair sentences in a corpus with their corresponding trees.

Parsing

- Parsing with CFGs refers to the task of assigning proper trees to input strings
- Proper here means a tree that covers all and only the elements of the input and has an S at the top
- It doesn't actually mean that the system can select the correct tree from among all the possible trees

Automatic Syntactic Parse



For Now

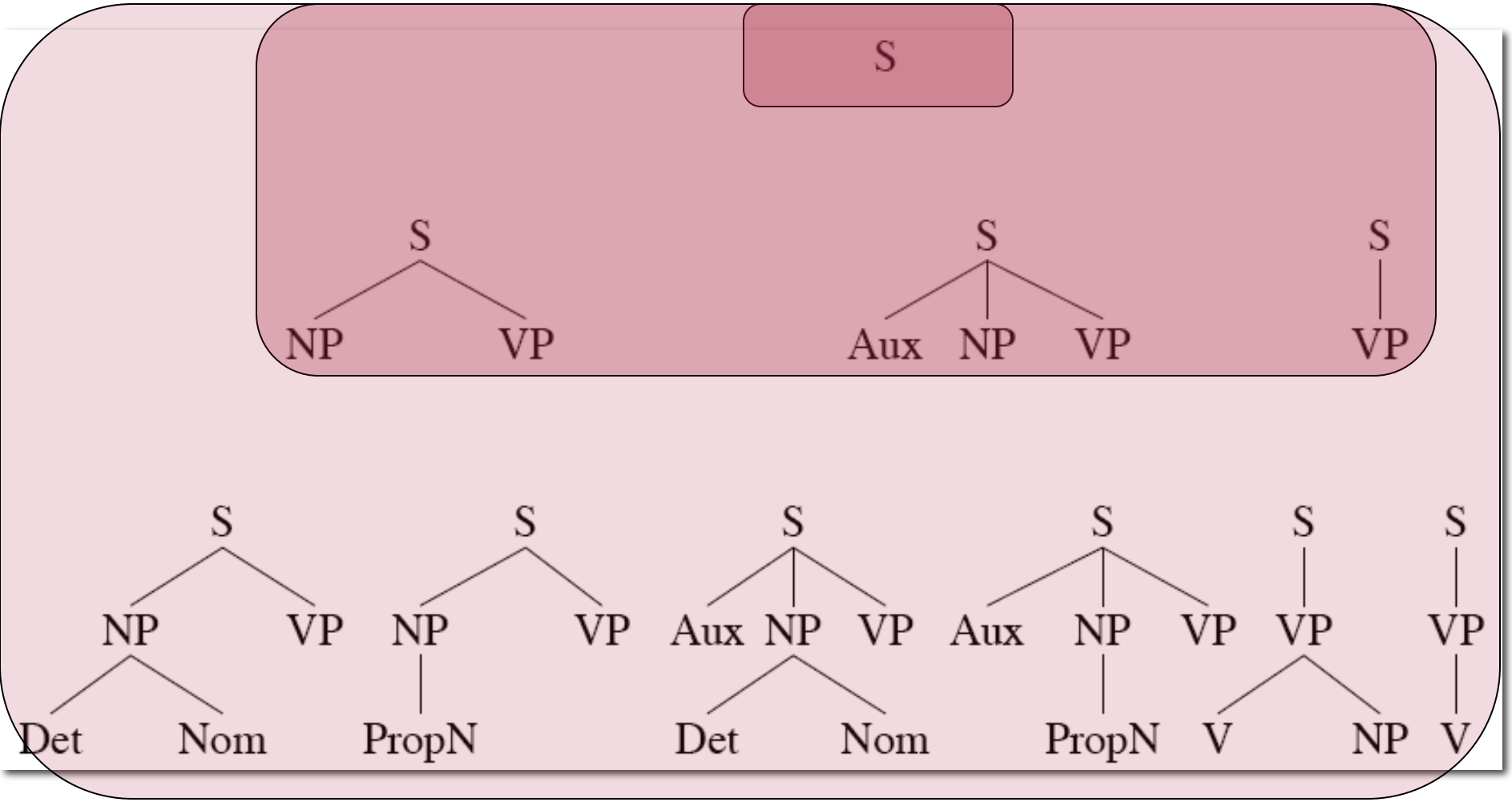
- Assume…
  - You have all the words already in some buffer
  - The input is not POS tagged prior to parsing
  - We won't worry about morphological analysis
  - All the words are known

  - These are all problematic in various ways, and would have to be addressed in real applications.

Top-Down Search

- Since we're trying to find trees rooted with an *S* (Sentences), why not start with the rules that give us an *S*.
- Then we can work our way down from there to the words.

Top Down Space



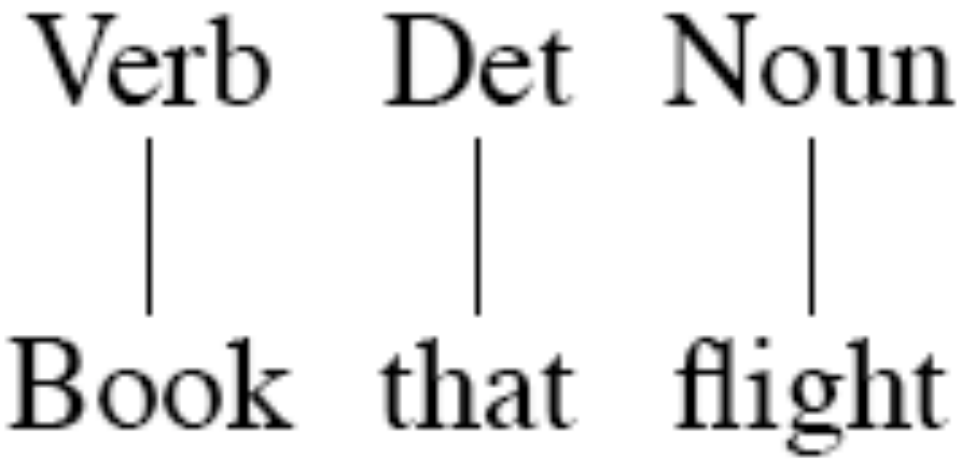
Bottom-Up Parsing

- Of course, we also want trees that cover the input words. So we might also start with trees that link up with the words in the right way.
- Then work your way up from there to larger and larger trees.

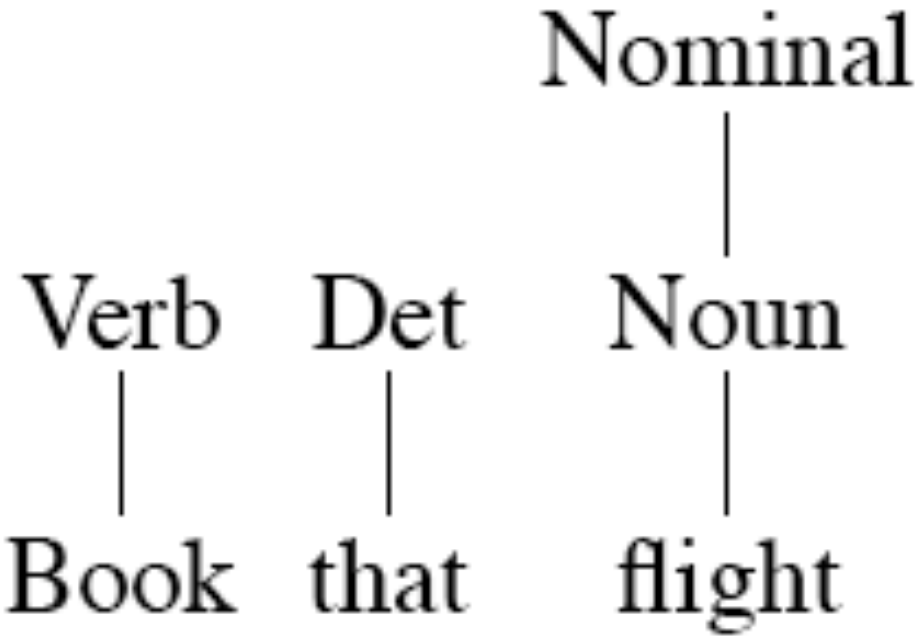
Bottom-Up Search

Book that flight

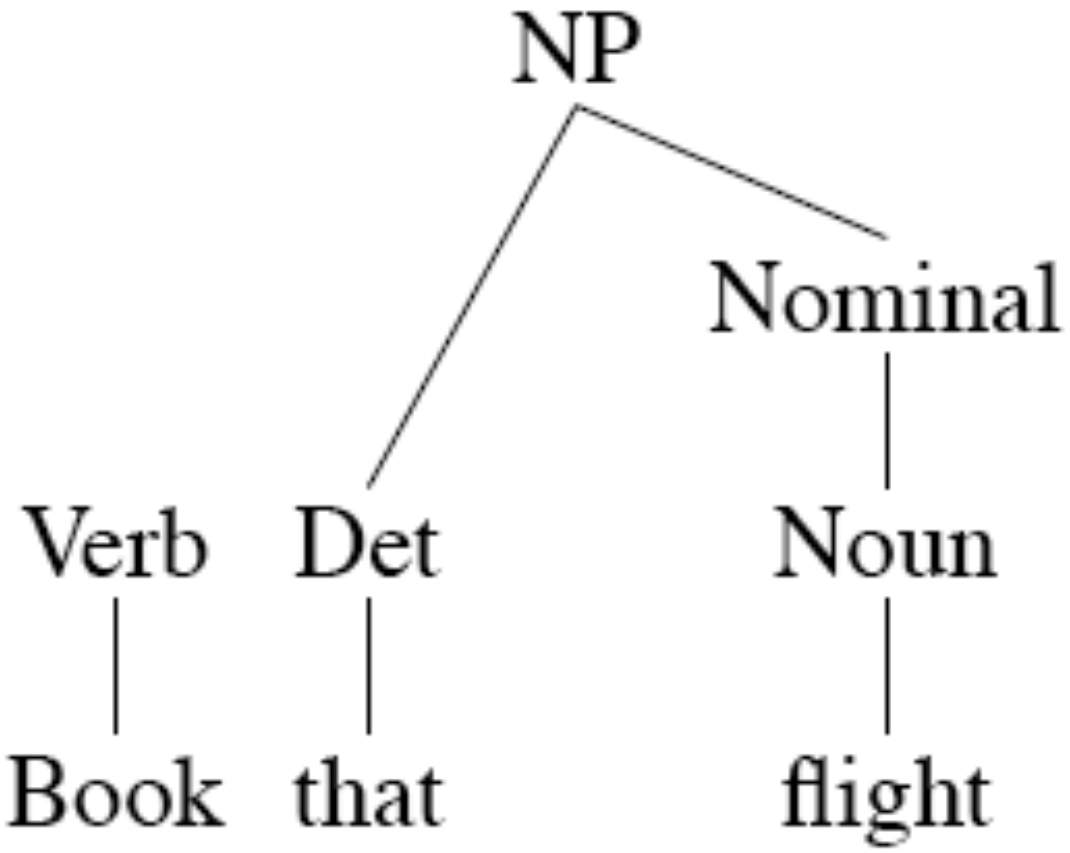
Bottom-Up Search



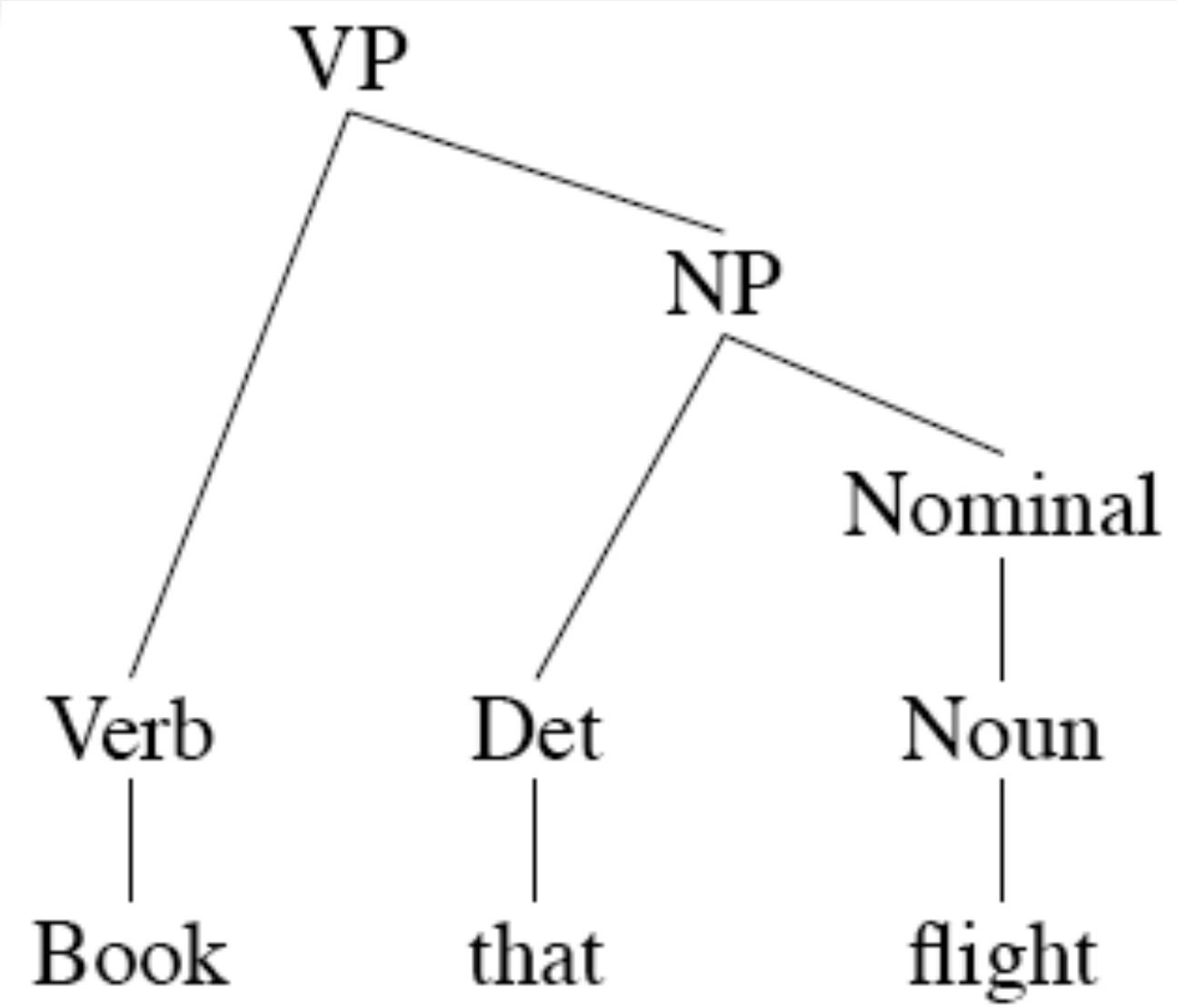
Bottom-Up Search



Bottom-Up Search



Bottom-Up Search



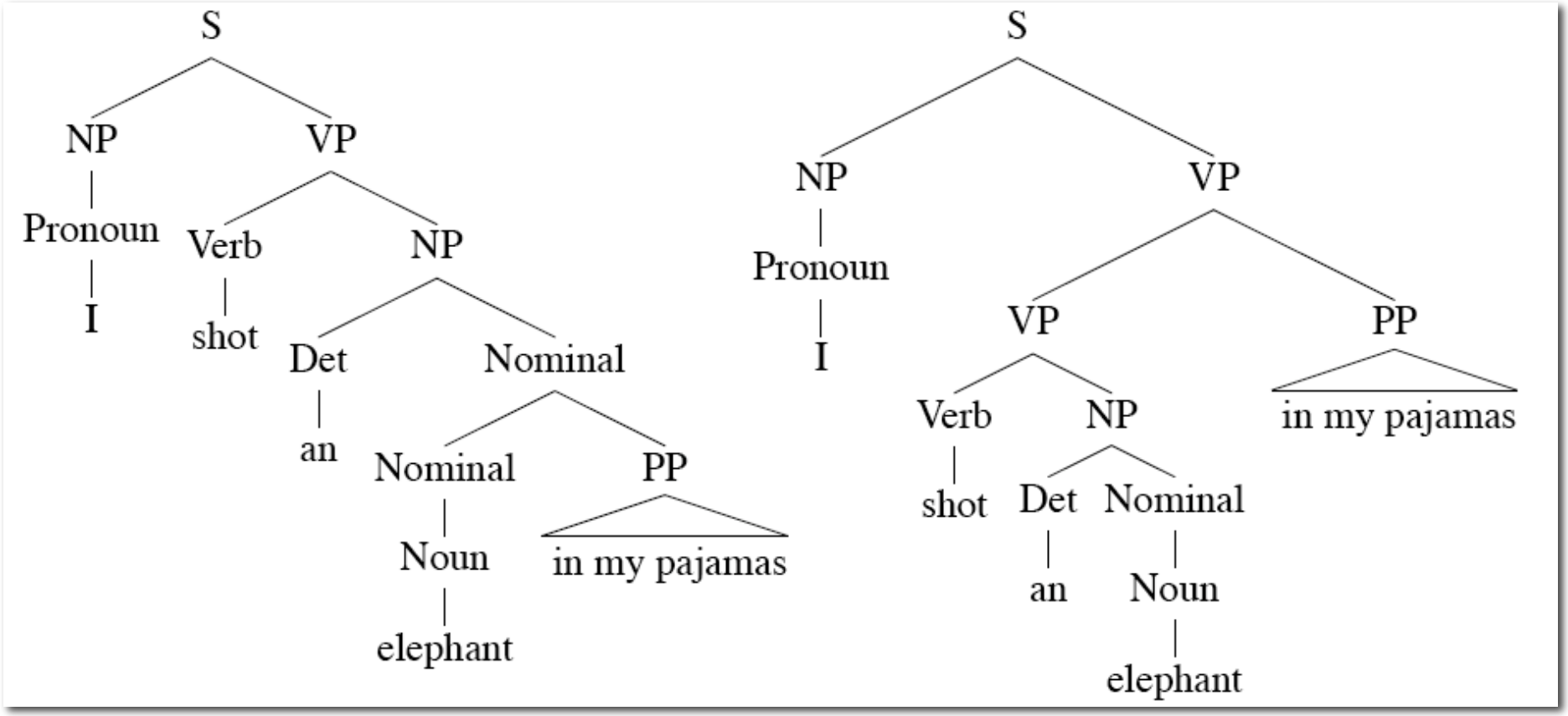
Control

- Of course, in both cases we left out how to keep track of the search space and how to make choices
  - Which node to try to expand next
  - Which grammar rule to use to expand a node
- One approach is called backtracking.
  - Make a choice, if it works out then fine
  - If not then back up and make a different choice
    - Same as with ND-Recognize

Problems

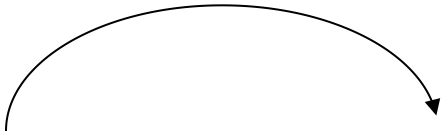
- Even with the best filtering, backtracking methods are doomed because of two inter-related problems
  - Ambiguity and search control (choice)
  - Shared subproblems

Ambiguity


S
NP
VP
Pronoun
Verb
NP
I
shot
Det
Nominal
an
Nominal
PP
Noun
in my pajamas
elephant
S
NP
VP
Pronoun
VP
PP
I
Verb
NP
in my pajamas
shot
Det
Nominal
an
Noun
elephant

Structural Ambiguities

- Its very important to separate PP's that are part of the verb subcategorization frame from PP's that modify the entire event.



- *The man saw the woman on the hill with the telescope.*

Woman has telescope

- *The man saw the woman on the hill with the telescope.*

Man has telescope

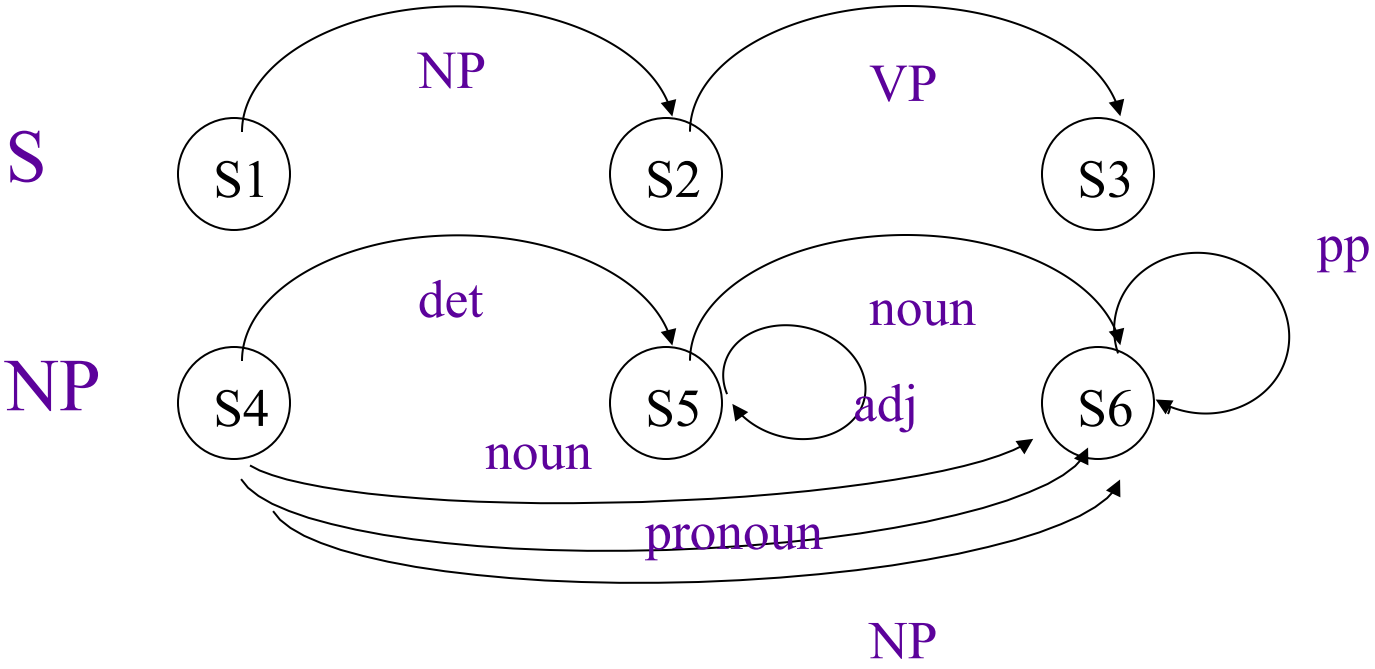
Shared Sub-Problems

- No matter what kind of search (top-down or bottom-up or mixed) that we choose...
  - We can't afford to redo work we've already done.
  - Without some help naïve backtracking will lead to such duplicated work.

Sample L1 Grammar

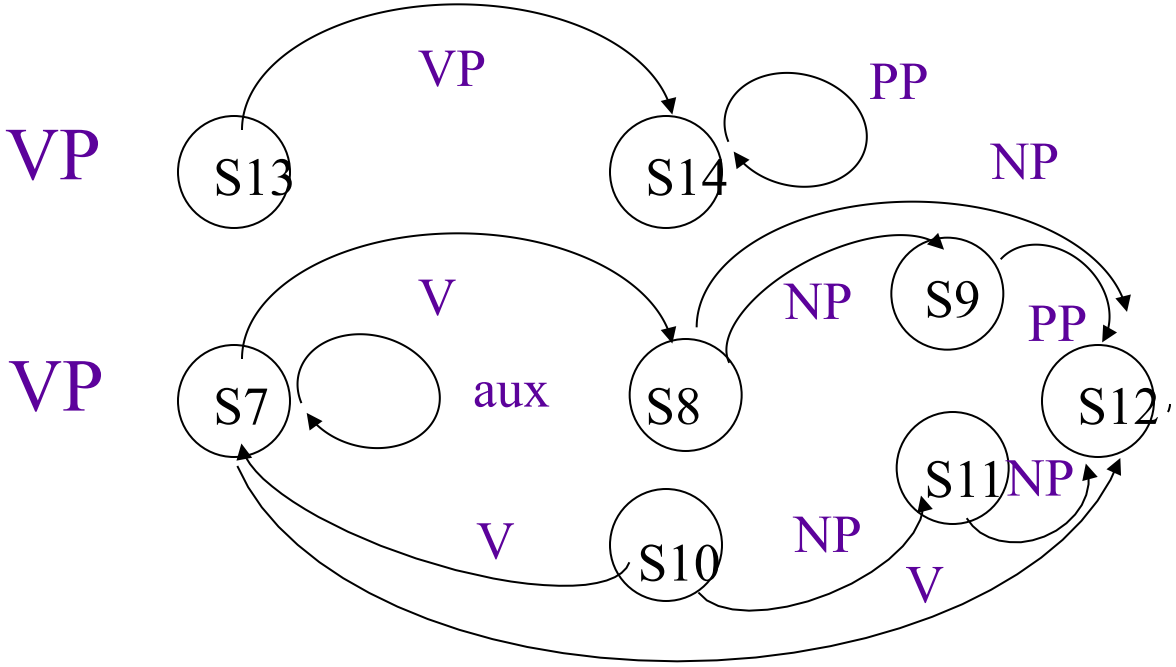
Grammar	Lexicon
$S \rightarrow NP\ VP$	$Det \rightarrow that \mid this \mid a$
$S \rightarrow Aux\ NP\ VP$	$Noun \rightarrow book \mid flight \mid meal \mid money$
$S \rightarrow VP$	$Verb \rightarrow book \mid include \mid prefer$
$NP \rightarrow Pronoun$	$Pronoun \rightarrow I \mid she \mid me$
$NP \rightarrow Proper\text{-}Noun$	$Proper\text{-}Noun \rightarrow Houston \mid NWA$
$NP \rightarrow Det\ Nominal$	$Aux \rightarrow does$
$Nominal \rightarrow Noun$	$Preposition \rightarrow from \mid to \mid on \mid near \mid through$
$Nominal \rightarrow Nominal\ Noun$	
$Nominal \rightarrow Nominal\ PP$	
$VP \rightarrow Verb$	
$VP \rightarrow Verb\ NP$	
$VP \rightarrow Verb\ NP\ PP$	
$VP \rightarrow Verb\ PP$	
$VP \rightarrow VP\ PP$	
$PP \rightarrow Preposition\ NP$	

State space representations: Recursive transition nets



- s :- np,vp.
- np:- pronoun; noun; det,adj, noun; np,pp.

State space representations: Recursive transition nets, cont.

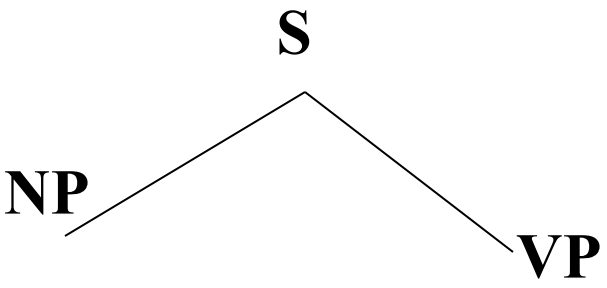


- VP:- VP, PP.
- VP:- V; V,NP; V,NP,NP; V,NP,PP.

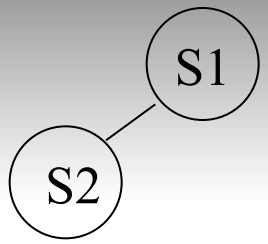
S1

Parses

The cat sat on the mat



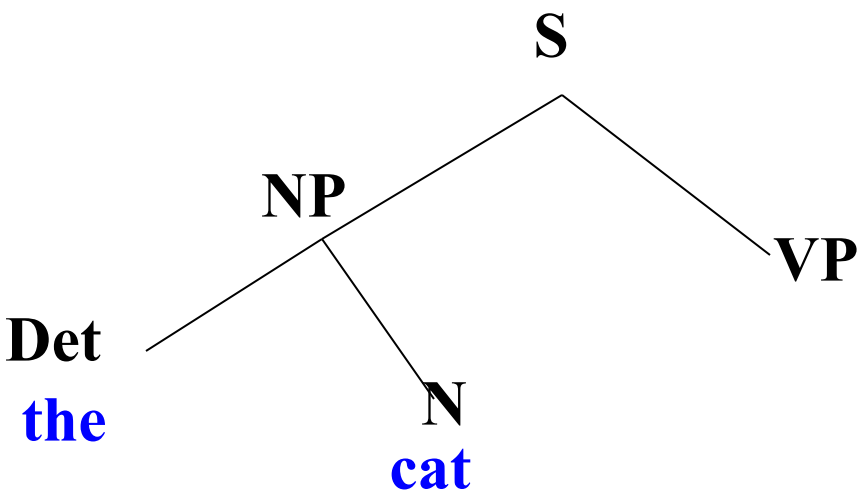
S1: S → NP, VP

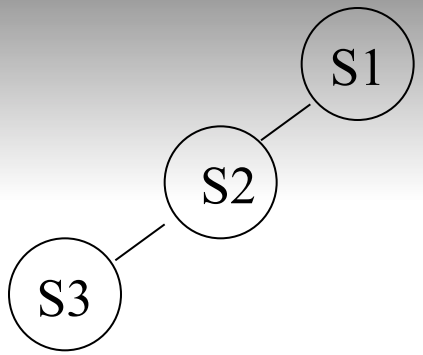


Parses

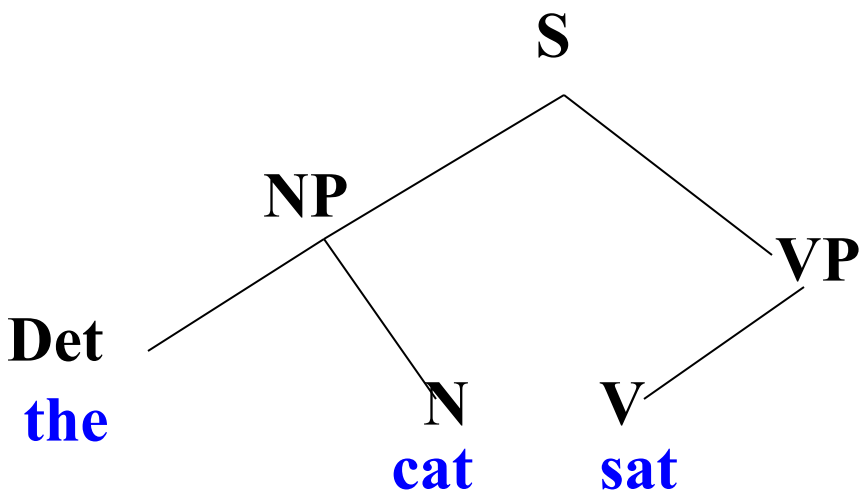
The cat sat on the mat

S1: S → NP, VP
S2: NP → Det, N





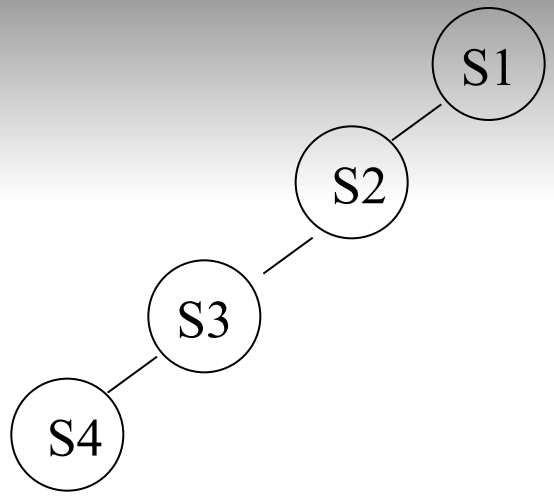
Parses



The cat sat on the mat

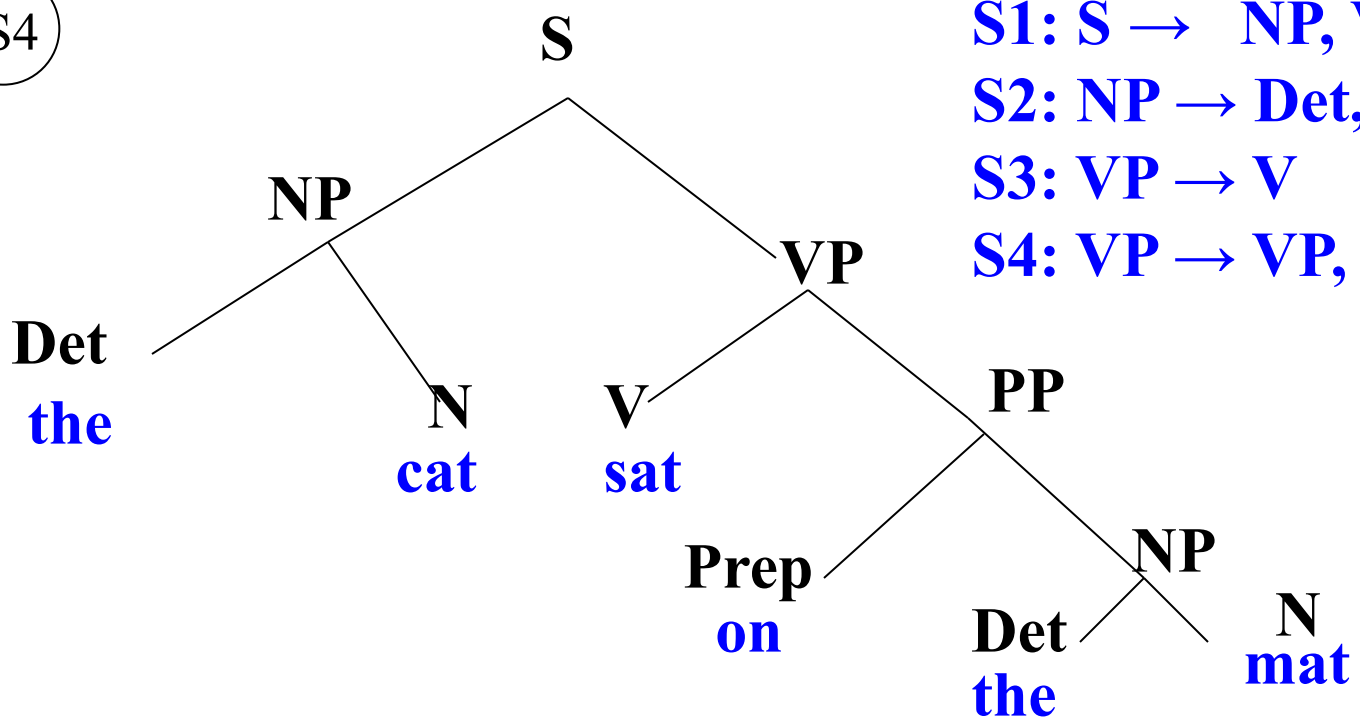
S1: S → NP, VP
S2: NP → Det, N
S3: VP → V

Parses

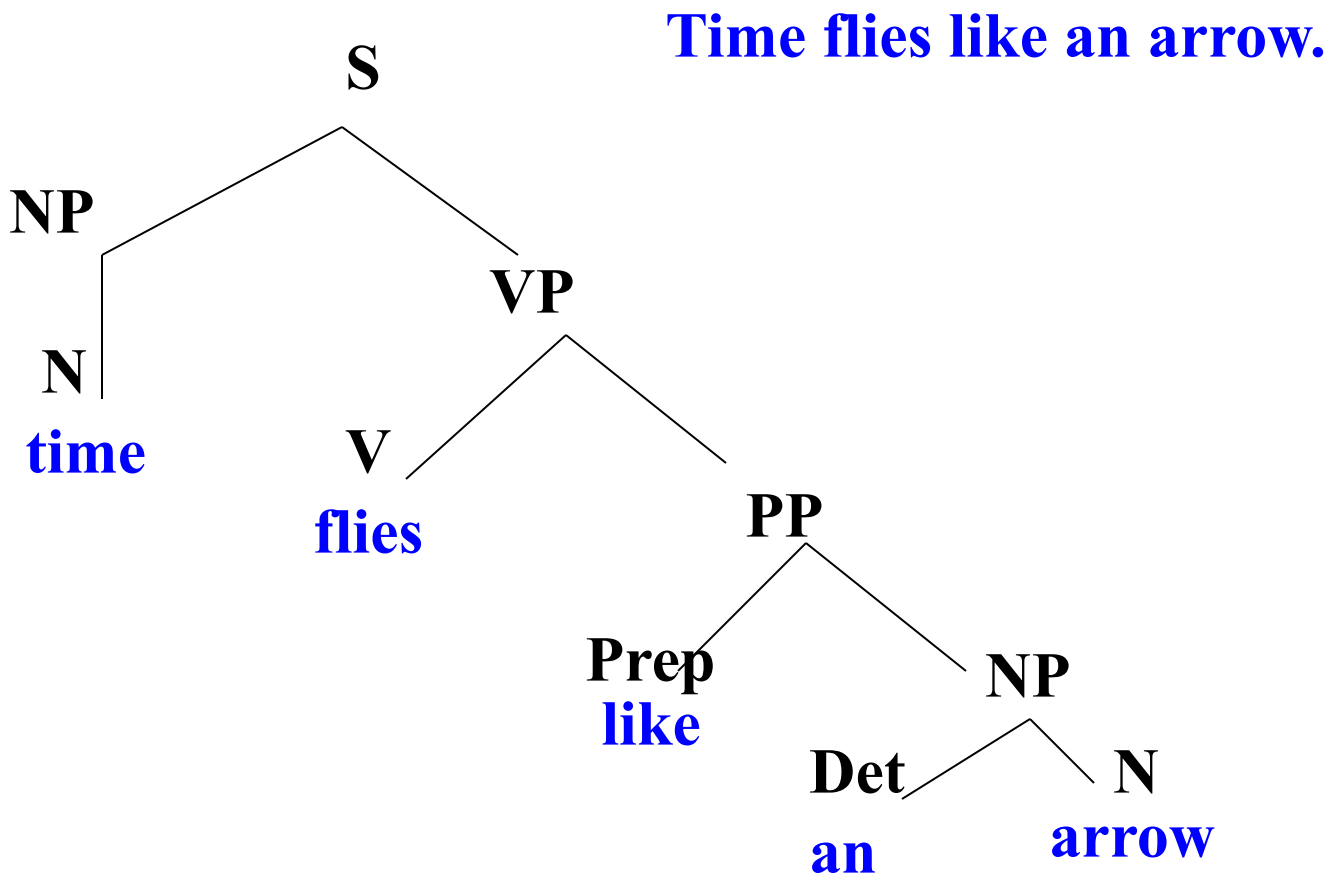


The cat sat on the mat

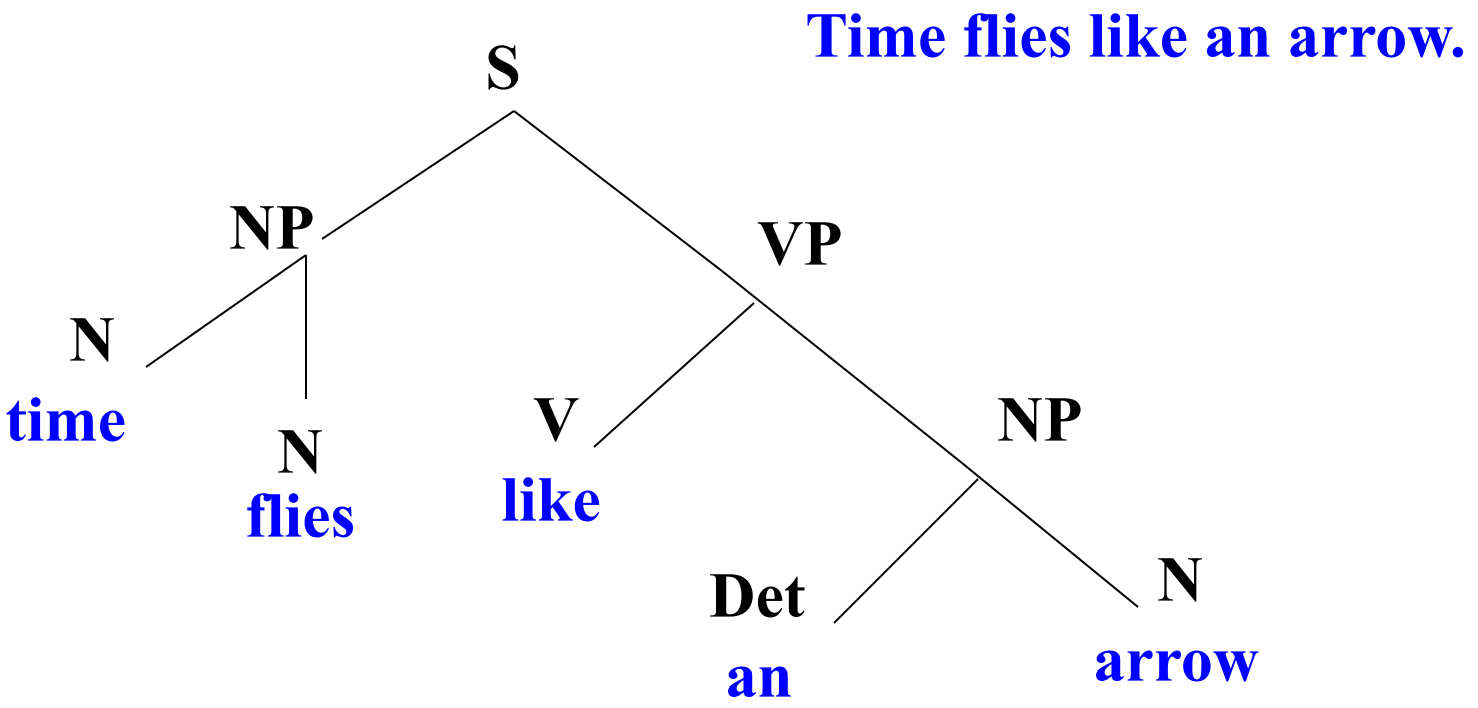
S1: S → NP, VP
S2: NP → Det, N
S3: VP → V
S4: VP → VP, PP



Multiple parses for a single sentence



Multiple Parses for a single sentence



Lexicon

noun(cat).

noun(mat).

det(the).

det(a).

verb(sat).

prep(on).

noun(flies).

noun(time).

noun(arrow).

det(an).

verb(flies).

verb(time).

prep(like).

Lexicon with Roots

noun(cat,cat).

noun(mat,mat).

det(the,the)

det(a,a).

verb(sat,sit).

prep(on,on).

noun(flies,fly).

noun(time,time).

noun(arrow,arrow).

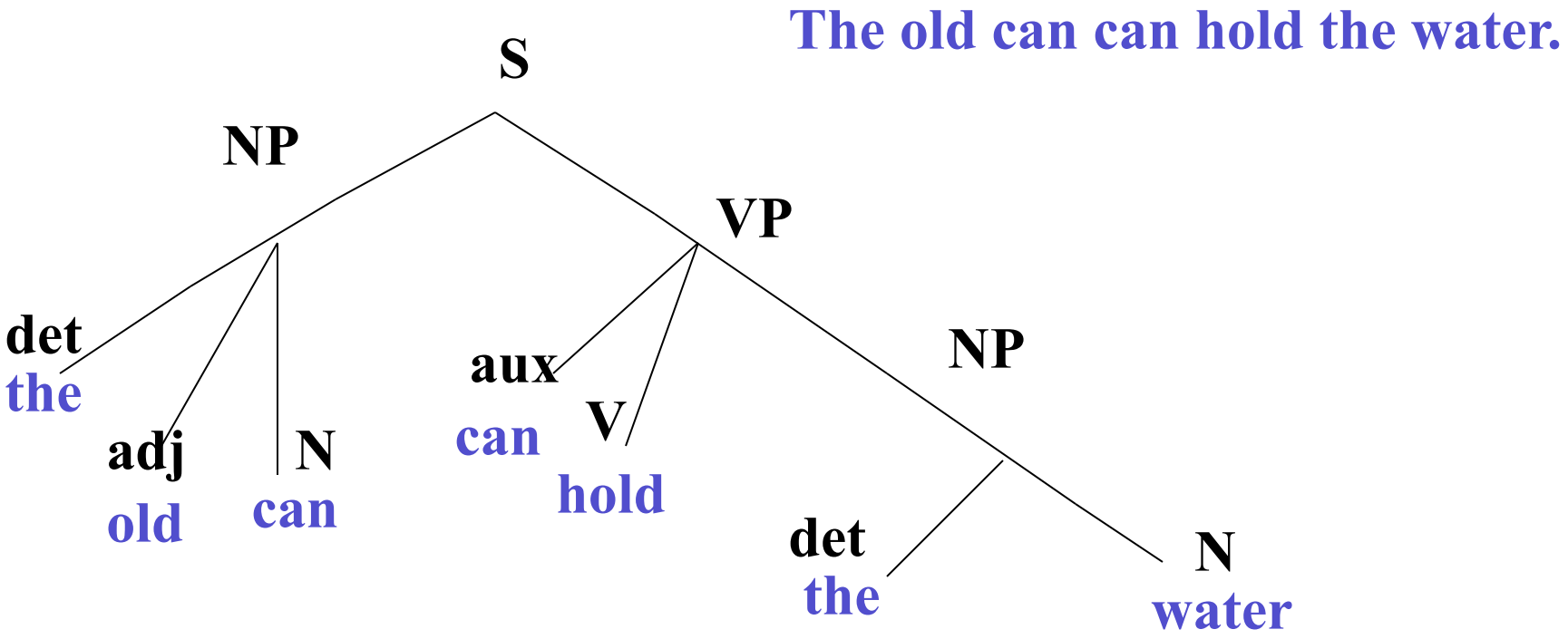
det(an,an).

verb(flies,fly).

verb(time,time).

prep(like,like).

Parses


The old can can hold the water.
S
NP
VP
det
the
adj
old
N
can
aux
can
V
hold
NP
det
the
N
water

Structural ambiguities

- *That factory can can tuna.*
- *That factory cans cans of tuna and salmon.*

Lexicon

The old can can hold the water.

Noun(can,can)

Noun(cans,can)

Noun(water,water)

Noun(hold,hold)

Noun(holds,hold)

Det(the,the)

Verb(hold,hold)

Verb(holds,hold)

Verb(can, can)

Aux(can,can)

Adj(old,old)

Noun(old, old)

Simple Context Free Grammar in BNF

S → NP VP

NP → Pronoun
| Noun
| Det Adj Noun
|NP PP

PP → Prep NP

V → Verb
| Aux Verb

VP → V
| V NP
| V NP NP
| V NP PP
| VP PP

Top-down parse in progress

[The, old, can, can, hold, the, water]

S → NP VP

NP → NP?

NP → Pronoun?

Pronoun? fail

NP → Noun?

Noun? fail

NP → Det Adj Noun?

Det? **the**

ADJ?**old**

Noun? **Can**

Succeed.

Succeed.

VP?

Top-down parse in progress

[can, hold, the, water]

VP → VP?

V → Verb?

Verb? **fail**

V → Aux Verb?

Aux? **can**

Verb? **hold**

succeed

succeed

fail **[the, water]**

Top-down parse in progress

[can, hold, the, water]

VP → V NP?

V → Verb?

Verb? **fail**

V → Aux Verb?

Aux? **can**

Verb? h**old**

NP → Pronoun?

Pronoun? fail

NP → Noun?

Noun? fail

NP → Det Adj Noun?

Det? **the**

Noun? w**ater**

SUCCEED

SUCCEED

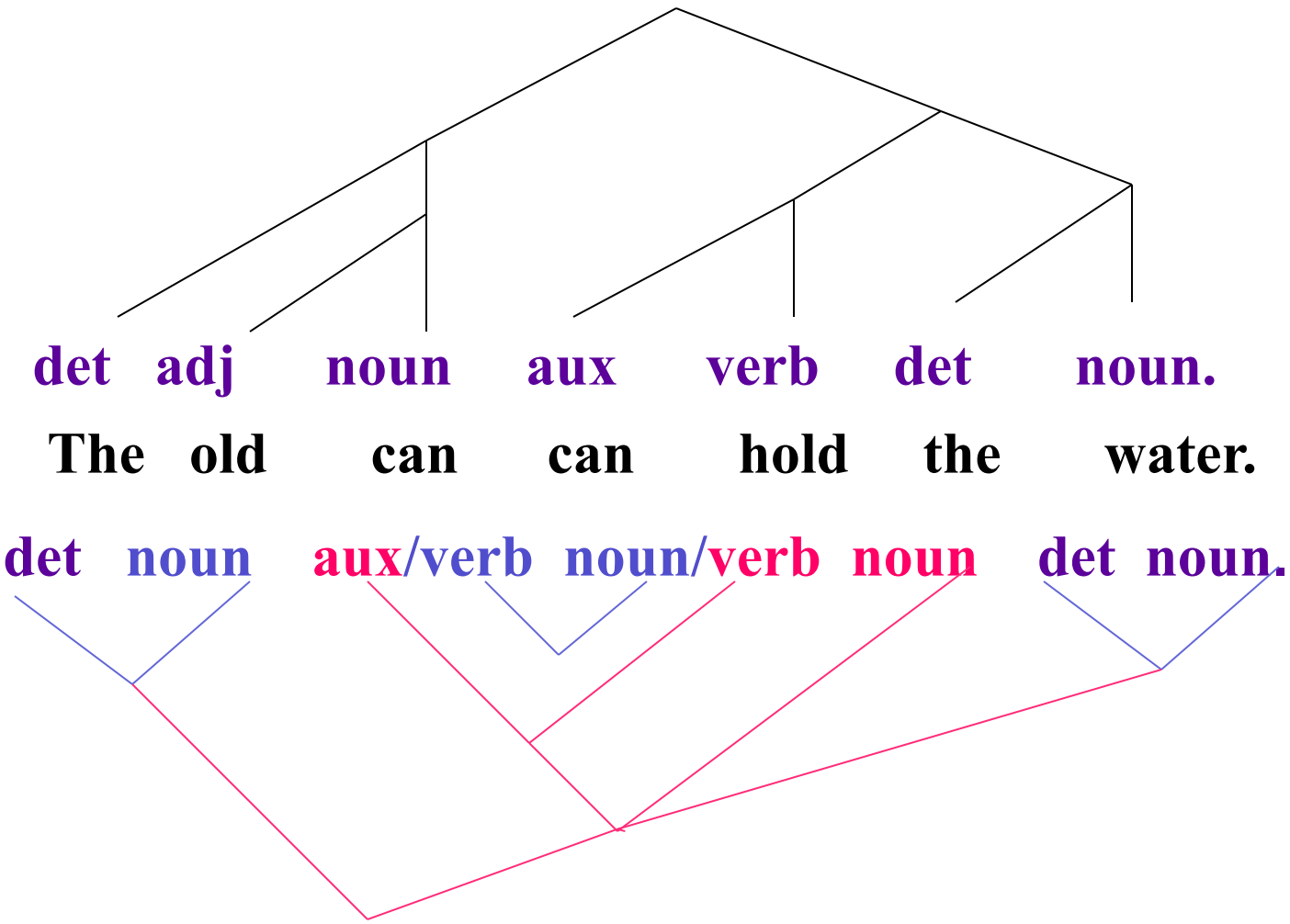
Top-down approach

- Start with goal of sentence
  - S → NP VP
  - S → Wh-word Aux NP VP
- Will try to find an NP 4 different ways before trying a parse where the verb comes first.
- What would be better?

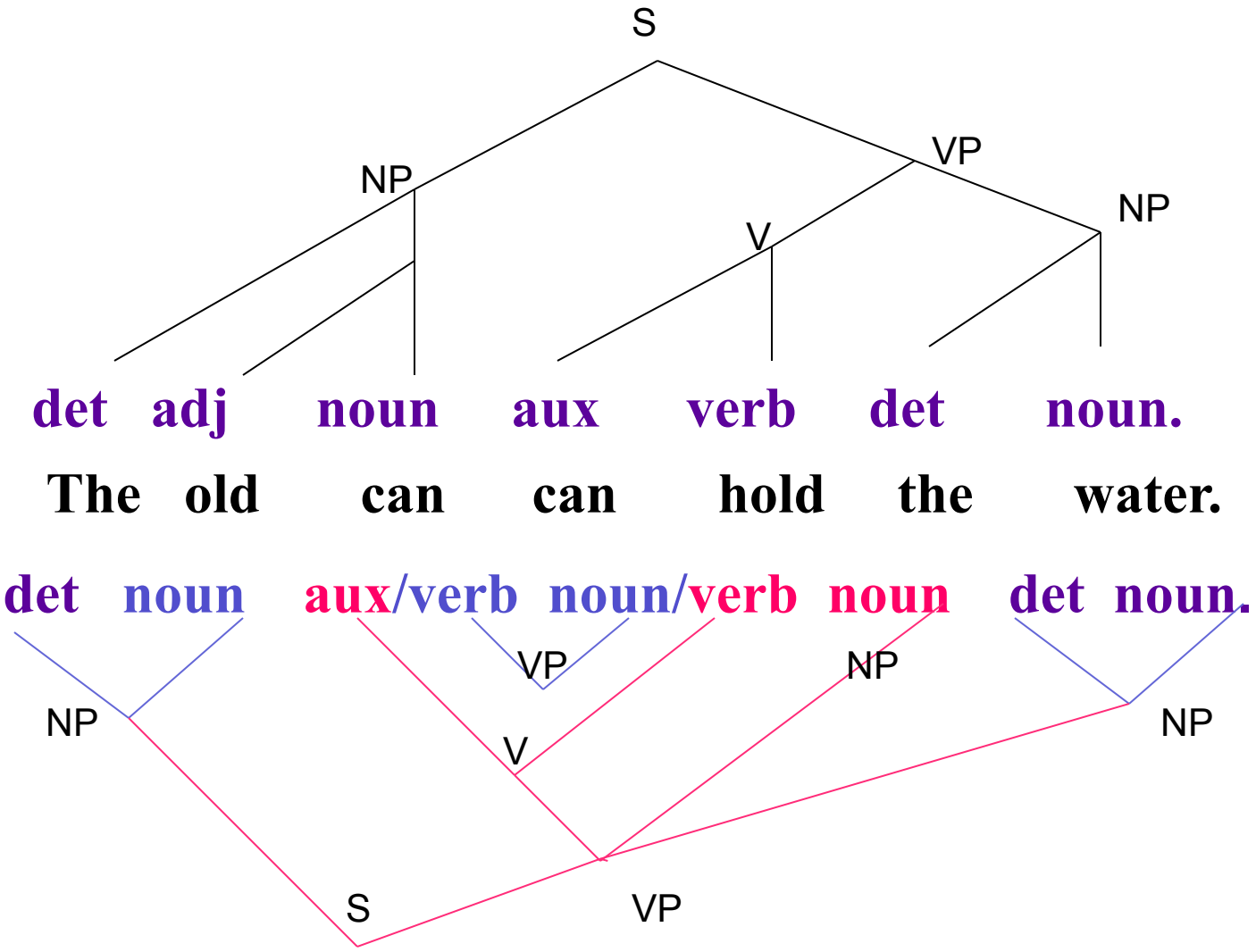
Bottom-up approach

- Start with words in sentence.
- What structures do they correspond to?
- Once a structure is built, kept on a CHART.

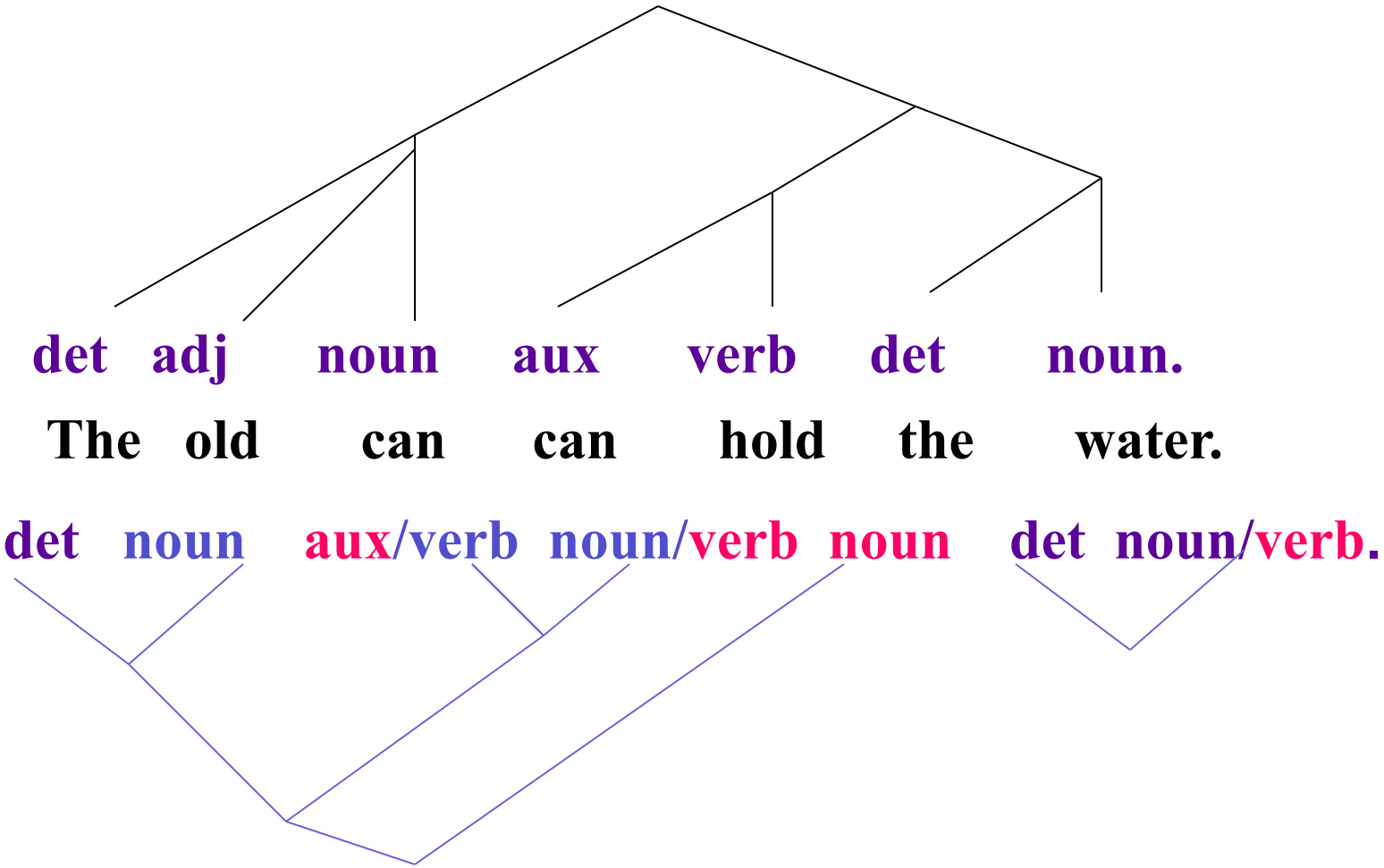
Bottom-up parse in progress



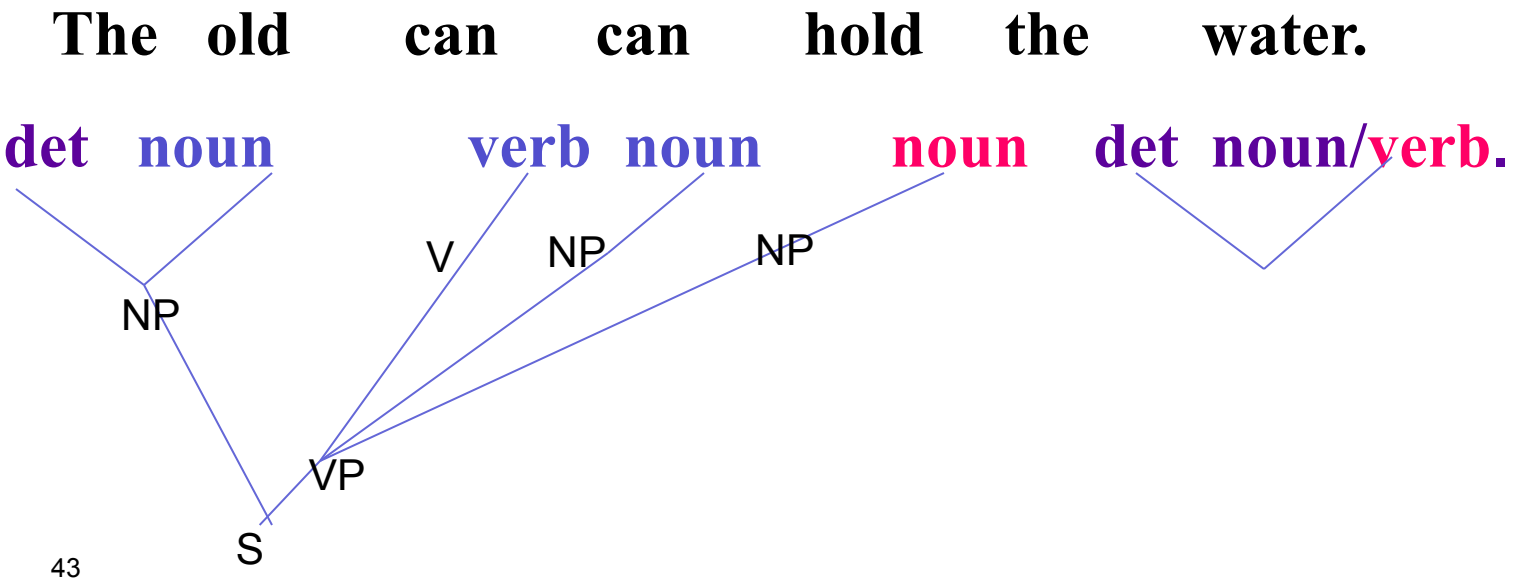
Bottom-up parse in progress



Bottom-up parse in progress – What is wrong w/ bottom parse?



Bottom-up parse, corrected



Headlines

- Police Begin Campaign To Run Down Jaywalkers
- Iraqi Head Seeks Arms
- Teacher Strikes Idle Kids
- Miners Refuse To Work After Death
- Juvenile Court To Try Shooting Defendant

Headlines

- Drunk Gets Nine Months in Violin Case
- Enraged Cow Injures Farmer with Ax
- Hospitals are Sued by 7 Foot Doctors
- Milk Drinkers Turn to Powder
- Lung Cancer in Women Mushrooms

Top-down vs. Bottom-up

- Helps with POS ambiguities – only consider relevant POS
- Rebuilds the same structure repeatedly
- Spends a lot of time on impossible parses (trees that are not consistent with any of the words)

- Has to consider every POS
- Builds each structure once
- Spends a lot of time on useless structures (trees that make no sense globally)

What would be better?