Language and Computation

week 9, Tuesday, March 25

Tamás Biró
Yale University
tamas.biro@yale.edu
http://www.birot.hu/courses/2014-LC/

Practical matters

- **Pre/post-reading:** JM 12, JM 13, 14.1.
- **To come / background** JM 14.
- `http://birot.hu/courses/2014-LC/readings.txt`
- Midterm due Thursday.
- (To come: Viterbi and Forward-Backward – an example)

Today

- Context-free grammars
- Parsing CFGs
- Probabilistic CFGs

Next time: More (non-prob) parsers, as well as probabilistic CFGs, parsing PCFGs, learning PCFGs.

Formal languages (recap)

Given finite alphabet Σ, for instance

- $\Sigma =$ letters of the alphabet (orthographic)
- $\Sigma =$ segments in a phonological system (phonemes and/or allophones?)
- $\Sigma =$ words in a finite vocabulary
- $\Sigma =$ atoms in the formalism of some logic (first order, modal, etc.) employed to describe natural language semantics.

A language is $L \subseteq \Sigma^*$.

Formal languages (recap)

Given finite alphabet Σ, a language is $L \subseteq \Sigma^*$.

- Given formalism $\mathcal{F}$, such as
regular expressions, finite state automata, regular grammars, context free grammars, tree adjoining grammars, etc.

- let $F \in \mathcal{F}$ be an instance of such a formalism.

- $L(F)$: language accepted / generated by F.

- What is language class $\mathcal{L}(\mathcal{F}) = \{L(F) | F \in \mathcal{F}\}$?

Chomsky hierarchy

Type	Common Name	Rule Skeleton	Linguistic Example
0	Turing Equivalent	$\alpha \rightarrow \beta$, s.t. $\alpha \neq \epsilon$	HPSG, LFG, Minimalism
1	Context Sensitive	$\alpha A \beta \rightarrow \alpha \gamma \beta$, s.t. $\gamma \neq \epsilon$	
–	Mildly Context Sensitive		TAG, CCG
2	Context Free	$A \rightarrow \gamma$	Phrase-Structure Grammars
3	Regular	$A \rightarrow xB$ or $A \rightarrow x$	Finite-State Automata

NB:

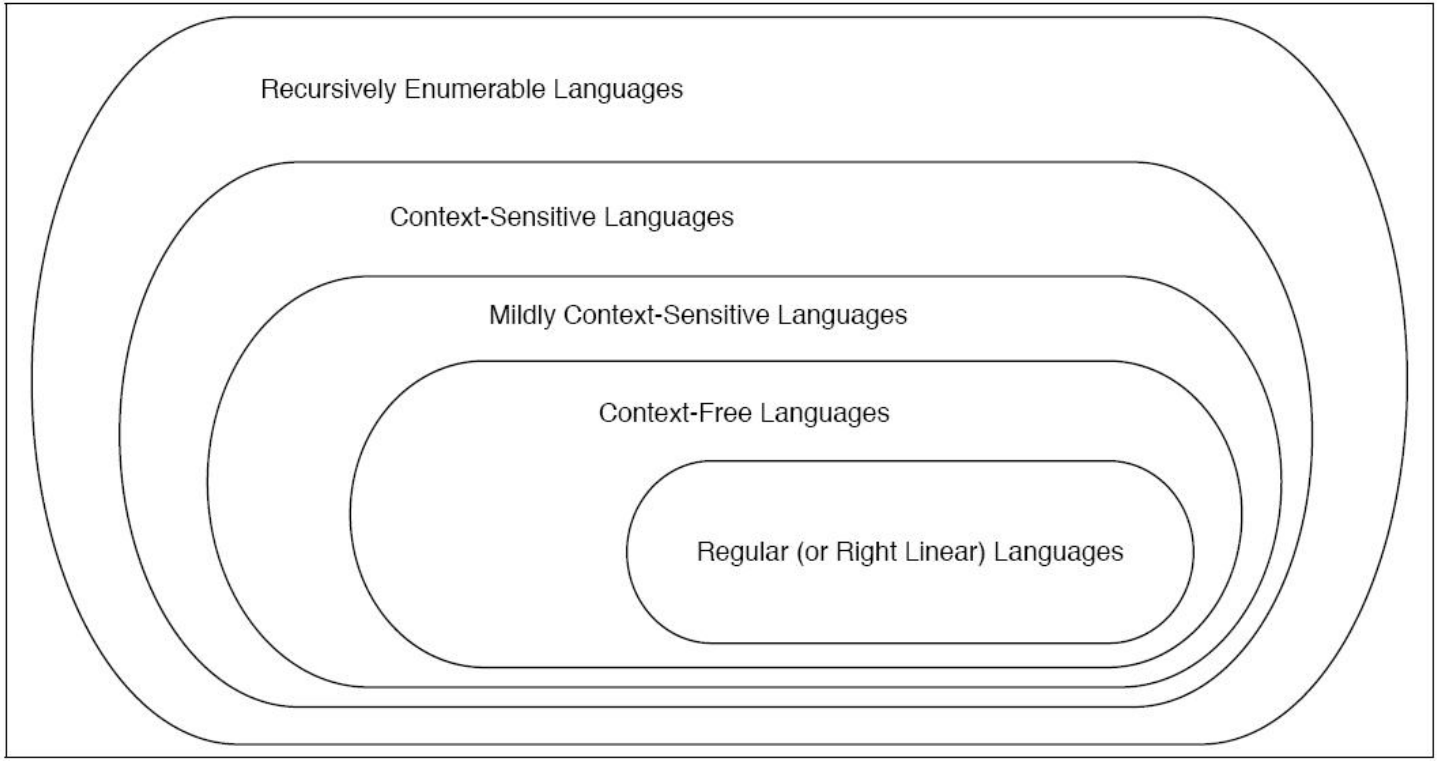
0: Turing machine

1: Linear bounded automaton

2: Non-deterministic push-down automaton

3: Finite-state automaton

The Chomsky Hierarchy



Chomsky hierarchy: examples

$\Sigma = \{a, b, c\}$ (or larger)

- $a^n b^m$ is not a finite language, but a regular language.
  RE, FSA and regular grammar provided during lecture 03/06.

- $a^n b^n$ is a $\text{context free language}$, but not a regular language.
  Not a regular language: proof based on the Pumping Lemma.
  It is a context free language: $S \rightarrow a\ S\ b$, $S \rightarrow a\ b$.

- $a^n b^n c^n$ is a $\text{context sensitive language}$, but not context free.
  Not a context free: proof based on the Bar-Hillel Lemma.

- $\{a^n | n \text{ is a prime number}\}$ is not context sensitive.

Context free languages

Context Free Grammars

A formal grammar is $G = (N, \Sigma, R, S)$.

Specifically, a CFG has a rewrite rule skeleton: $A \rightarrow \gamma$,
where $A \in N$ (non-terminal), and $\gamma \in (N \cup \Sigma)^*$ (any string).

Two perspectives:

- Generating a string: $S \Rightarrow^* s \in \Sigma^*$.
- Generating a tree: for each application of rule $A \rightarrow \gamma$, have a subtree with root A and daughters in γ.

Context Free Grammars

Two grammars, G_1 and G_2, are

- **weakly equivalent** if they generate the same language: $L(G_1) = L(G_2)$.
- **strongly equivalent** if they do so by generating the same trees. They assign same *phrase structure* to each sentence (allowing for renaming non-terminals).

Example from previous week $\{a^n b^m | n, m \in \mathbb{N}^+\}$:

- Regular grammar G_1: $S \rightarrow a\ S$, $S \rightarrow a\ A$, $A \rightarrow b\ A$, $A \rightarrow b$
- CFG G_2: $S \rightarrow A\ B$, $A \rightarrow A\ A$, $B \rightarrow B\ B$, $A \rightarrow a$, $B \rightarrow b$

Phrase structure: what is in a CFG rule?

John	eats	soup	with noodles	with a friend.
John	eats	soup		with a friend.
John	eats			with a friend.
John	eats.			
	Eat!			

(John ((eats (soup (with noodles))) (with a friend)))

Phrase structure: what is in a CFG rule?

$[_S$ *John* $[_{VP}$ $[_V$ *eats* $]$ $[_{NP}$ *soup* $[_{PP}$ *with noodles* $]]$ $[_{PP}$*with a friend* $]$ $]$ $]$

- A phrase: unit larger than word and smaller than sentence: *[soup with noodles]* (but not *soup with a friend*).
- The application of the rewrite rule: $VP \rightarrow V \;\; NP \;\; (PP)$, $\;NP \rightarrow NP \;\; PP$, $\;PP \rightarrow P \;\; NP$.
- A subtree in the sentence tree.
- Tree representation $=$ bracketing representation.

Chomsky Normal Form

Binary branching trees down to the prelexical nodes.

Rewrite rule skeletons:

- $A \rightarrow B \quad C$ (non-terminals only)
- $A \rightarrow a$ (lexical insertion of terminal symbols)

Theorem: Any context-free grammar can be converted into a weakly equivalent Chomsky Normal Form grammar.

Parsing context-free grammars

Parsing

Parsing: (1) recognizing an input string, as well as
(2) assigning some **structure** to it.

Context-free parsing: Given a CFG G and a string s,
(1) decide whether $s \in L(G)$, and
(2) identify the **phrases** and their categories in s.

Applications:

- Grammar checking
- Intermediate step toward semantic analysis, machine translation, question answering, information extraction, etc.

Parsing

Due to inherent **ambiguity** in language,

- Usually more than one parse is possible.
- The wider the coverage of a grammar, the more parses.

ALPINO (wide-coverage dependency parser, Dutch): 521,472 parses for

Aan Charles Masterman, een collega uit de eerste ministeriele jaren, was die neiging al eerder opgevallen (Mullen, 2002).

- Most of the parses are non-sense to the human judges.
- Solutions: filters; probabilistic CFGs.

Parsing

Shallow parsing (a.k.a. partial parsing, or **chunking**):

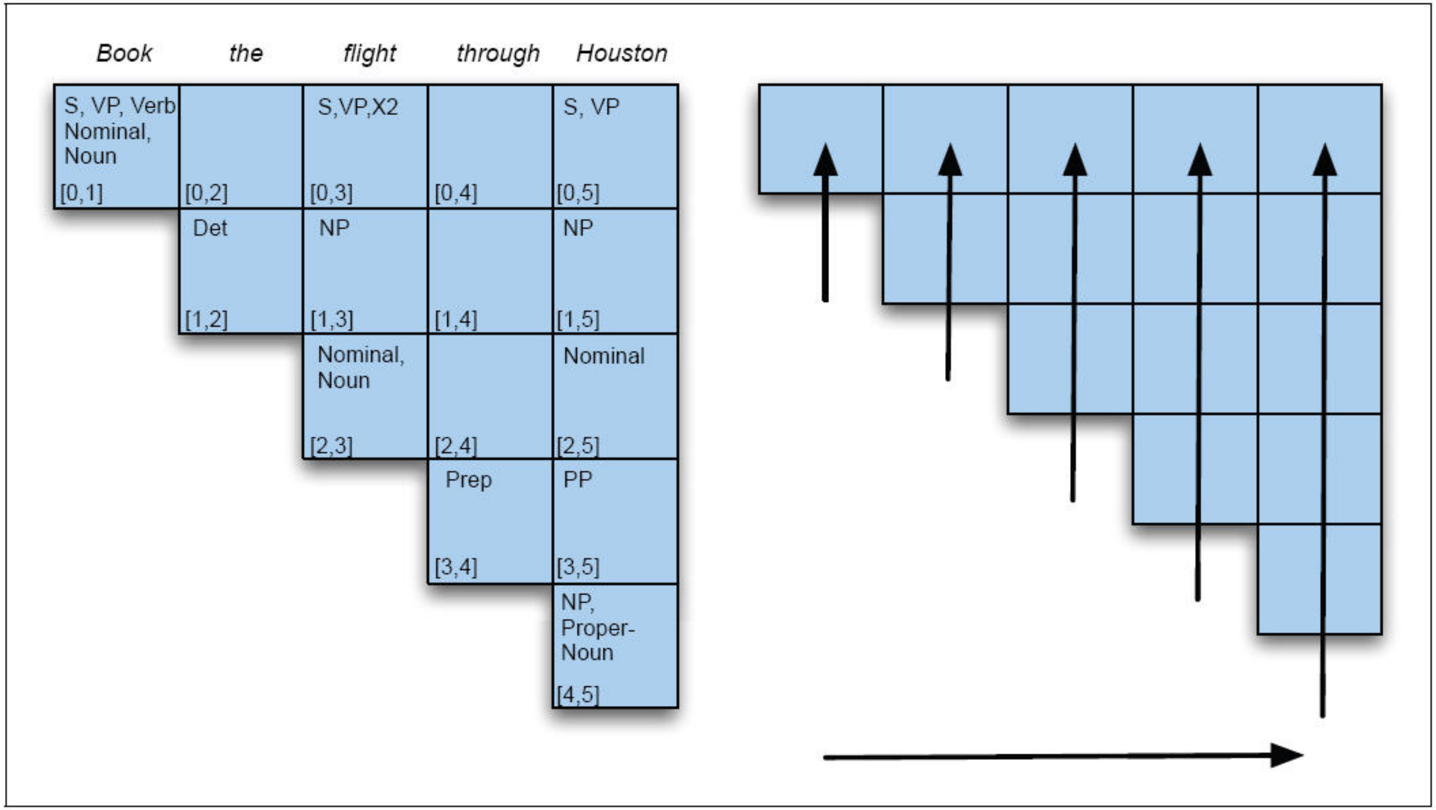
- Fast heuristic techniques useful for various NLP tasks (information extraction, statistical MT, etc.)
- that do not really need full parses.
- Less ambiguity.
- E.g., cascades of finite-state, rule-based transducers.
- E.g., various machine learning approaches.

Parsing

Full parsing as a search

- Bottom-up search strategy:
  - CKY (Cocke-Kasami-Younger) Algorithm
- Top-down search strategy:
  - Earley's Parser
- Chart parsing

CKY (Cocke-Kasami-Younger) Algorithm

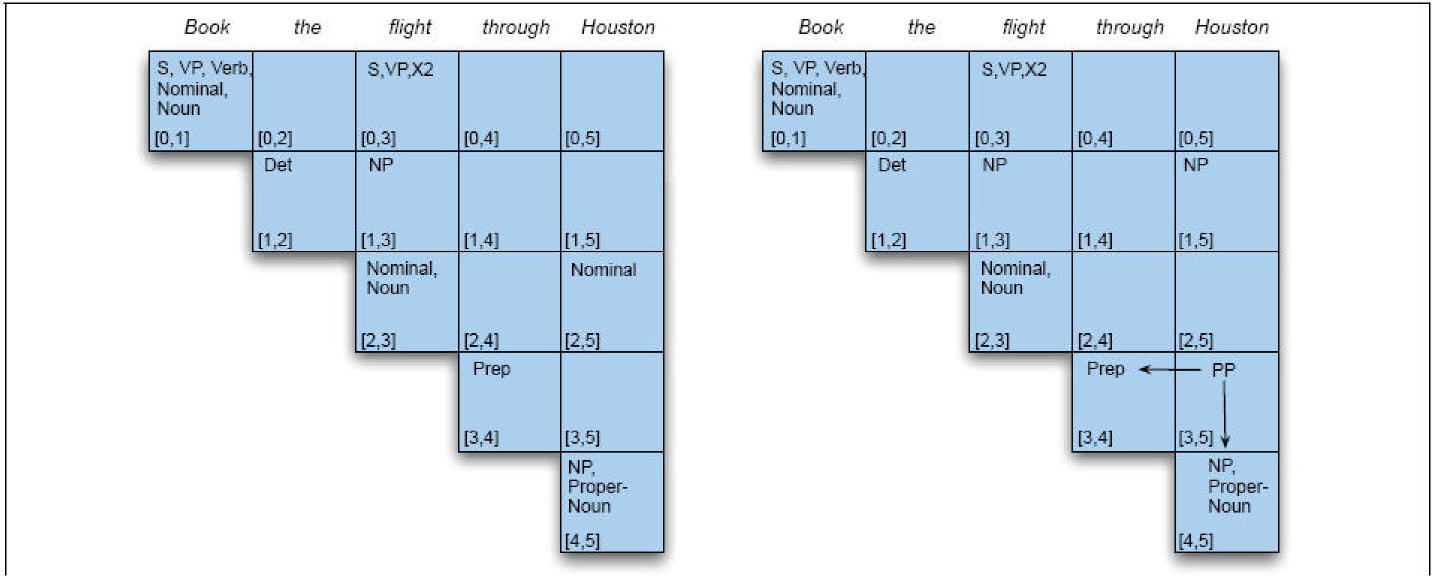


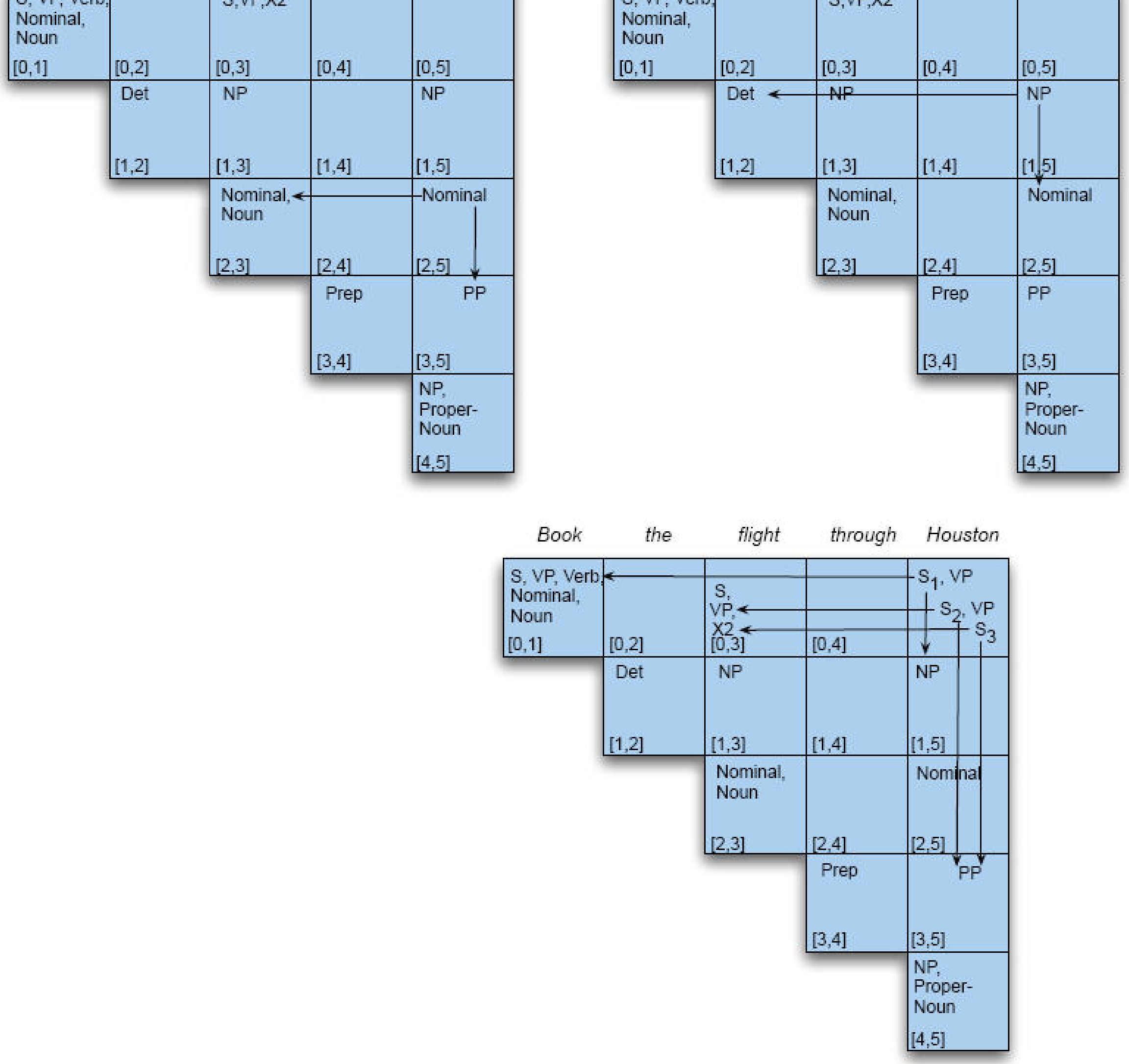
CKY (Cocke-Kasami-Younger) Algorithm

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

  for j ← from 1 to LENGTH(words) do
      table[j−1, j] ← {A | A → words[j] ∈ grammar}
      for i ← from j−2 downto 0 do
          for k ← i+1 to j−1 do
              table[i,j] ← table[i,j] ∪
                            {A | A → BC ∈ grammar,
                                 B ∈ table[i,k],
                                 C ∈ table[k,j]}
```

CKY (Cocke-Kasami-Younger) Algorithm





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

CKY (Cocke-Kasami-Younger) Algorithm

- CKY requires using CNF!
- CKY recognition: is there an S in cell $[0, N]$?
- CKY parsing: backpointers in order to recover the path leading to S.

Probabilistic Context Free Grammars

Probabilistic Context Free Grammars

N	a set of **non-terminal symbols** (or **variables**)
Σ	a set of **terminal symbols** (disjoint from N)
R	a set of **rules** or productions, each of the form $A \rightarrow \beta\ [p]$, where A is a non-terminal, β is a string of symbols from the infinite set of strings $(\Sigma \cup N)*$, and p is a number between 0 and 1 expressing $P(\beta \vert A)$
S	a designated **start symbol**

$$\sum_{\beta \in (N \cup \Sigma)^*} P(A \rightarrow \beta) = 1$$

Probabilistic Context Free Grammars

Probability of tree T (which yields sentence S):

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

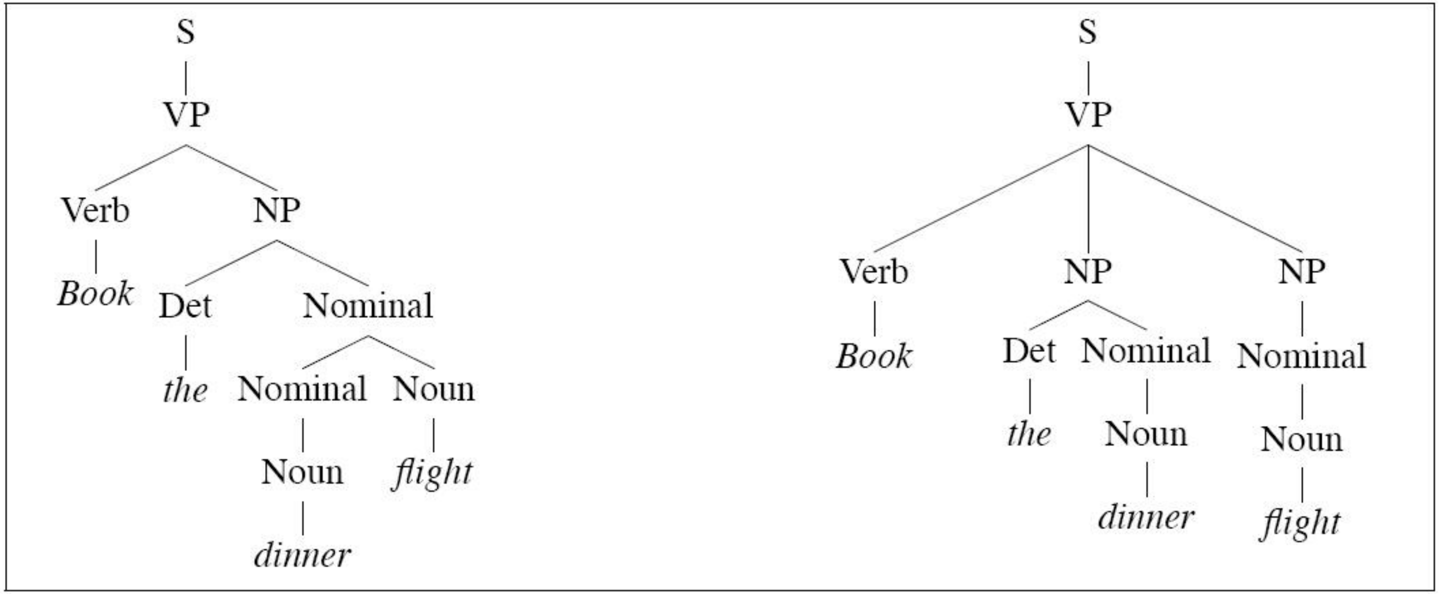
the product of the probabilities of the n rules used to expand each of the n non-terminal nodes in parse tree T (J&M 14.1.1).

Probabilistic CFG: an example

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]	| $meal$ [.15] | $money$ [.05]
$NP \rightarrow Pronoun$	[.35]	| $flights$ [.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]	

Probabilistic Context Free Grammars

An example:



(booking a flight serving dinner vs. booking a flight on behalf of 'dinner'.)

Parsing and grammar learning

- **Parsing:** Given (probabilistic) CFG G, given sentence s, find (possible/most probable) parse(s) tree for s in G.

- **Learning:** Given set of (parsed/unparsed) sentences, build a (probabilistic) context free grammar.

See you on Thursday!