CSCI 5832
Natural Language Processing

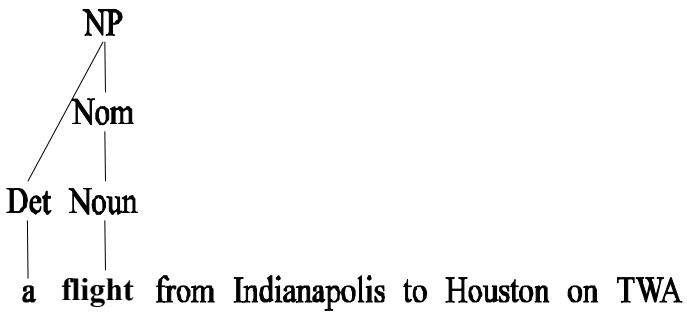
Lecture 12
Jim Martin

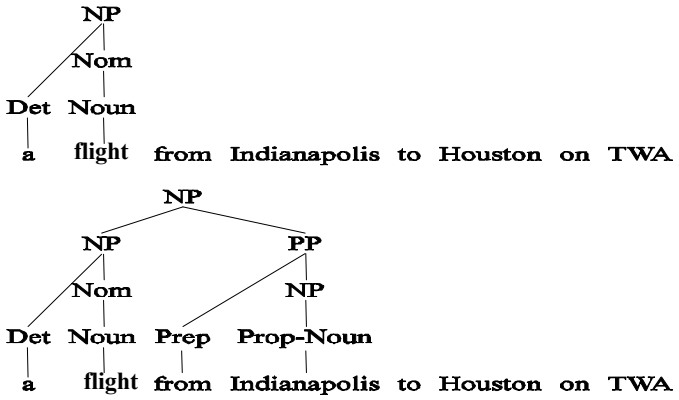
Today: 2/27

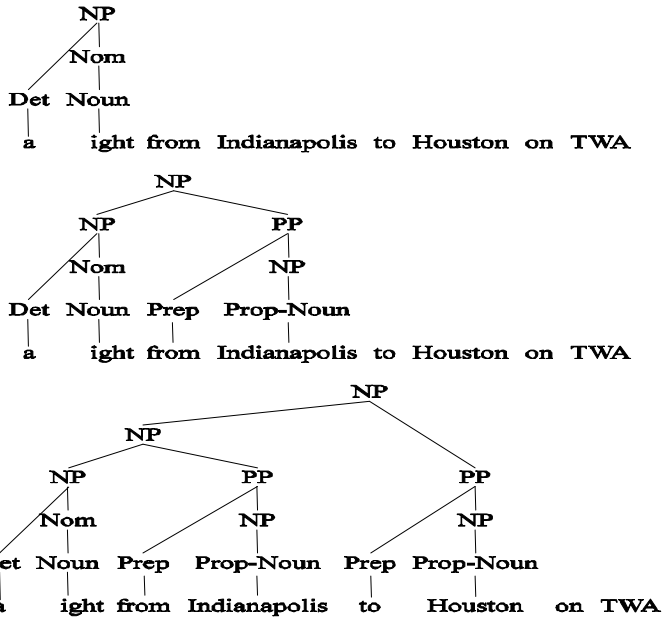
- **Review**
- **Treebanks**
- **Parsing**
- **Break**
- **More on projects**

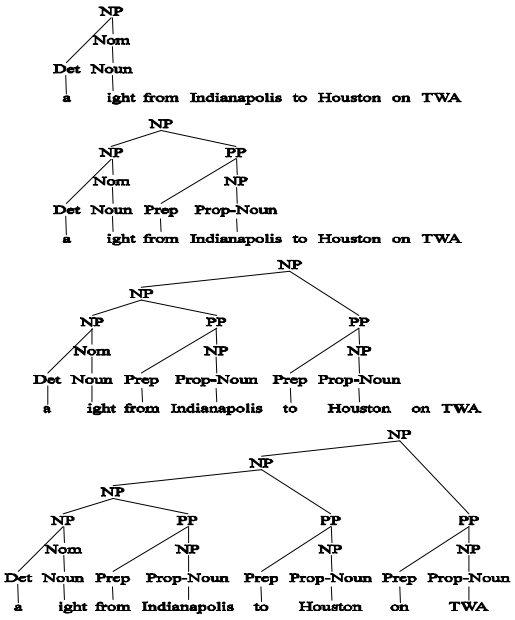
Avoiding Repeated Work

- Parsing is hard, and slow. It's wasteful to redo stuff over and over and over.
- Grammars are ambiguous both locally and globally exacerbating the parsing problems.




NP
Nom
Det Noun
a flight from Indianapolis to Houston on TWA
NP
NP PP
Nom NP
Det Noun Prep Prop-Noun
a flight from Indianapolis to Houston on TWA

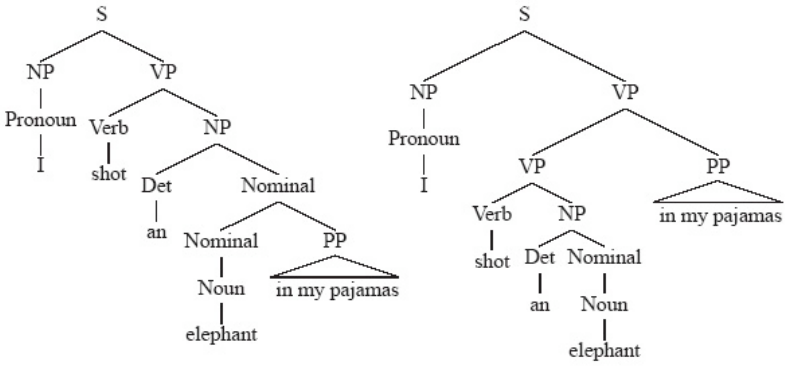

NP
Nom
Det Noun
a ight from Indianapolis to Houston on TWA
NP
NP PP
Nom NP
Det Noun Prep Prop-Noun
a ight from Indianapolis to Houston on TWA
NP
NP
NP PP PP
Nom NP NP
Det Noun Prep Prop-Noun Prep Prop-Noun
a ight from Indianapolis to Houston on TWA



Ambiguity

- For that example, the problem was local ambiguity; at the point a decision was being made the information needed to make the right decision wasn't there.
- What about global ambiguity?

Ambiguity



Ambiguity

- **Local ambiguity means that we have to deal with multiple plausible choices during the parsing process.**
- **Global ambiguity means that the grammar can't tell us which of several (many?) possible parses is the correct one.**
- **To deal with these problems we're going to…**
  - **Pursue all possible choices in parallel**
  - **Store (but not necessarily return) all globally consistent parse trees.**

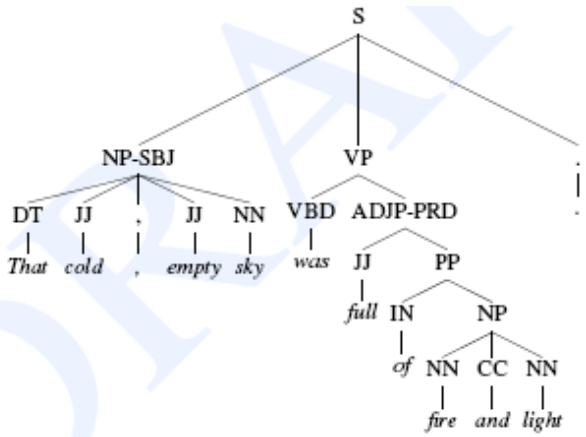
Grammars

- Before you can parse you need a grammar.
- So where do grammars come from?
  - Grammar Engineering
    - Lovingly hand-crafted decades-long efforts by humans to write grammars (typically in some particular grammar formalism of interest to the linguists developing the grammar).
  - TreeBanks
    - Semi-automatically generated sets of parse trees for the sentences in some corpus. Typically in a generic lowest common denominator formalism (of no particular interest to any modern linguist).

TreeBank Grammars

- Reading off the grammar…
- The grammar is the set of rules (local subtrees) that occur in the annotated corpus
- They tend to avoid recursion (and elegance and parsimony)
  - Ie. they tend to the flat and redundant
- Penn TreeBank (III) has about 17500 grammar rules under this definition.

TreeBanks



TreeBanks

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

Sample Rules

```
NP → DT JJ NNS
NP → DT JJ NN NN
NP → DT JJ JJ NN
NP → DT JJ CD NNS
NP → RB DT JJ NN NN
NP → RB DT JJ JJ NNS
NP → DT JJ JJ NNP NNS
NP → DT NNP NNP NNP NNP JJ NN
NP → DT JJ NNP CC JJ JJ NN NNS
NP → RB DT JJS NN NN SBAR
NP → DT VBG JJ NNP NNP CC NNP
NP → DT JJ NNS , NNS CC NN NNS NN
NP → DT JJ JJ VBG NN NNP NNP FW NNP
NP → NP JJ , JJ `` SBAR '' NNS
```


Example

```
NP → NP JJ , JJ `` SBAR '' NNS
```

(11.10) [NP Shearson's] [JJ easy-to-film], [JJ black-and-white] "[SBAR Where We Stand]" [NNS commercials]

TreeBanks

- TreeBanks provide a grammar (of a sort).
- As we'll see they also provide the training data for various ML approaches to parsing.
- But they can also provide useful data for more purely linguistic pursuits.
  - You might have a theory about whether or not something can happen in particular language.
  - Or a theory about the contexts in which something can happen.
  - TreeBanks can give you the means to explore those theories. If you can formulate the questions in the right way and get the data you need.

Tgrep

- You might for example like to grep through a file filled with trees.

```
NP < JJ . VP
```

```
(NP (NP (DT the) (JJ austere) (NN company) (NN dormitory))
  (VP (VBN run)
      (PP (IN by) (NP (DT a) (JJ prying) (NN caretaker)))))
```

TreeBanks

- Finally, you should have noted a bit of a circular argument here.
- Treebanks provide a grammar because we can read the rules of the grammar out of the treebank.
- But how did the trees get in there in the first place? There must have been a grammar theory in there someplace…

TreeBanks

- Typically, not all of the sentences are hand-annotated by humans.
- They're automatically parsed and then hand-corrected.

Break

- Plan is to have everybody in a group and all the groups with projects by Friday.
- We have a pretty good start on that already.
- Google semeval 2007 and CONLL to get ideas on some interesting tasks.

Parsing

- We're going to cover from Chapter 12
  - CKY (today)
  - Earley (Thursday)
- Both are dynamic programming solutions that run in O(n**3) time.
  - CKY is bottom-up
  - Earley is top-down

Sample Grammar

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

Dynamic Programming

- **DP methods fill tables with partial results and**
  - **Do not do too much avoidable repeated work**
  - **Solve exponential problems in polynomial time (sort of)**
  - **Efficiently store ambiguous structures with shared sub-parts.**

CKY Parsing

- First we'll limit our grammar to epsilon-free, binary rules (more later)
- Consider the rule A -> BC
  - If there is an A in the input then there must be a B followed by a C in the input.
  - If the A spans from i to j in the input then there must be some k st. i<k<j
    - Ie. The B splits from the C someplace.

CKY

- So let's build a table so that an A spanning from i to j in the input is placed in cell [i,j] in the table.
- So a non-terminal spanning an entire string will sit in cell [0, n]
- If we build the table bottom up we'll know that the parts of the A must go from i to k and from k to j

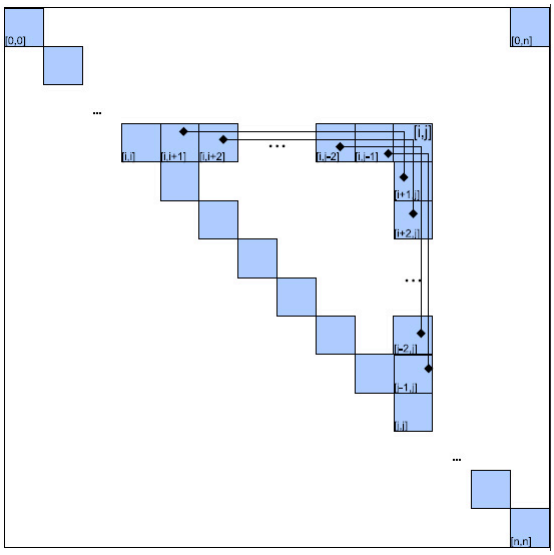
CKY

- Meaning that for a rule like A -> B C we should look for a B in [i,k] and a C in [k,j].
- In other words, if we think there might be an A spanning i,j in the input… AND
- A -> B C is a rule in the grammar THEN
- There must be a B in [i,k] and a C in [k,j] for some i<k<j

CKY

- So to fill the table loop over the cell[i,j] values in some systematic way
  - What constraint should we put on that?
  - For each cell loop over the appropriate k values to search for things to add.

CKY Table



CKY Algorithm

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

for $j \leftarrow$ **from** 1 **to** LENGTH(*words*) **do**
  $table[j-1, j] \leftarrow \{A \mid A \rightarrow words[j] \in grammar\}$
  for $i \leftarrow$ **from** $j-2$ **downto** 0 **do**
    for $k \leftarrow i+1$ **to** $j-1$ **do**
      $table[i,j] \leftarrow table[i,j] \cup$
        $\{A \mid A \rightarrow BC \in grammar,$
          $B \in table[i,k],$
          $C \in table[k,j]\}$

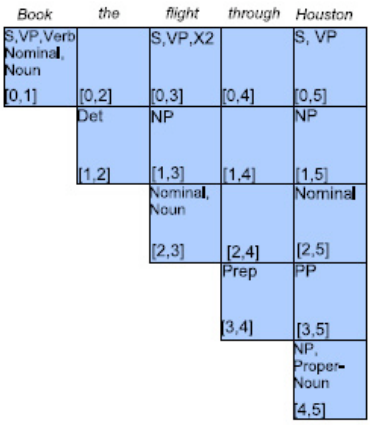
CKY Parsing

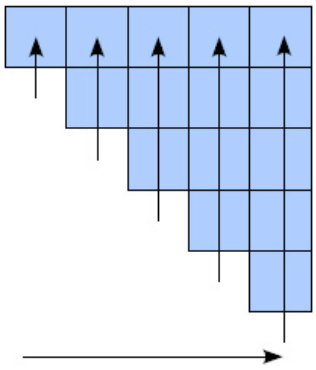
- Is that really a parser?

Note

- We arranged the loops to fill the table a column at a time, from left to right, bottom to top.
  - This assures us that whenever we're filling a cell, the parts needed to fill it are already in the table (to the left and below)

Example

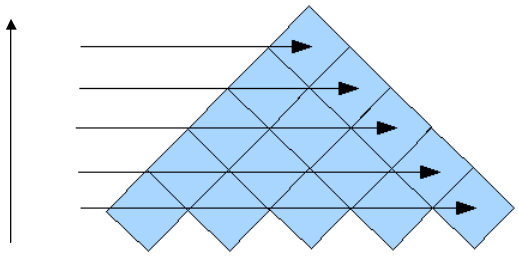




Other Ways to Do It?

- **Are there any other sensible ways to fill the table that still guarantee that the cells we need are already filled?**

Other Ways to Do It?



Sample Grammar

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

Problem

- What if your grammar isn't binary?
  - As in the case of the TreeBank grammar?
- Convert it to binary… any arbitrary CFG can be rewritten into Chomsky-Normal Form automatically.
- What does this mean?
  - The resulting grammar accepts (and rejects) the same set of strings as the original grammar.
  - But the resulting derivations (trees) are different.

Problem

- More specifically, rules have to be of the form

  A -> B C

  Or

  A -> w

  That is rules can expand to either 2 non-terminals or to a single terminal.

Binarization Intuition

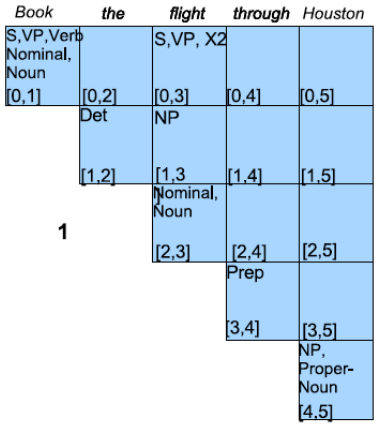
- Eliminate chains of unit productions.
- Introduce new intermediate non-terminals into the grammar that distribute rules with length > 2 over several rules. So…
  - S -> A B C
    - Turns into
  - S -> X C
  - X - A B

  Where X is a symbol that doesn't occur anywhere else in the the grammar.

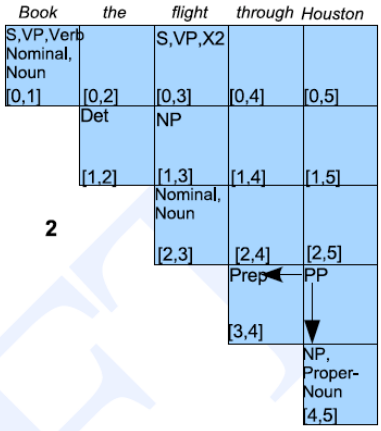
CNF Conversion

$S \rightarrow NP\ VP$	$S \rightarrow NP\ VP$
$S \rightarrow Aux\ NP\ VP$	$S \rightarrow X1\ VP$
	$X1 \rightarrow Aux\ NP$
$S \rightarrow VP$	$S \rightarrow book \mid include \mid prefer$
	$S \rightarrow Verb\ NP$
	$S \rightarrow X2\ PP$
	$S \rightarrow Verb\ PP$
	$S \rightarrow VP\ PP$
$NP \rightarrow Pronoun$	$NP \rightarrow I \mid she \mid me$
$NP \rightarrow Proper\text{-}Noun$	$NP \rightarrow TWA \mid Houston$
$NP \rightarrow Det\ Nominal$	$NP \rightarrow Det\ Nominal$
$Nominal \rightarrow Noun$	$Nominal \rightarrow book \mid flight \mid meal \mid money$
$Nominal \rightarrow Nominal\ Noun$	$Nominal \rightarrow Nominal\ Noun$
$Nominal \rightarrow Nominal\ PP$	$Nominal \rightarrow Nominal\ PP$
$VP \rightarrow Verb$	$VP \rightarrow book \mid include \mid prefer$
$VP \rightarrow Verb\ NP$	$VP \rightarrow Verb\ NP$
$VP \rightarrow Verb\ NP\ PP$	$VP \rightarrow X2\ PP$
	$X2 \rightarrow Verb\ NP$
$VP \rightarrow Verb\ PP$	$VP \rightarrow Verb\ PP$
$VP \rightarrow VP\ PP$	$VP \rightarrow VP\ PP$
$PP \rightarrow Preposition\ NP$	$PP \rightarrow Preposition\ NP$

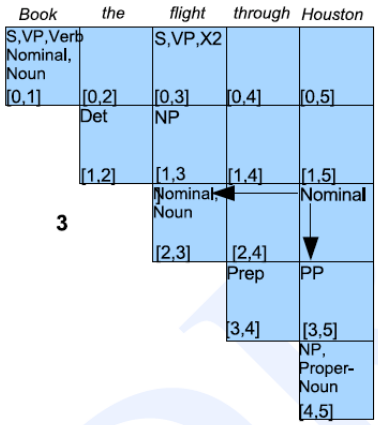
Example



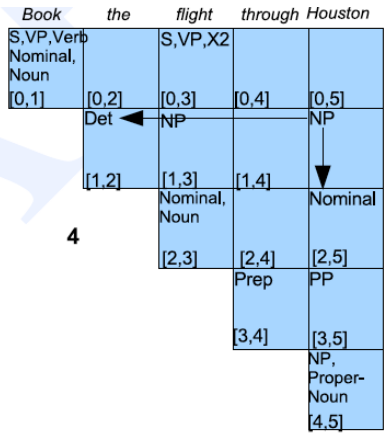
Example



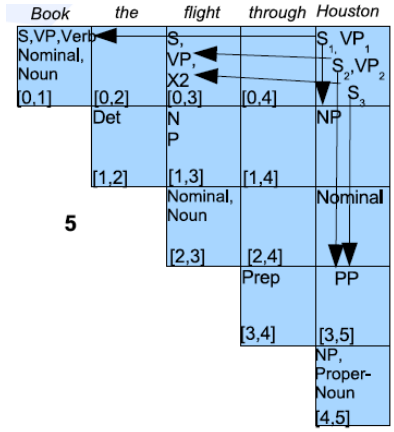
Example



Example



Example


Book
the
flight
through
Houston
S,VP,Verb
Nominal,
Noun
[0,1]
[0,2]
Det
[1,2]
S,
VP,
X2
[0,3]
N
P
[1,3]
Nominal,
Noun
[2,3]
[0,4]
[1,4]
[2,4]
Prep
[3,4]
S1, VP1
S2, VP2
S3
NP
Nominal
PP
[3,5]
NP,
Proper-
Noun
[4,5]
5