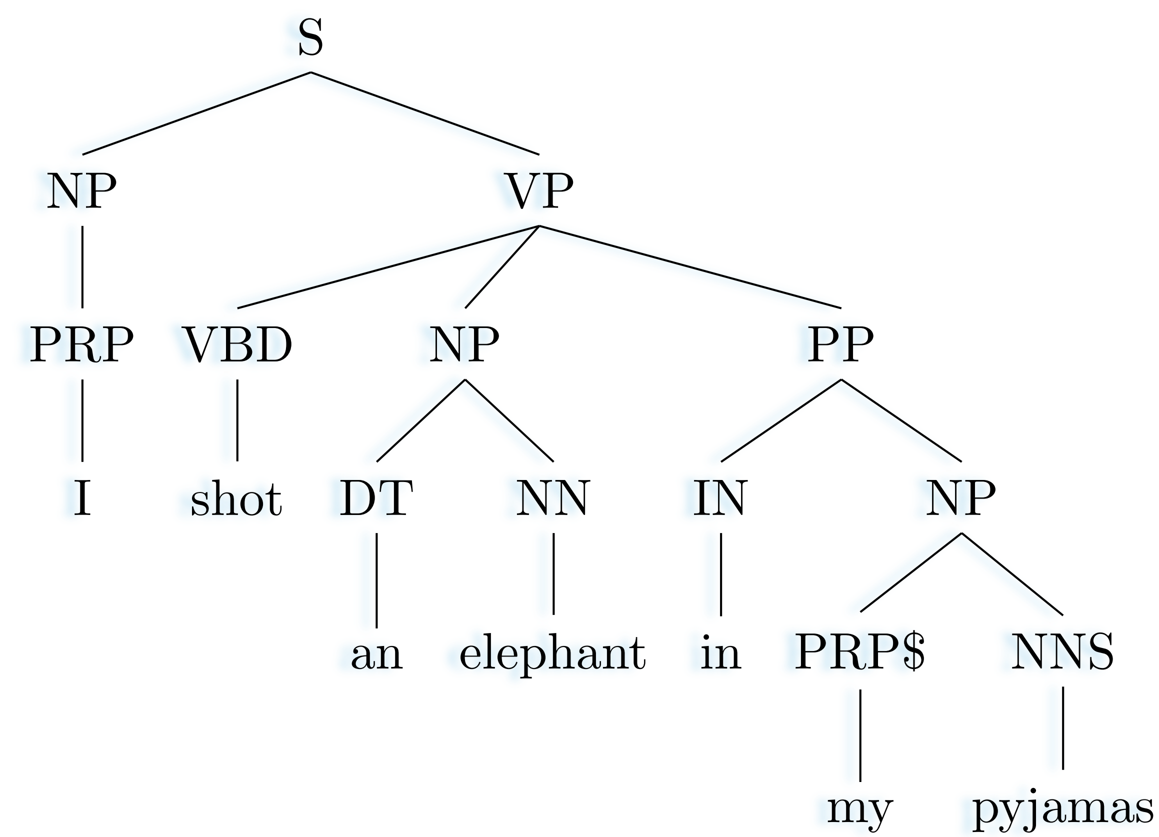
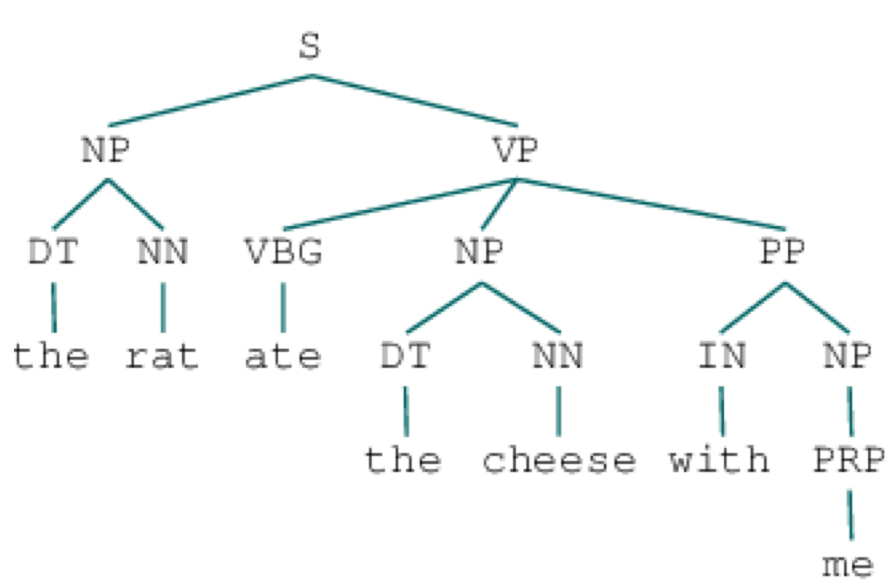


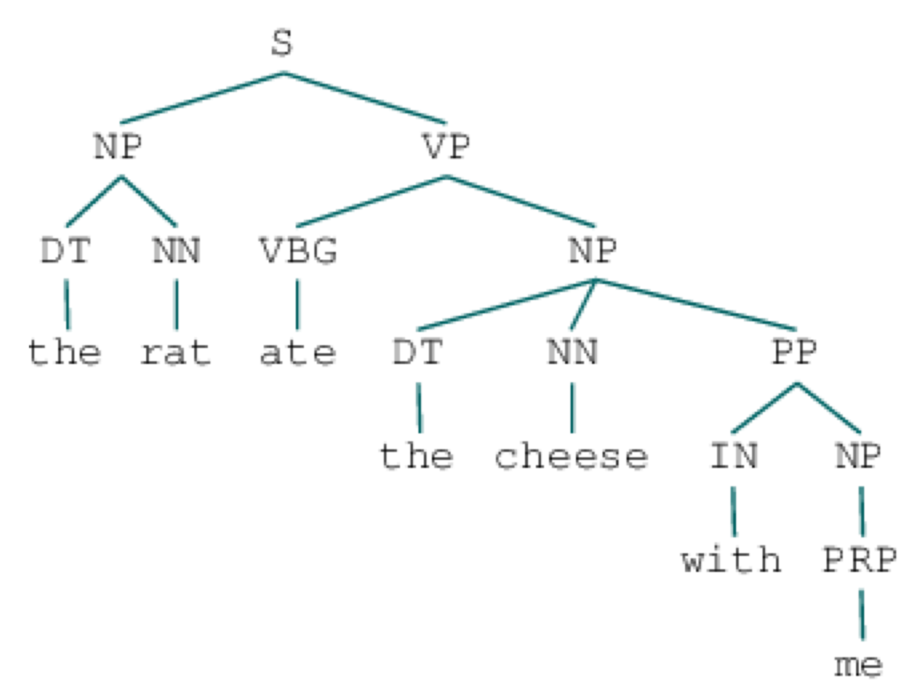
COMP90042 LECTURE 9

PROBABILISTIC PARSING

AMBIGUITY IN PARSING

- Context-free grammars assign hierarchical structure to language
  - Linguistic notion of a '*syntactic constituent*'
  - Formulated as generating all strings in the language; or
  - Predicting the structure(s) for a given string
- Raises problem of ambiguity, e.g., which is better?





OUTLINE

- Probabilistic context-free grammars (PCFGs)
- Parsing using dynamic programming
- Limitations of ‘context-free’ assumption and some solutions:
  - parent annotation
  - head lexicalisation

BASICS OF PROBABILISTIC CFGS

- As for CFGs, same symbol set:
  - Terminals: words such as *book*
  - Non-terminal: syntactic labels such as NP or NN
- Same productions (rules)
  - LHS non-terminal $\rightarrow$ ordered list of RHS symbols
- In addition, store a **probability** with each production
  - NP $\rightarrow$ DT NN [p = 0.45]
  - NN $\rightarrow$ cat [p = 0.02]
  - NN $\rightarrow$ leprechaun [p = 0.00001]
  - …

PROBABILISTIC CFGS

- Probability values denote *conditional*
  - Pr(RHS | LHS)
- Consequently they:
  - must be positive values, between 0 and 1
  - must sum to one for given LHS
- E.g.,
  - NN $\rightarrow$ aadvark [p = 0.0003]
  - NN $\rightarrow$ leprechaun [p = 0.0001]
  - NN $\rightarrow$ Zanzibar [p = 0.0025]
  - $\sum_x \Pr(\text{NN} \rightarrow x \mid \text{NN}) = 1$

A PROBABILISTIC GRAMMAR

Grammar		Lexicon
$S \rightarrow NP\ VP$	[.80]	$Det \rightarrow that$ [.10] | a [.30] | the [.60]
$S \rightarrow Aux\ NP\ VP$	[.15]	$Noun \rightarrow book$ [.10] | ~~*flight*~~ [.30]
$S \rightarrow VP$	[.05]	| $meal$ [.015] | $money$ [.05]
$NP \rightarrow Pronoun$	[.35]	| $flight$ [.40] | $dinner$ [.10]
$NP \rightarrow Proper\text{-}Noun$	[.30]	$Verb \rightarrow book$ [.30] | $include$ [.30]
$NP \rightarrow Det\ Nominal$	[.20]	| $prefer$ [.40]
$NP \rightarrow Nominal$	[.15]	$Pronoun \rightarrow I$ [.40] | she [.05]
$Nominal \rightarrow Noun$	[.75]	| me [.15] | you [.40]
$Nominal \rightarrow Nominal\ Noun$	[.20]	$Proper\text{-}Noun \rightarrow Houston$ [.60]
$Nominal \rightarrow Nominal\ PP$	[.05]	| NWA [.40]
$VP \rightarrow Verb$	[.35]	$Aux \rightarrow does$ [.60] | can [40]
$VP \rightarrow Verb\ NP$	[.20]	$Preposition \rightarrow from$ [.30] | to [.30]
$VP \rightarrow Verb\ NP\ PP$	[.10]	| on [.20] | $near$ [.15]
$VP \rightarrow Verb\ PP$	[.15]	| $through$ [.05]
$VP \rightarrow Verb\ NP\ NP$	[.05]	
$VP \rightarrow VP\ PP$	[.15]	
$PP \rightarrow Preposition\ NP$	[1.0]	

Source JM3, Fig 13.1

STOCHASTIC GENERATION WITH PCFGS

Déjà vu, it's almost the same as for CFG, with one twist:

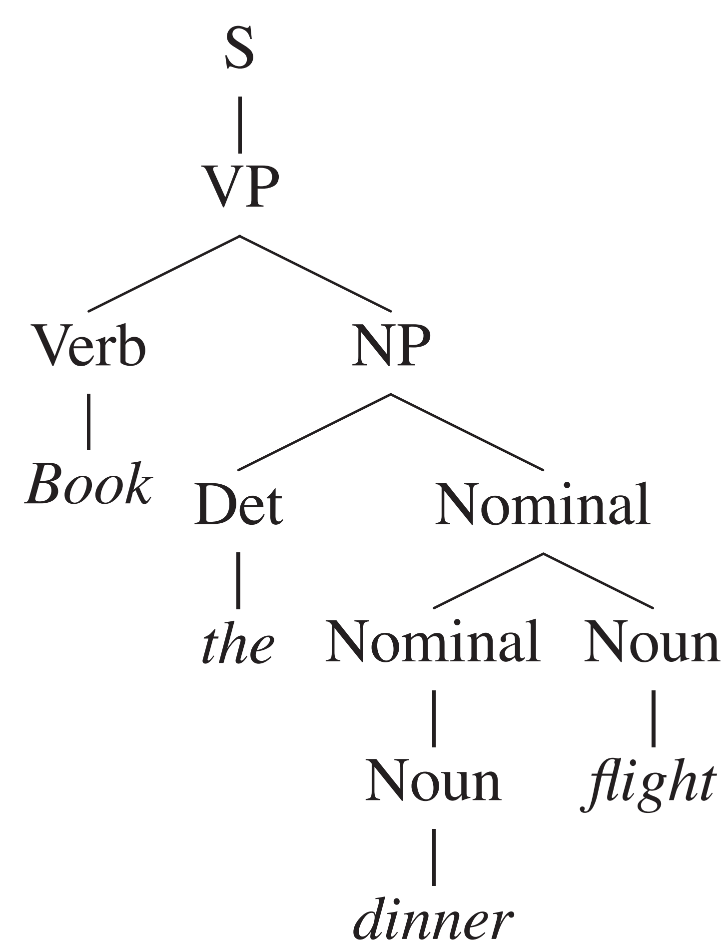
1. Start with S, the sentence symbol
2. Choose a rule with S as the LHS
   - **Randomly select a RHS** according to Pr(RHS | LHS) e.g., $S \rightarrow VP$
   - Apply this rule, e.g., substitute VP for S
3. Repeat step 2 for each non-terminal in the string (here, VP)
4. Stop when no non-terminals remain

Gives us a tree, as before, with a sentence as the yield

HOW LIKELY IS A TREE?

- Given a tree, we can compute its probability
  - Decomposes into probability of each production

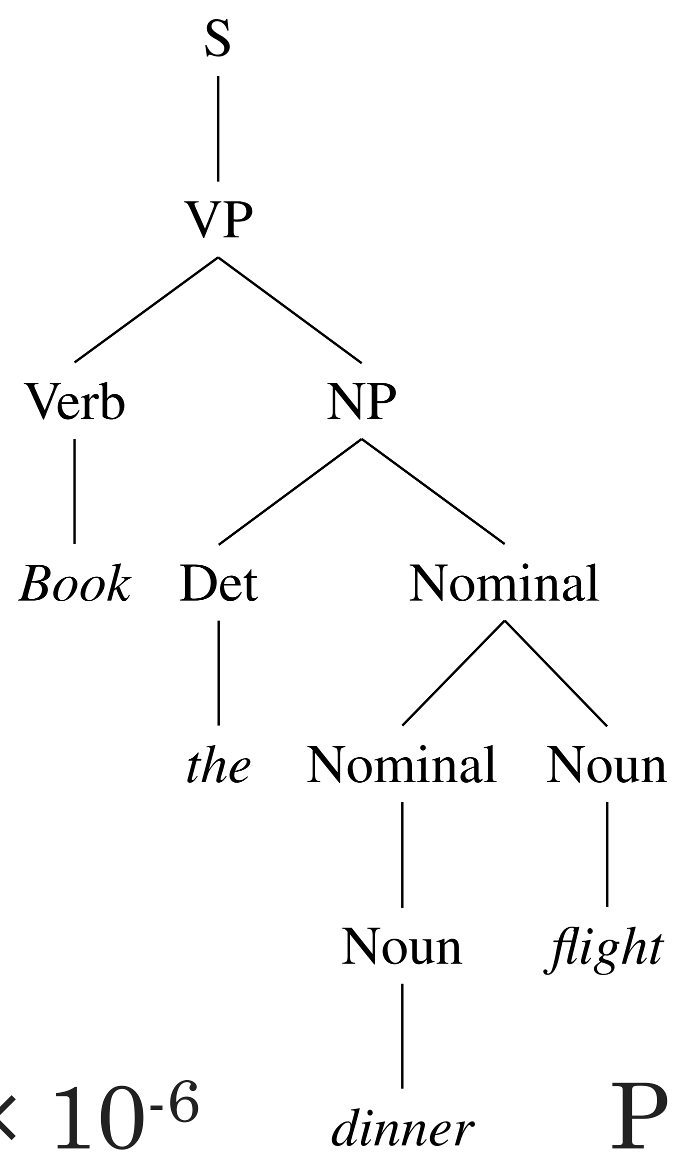
$$P(T|S) = \prod_{i=1}^{n} P(RHS_i|LHS_i)$$

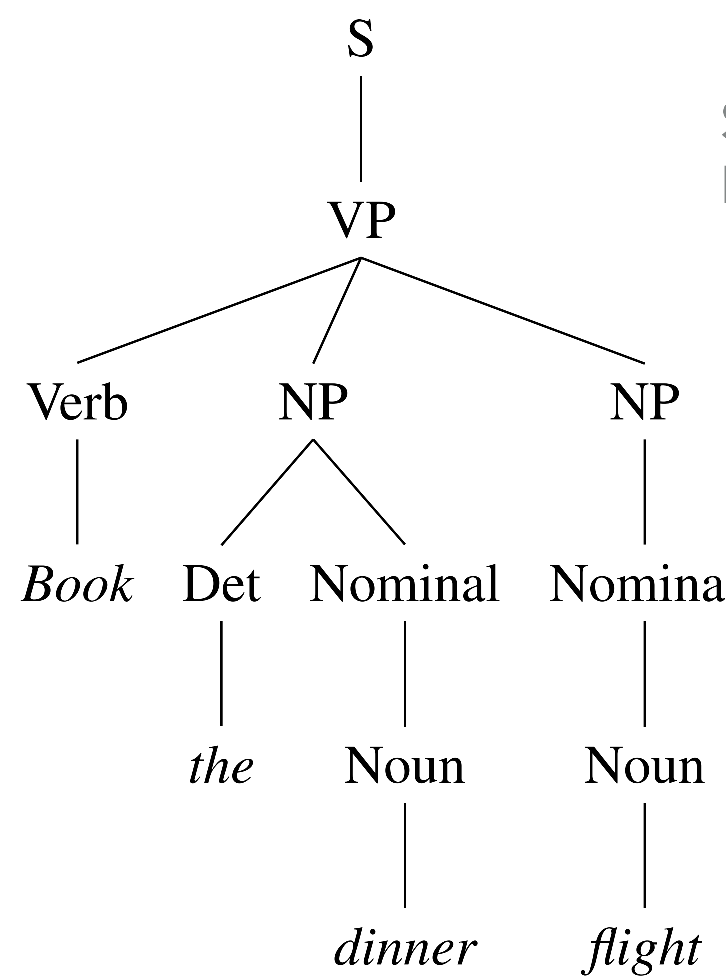


- E.g., for tree on right,
  - P(T | S) = P(S → VP) ×
    P(VP → Verb NP) ×
    P(Verb → *Book*) ×
    P(NP → Det Nominal) ×
    P(Det → *the*) ×
    P(Nominal → Nominal Noun) ×
    P(Noun → *dinner*) ×
    P(Noun → *flight*) = 2.2 × 10^{-6}

RESOLVING PARSE AMBIGUITY

- Can select between different trees based on P(T)





Source: JM3 Fig 13.2

- $P = 2.16 \times 10^{-6}$ $P = 3.04 \times 10^{-7}$

Mistakes in textbook

- Note that some structures are the same ($S \rightarrow VP$, $Verb \rightarrow$ *Book*…)

PARSING PCFGS

- Instead of selecting between two trees, can we select a tree from the set of all possible trees?
- Before we looked at
  - CYK and Early
  - for unweighted grammars (CFGs)
  - finds **all possible trees**
- But there are often 1000s, many completely nonsensical
- Can we solve for the **most probable tree**?

$$T = \text{argmax}_{T \text{ s.t. } \text{yield}(T)=\mathbf{w}} P(T|S)$$

CYK FOR PCFGS

- CYK finds *all trees* for a sentence; we want best tree
- Prob. CYK follows similar process to standard CYK
- Convert grammar to Chomsky Normal Form (CNF)
  - E.g., $\quad \text{VP} \rightarrow \text{Verb NP NP} \quad [0.05]$

    becomes

    $$\text{VP} \rightarrow \text{Verb X} \quad [??]$$
    $$\text{X} \rightarrow \text{NP NP} \quad [??]$$

    where X is a new symbol.
  - But what happens to the probability?
- Issues with unary productions *(see ipython notebook)*

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

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

Figure 12.5 The CKY algorithm.

function PROBABILISTIC-CKY(*words,grammar*) **returns** most probable parse and its probability

for $j \leftarrow$ **from** 1 **to** LENGTH(*words*) **do**
  for all $\{ A \mid A \rightarrow words[j] \in grammar\}$
    $table[j-1, j, A] \leftarrow P(A \rightarrow words[j])$
  for $i \leftarrow$ **from** $j-2$ **downto** 0 **do**
    for $k \leftarrow i+1$ **to** $j-1$ **do**
      for all $\{ A \mid A \rightarrow BC \in grammar,$ **and** $table[i,k,B] > 0$ **and** $table[k,j,C] > 0 \}$
        if $(table[i,j,A] < P(A \rightarrow BC) \times table[i,k,B] \times table[k,j,C])$ **then**
          $table[i,j,A] \leftarrow P(A \rightarrow BC) \times table[i,k,B] \times table[k,j,C]$
          $back[i,j,A] \leftarrow \{k, B, C\}$
return BUILD_TREE(*back*[1, LENGTH(*words*), *S*]), *table*[1, LENGTH(*words*), *S*]

Figure 13.3 The probabilistic CKY algorithm for finding the maximum probability parse

Source: JM3 Ch 12,13

ILLUSTRATION

- Insert preterminal productions of type POS $\rightarrow$ *word*

Book	*the*	*dinner*	*flight*
Verb [0.3] Noun [0.1] VP [0.105] Nominal [0.075] NP [0.01125] S [0.00525] [0,1]	[0,2]	[0,3]	[0,4]
	Det [0.6] [1,2]	[1,3]	[1,4]
		Noun [0.1] Nominal [0.075] NP [0.075] [2,3]	[2,4]
			Noun [0.3] Nominal [0.225] NP [0.03375] [3,4]

Verb $\rightarrow$ *book* [0.3]
Noun $\rightarrow$ *book* [0.1]
Noun $\rightarrow$ *dinner* [0.1]
Noun $\rightarrow$ *flight* [0.3]

VP $\rightarrow$ Verb [0.35]
S $\rightarrow$ VP [0.05]
Nominal $\rightarrow$ Noun [0.75]
NP $\rightarrow$ Nominal [0.15]

ILLUSTRATION

NP → Det Nominal [0.20]
score = 0.6 x 0.075 x 0.2
= 0.09

Book	*the*	*dinner*	*flight*
Verb [0.3] Noun [0.1] VP [0.105] Nominal [0.075] NP [0.01125] S [0.00525] [0,1]	[0,2]	[0,3]	[0,4]
	Det [0.6] [1,2]	NP [0.09] [1,3]	[1,4]
		Noun [0.1] **Nominal [0.075]** NP [0.01125] [2,3]	[2,4]
			Noun [0.3] Nominal [0.225] NP [0.03375] [3,4]

ILLUSTRATION

Nominal → Nominal Noun [0.20]

score = 0.075 x 0.3 x 0.2

= 0.0045

Book	*the*	*dinner*	*flight*
Verb [0.3] Noun [0.1] VP [0.105] Nominal [0.075] NP [0.01125] S [0.00525] [0,1]	[0,2]	[0,3]	[0,4]
	Det [0.6] [1,2]	NP [0.09] [1,3]	[1,4]
		Noun [0.1] **Nominal [0.075]** NP [0.01125] [2,3]	Nominal [0.0045] [2,4]
			Noun [0.3] Nominal [0.225] NP [0.03375] [3,4]

ILLUSTRATION

NP → Det Nominal [0.20]
score = 0.6 x 0.0045 x 0.2
= 0.00054

Book	*the*	*dinner*	*flight*
Verb [0.3] Noun [0.1] VP [0.105] Nominal [0.075] NP [0.01125] S [0.00525] [0,1]	[0,2]	[0,3]	[0,4]
	Det [0.6] [1,2]	NP [0.09] [1,3]	NP [0.00054] [1,4]
		Noun [0.1] Nominal [0.075] NP [0.01125] [2,3]	**Nominal [0.0045]** [2,4]
			Noun [0.3] Nominal [0.225] NP [0.03375] [3,4]

ILLUSTRATION

VP → Verb NP [0.20]

score = 0.3 x 0.00054 x 0.2

= 0.000032

Book	*the*	*dinner*	*flight*
Verb [0.3] Noun [0.1] VP [0.105] Nominal [0.075] NP [0.01125] S [0.00525] [0,1]	[0,2]	[0,3]	**VP [0.000032]** [0,4]
	Det [0.6] [1,2]	NP [0.09] [1,3]	**NP [0.00054]** [1,4]
		Noun [0.1] Nominal [0.075] NP [0.01125] [2,3]	**Nominal [0.0045]** [2,4]
			Noun [0.3] Nominal [0.225] NP [0.03375] [3,4]

ILLUSTRATION: COMPETING ANALYSIS

- Length j = 4

X → NP NP [1]
VP → Verb X [0.05]

Book	*the*	*dinner*	*flight*
Verb [0.3] Noun [0.1] VP [0.105] Nominal [0.075] NP [0.01125] S [0.00525] [0,1]	[0,2]	[0,3]	**VP** **[0.00015]** [0,4]
	Det [0.6] [1,2]	**NP** **[0.09]** [1,3]	NP [0.00054] X [0.003] [1,4]
		Noun [0.1] Nominal [0.075] NP [0.01125] [2,3]	Nominal [0.0045] [2,4]
			Noun [0.3] Nominal [0.225] **NP** **[0.03375]** [3,4]

outscores existing analysis for [0,4; VP]

PROB CYK: RETRIEVING THE PARSES

- S in the top-right corner of parse table indicates success
- Retain back-pointer to best analysis
  - for each chart cell, store the split point and the non-terminal for the left and right children
- To get parse(s), follow pointers back for each match
- Convert back from CNF by removing new non-terminals

COMPLEXITY OF CYK

- What's the space and time complexity of this algorithm?
  - in terms of n the length of the input sentence

PROBLEMS WITH (P)CFGS

- **poor independence assumptions:** rewrite decisions made independently, whereas inter-dependence is often needed to capture global structure.
  - E.g., NP $\rightarrow$ PRP used often as subject (first NP), much less often as object (second NP)
- **lack of lexical conditioning:** non-terminals representation behaviour of the actual words, but are much too coarse. Problems with
  - preposition attachment ambiguity;
  - subcategorisation ([*forgot NP*] vs [*forgot S*]);
  - coordinate structure ambiguities (*dogs in houses and cats)*

PP ATTACHMENT

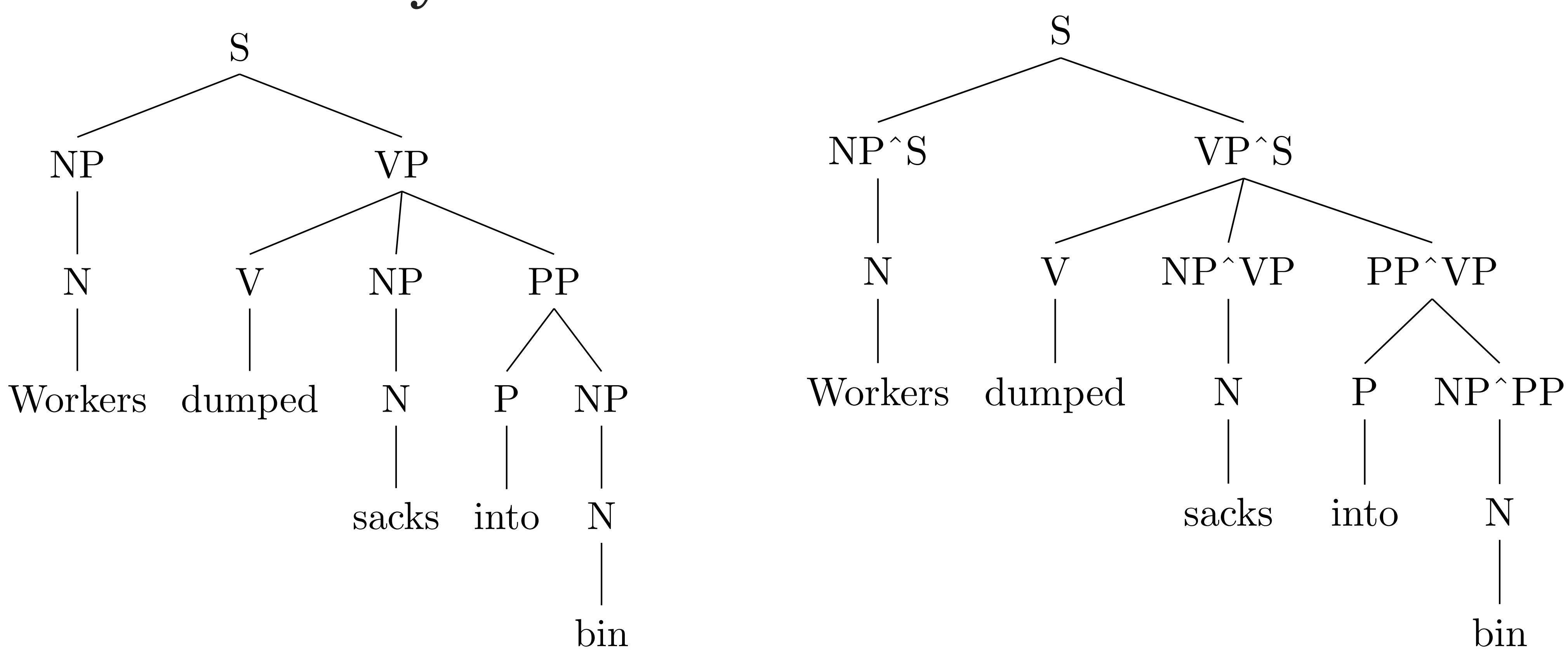
- Consider sentences (PP shown bracketed)

  (1) *Workers dumped sacks [into bin].*
  (2) *Fishermen caught tons [of herring].*

- Both have same POS tag sequence, but different structure
  - PP attaches either high (to the verb) or low (to the noun)
  - how to make this attachment decision? Difference between the two analyses minor:
    - VP → Verb NP PP vs. VP → Verb NP; NP → NP PP
- The probabilities of these three rules drive attachment, *irrespective of the verb, preposition and noun*

ONE SOLUTION: PARENT CONDITIONING

- Make non-terminals more explicit by incorporating parent symbol into each symbol

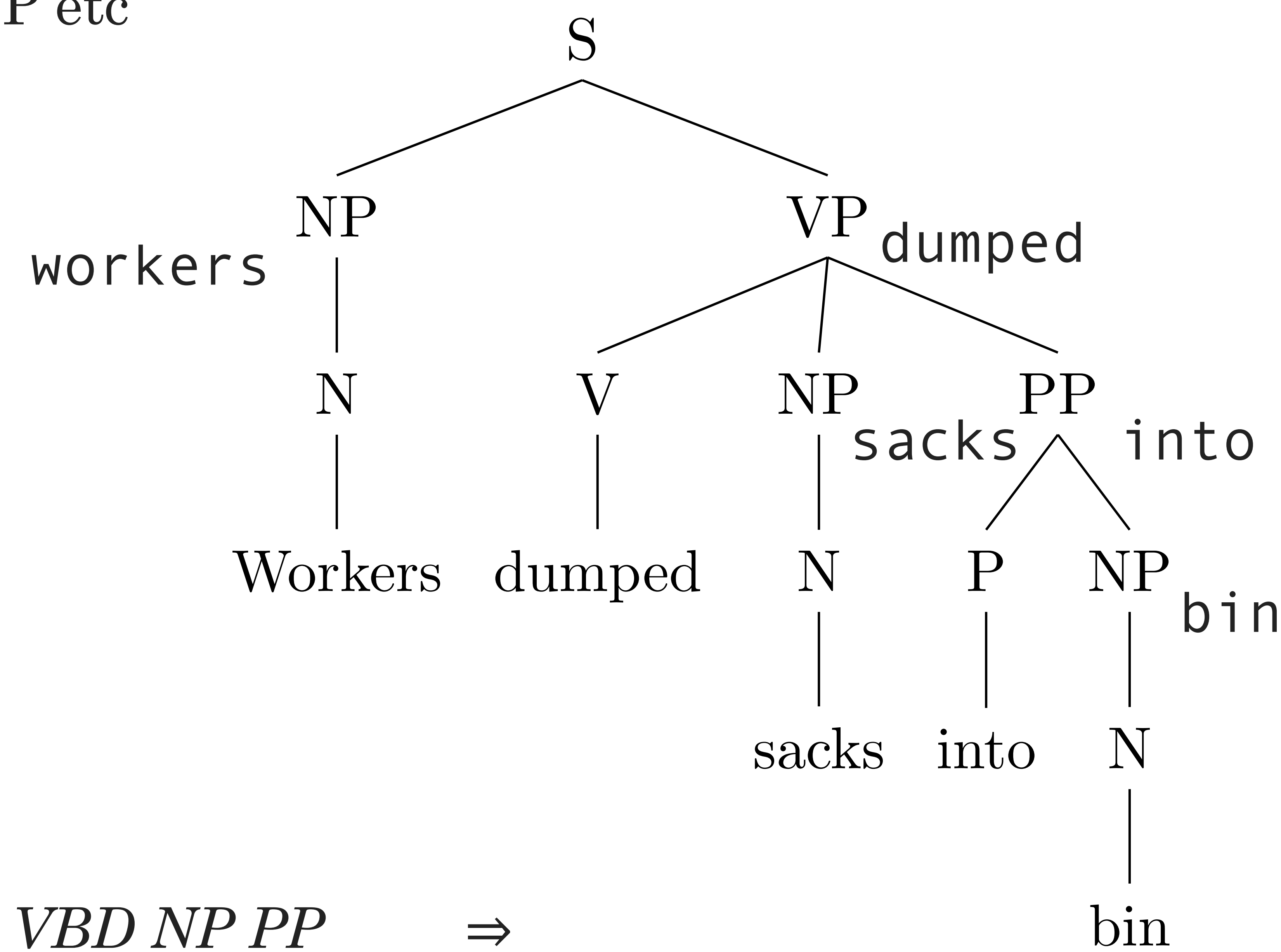


- NP^S represents subject position (left); NP^VP denotes object position (right)

- Helps to make general tags more specific, used for a number of different purposes, e.g., *He said* ***that*** *I saw* …

ANOTHER SOLUTION: HEAD LEXICALISATION

- Record **head word** with parent symbols
  - the most salient child of a constituent, usually the noun in a NP, verb in a VP etc



- $VP \rightarrow VBD\ NP\ PP \quad \Rightarrow$
  $VP(dumped) \rightarrow VBD(dumped)\ NP(sacks)\ PP(into)$

HEAD LEXICALISATION

- Incorporate head words into productions, such that the most important links between words is captured
  - rule captures correlations between head tokens of phrases
- Grammar symbol inventory expands massively!
  - Many of the productions much too specific, seen very rarely
  - Learning more involved to avoid sparsity problems (e.g., zero probabilities)

A FINAL WORD

- PCFGs widely used, and some of the best performing parsers available. E.g.,
  - Collins parser, Berkeley parser, Stanford parser
  - all use some form of lexicalisation or change to non-terminal set with CFGs

REQUIRED READING

- J&M3 Ch. 13 – 13.6