CS689: Computational Linguistics for Indian Languages
Parsing

Arnab Bhattacharya
arnabb@cse.iitk.ac.in

Computer Science and Engineering,
Indian Institute of Technology, Kanpur
http://web.cse.iitk.ac.in/~cs689/

2nd semester, 2023-24

Tue 10:30–11:45, Thu 12:00–13:15 at RM101/KD102

Parsing

- Parsing is finding *structures* and *relationships* in a sentence
- Two main types
- Constituency Parsing
  - Breaks a sentence into an *ordered* set of *constituents*
  - Uses a grammar
  - Produces a *constituent parse tree*
  - Useful for rigid-order languages
- Dependency Parsing
  - Finds *directed relationships* between pairs of words
  - *Head* word of a sentence
  - Produces a *dependency parse tree*
  - Useful for free-order languages

Grammar

- Constituency parsing uses parts-of-speech (POS) tags
- POS tags with a certain *ordered combination* form a phrase
  - Noun phrase, Verb phrase, Prepositional phrase
- *Order* of phrases is dictated by a grammar
- Constituency parsing uses context-free grammars (CFG)
  - Also called phrase-structure grammars

Context-Free Grammar (CFG)

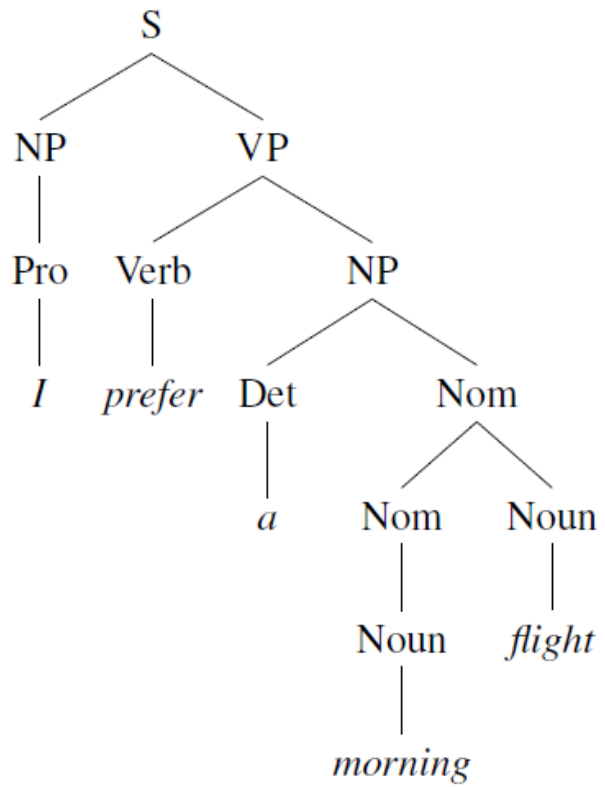
- A CFG consists of
  - A set of non-terminals, denoted by N (capital letters)
  - A set of terminals, denoted by Σ (small letters)
  - A set of rules of the form $A \rightarrow B$ where A is a non-terminal, and B is any string composed of any number of terminals and non-terminals
  - A start non-terminal, denoted by S

$$
\begin{aligned}
S &\rightarrow NP\ VP \\
NP &\rightarrow Pronoun \\
&\mid Proper\text{-}Noun \\
&\mid Det\ Nominal \\
Nominal &\rightarrow Nominal\ Noun \\
&\mid Noun \\
VP &\rightarrow Verb \\
&\mid Verb\ NP \\
&\mid Verb\ NP\ PP \\
&\mid Verb\ PP \\
PP &\rightarrow Preposition\ NP
\end{aligned}
$$

$$
\begin{aligned}
Noun &\rightarrow flights \mid flight \mid breeze \mid trip \mid morning \\
Verb &\rightarrow is \mid prefer \mid like \mid need \mid want \mid fly \mid do \\
Adjective &\rightarrow cheapest \mid non\text{-}stop \mid first \mid latest \\
&\mid other \mid direct \\
Pronoun &\rightarrow me \mid I \mid you \mid it \\
Proper\text{-}Noun &\rightarrow Alaska \mid Baltimore \mid Los\ Angeles \\
&\mid Chicago \mid United \mid American \\
Determiner &\rightarrow the \mid a \mid an \mid this \mid these \mid that \\
Preposition &\rightarrow from \mid to \mid on \mid near \mid in \\
Conjunction &\rightarrow and \mid or \mid but
\end{aligned}
$$

Parsing using a CFG

- Why CFG?
- Expressive enough
- Computationally feasible to handle
- Parsing using a CFG produces a constituent parse tree, sometimes called a parse tree or simply a parse
- "I prefer a morning flight" has the parse tree



Representation of a Parse Tree

- Bracket flat (linear) notation

$$[_S\ [_{NP}\ [_{Pro}\ \text{I}]]\ [_{VP}\ [_V\ \text{prefer}]\ [_{NP}\ [_{Det}\ \text{a}]\ [_{Nom}\ [_N\ \text{morning}]\ [_{Nom}\ [_N\ \text{flight}]]]]]]$$

- Bracket tree (hierarchical) notation

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

```
((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 )))))
```

How to Write a Grammar?

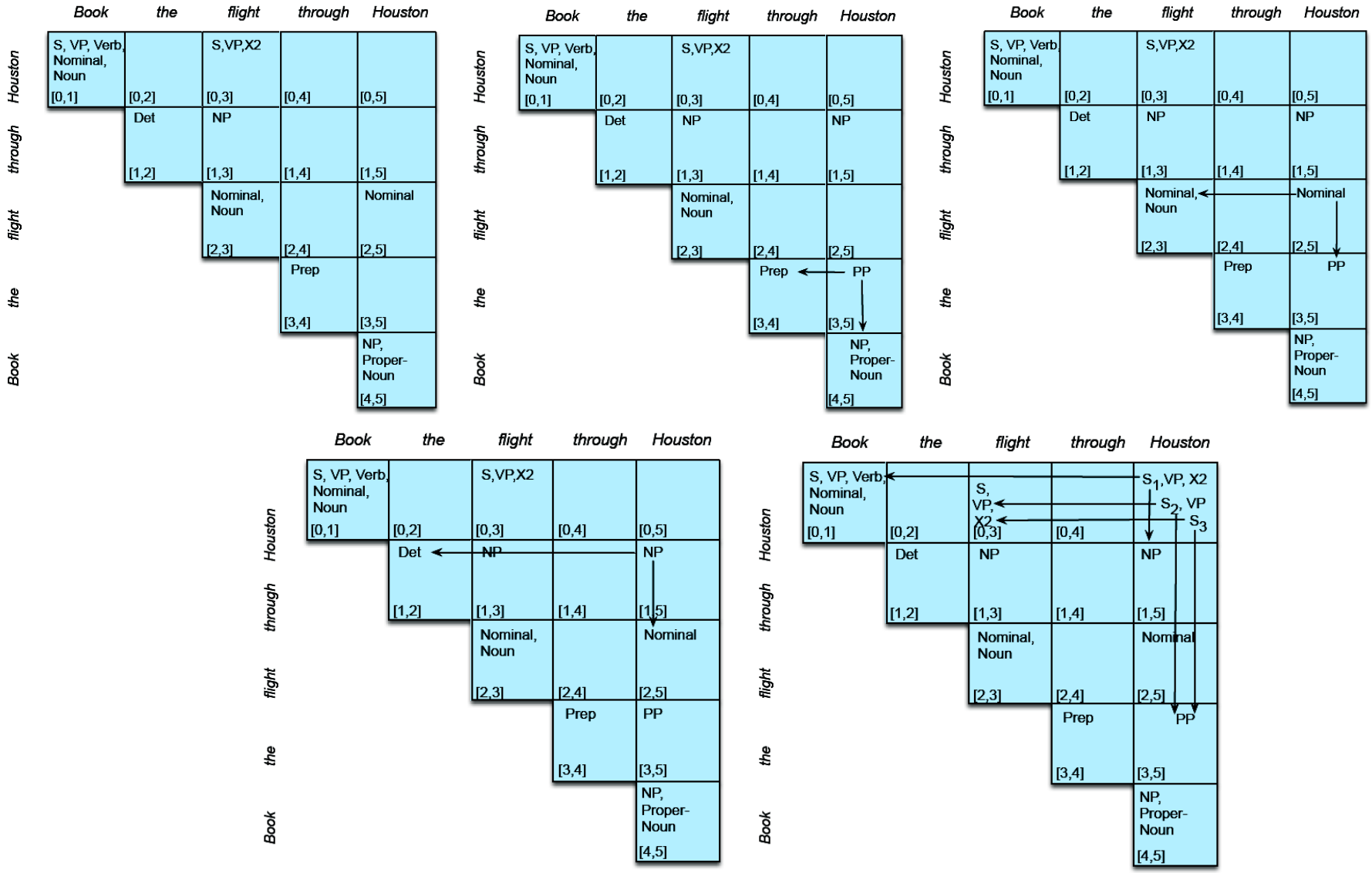
- Rules of the language
- *Learned* grammar from an *annotated corpus*
- Parse tree annotated corpus is called treebank
  - Penn Treebank for English
  - Universal Dependencies (UD) Treebank for multiple languages
- Rule derived from *every* sentence is a grammar rule
- *No* other rule is a grammar rule
- *Every* word is part of the lexicon for the corresponding POS tag
- *No* other word is part of the lexicon for a POS tag

Algorithm

- *Dynamic programming* solution: Cocke-Kasami-Younger (CKY) algorithm
- Converts any CFG into Chomsky Normal Form
  - Rules are either $A \rightarrow BC$ or $A \rightarrow a$
  - A non-terminal expands to either *exactly two* non-terminals or a *single* terminal
- Sentence of length n broken into $n+1$ *gaps* or fenceposts
- For a cell (i, j) in DP, consider all possible rules
  - Either, it is a terminal: single word, i.e., $i = j - 1$
  - Or, consider all k such that $i < k < j$ and $w_i \ldots w_k$ and $w_k \ldots w_j$ is a valid parse
- Finally, cell $(0, n)$ gives *all* the parses

Example

- "Book the flight through Houston"



Evaluating a Parse

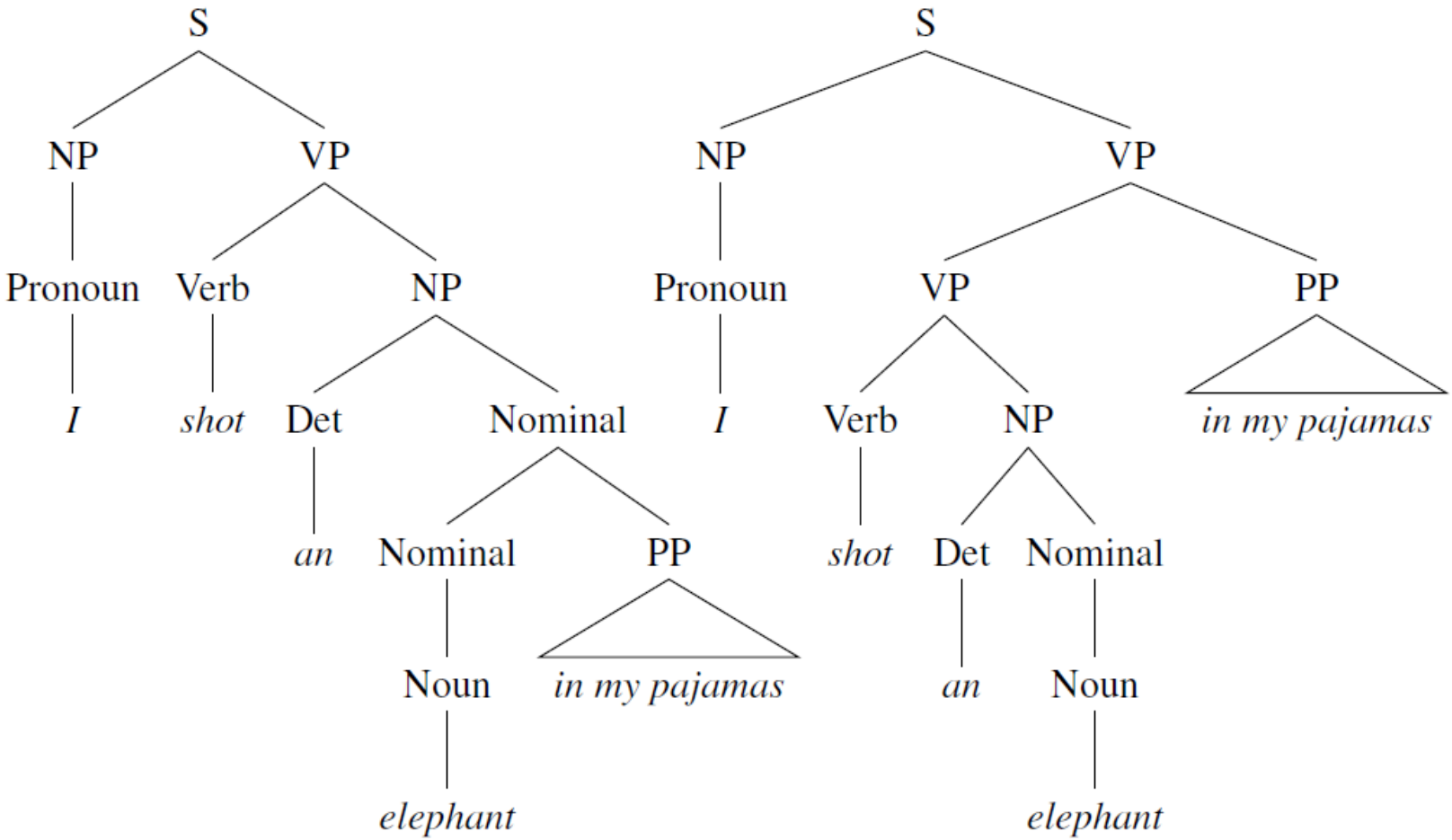
- How to evaluate a parse tree?
- Consider a parse tree generated by an algorithm A and a golden truth parse tree T
- Find the set of *constituents* C_A in A and C_T in T
  - A constituent is a non-terminal along with the starting and ending word positions
  - The cardinality of the two sets may not be the same
- Find set of common constituents $C_c = C_A \cap C_T$
- Metrics of evaluation

$$\text{Precision } P = \frac{|C_c|}{|C_A|}; \qquad \text{Recall } R = \frac{|C_c|}{|C_T|};$$

$$\text{F-score } F = \text{Harmonic Mean} = \frac{2 \cdot P \cdot R}{P + R} = \frac{2|C_c|}{|C_A| + |C_T|}$$

Ambiguity in Parsing

- Multiple parse trees can be possible
- Ambiguity in sentence meaning
- "One morning I shot an elephant in my pajamas"



Problems with Constituency Parsing

- Assumes a rather rigid sentence structure
- Phrases are less frequent in inflectional languages
  - They become inflected words or compound words or `samasta-pada`
  - अहं ज्ञातुम् इच्छामि becomes अहं जिज्ञासुः
- Can produce multiple valid parse trees
- Ways to choose a parse tree over other
  - Weights of a rule
  - Neural-network based
  - Probability-based: rules have probabilities, leading to probabilistic CFGs
  - Statistical constituency parsing

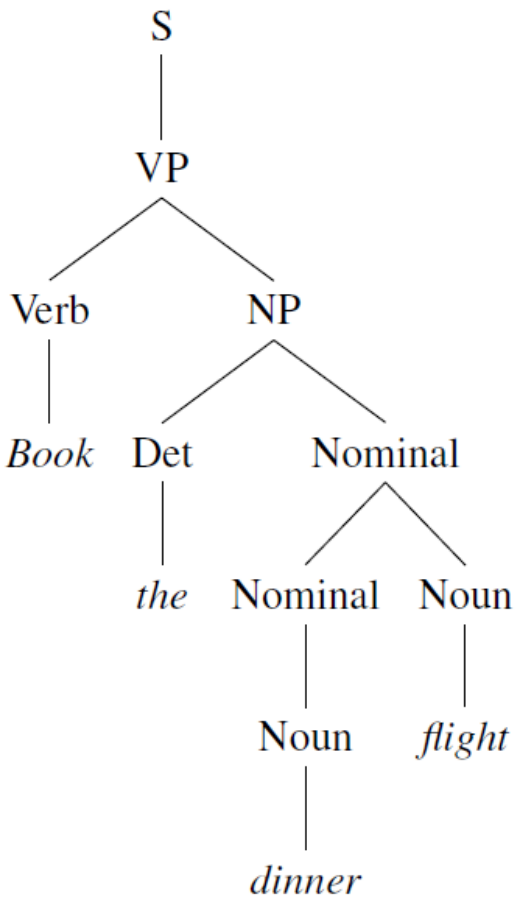
Probabilistic Context-Free Grammar

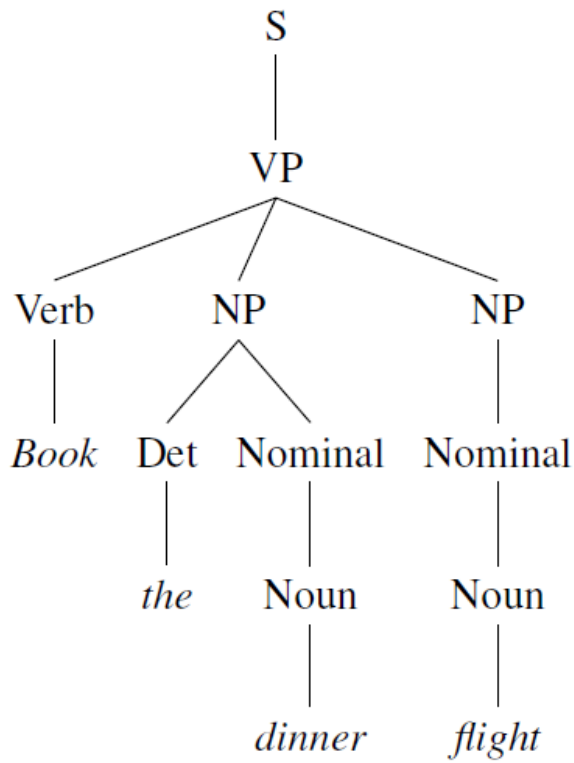
- In a probabilistic CFG, rules have probabilities
- All possible expansions of a rule add up to 1

Grammar		Lexicon
$S \rightarrow NP\ VP$	[.80]	$Det \rightarrow that\ [.10] \mid a\ [.30] \mid the\ [.60]$
$S \rightarrow Aux\ NP\ VP$	[.15]	$Noun \rightarrow book\ [.10] \mid trip\ [.30]$
$S \rightarrow VP$	[.05]	$\mid meal\ [.05] \mid money\ [.05]$
$NP \rightarrow Pronoun$	[.35]	$\mid flight\ [.40] \mid dinner\ [.10]$
$NP \rightarrow Proper\text{-}Noun$	[.30]	$Verb \rightarrow book\ [.30] \mid include\ [.30]$
$NP \rightarrow Det\ Nominal$	[.20]	$\mid prefer\ [.40]$
$NP \rightarrow Nominal$	[.15]	$Pronoun \rightarrow I\ [.40] \mid she\ [.05]$
$Nominal \rightarrow Noun$	[.75]	$\mid me\ [.15] \mid you\ [.40]$
$Nominal \rightarrow Nominal\ Noun$	[.20]	$Proper\text{-}Noun \rightarrow Houston\ [.60]$
$Nominal \rightarrow Nominal\ PP$	[.05]	$\mid NWA\ [.40]$
$VP \rightarrow Verb$	[.35]	$Aux \rightarrow does\ [.60] \mid can\ [.40]$
$VP \rightarrow Verb\ NP$	[.20]	$Preposition \rightarrow from\ [.30] \mid to\ [.30]$
$VP \rightarrow Verb\ NP\ PP$	[.10]	$\mid on\ [.20] \mid near\ [.15]$
$VP \rightarrow Verb\ PP$	[.15]	$\mid through\ [.05]$
$VP \rightarrow Verb\ NP\ NP$	[.05]	
$VP \rightarrow VP\ PP$	[.15]	
$PP \rightarrow Preposition\ NP$	[1.0]	

Choosing a Parse

- The parse tree with the *highest joint probability* is the *best* parse
- "Book the dinner flight"





$$P(T_{left}) = .05 * .20 * .20 * .20 * .75 * .30 * .60 * .10 * .40 = \mathbf{2.2 \times 10^{-6}}$$

$$P(T_{right}) = .05 * .10 * .20 * .15 * .75 * .75 * .30 * .60 * .10 * .40 = \mathbf{6.1 \times 10^{-7}}$$