

Parsing with Context Free Grammar

Sudeshna Sarkar

16 AUG 2019

Parsing

- Parsing is the process of taking a string and a grammar and returning parse tree(s) for that string

“The old dog the footsteps of the young.”

$S \rightarrow NP VP$	$VP \rightarrow V$
$S \rightarrow Aux NP VP$	$VP \rightarrow V PP$
$S \rightarrow VP$	$PP \rightarrow Prep NP$
$NP \rightarrow Det Nom$	$N \rightarrow \text{old} \mid \text{dog} \mid \text{footsteps} \mid \text{young}$
$NP \rightarrow PropN$	$V \rightarrow \text{dog} \mid \text{eat} \mid \text{sleep} \mid \text{bark} \mid \text{meow}$
$Nom \rightarrow Adj N$	$Aux \rightarrow \text{does} \mid \text{can}$
$Nom \rightarrow N$	$Prep \rightarrow \text{from} \mid \text{to} \mid \text{on} \mid \text{of}$
$Nom \rightarrow N Nom$	$PropN \rightarrow \text{Fido} \mid \text{Felix}$
$Nom \rightarrow Nom PP$	$Det \rightarrow \text{that} \mid \text{this} \mid \text{a} \mid \text{the}$
$VP \rightarrow V NP$	$Adj \rightarrow \text{old} \mid \text{happy} \mid \text{young}$

Parsing

- Parsing with CFGs refers to the task of assigning proper trees to input strings
- Proper: a tree that covers **all and only the elements of the input** and **has an S at the top**

Syntactic Analysis (Parsing)

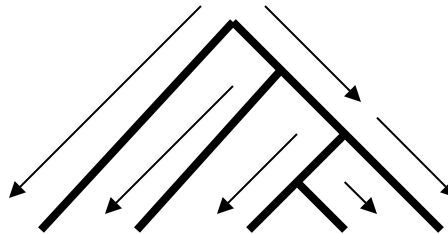
- Automatic methods of finding the syntactic structure for a sentence
 - Symbolic methods: a phrase grammar or another description of the structure of language is required. The chart parser.
 - Statistical methods: a text corpus with syntactic structures is needed (a **treebank**)

Search Framework

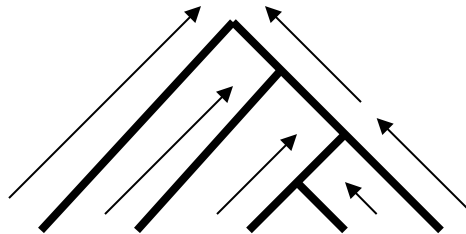
- Think about parsing as a form of search...
 - A search through the space of possible trees given an input sentence and grammar

How to parse

- **Top-down:** Start at the top of the tree with an S node, and work your way down to the words.



- **Bottom-up:** Look for small pieces that you know how to assemble, and work your way up to larger pieces.



Top-Down Search

- Builds from the root S node to the leaves
- Expectation-based
- Common top-down search strategy
 - Top-down, left-to-right, with backtracking
 - Try first rule s.t. LHS is S
 - Next expand all constituents on RHS
 - Iterate until all leaves are POS
 - Backtrack when candidate POS does not match POS of current word in input string

“The old dog the footsteps of the young.”

$S \rightarrow NP VP$	$VP \rightarrow V$
$S \rightarrow Aux NP VP$	$VP \rightarrow V PP$
$S \rightarrow VP$	$PP \rightarrow Prep NP$
$NP \rightarrow Det Nom$	$N \rightarrow \text{old} \mid \text{dog} \mid \text{footsteps} \mid \text{young}$
$NP \rightarrow PropN$	$V \rightarrow \text{dog} \mid \text{eat} \mid \text{sleep} \mid \text{bark} \mid \text{meow}$
$Nom \rightarrow Adj N$	$Aux \rightarrow \text{does} \mid \text{can}$
$Nom \rightarrow N$	$Prep \rightarrow \text{from} \mid \text{to} \mid \text{on} \mid \text{of}$
$Nom \rightarrow N Nom$	$PropN \rightarrow \text{Fido} \mid \text{Felix}$
$Nom \rightarrow Nom PP$	$Det \rightarrow \text{that} \mid \text{this} \mid \text{a} \mid \text{the}$
$VP \rightarrow V NP$	$Adj \rightarrow \text{old} \mid \text{happy} \mid \text{young}$

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.

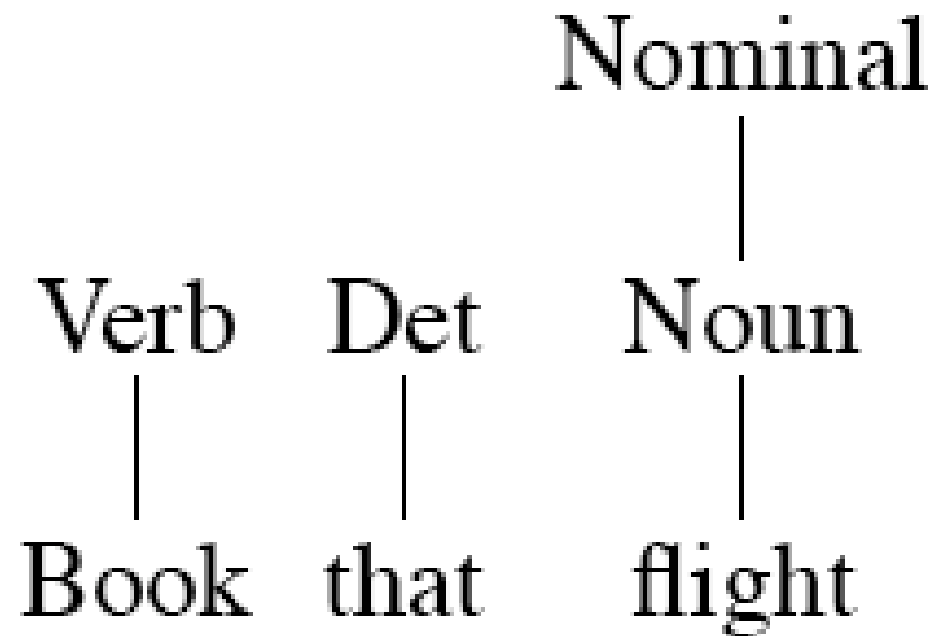
Bottom-Up Search

Book that flight

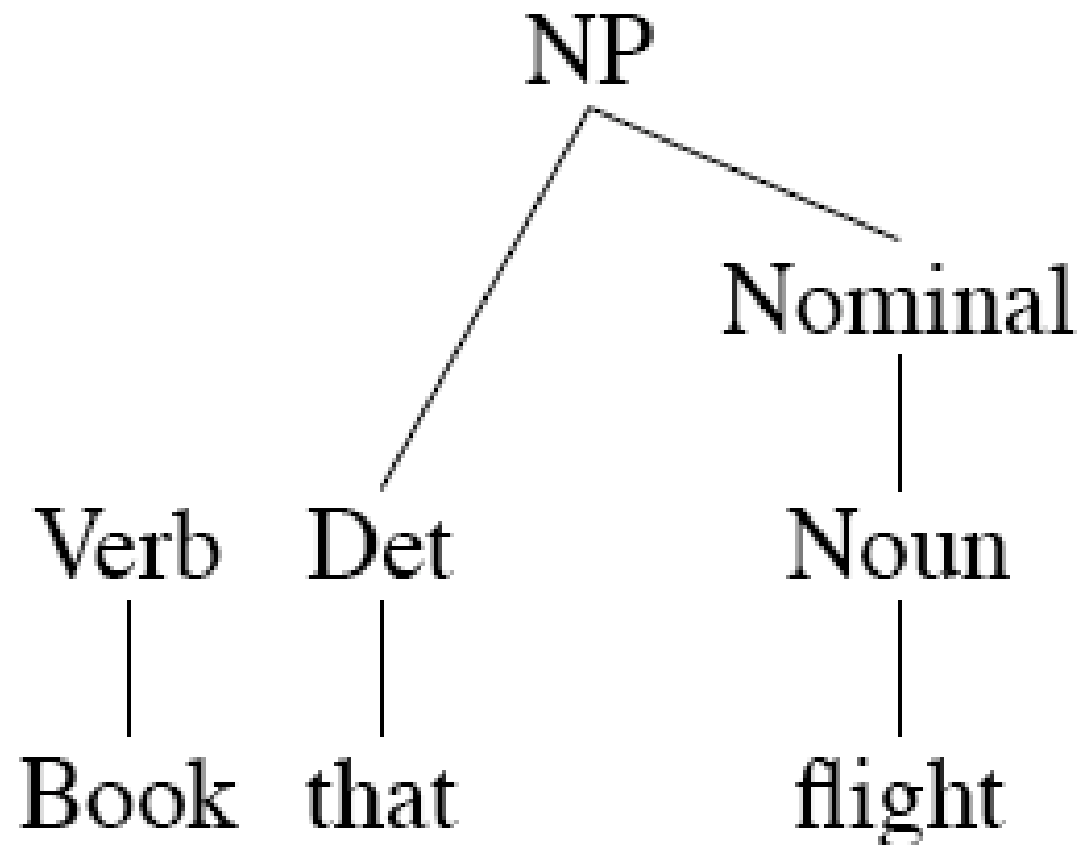
Bottom-Up Search

Verb	Det	Noun
Book	that	flight

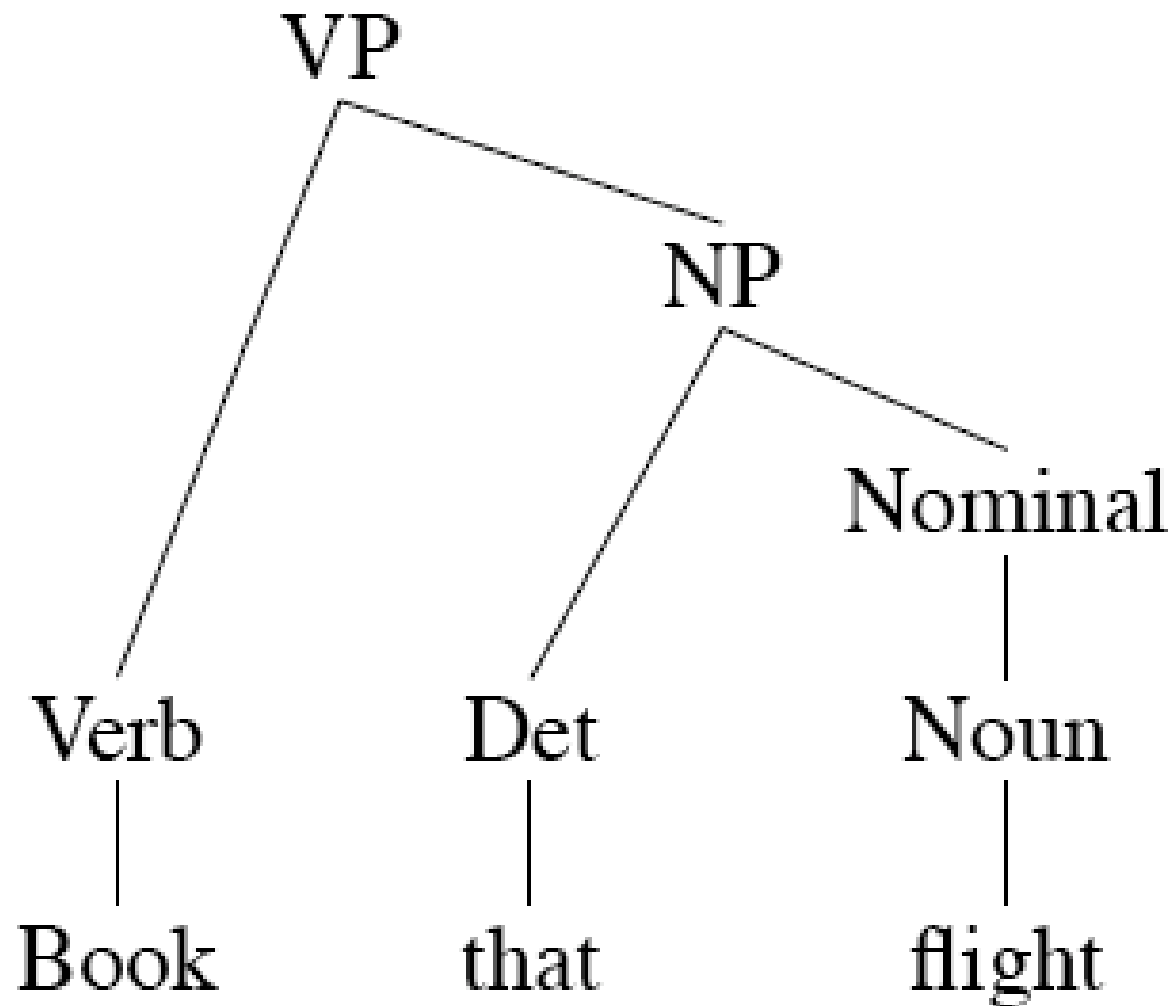
Bottom-Up Search



Bottom-Up Search



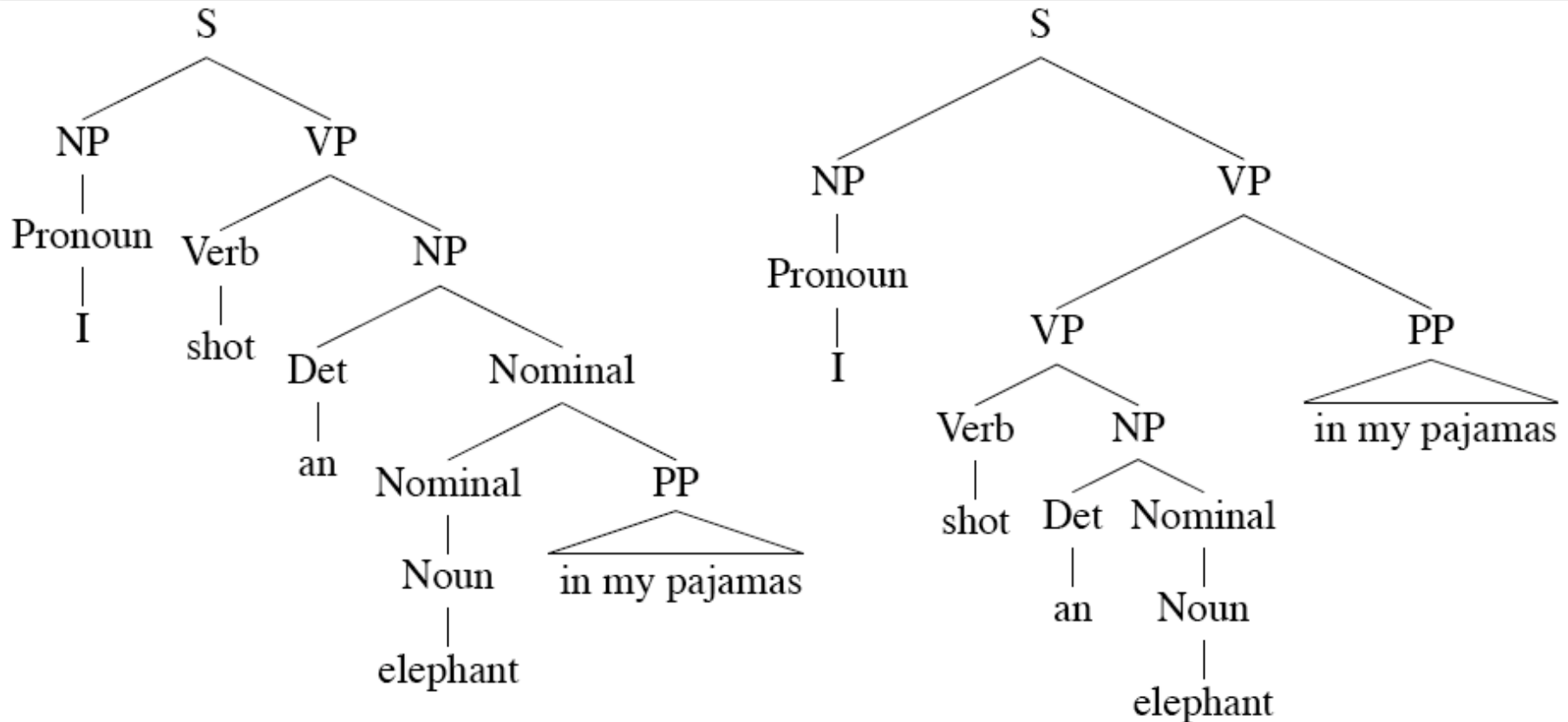
Bottom-Up Search



Issues

- Ambiguity
- Shared subproblems

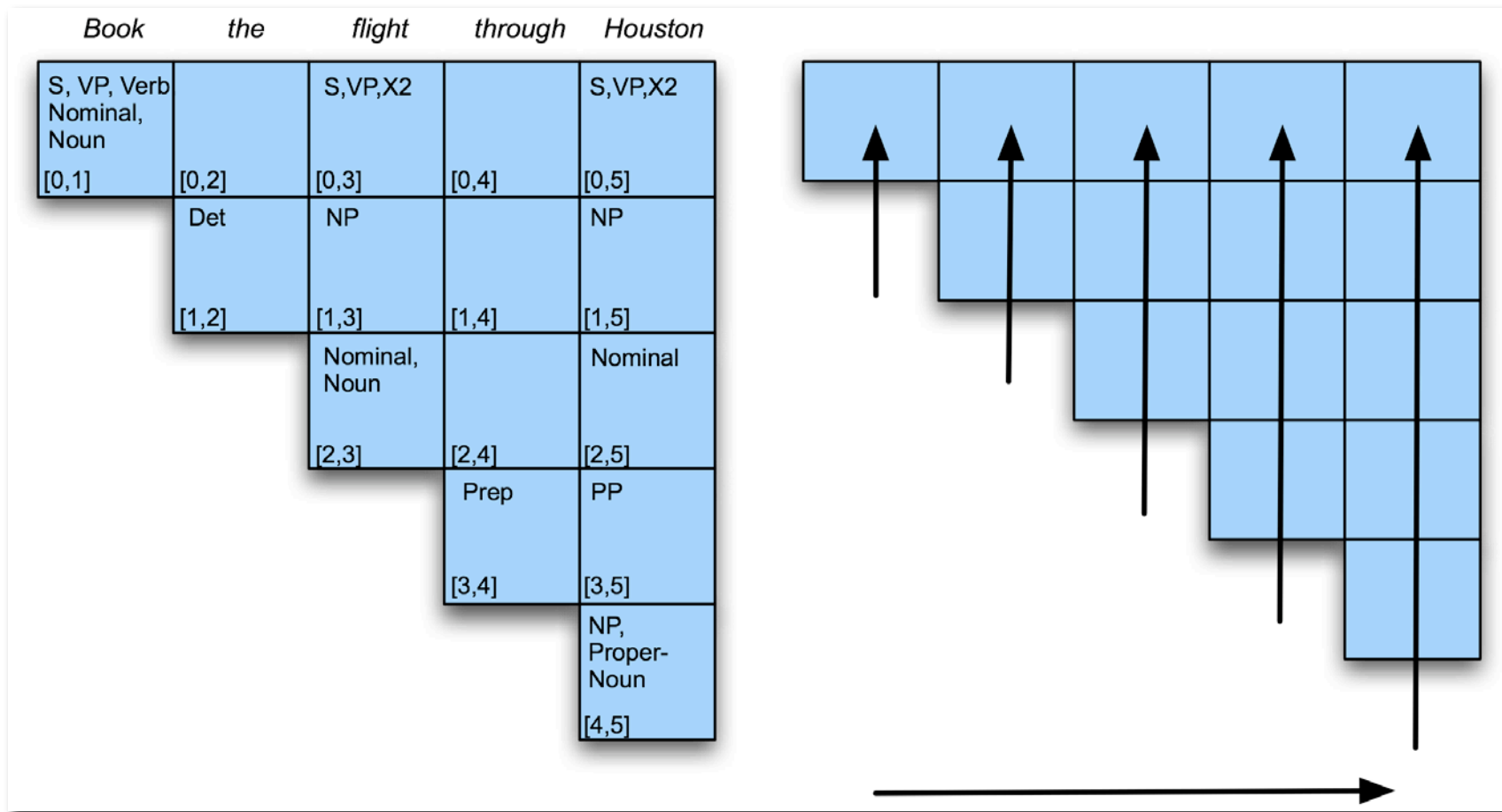
Ambiguity



Dynamic Programming

- DP search methods fill tables with partial results and thereby
 - Avoid doing avoidable repeated work
 - Solve exponential problems in polynomial time (ok, not really)
 - Efficiently store ambiguous structures with shared sub-parts.
- We'll cover one approach that corresponds to a bottom-up strategy
 - CKY

CKY Algorithm



CKY Algorithm

function CKY-PARSE(*words*, *grammar*) **returns** *table*

for $j \leftarrow$ **from** 1 **to** LENGTH(*words*) **do**

Looping over the columns

$table[j-1, j] \leftarrow \{A \mid A \rightarrow words[j] \in grammar\}$

Filling the bottom cell

for $i \leftarrow$ **from** $j-2$ **downto** 0 **do**

Filling row i in column j

for $k \leftarrow i+1$ **to** $j-1$ **do**

Looping over the possible split locations between i and j .

$table[i, j] \leftarrow table[i, j] \cup$

Check the grammar for rules that link the constituents in $[i, k]$ with those in $[k, j]$. For each rule found store the LHS of the rule in cell $[i, j]$.

$\{A \mid A \rightarrow BC \in grammar,$
 $B \in table[i, k],$
 $C \in table[k, j]\}$

Treebank

- A syntactically annotated corpus where every sentence is paired with a corresponding tree.
- The Penn Treebank project
 - treebanks from the Brown, Switchboard, ATIS, and Wall Street Journal corpora of English
 - treebanks in Arabic and Chinese.
- Others
 - the Prague Dependency Treebank for Czech,
 - the Negra treebank for German, and
 - the Susanne treebank for English
 - Universal Dependencies Treebank

Penn Treebank

- Penn TreeBank is a widely used treebank.

Most well known part is the Wall Street Journal section of the Penn TreeBank.

- 1 M words from the 1987-1989 Wall Street Journal.

```
( (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)
                (S
                  (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) ))
                ( . . ) ))
```

```
((S
  (NP-SBJ (DT That)
    (JJ cold) (, ,)
    (JJ empty) (NN sky) )
  (VP (VBD was)
    (ADJP-PRD (JJ full)
      (PP (IN of)
        (NP (NN fire)
          (CC and)
          (NN light) ))))
  (. .) ))
```

(a)

```
((S
  (NP-SBJ The/DT flight/NN )
  (VP should/MD
    (VP arrive/VB
      (PP-TMP at/IN
        (NP eleven/CD a.m/RB ))
      (NP-TMP tomorrow/NN )))))
```

(b)

Figure 11.7 Parsed sentences from the LDC Treebank3 version of the Brown (a) and ATIS (b) corpora.

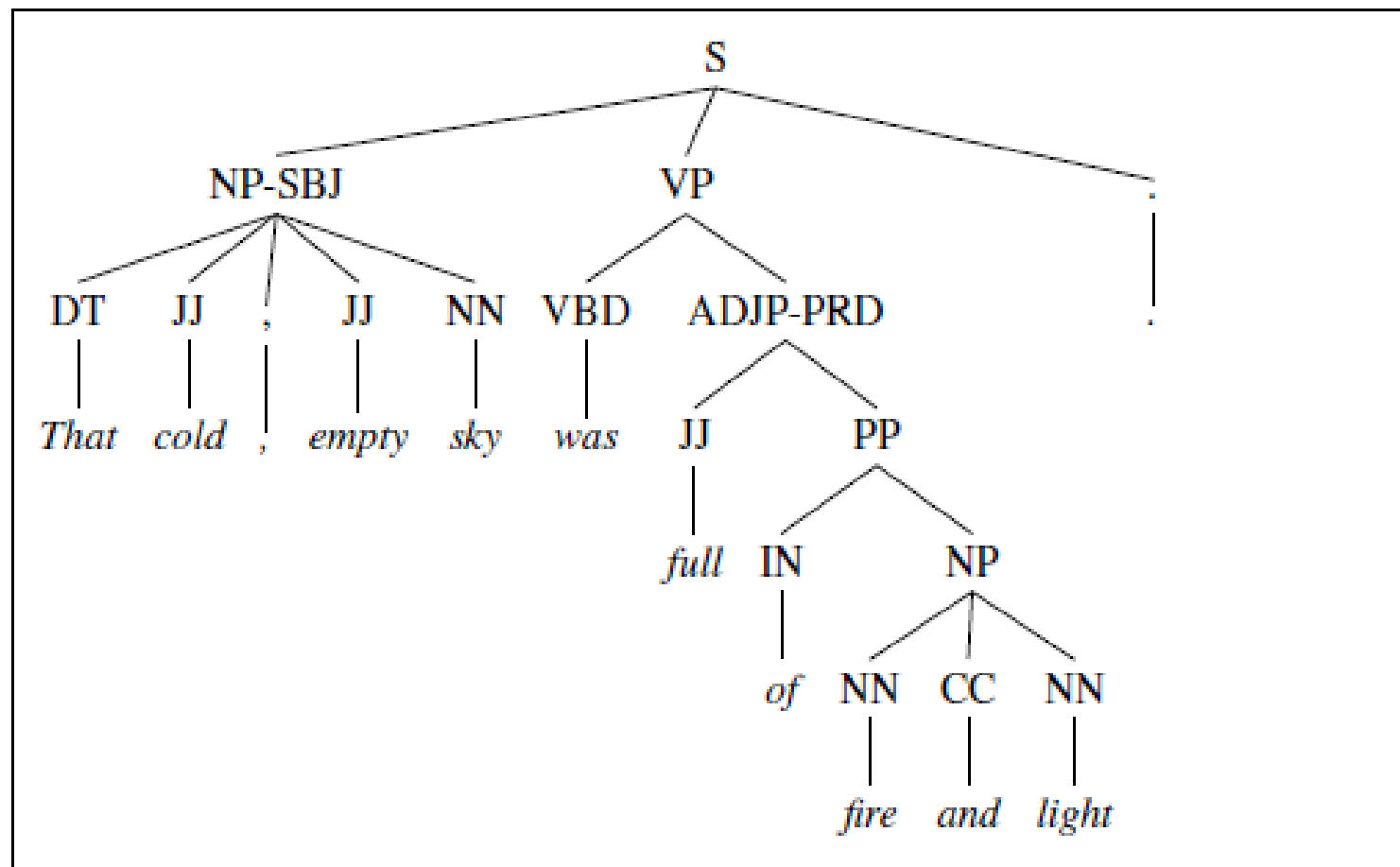


Figure 11.8 The tree corresponding to the Brown corpus sentence in the previous figure.

[REDACTED]

[REDACTED]

[REDACTED]

Treebanks as Grammars

- The sentences in a treebank implicitly constitute a grammar of the language represented by the corpus being annotated.
- Simply take the local rules that make up the subtrees in all the trees in the collection and you have a grammar
 - The WSJ section gives us about 12k rules if you do this
- Treebanks (and head-finding) are particularly critical to the development of statistical parsers