Amirkabir University of Technology
(Tehran Polytechnic)

Natural Language Processing

Lecture 14: Constituency Grammar and Constituency Parsing

Amirkabir University of Technology
Dr Momtazi

Parsing

- Finding structural relationship between words in a sentence

Parsing

- Finding structural relationship between words in a sentence
- Applications
  - Spell checking
  - Speech recognition
  - Machine translation
  - Language modeling

Outline

- **Introduction**
- Context Free Grammar
  - Some Grammar Rules
  - Syntactic Parsing
- Probabilistic Context Free Grammar
  - Statistical Parsing
- Lexicalized PCFG
- Partial Parsing
- Evaluation

Constituency

- Working based on Constituency (Phrase structure)
  - Organizing words into nested constituents
  - Showing that groups of words within utterances can act as single units
  - Forming coherent classes from these units that can behave in similar ways
    - With respect to their internal structure
    - With respect to other units in the language
  - Considering a head word for each constituent

Constituency

the writer talked to the audiences about his new book.

Constituency

the writer talked to the audiences about his new book.

Constituency

the writer talked to the audiences about his new book.

the writer talked about his new book to the audiences. ✓

about his new book the writer talked to the audiences. ✓

Constituency

the writer talked to the audiences about his new book.

the writer talked about his new book to the audiences. ✓

about his new book the writer talked to the audiences. ✓

the writer talked book to the audiences about his new. ✗

Outline

- Introduction
- **Context Free Grammar**
  - Some Grammar Rules
  - Syntactic Parsing
- Probabilistic Context Free Grammar
  - Statistical Parsing
- Lexicalized PCFG
- Partial Parsing
- Evaluation

Context Free Grammer (CFG)

- Grammar G consists of
  - Terminals (T)
  - Non-terminals (N)
  - Start symbol (S)
  - Rules (R)

CFG

- Terminals
  - The set of words in the text
- Non-Terminals
  - The constituents in a language (noun phrase, verb phrase, ...)
- Start symbol
  - The main constituent of the language (sentence)
- Rules
  - Equations that consist of a single non-terminal on the left and any number of terminals and non-terminals on the right

CFG

S → NP VP

S → VP

NP → N

NP → Det N

NP → NP NP

NP → NP PP

VP → V

VP → VP PP

VP → VP NP

PP → Prep NP

N → book

V → book

Det → the

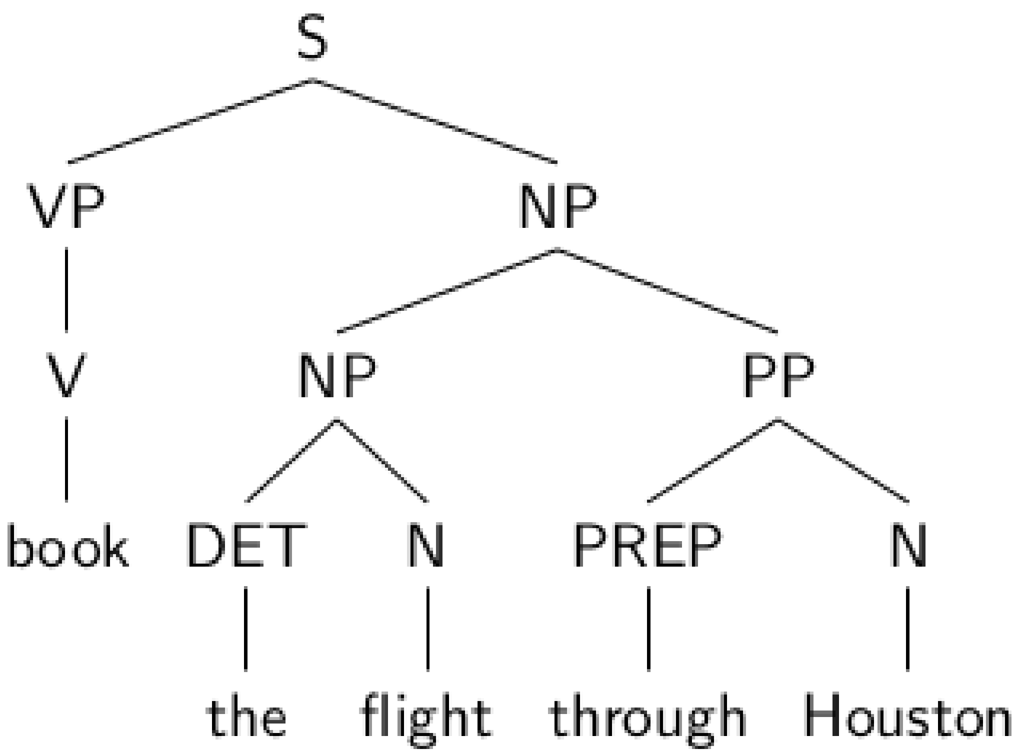
N → flight

Prep → through

N → Houston

CFG

CFG



Outline

- Introduction
- **Context Free Grammar**
  - **Some Grammar Rules**
  - Syntactic Parsing
- Probabilistic Context Free Grammar
  - Statistical Parsing
- Lexicalized PCFG
- Partial Parsing
- Evaluation

Main Grammar Fragments

- Sentence
- Noun Phrase
  - Agreement
- Verb Phrase
  - Sub-categorization

Main Grammar Fragments

- Declaratives

  A plane left.

  S → NP VP

- Imperatives

  Leave!

  S → VP

- Yes-No Questions

  Did the plane leave?

  S → Aux NP VP

- WH Questions

  When did the plane leave?

  S → NP_{WH} Aux NP VP

Main Grammar Fragments

- Each NP has a central critical noun called head
- The head of an NP can be expressed using
  - Pre-nominals: the words that can come before the head
  - Post-nominals: the words that can come after the head

Grammar Fragments: NP

- Pre-nominals
  - Simple lexical items: the, this, a, an, ...

    a car
- Simple possessives
  - John's car
- Complex recursive possessives
  - John's sister's friend's car
- Quantifiers, cardinals, ordinals...
  - three cars
- Adjectives
  - large cars

Grammar Fragments: NP

- Post-nominals
  - Prepositional phrases

    flight from Seattle
  - Non-finite clauses

    flight arriving before noon
  - Relative clauses

    flight that serves breakfast

Agreement

- Having constraints that hold among various constituents
- Considering these constraints in a rule or set of rules

Agreement

- Having constraints that hold among various constituents
- Considering these constraints in a rule or set of rules

Example: determiners and the head nouns in NPs have to agree in number

This flight ✓

Those flights ✓

This flights ✘

Those flight ✘

Agreement

- Having constraints that hold among various constituents
- Considering these constraints in a rule or set of rules

Example: determiners and the head nouns in NPs have to agree in number

This flight ✓

Those flights ✓

This flights ✗

Those flight ✗

- Grammars that do not consider constraints will over-generate
  - Accepting and assigning correct structures to grammatical examples (this flight)
  - But also accepting incorrect examples (these flight)

Agreement at sentence level

- Considering similar constraints at sentence level

Agreement at sentence level

- Considering similar constraints at sentence level

Example: subject and verb in sentences have to agree in number and person

John flies ✓

We fly ✓

John fly ✗

We flies ✗

Agreement

- Possible CFG solution

$S_{sg} \rightarrow NP_{sg}\ \ VP_{sg}$

$S_{pl} \rightarrow NP_{pl}\ \ VP_{pl}$

$NP_{sg} \rightarrow DET_{sg}\ \ N_{sg}$

$NP_{pl} \rightarrow DET_{pl}\ \ N_{pl}$

$VP_{sg} \rightarrow VP_{sg}\ \ NP_{sg}$

$VP_{pl} \rightarrow VP_{pl}\ \ NP_{pl}$

…

Agreement

- Possible CFG solution

$S_{sg} \rightarrow NP_{sg}\ \ VP_{sg}$

$S_{pl} \rightarrow NP_{pl}\ \ VP_{pl}$

$NP_{sg} \rightarrow DET_{sg}\ \ N_{sg}$

$NP_{pl} \rightarrow DET_{pl}\ \ N_{pl}$

$VP_{sg} \rightarrow VP_{sg}\ \ NP_{sg}$

$VP_{pl} \rightarrow VP_{pl}\ \ NP_{pl}$

…

- Shortcoming:
  - Introducing many rules in the system

Grammar Fragments: VP

- VPs consist of a head verb along with zero or more constituents called arguments

VP → V	Type equation here.disappear
VP → V NP	prefer a morning flight
VP → V PP	fly on Thursday
VP → V NP PP	leave Boston in the morning
VP → V NP NP	give me the flight number

Grammar Fragments: VP

- VPs consist of a head verb along with zero or more constituents called arguments

VP → V	Type equation here.disappear
VP → V NP	prefer a morning flight
VP → V PP	fly on Thursday
VP → V NP PP	leave Boston in the morning
VP → V NP NP	give me the flight number

- Arguments
  - Obligatory: complement
  - Optional: adjunct

Sub-categorization

- Even though there are many valid VP rules, not all verbs are allowed to participate in all VP rules

disappear a morning flight ✗

Sub-categorization

- Even though there are many valid VP rules, not all verbs are allowed to participate in all VP rules

  disappear a morning flight ✘

- Solution:
  - Subcategorizing the verbs according to the sets of VP rules that they can participate in
  - This is a modern take on the traditional notion of transitive/intransitive
  - Modern grammars may have 100s or such classes

Sub-categorization

- Example:

Sneeze	John sneezed
Find	Please find [a flight to NY]NP
Give	Give [me]NP[a cheaper fair]NP
Help	Can you help [me]NP[with a flight]PP
Prefer	I prefer [to leave earlier]TO-VP
Told	I was told [United has a flight]S

John sneezed the book ✘

I prefer United has a flight ✘

Give with a flight ✘

Sub-categorization

- The over-generation problem also exists in VP rules
  - Permitting the presence of strings containing verbs and arguments that do not go together

John sneezed the book

VP → V NP

- Solution:
  - Similar to agreement phenomena, we need a way to formally express the constraints

Outline

- Introduction
- **Context Free Grammar**
  - Some Grammar Rules
  - **Syntactic Parsing**
- Probabilistic Context Free Grammar
  - Statistical Parsing
- Lexicalized PCFG
- Partial Parsing
- Evaluation

Parsing

- Taking a string and a grammar and returning proper parse tree(s) for that string
- Covering all and only the elements of the input string
- Reaching the start symbol at the top of the string

Parsing

- Taking a string and a grammar and returning proper parse tree(s) for that string
- Covering all and only the elements of the input string
- Reaching the start symbol at the top of the string

- The system cannot select the correct tree among all the possible trees

Parsing Algorithms

- Top-Down
  - Starting with the rules that give us an S, since trees should be rooted with an S
  - Working on the way down from S to the words

- Bottom-Up

Parsing Algorithms

- Top-Down
  - Starting with the rules that give us an S, since trees should be rooted with an S
  - Working on the way down from S to the words

- Bottom-Up
  - Starting with trees that link up with the words, since trees should cover the input words
  - Working on the way up from words to larger and larger trees

Top-Down vs. Bottom-up

- Top-Down
  - Only searches for trees that can be answers (i.e. S’s)
  - But also suggests trees that are not consistent with any of the Words
- Bottom-Up
  - Only forms trees consistent with the words
  - But suggests trees that make no sense globally

Top-Down vs. Bottom-up

- In both cases, we left out how to keep track of the search space and how to make choices

Top-Down vs. Bottom-up

- In both cases, we left out how to keep track of the search space and how to make choices

- Solutions
  - Backtracking
    - Making a choice, if it works out then fine
    - If not, then back up and make a different choice

      ⇨duplicated work

Top-Down vs. Bottom-up

- In both cases, we left out how to keep track of the search space and how to make choices

- Solutions
  - Backtracking
    - Making a choice, if it works out then fine
    - If not, then back up and make a different choice

      ⇨duplicated work
  - Dynamic programming
    - Avoiding repeated work
    - Solving exponential problems in polynomial time
    - Storing ambiguous structures efficiently

Dynamic Programming Methods

- CKY: bottom-up
- Early: top-down

Chomsky Normal Form

- Each grammar can be represented by a set of binary rules

$$A \rightarrow B\ C$$

$$A \rightarrow w$$

A, B, C are noun-terminals w is a terminal

Chomsky Normal Form

- Converting to Chomsky normal form

$$A \rightarrow B\ C\ D$$

Chomsky Normal Form

- Converting to Chomsky normal form

$$A \rightarrow B\ C\ D$$

$$X \rightarrow B\ C$$

$$A \rightarrow X\ D$$

X does not occur anywhere else in the the grammar

Chomsky Normal Form

- Converting to Chomsky normal form

$$A \rightarrow B$$

$$B \rightarrow C\ D$$

Chomsky Normal Form

- Converting to Chomsky normal form

$$A \to B$$

$$B \to C\ D$$

$$A \to C\ D$$

CKY Parsing

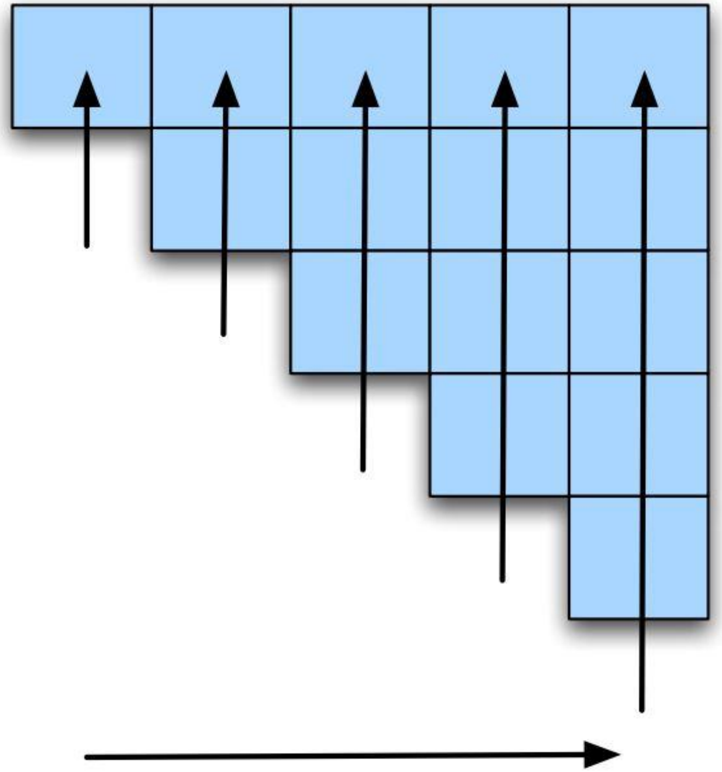
$$A \rightarrow B\ C$$

- If there is an A somewhere 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 a k such that $i < k < j$
  - B spans from i to k
  - C spans from k to j

CKY Parsing

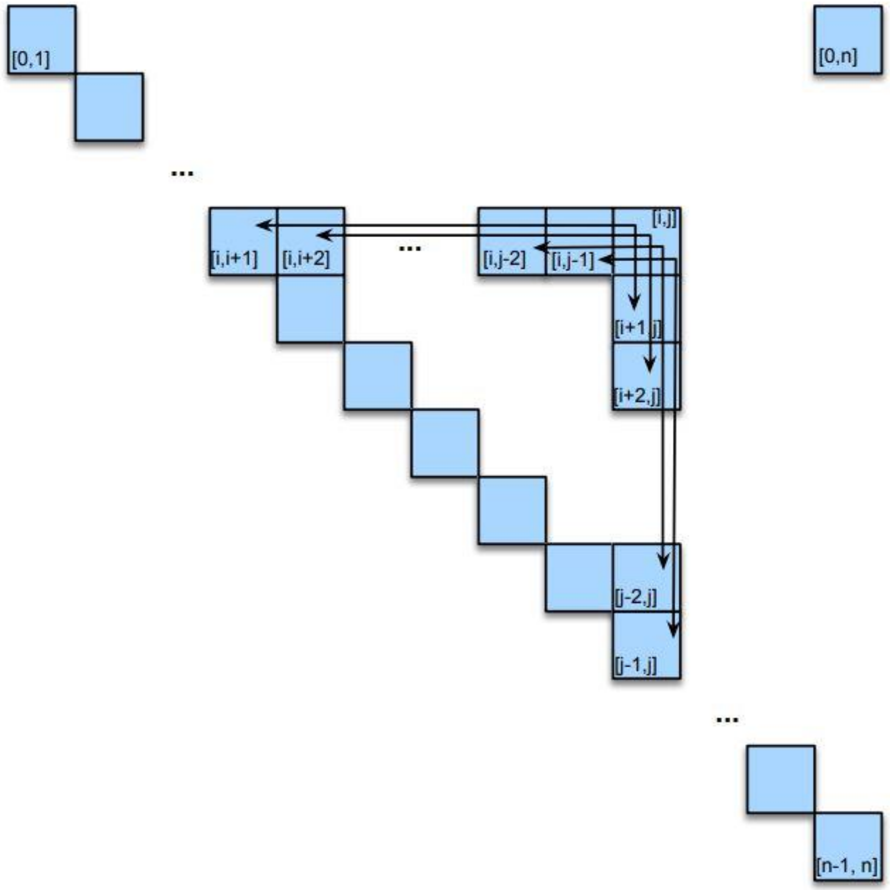
- Completed parse table for Book the flight through Houston.

Book	the	flight	through	Houston
S, VP, Verb, Nominal, Noun [0,1]	[0,2]	S,VP,X2 [0,3]	[0,4]	S,VP,X2 [0,5]
	Det [1,2]	NP [1,3]	[1,4]	NP [1,5]
		Nominal, Noun [2,3]	[2,4]	Nominal [2,5]
			Prep [3,4]	PP [3,5]
				NP, Proper-Noun [4,5]



CKY Parsing

- All the ways to fill the [i, j]th cell in the CKY table.



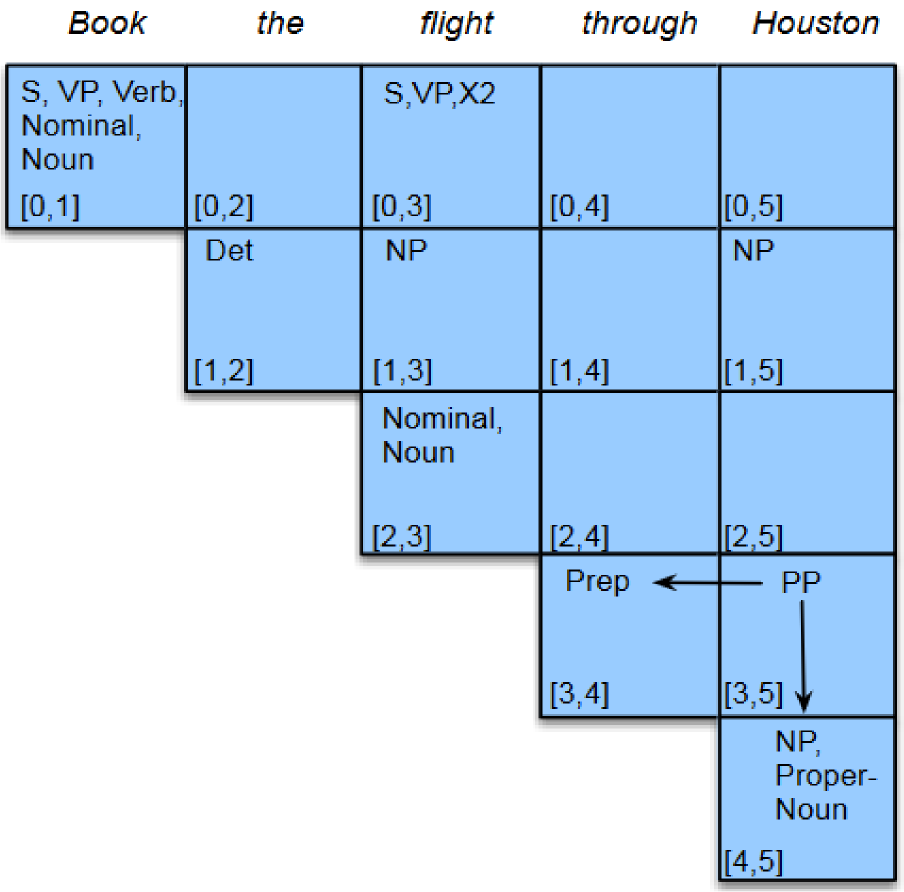
CKY Parsing

- Filling the cells of column 5 after reading the word Houston.

Book	*the*	*flight*	*through*	*Houston*
S, VP, Verb, Nominal, Noun [0,1]	[0,2]	S,VP,X2 [0,3]	[0,4]	[0,5]
	Det [1,2]	NP [1,3]	[1,4]	[1,5]
		Nominal, Noun [2,3]	[2,4]	Nominal [2,5]
			Prep [3,4]	[3,5]
				NP, Proper-Noun [4,5]

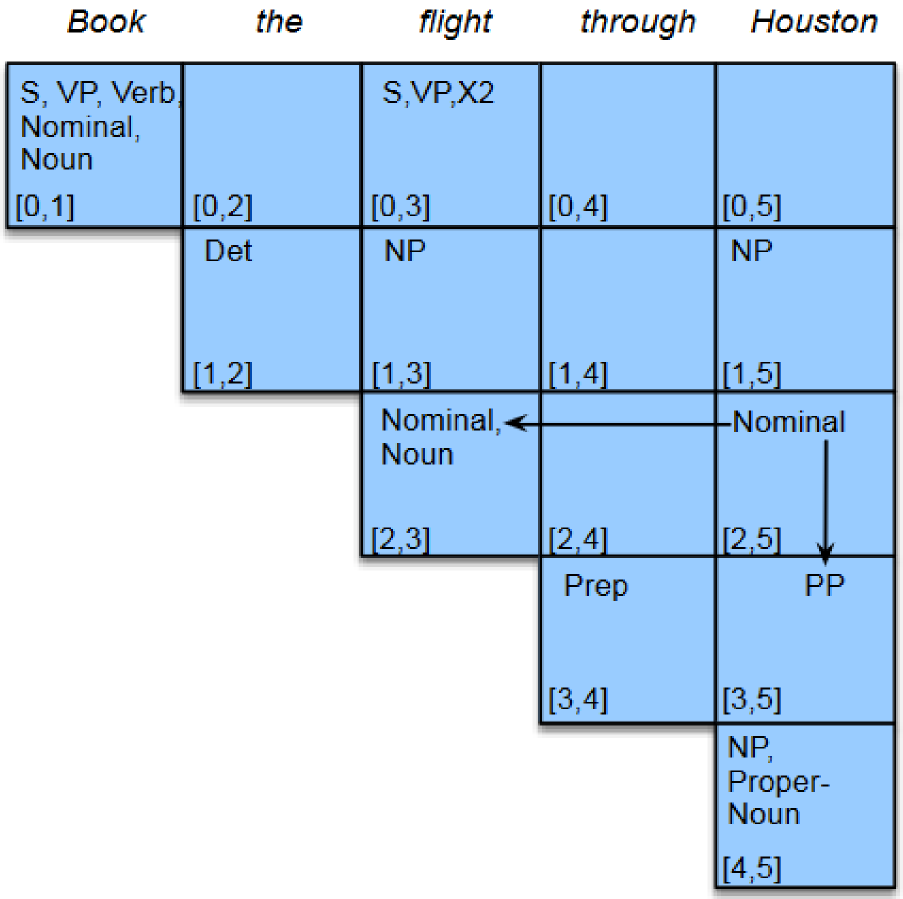
CKY Parsing

- Filling the cells of column 5 after reading the word Houston.



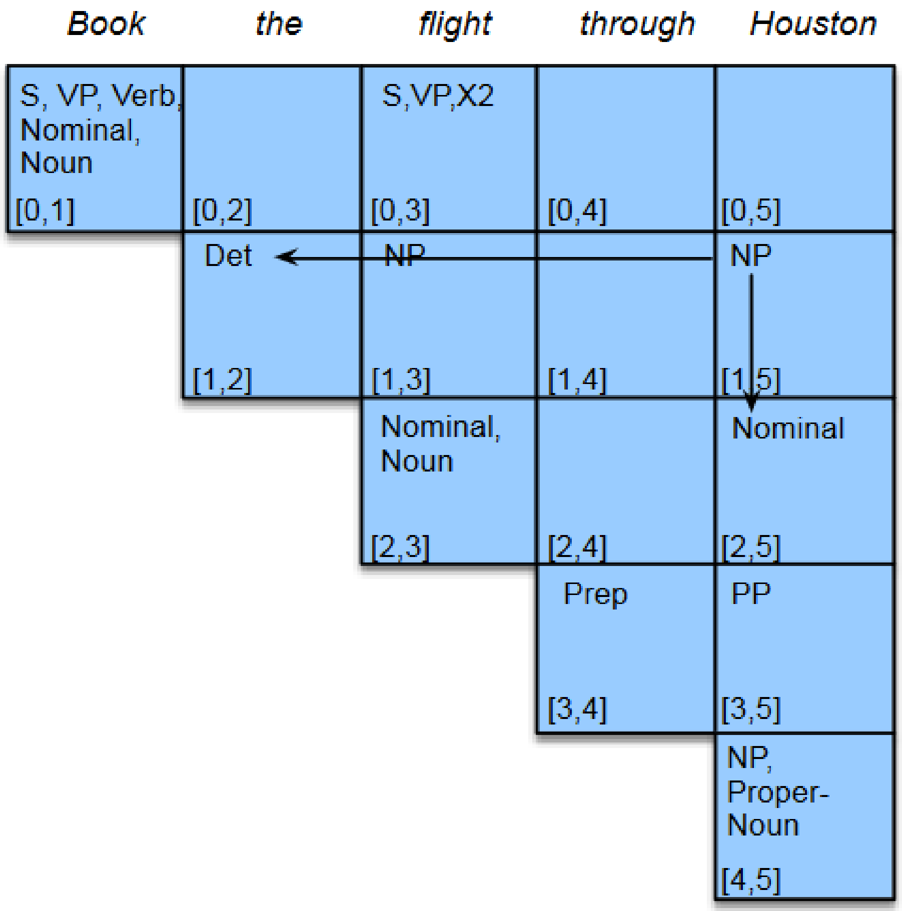
CKY Parsing

- Filling the cells of column 5 after reading the word Houston.



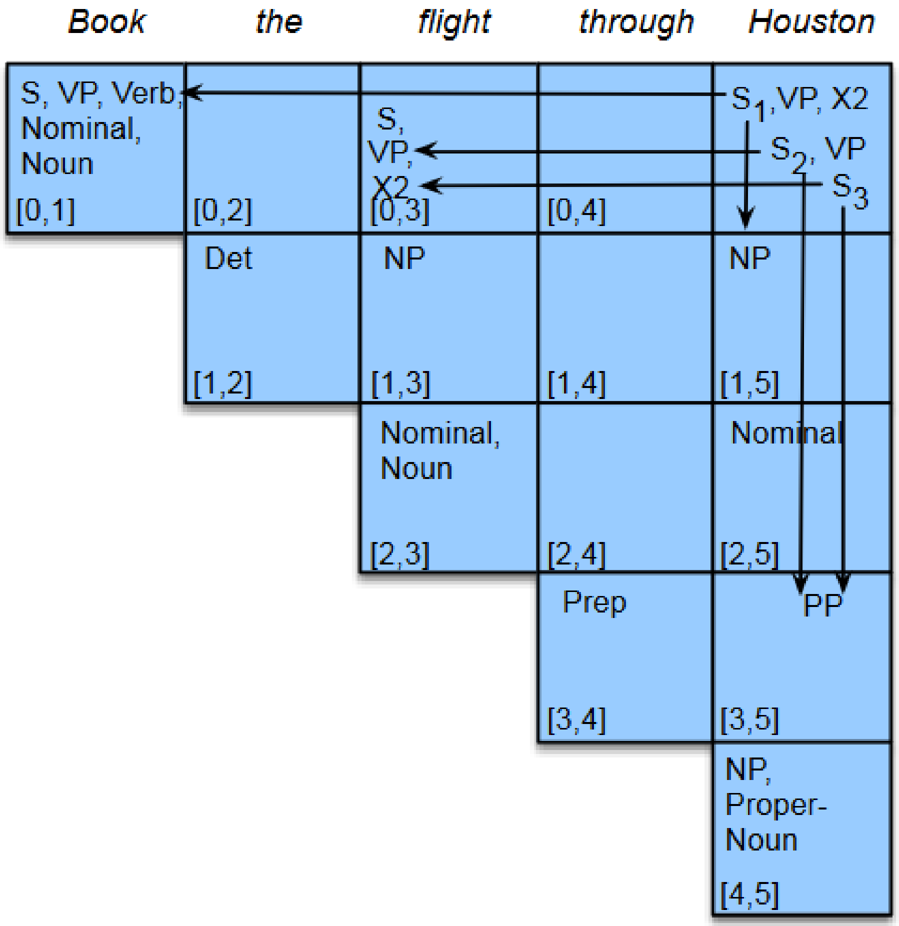
CKY Parsing

- Filling the cells of column 5 after reading the word Houston.



CKY Parsing

- Filling the cells of column 5 after reading the word Houston.



Outline

- Introduction
- Context Free Grammar
  - Some Grammar Rules
  - Syntactic Parsing
- **Probabilistic Context Free Grammar**
  - Statistical Parsing
- Lexicalized PCFG
- Partial Parsing
- Evaluation

Probabilistic Context Free Grammer (PCFG)

- Grammar G consists of
  - Terminals (T)
  - Non-terminals (N)
  - Start symbol (S)
  - Rules (R)

Probabilistic Context Free Grammer (PCFG)

- Grammar G consists of
  - Terminals (T)
  - Non-terminals (N)
  - Start symbol (S)
  - Rules (R)
  - Probability function (P)
    - $\mathrm{P} : \mathrm{R} \rightarrow [0; 1]$
    - $\forall X \in N, \sum_{X \rightarrow \lambda \in r} P(X \rightarrow \lambda) = 1$

CFG

$S \rightarrow NP\ VP$

$S \rightarrow VP$

$NP \rightarrow N$

$NP \rightarrow Det\ N$

$NP \rightarrow NP\ NP$

$NP \rightarrow NP\ PP$

$VP \rightarrow V$

$VP \rightarrow VP\ PP$

$VP \rightarrow VP\ NP$

$PP \rightarrow Prep\ NP$

$N \rightarrow$ book

$V \rightarrow$ book

$Det \rightarrow$ the

$N \rightarrow$ flight

$Prep \rightarrow$ through

$N \rightarrow$ Houston

PCFG

$S \rightarrow NP\ VP$	0.9	$N \rightarrow$ book	0.5
$S \rightarrow VP$	0.1	$V \rightarrow$ book	1.0
$NP \rightarrow N$	0.3	$Det \rightarrow$ the	1.0
$NP \rightarrow Det\ N$	0.4	$N \rightarrow$ flight	0.4
$NP \rightarrow NP\ NP$	0.1	$Prep \rightarrow$ through	1.0
$NP \rightarrow NP\ PP$	0.2	$N \rightarrow$ Houston	0.1
$VP \rightarrow V$	0.1		
$VP \rightarrow VP\ PP$	0.3		
$VP \rightarrow VP\ NP$	0.6		
$PP \rightarrow Prep\ NP$	1.0		

Treebank

- A treebank is a corpus in which each sentence has been paired with a parse tree
- These are generally created by
  - Parsing the collection with an automatic parser
  - Correcting each parse by human annotators if required
- Requirement:

  detailed annotation guidelines that provide
  - A POS tagset
  - A grammar
  - Annotation schema
    - Instructions for how to deal with particular grammatical constructions

Penn Treebank

- Penn Treebank is a widely used treebank for English
  - Most well-known section: Wall Street Journal Section
    - 1 M words from 1987-1989

```
(S (NP (NNP John))
        (VP (VPZ flies)
                (PP (IN to)
                        (NNP Paris)))
(. .))
```

Annotated Data

- Providing a valuable linguistic resource
- Can be used by many taggers and parsers (reusability)
- Broad coverage
- Providing various information
  - Frequencies and distributions
- Can be used for evaluating systems

Outline

- Introduction
- Context Free Grammar
  - Some Grammar Rules
  - Syntactic Parsing
- **Probabilistic Context Free Grammar**
  - **Statistical Parsing**
- Lexicalized PCFG
- Neural Constituency Parsing
- Partial Parsing
- Evaluation

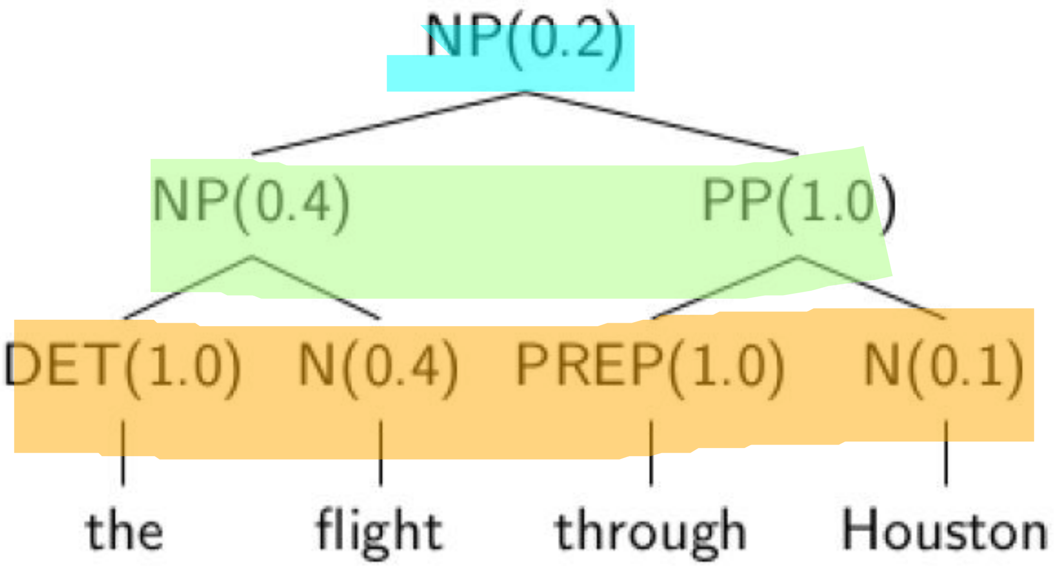
Statistical Parsing

- Considering the corresponding probabilities while parsing a sentence
- Selecting the parse tree which has the highest probability
- P(t): the probability of a tree t
  - Product of the probabilities of the rules used to generate the tree

PCFG

S → NP VP	0.9	N → book	0.5
S → VP	0.1	V → book	1.0
NP → N	0.3	Det → the	1.0
NP → Det N	0.4	N → flight	0.4
NP → NP NP	0.1	Prep → through	1.0
NP → NP PP	0.2	N → Houston	0.1
VP → V	0.1		
VP → VP PP	0.3		
VP → VP NP	0.6		
PP → Prep NP	1.0		

Statistical Parsing



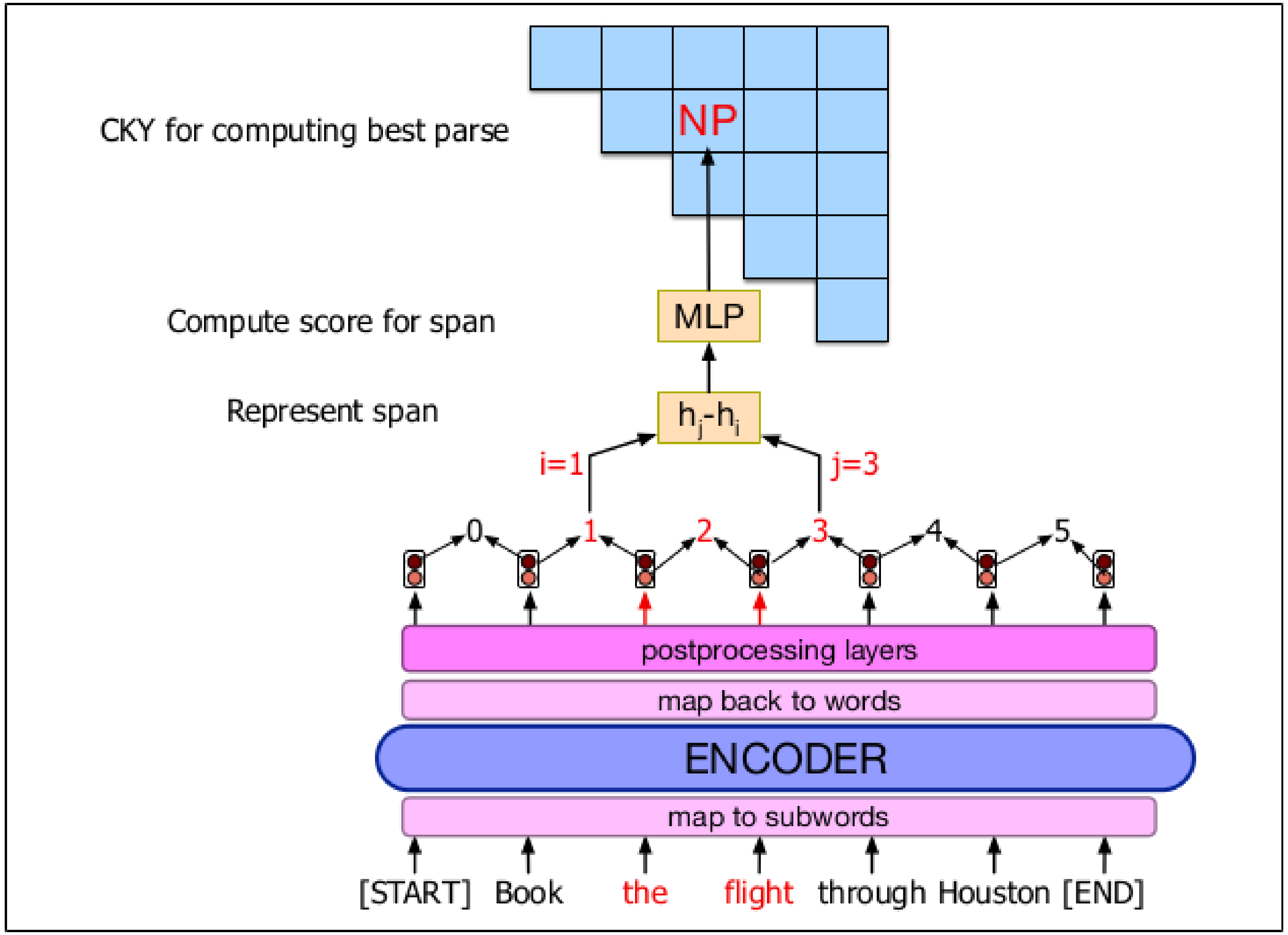
$P(t) = 0.2 \times 0.4 \times 1.0 \times 1.0 \times 0.4 \times 1.0 \times 0.12 = 0.0032$

Probabilistic CKY Parsing

0.5 N → book$_{[0,1]}$ 1.0 V → book$_{[0,1]}$ 0.3*0.5=0.15 NP → N$_{[0,1]}$ 0.1*1.0=0.1 VP → V$_{[0,1]}$ 0.1*0.1=0.01 S → VP$_{[0,1]}$ [0,1]	[0,2]	0.1*0.15*0.16=0.0024 NP → NP$_{[0,1]}$, NP$_{[1,3]}$ 0.6*0.1*0.16=0.0096 VP → VP$_{[0,1]}$, NP$_{[1,3]}$ 0.1*0.0096=0.00096 S → VP$_{[0,3]}$ [0,3]	[0,4]	VP → VP$_{[0,1]}$, NP$_{[1,5]}$ VP' → VP$_{[0,3]}$, PP$_{[3,5]}$ S → VP$_{[0,5]}$ S → VP'$_{[0,5]}$ [0,5]
	1.0 Det → the$_{[1,2]}$ [1,2]	NP → Det$_{[1,2]}$, N$_{[2,3]}$ 0.4*1.0*0.4=0.16 [1,3]	[1,4]	NP → NP$_{[1,3]}$, PP$_{[3,5]}$ [1,5]
		0.4 N → flight$_{[2,3]}$ 0.3*0.4=0.12 NP → N$_{[2,3]}$ [2,3]	[2,4]	NP → NP$_{[2,3]}$, PP$_{[3,5]}$ [2,5]
			Prep → through$_{[3,4]}$ [3,4]	PP → Prep$_{[3,4]}$,NP$_{[4,5]}$ [3,5]
				N → houston$_{[4,5]}$ NP → N$_{[4,5]}$ [4,5]

Neural Constituency Parsing

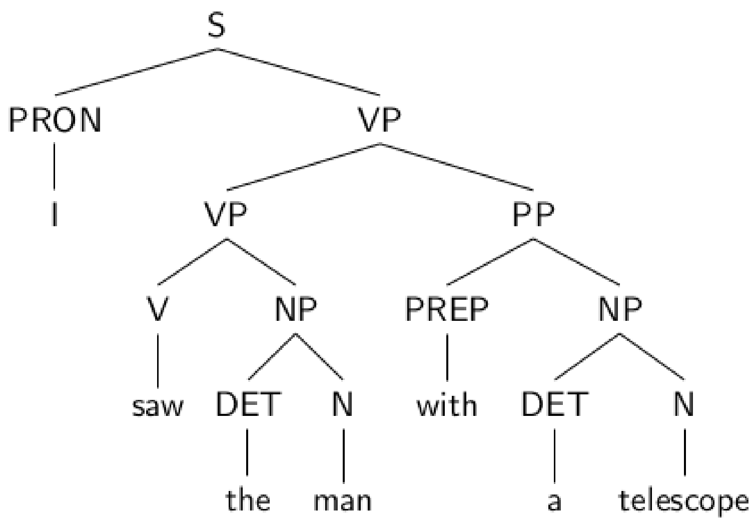
- Span-based model

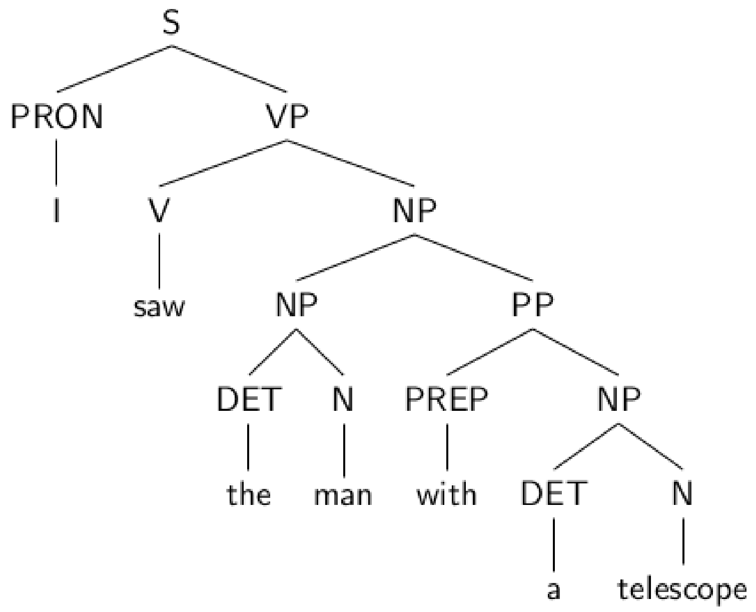


Parsing Ambiguity

- The main source of ambiguities in parsing
  - Finding the correct place of attachments

I saw the man with a telescope.



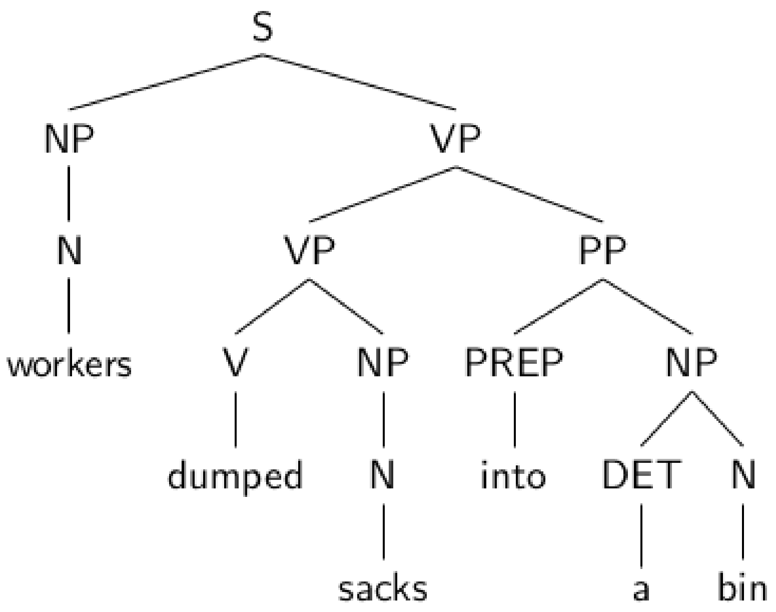


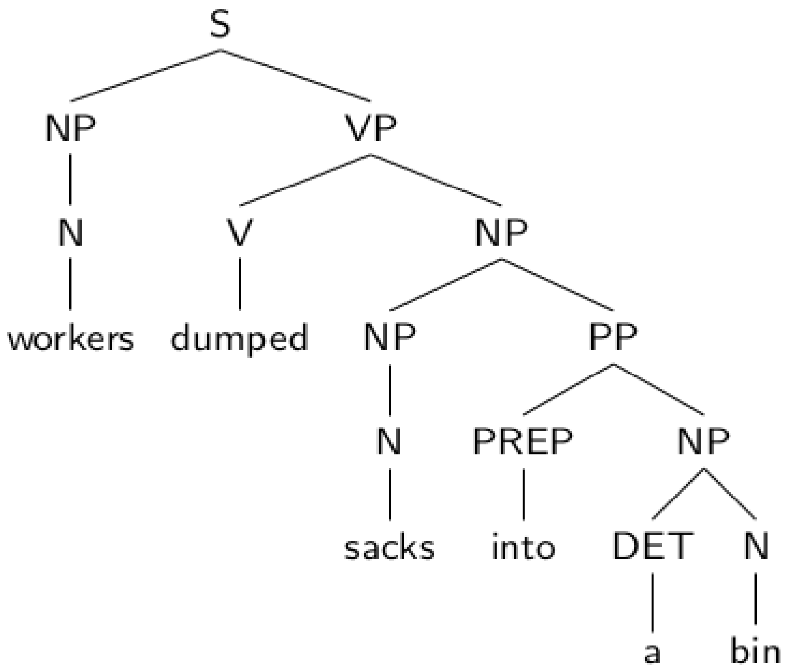
Problems

- PCFG considers no knowledge about the words
- Strong independent assumption in parse trees

Problems

- PCFG considers no knowledge about the words
- Strong independent assumption in parse trees





Problems

(a)

Rules
S → NP VP
NP → NNS
VP → VP PP
VP → VBD NP
NP → NNS
PP → IN NP
NP → DT NN
NNS → workers
VBD → dumped
NNS → sacks
IN → into
DT → a
NN → bin

(b)

Rules
S → NP VP
NP → NNS
NP → NP PP
VP → VBD NP
NP → NNS
PP → IN NP
NP → DT NN
NNS → workers
VBD → dumped
NNS → sacks
IN → into
DT → a
NN → bin

Problems

- Knowing about lexicon help us to select a better option for merging constituents

Local Tree	*come*	*take*	*think*	*want*
VP → V	9.5%	2.6%	4.6%	5.7%
VP → V NP	1.1%	32.1%	0.2%	13.9%
VP → V PP	34.5%	3.1%	7.1%	0.3%
VP → V SBAR	6.6%	0.3%	73.0%	0.2%
VP → V S	2.2%	1.3%	4.8%	70.8%
VP → V NP S	0.1%	5.7%	0.0%	0.3%
VP → V PRT NP	0.3%	5.8%	0.0%	0.0%
VP → V PRT PP	6.1%	1.5%	0.2%	0.0%

Outline

- Introduction
- Context Free Grammar
  - Some Grammar Rules
  - Syntactic Parsing
- Probabilistic Context Free Grammar
  - Statistical Parsing
- **Lexicalized PCFG**
- Partial Parsing
- Evaluation

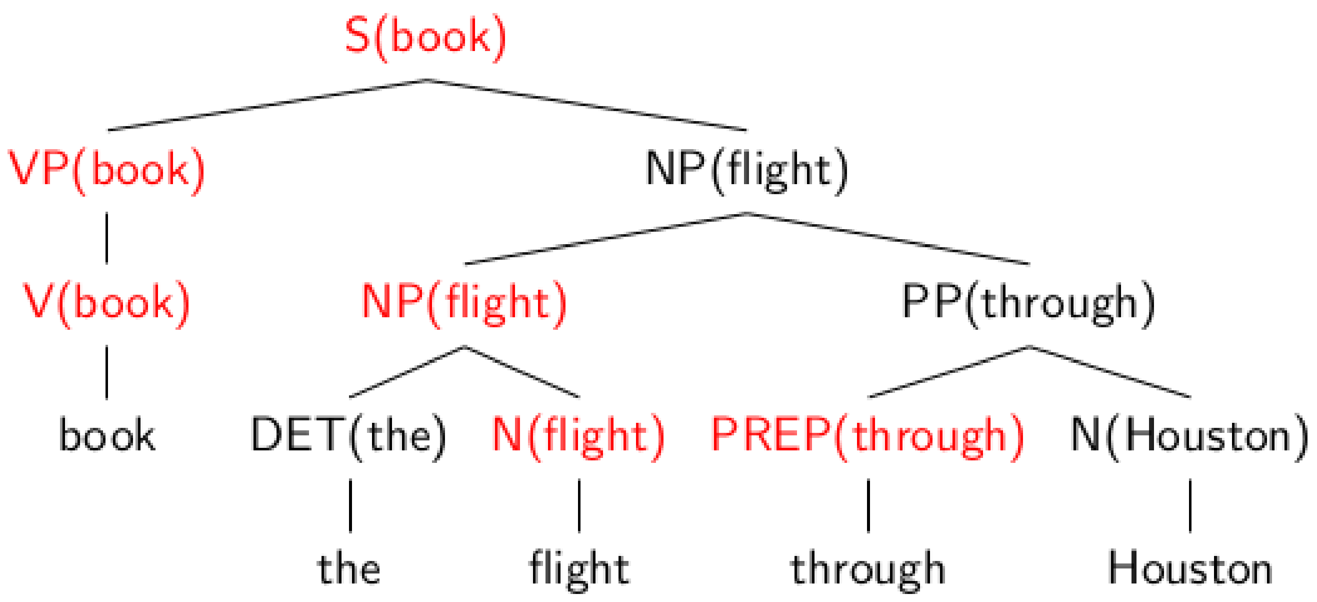
Headed CFG

- Each context-free rule has one special child that is the head of the rule

S → NP VP
S → VP
NP → N
NP → Det N
NP → NP NP
NP → NP PP
VP → V
VP → VP PP
VP → VP NP
PP → Prep NP

Lexicalization of a Tree

- Trees with headwords
  - Assign a head word to each constituent and transfer it to the parents
  - Each constituent received its headword from its head child



Lexicalized PCFG

- Grammar G consists of
  - Terminals (T)
  - Non-terminals (N)
  - Start symbol (S)
  - Rules (R)

- Rules in CFG are written as follows:
  - $N \rightarrow N_1\ N_2$
  - $N \rightarrow T$

Lexicalized PCFG

- Grammar G consists of
  - Terminals (T)
  - Non-terminals (N)
  - Start symbol (S)
  - Rules (R)

- Rules in CFG are written as follows:
  - $N \rightarrow N_1\ N_2$
  - $N \rightarrow T$

- Rules in lexicalized PCFG are written as follows:
  - $N(X) \rightarrow N_1(X)\ N_2(Y)$
  - $N(Y) \rightarrow N_1(X)\ N_2(Y)$
  - $N(T) \rightarrow T$

Lexicalized PCFG

S → NP VP

S → VP

NP → N

NP(flight) → Det(the) N(flight)

NP → NP NP

NP(flight) → NP(flight) PP(through)

VP(book) → V(book)

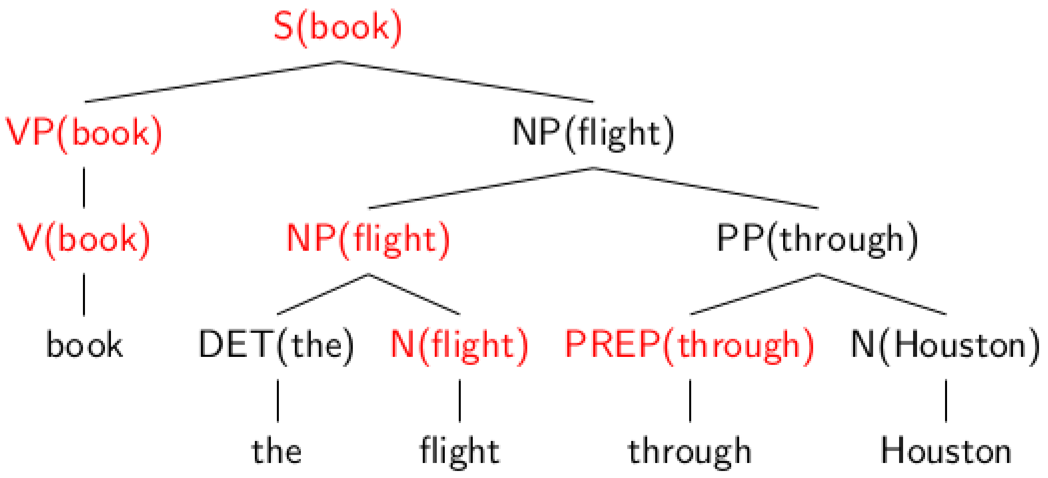
VP → VP PP

VP → VP(book) NP(flight)

PP(through) → Prep(through) NP(Hauston)

Parameter Estimation in LPCFG

S(book) → VP(book) NP(flight)



$P(S \rightarrow VP\ NP)$

$P(S(book) \rightarrow_1 VP(book)\ NP(flight))$

Lexicalized PCFG

- Number of rules increases dramatically
- # rules
  - In CFG: $O(|N|^3)$
  - In LPCFG: $O(|\Sigma|^2 \times |N|^3)$
- Running time for parsing an n words sentence
  - In CFG: $O(n^3|N|^3)$
  - In LPCFG: $O(n^3|\Sigma|^2|N|^3)$

Lexicalized PCFG

- Number of rules increases dramatically
- # rules
  - In CFG: $O(|N|^3)$
  - In LPCFG: $O(|\Sigma|^2 \times |N|^3)$
- Running time for parsing an n words sentence
  - In CFG: $O(n^3|N|^3)$
  - In LPCFG: $O(n^3|\Sigma|^2|N|^3)$
- But:

  Considering the observed words in the sentence, the number of rules will be reduced to $O(n^2|N|^3)$ for parsing each sentence

  ⇨running time of LPCFG: $O(n^5|N|^3)$

Outline

- Introduction
- Context Free Grammar
  - Some Grammar Rules
  - Syntactic Parsing
- Probabilistic Context Free Grammar
  - Statistical Parsing
- Lexicalized PCFG
- **Partial Parsing**
- Evaluation

Partial Parsing

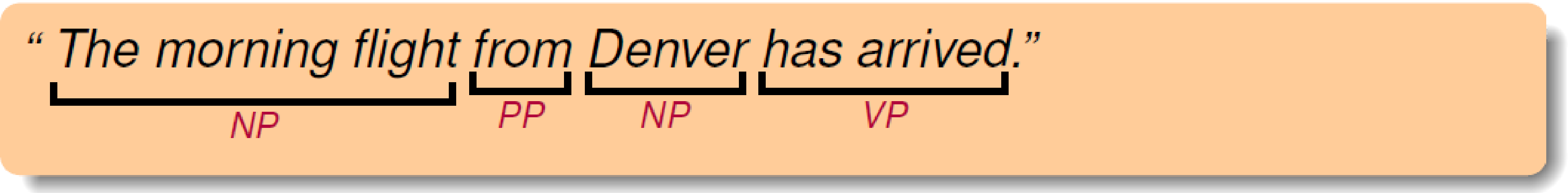
- Many NLP tasks do not require complex, complete parse trees for all inputs.
- For these tasks, a partial parse (shallow parse) of input sentences may be sufficient.

- Example:
  - Information extraction do not extract all the possible information from a text: they simply identify and classify the segments in a text that are likely to contain valuable information.

- Possible approaches:
  - Using cascades of finite state transducers to produce tree-like representations.
    - Produce flatter trees than complete pars trees
  - Chunking

Chunking

- The process of identifying and classifying the flat, non-overlapping segments of a sentence that constitute the basic non-recursive phrases corresponding to the major content-word parts-of-speech:
  - Noun phrases
  - Verb phrases
  - Adjective phrases
  - Prepositional phrase
- Since chunked texts lack a hierarchical structure, a simple bracketing notation is sufficient to denote the location and the type of the chunks



Chunking

- Two fundamental tasks are involved in chunking:
  - Segmenting (finding the non-overlapping extents of the chunks)
  - Labeling (assigning the correct tag to the discovered chunks)
- Some input words may not be part of any chunk

Chunking

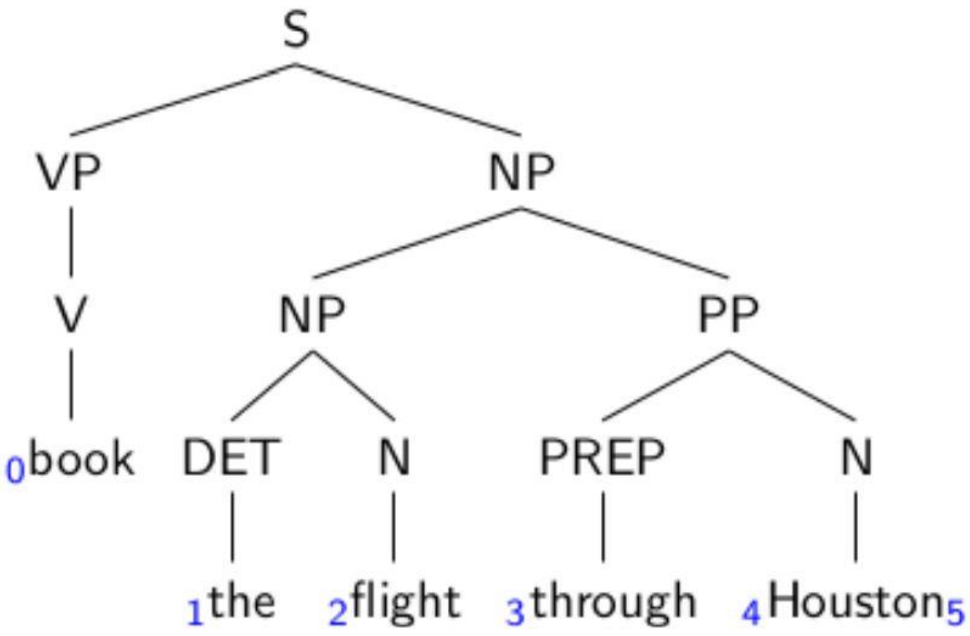
- Main approaches:
  - Rule-based models
  - Sequence modeling) very similar to the NER approaches
- Evaluation:
  - Precision
  - Recall
  - F-measure

Outline

- Introduction
- Context Free Grammar
  - Some Grammar Rules
  - Syntactic Parsing
- Probabilistic Context Free Grammar
  - Statistical Parsing
- Lexicalized PCFG
- Partial Parsing
- **Evaluation**

Constituents in Parse Tree

Label	Start	End
S	0	5
VP	0	1
NP	1	5
NP	1	3
PP	3	5



Precision/Recall Evaluation

Gold:

Label	Start	End
S	0	5
VP	0	1
NP	1	5
NP	1	3
PP	3	5

Prediction:

Label	Start	End
S	0	5
VP	0	3
NP	1	3
PP	3	5

$$\text{Labeled Precision} = \frac{3}{4} = 0.75$$

$$\text{Labeled Recall} = \frac{3}{5} = 0.6$$

$$\mathrm{F}_1 = \frac{2*\mathrm{P}*\mathrm{R}}{\mathrm{P}+\mathrm{R}} = 0.67$$

Precision/Recall Evaluation

- Labeled precision and labeled recall consider the beginning and end of brackets as well as the label of detected constituencies
- Unlabeled precision and unlabeled recall only match the beginning and end of brackets regardless of their label

Results

- Available results on Penn treebank (Wall Street Journal)
  - # train sentences: 40,000
  - # test sentences: 2,400

  - PCFG: 70.6% Recall & 74.8% Precision
  - LPCFG: 88.1% Recall & 88.3% Precision

  - More results:
    - (Charniak & Johnson 2005): 91.2% Recall & 91.8% Precision
    - (Carreras et al. 2008): 90.7% Recall & 91.4% Precision
    - (Petrov 2010): 91.7% Recall & 92.0% Precision

Further Reading

- Speech and Language Processing (3rd ed. draft)
  - Chapters 17 + Appendix C & D