Computational Linguistics

7. Statistical Parsing and Dependency Parsing

Xiaojing Bai

Tsinghua University

[**https://bxjthu.github.io/CompLing**](https://bxjthu.github.io/CompLing)

Recap: A formal definition of context-free grammars

S: a designated start symbol;

Σ: a set of terminal symbols;

N: a set of non-terminal symbols;

R: a set of rewrite rules of the form $A \rightarrow \beta$
where A is a non-terminal
and β is a string of elements from the infinite set $(\Sigma \cup N)^*$.

Or most commonly written as **Chomsky Normal Form** (CNF):

$A \rightarrow BC$, or, $A \rightarrow a$, where, A, B, $C \in N$, and, $a \in \Sigma$.

Recap exercise

Using the set of terminals {can, fish, rivers, pools, December, Scotland, it, they, in} and non-terminals {NP, VP, PP, V, P, S} with starting symbol S:

- Design a CFG in Chomsky Normal Form capable of generating the following sentences: *they can fish*, *they fish*, *they fish in rivers*, *they fish in rivers in December*.
- Give a formal definition of your CFG.
- Discuss any weaknesses of it in terms of over/under-production.

Sample codes

At the end of this session you will

- reflect on how humans process sentences;
- know about different kinds of ambiguities;
- know how to describe a language using a probabilistic context-free grammar;
- understand how treebanks present syntactic knowledge implicitly;
- know about the problems with PCFGs and the possible solutions;
- know how to describe a language using a dependency grammar;
- learn more about parsing and generating sentences with Python.

One way to avoid discovery is to make the changes during vacation.

One way to avoid confusion is to make the changes during vacation.

When humans are "processing" sentences ...

- When humans read, the predictability of a word seems to influence the reading time.

  Evidence for the predictability of an upcoming word given

  - the preceding words
  - the syntactic parse of the preceding sentence prefix
  - the morphological structure of the word

  $\Rrightarrow$ Voluntary research task:

  - Scott and Shillcock (2003)
  - Hale (2001), Levy (2008)
  - Moscoso del Prado Martin et al. (2004)

When humans are “processing” sentences ...

- When humans disambiguate sentences that have multiple possible parses, the more probable parse is preferred.

  Garden-path sentences

  E.g.

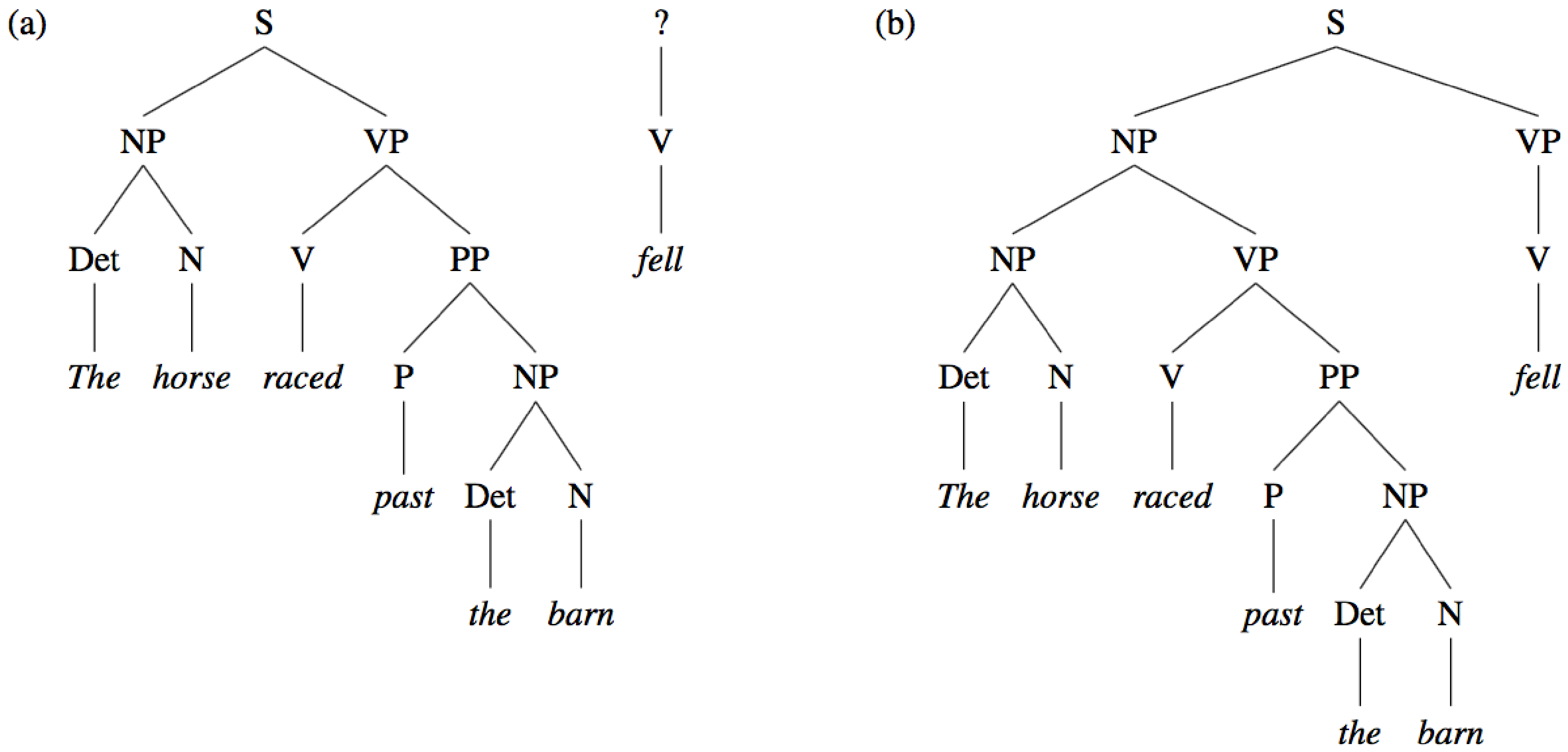
  The horse raced past the barn fell.

  The complex houses married and single students and their families.

  The student forgot the solution was in the back of the book.

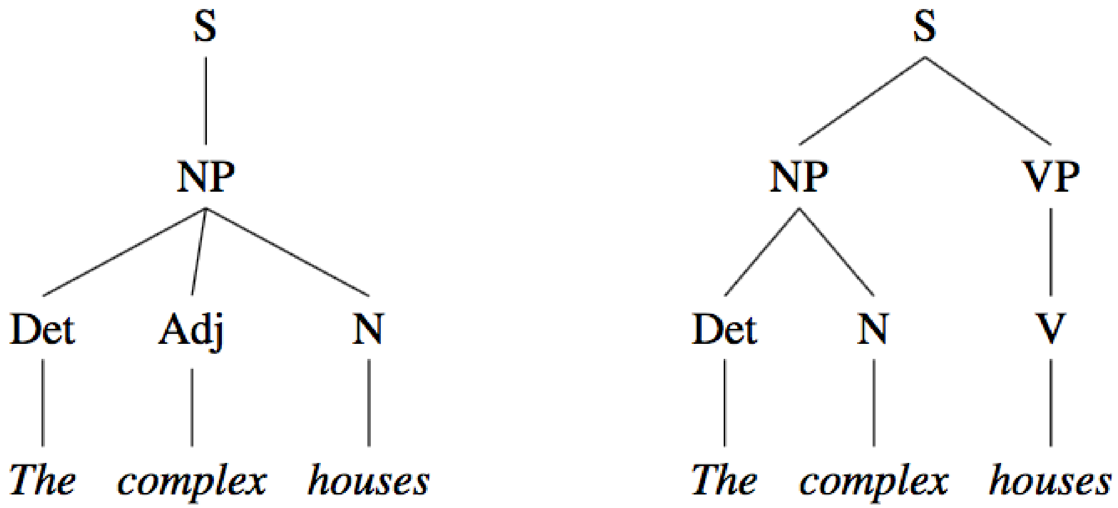
Garden-path sentences

E.g. *The horse raced past the barn fell.*



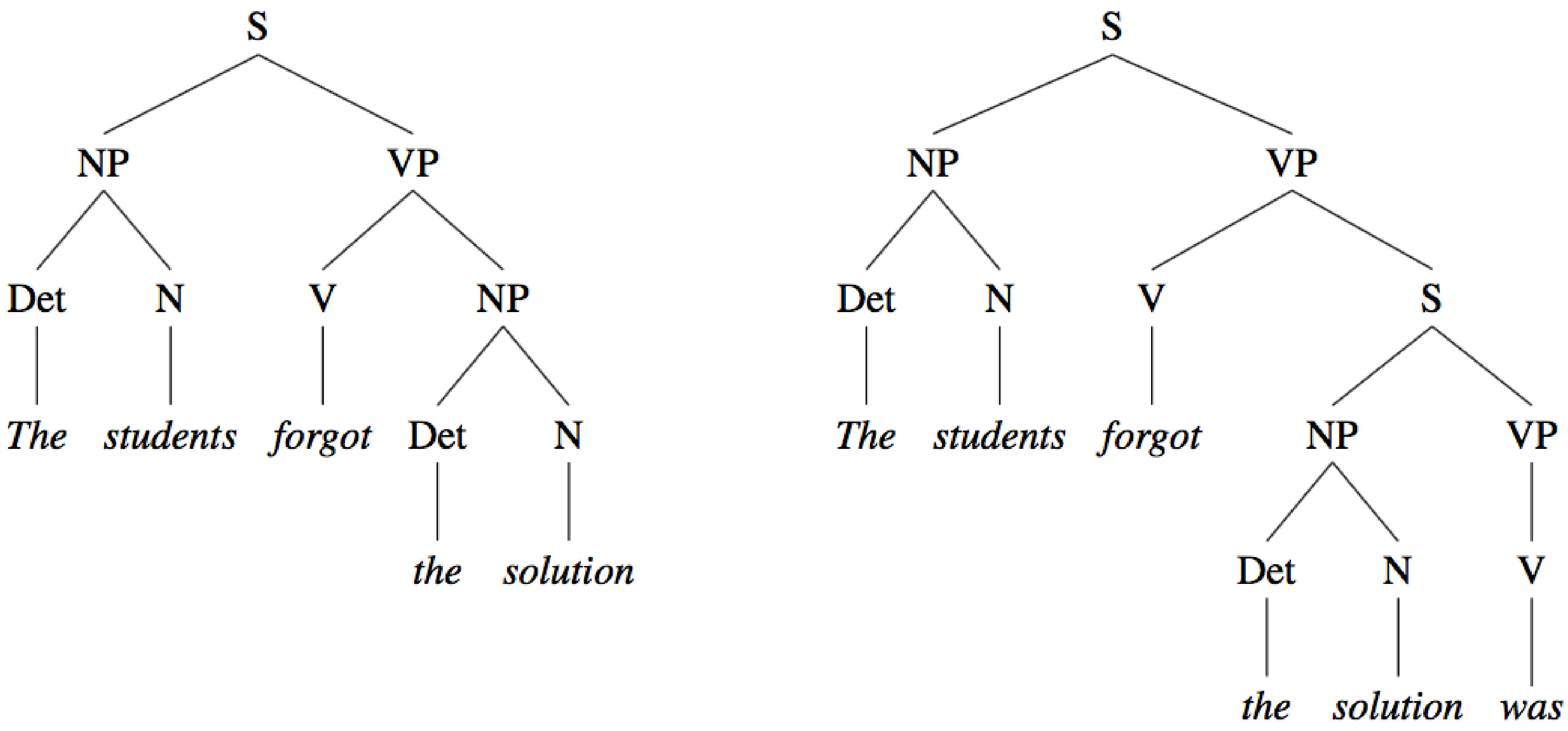
Garden-path sentences

E.g. *The complex houses married and single students and their families.*



Garden-path sentences

E.g. *The student forgot the solution was in the back of the book.*



Ambiguity

- POS ambiguity and POS disambiguation
  - *E.g. book that flight*
  - POS ambiguity in the Brown and WSJ corpora

		WSJ		Brown	
Types:					
Unambiguous	(1 tag)	44,432	(**86%**)	45,799	(**85%**)
Ambiguous	(2+ tags)	7,025	(**14%**)	8,050	(**15%**)
Tokens:					
Unambiguous	(1 tag)	577,421	(**45%**)	384,349	(**33%**)
Ambiguous	(2+ tags)	711,780	(**55%**)	786,646	(**67%**)

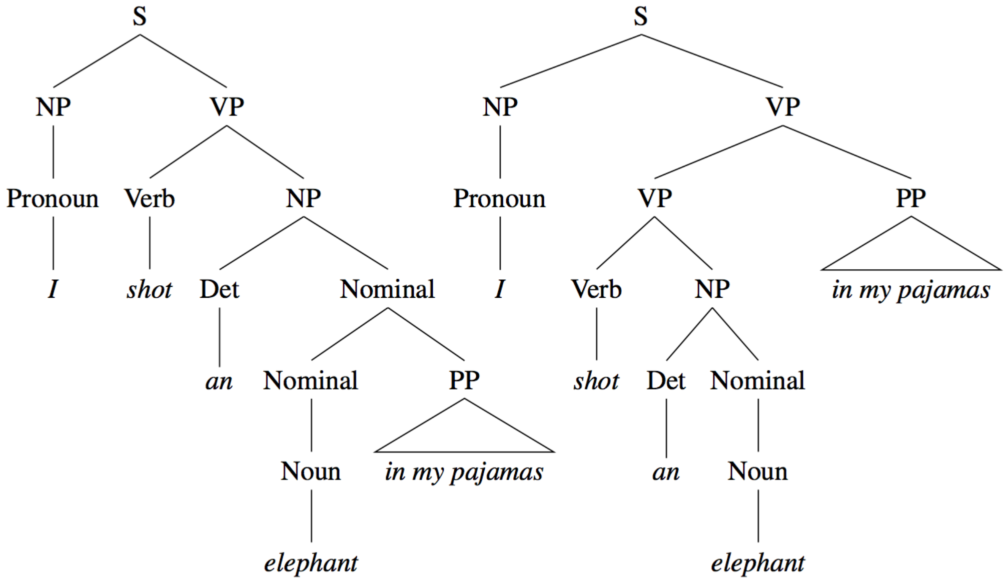
Ambiguity

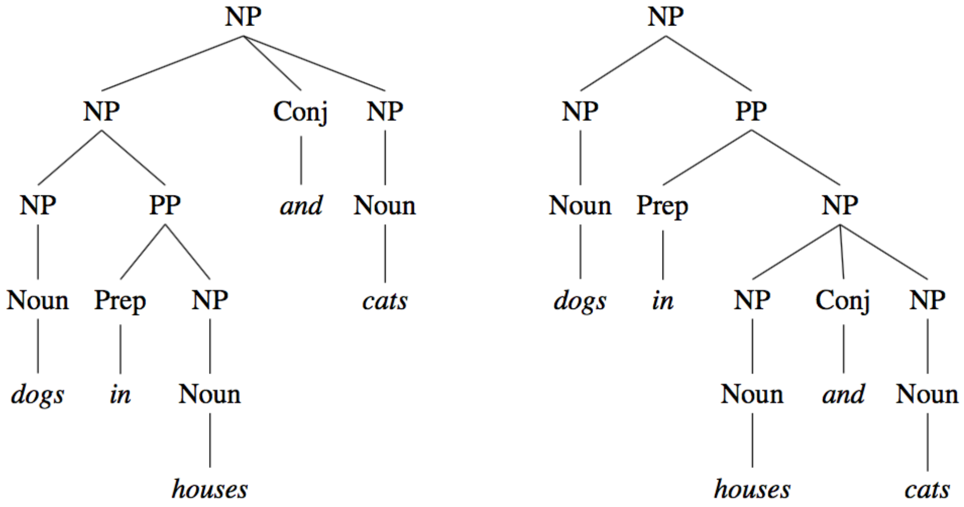
- Structural ambiguity
  - Attachment ambiguity

  E.g. *I shot an elephant in my pajamas.*
  - Coordination ambiguity

  E.g. *dogs in houses and cats*

Suggested topic for research:
Structural ambiguities in Chinese





Probabilistic context-free grammars (PCFG)

- A natural extension to context-free grammars
- Proposed by Booth (1969)
- Aka: Stochastic context-free grammar (SCFG)
- A context-free grammar G is defined by four parameters (N, S, Σ, R)

N: a set of non-terminal symbols;
S: a designated start symbol;
Σ: a set of terminal symbols;
R: a set of rewrite rules of the form $A \to \beta \ \ [p]$
where A is a non-terminal,
β is a string of elements from the infinite set $(\Sigma \cup N)*$,
p is a number between 0 and 1 expressing $P(\beta | A)$

A toy PCFG

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]	

How are PCFGs used?

Assign a probability to each parse tree T of a sentence S

$$A \rightarrow \beta \quad [p]$$

$$\sum_{\beta} P(A \rightarrow \beta) = 1$$

PCFGs for disambiguation

Disambiguation algorithm

Select the parse with the highest PCFG probability

$$\hat{T}(S) = \underset{S=\text{yield}(T)}{\text{argmax}} \; P(T|S) = \underset{S=\text{yield}(T)}{\text{argmax}} \; P(T)$$

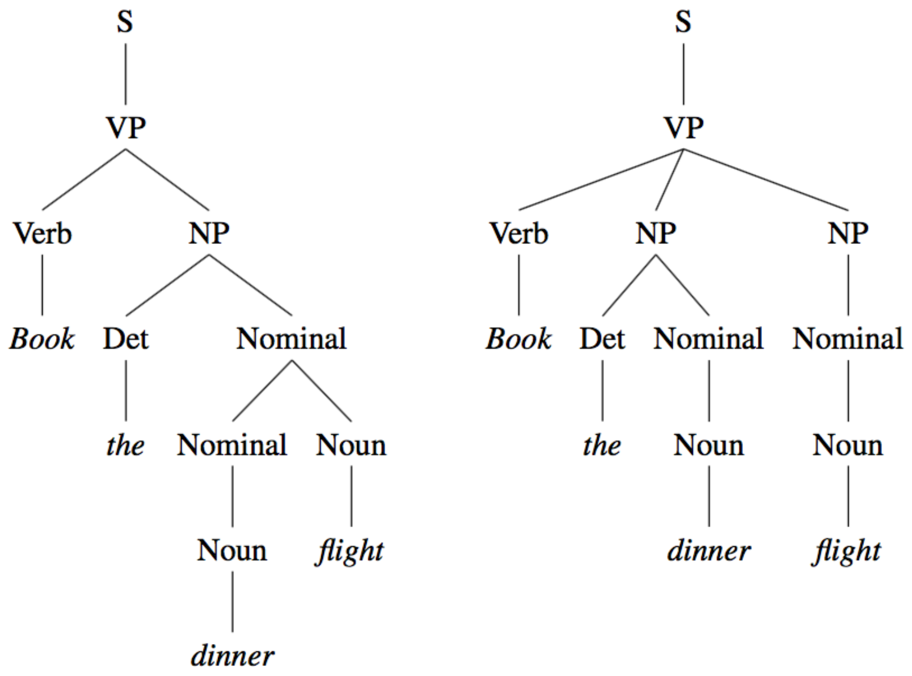
$$P(T|S) = \frac{P(T,S)}{P(S)} = P(T,S)$$

$$P(T,S) = P(T)P(S|T) = P(T)$$

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

Rule i: $LHS_i \rightarrow RHS_i$

Rules			P	Rules			P
S	$\rightarrow$	VP	.05	S	$\rightarrow$	VP	.05
VP	$\rightarrow$	Verb NP	.20	VP	$\rightarrow$	Verb NP NP	.10
NP	$\rightarrow$	Det Nominal	.20	NP	$\rightarrow$	Det Nominal	.20
Nominal	$\rightarrow$	Nominal Noun	.20	NP	$\rightarrow$	Nominal	.15
Nominal	$\rightarrow$	Noun	.75	Nominal	$\rightarrow$	Noun	.75
				Nominal	$\rightarrow$	Noun	.75
Verb	$\rightarrow$	book	.30	Verb	$\rightarrow$	book	.30
Det	$\rightarrow$	the	.60	Det	$\rightarrow$	the	.60
Noun	$\rightarrow$	dinner	.10	Noun	$\rightarrow$	dinner	.10
Noun	$\rightarrow$	flight	.40	Noun	$\rightarrow$	flight	.40



Learn PCFG rule probabilities with a non-probabilistic parser

- If sentences were unambiguous, increment a counter for every rule in the parse, and then normalize to get probabilities. (Not very probable!)
- If sentences are ambiguous, use the inside-outside algorithm.

Learn PCFG rule probabilities with a treebank

- Treebank: a syntactically annotated corpus
- Annotation: automatic parsers + human correction
- Treebanks in use
  - Penn Treebank project
    - Treebanks for English (the Brown, Switchboard, ATIS, WSJ corpora)
    - Treebanks for Arabic and Chinese
  - Other treebanks
    - The Sinica Treebank Corpus for Chinese
    - The Prague Dependency Treebank for Czech,
    - The Negra Treebank for German
    - The Susanne Treebank for English

Accessing treebanks

```
from nltk.corpus import treebank
t = treebank.parsed_sents('wsj_0001.mrg')[0]
print(t)
```

Or

```
import nltk
print(nltk.corpus.treebank.parsed_sents('wsj_0001.mrg')[0])
```

And

```
nltk.corpus.sinica_treebank.parsed_sents()[3450].draw()
```

Treebanks as implicit grammars

Given a treebank, compute the probability of each expansion of a non-terminal by counting the number of times that expansion occurs in the treebank and then normalizing.

$$P(\alpha \to \beta|\alpha) = \frac{Count(\alpha \to \beta)}{\sum_\gamma Count(\alpha \to \gamma)} = \frac{Count(\alpha \to \beta)}{Count(\alpha)}$$

Constructing and manipulating PCFGs

```
nltk.grammar.pcfg_demo()
```

More detailed instructions at

https://www.cs.bgu.ac.il/~elhadad/nlp16/NLTK-PCFG.html

Problems with PCFGs

- Lack of sensitivity to structural/contextual dependencies

  CFG independence assumption

  $$NP \rightarrow DT \quad NN \quad 0.28$$

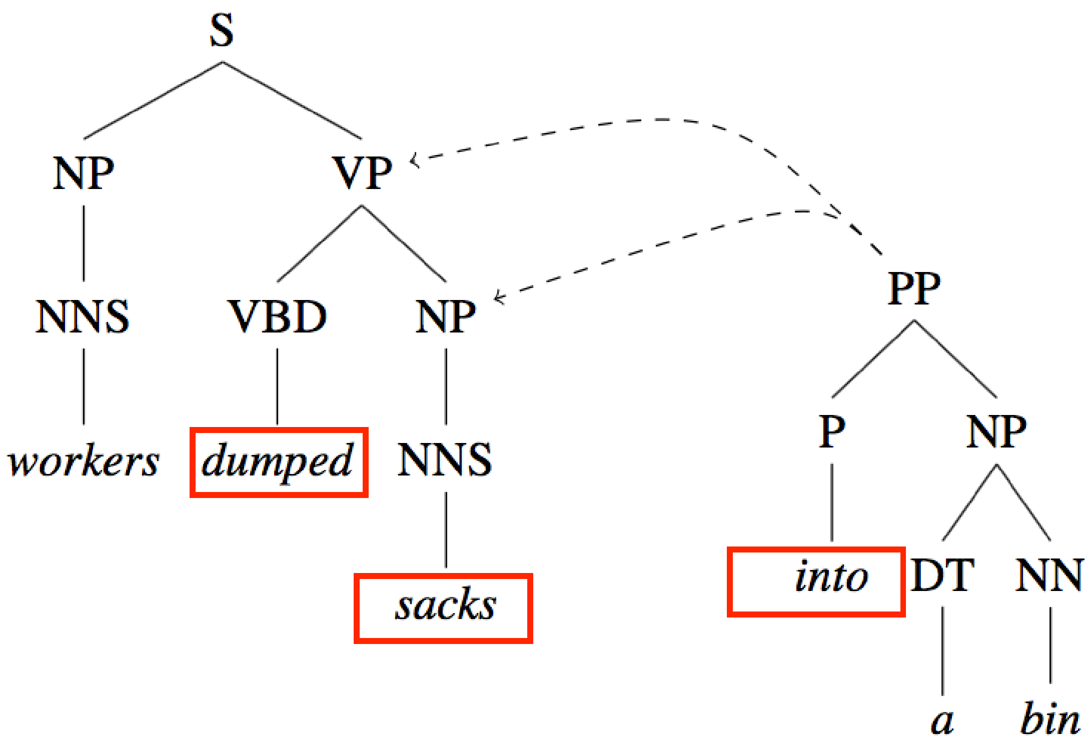
  $$NP \rightarrow PRP \quad 0.25$$

	Pronoun	**Non-pronoun**
Subject	0.91	0.9
Object	0.34	0.66

Problems with PCFGs

- Lack of sensitivity to lexical dependencies

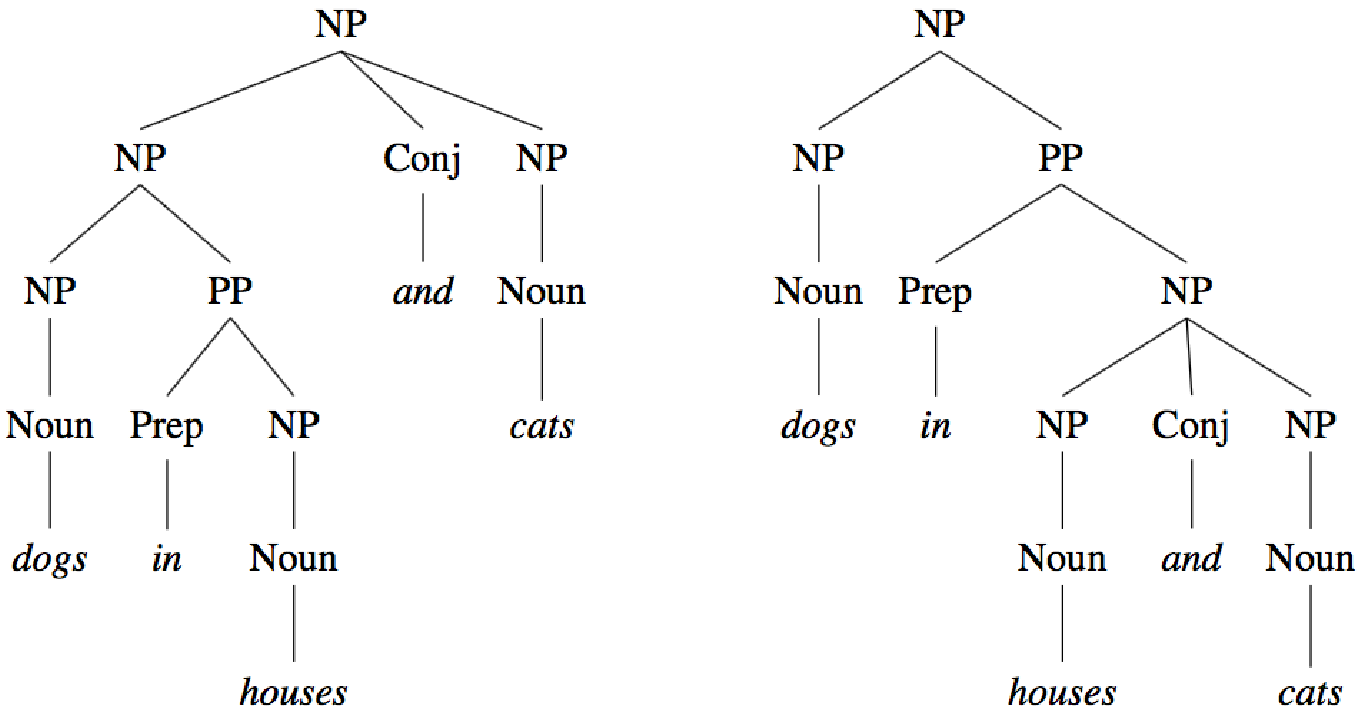
  E.g. *Workers dumped sacks into a bin.*



Problems with PCFGs

- Lack of sensitivity to lexical dependencies

  E.g. *dogs in houses and cats*



Improving PCFGs

- Lack of sensitivity to structural/contextual dependencies

  CFG independence assumption

$$NP \rightarrow DT \quad NN \quad 0.28$$

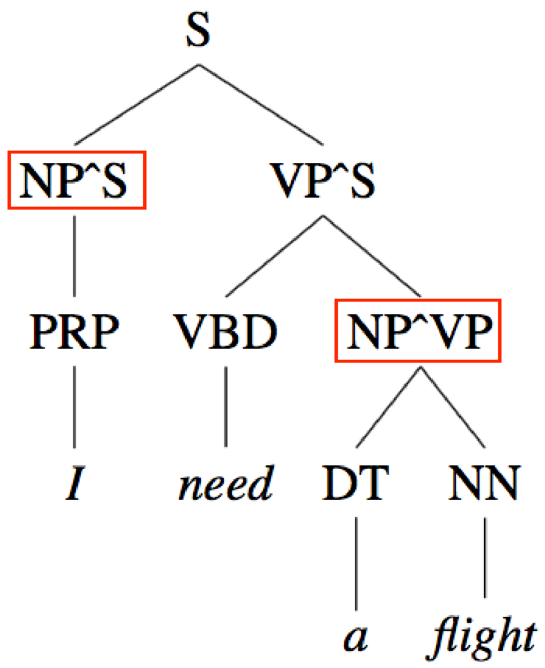
$$NP \rightarrow PRP \quad 0.25$$

	Pronoun	**Non-pronoun**
Subject	0.91	0.9
Object	0.34	0.66

Solution:
Splitting non-terminals

$$NP_{subject} \rightarrow PRP$$

$$NP_{object} \rightarrow PRP$$



Improving PCFGs

- Lack of sensitivity to lexical dependencies

  E.g. *Workers dumped sacks into a bin.*

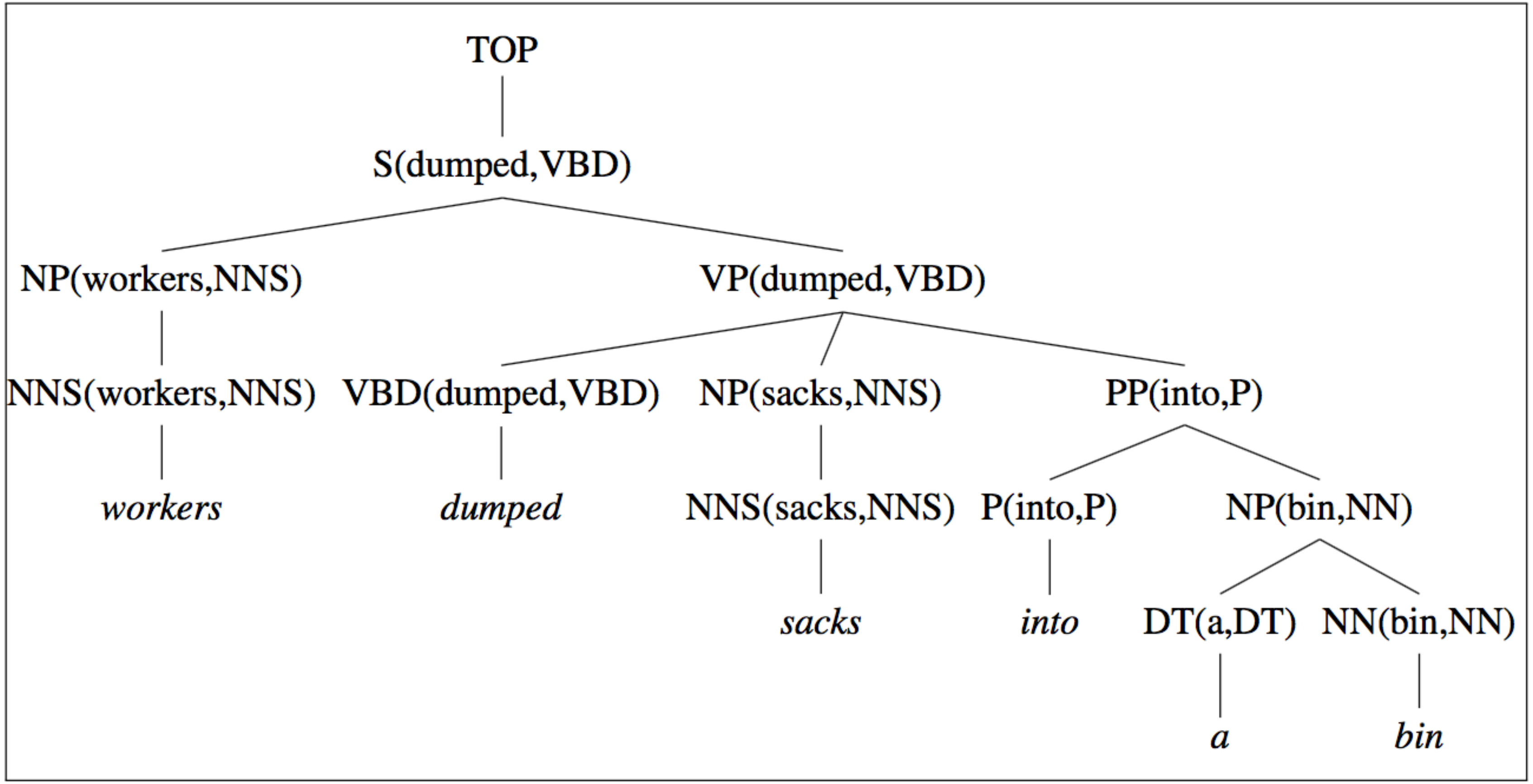
  Solution: Lexicalizing the rules

  $$VP \rightarrow VBD\ \ NP\ \ PP$$

  $$VP_{(dumped)} \rightarrow VBD_{(dumped)}\ \ NP_{(sacks)}\ \ PP_{(into)}$$

  $$VP_{(dumped, VBD)} \rightarrow VBD_{(dumped, VBD)}\ \ NP_{(sacks, NNS)}\ \ PP_{(into, P)}$$

  Lexicalized grammar, head tag



Internal Rules					**Lexical Rules**		
TOP	→	S(dumped,VBD)			NNS(workers,NNS)	→	workers
S(dumped,VBD)	→	NP(workers,NNS)	VP(dumped,VBD)		VBD(dumped,VBD)	→	dumped
NP(workers,NNS)	→	NNS(workers,NNS)			NNS(sacks,NNS)	→	sacks
VP(dumped,VBD)	→	VBD(dumped, VBD)	NP(sacks,NNS)	PP(into,P)	P(into,P)	→	into
PP(into,P)	→	P(into,P)	NP(bin,NN)		DT(a,DT)	→	a
NP(bin,NN)	→	DT(a,DT)	NN(bin,NN)		NN(bin,NN)	→	bin

Constituency

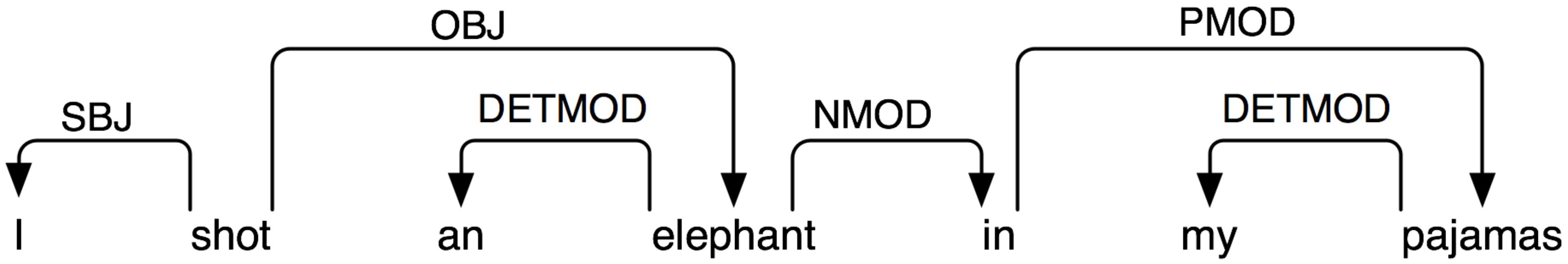
- Constituent-based language models vs. dependency-based language models
- Abstraction: the fundamental notion underlying the idea of constituency
    - A group of words behaving as a single unit
    - A group of words appearing in similar syntactic environments

E.g.
(1) *three parties from Brooklyn arrive* ...
(2) *a high-class spot such as Mindy's attracts* ...
(3) *the Broadway coppers love* ...
(4) *they sit* ...

(5) *On September seventeenth , I'd like to fly from Atlanta to Denver*
(6) *I'd like to fly on September seventeenth from Atlanta to Denver*
(7) *I'd like to fly from Atlanta to Denver on September seventeenth*

Dependency parsing

Describing the syntactic structure of a sentence in terms of the constituting words (or lemmas) and an associated set of directed binary grammatical relations that hold among them.



Arrows point from heads to their dependents.
Labels indicate the grammatical functions of the dependents.

Dependency-based parsing vs. constituent-based parsing

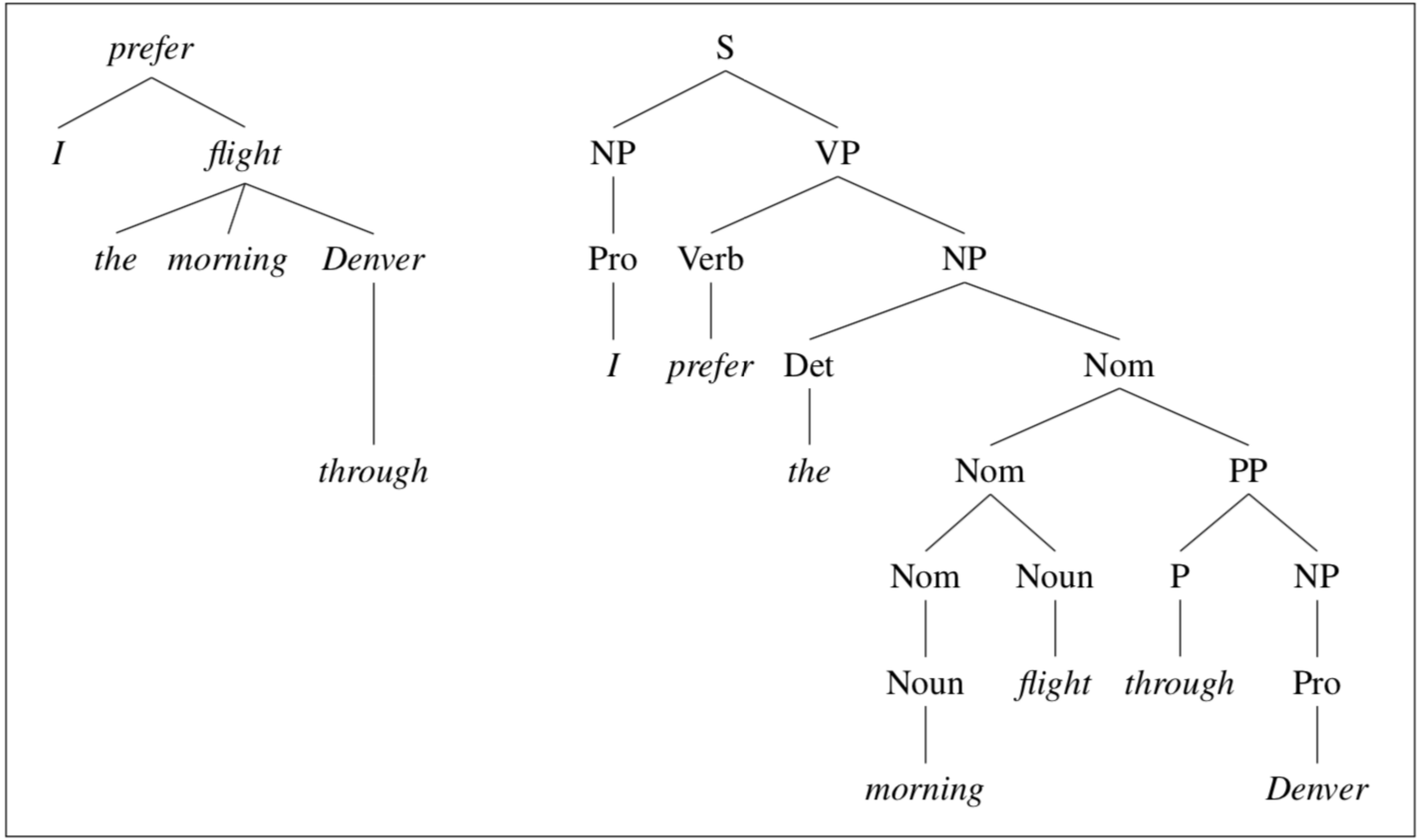


Figure 13.1 A dependency-style parse alongside the corresponding constituent-based analysis for *I prefer the morning flight through Denver.*

Dependency relations

Clausal Argument Relations	Description
NSUBJ	Nominal subject
DOBJ	Direct object
IOBJ	Indirect object
CCOMP	Clausal complement
XCOMP	Open clausal complement
Nominal Modifier Relations	**Description**
NMOD	Nominal modifier
AMOD	Adjectival modifier
NUMMOD	Numeric modifier
APPOS	Appositional modifier
DET	Determiner
CASE	Prepositions, postpositions and other case markers
Other Notable Relations	**Description**
CONJ	Conjunct
CC	Coordinating conjunction

Selected from the Universal Dependency set

Dependency relations

Relation	Examples with *head* and **dependent**
NSUBJ	**United** *canceled* the flight.
DOBJ	United *diverted* the **flight** to Reno.
	We *booked* her the first **flight** to Miami.
IOBJ	We *booked* **her** the flight to Miami.
NMOD	We took the **morning** *flight*.
AMOD	Book the **cheapest** *flight*.
NUMMOD	Before the storm JetBlue canceled **1000** *flights*.
APPOS	*United*, a **unit** of UAL, matched the fares.
DET	**The** *flight* was canceled.
	Which *flight* was delayed?
CONJ	We *flew* to Denver and **drove** to Steamboat.
CC	We flew to Denver **and** *drove* to Steamboat.
CASE	Book the flight **through** *Houston*.

Examples of core Universal Dependency relations

The Universal Dependencies project (Nivre et al., 2016)

provides an inventory of dependency relations that are linguistically motivated, computationally useful, and cross-linguistically applicable.

Dependency parsing algorithms

- Transition-based dependency parsing
- Graph-based dependency parsing

Models and algorithms

- Language models: correct parsing
  - Lexical-functional grammar
  - Head-driven phrase structure grammar
  - Link grammar
  - Dependency grammar
  - Probabilistic context-free grammar ...
- Parsing algorithms: efficient parsing
  - CYK parsing (Cocke-Younger-Kasami algorithm, Chomsky Normal Form)
  - Earley parsing
  - Chart parsing
  - Left–corner parsing
  - ATN parsing ...

At the end of this session you will

- reflect on how humans process sentences;
- know about different kinds of ambiguities;
- know how to describe a language using a probabilistic context-free grammar;
- understand how treebanks present syntactic knowledge implicitly;
- know about the problems with PCFGs and the possible solutions;
- know how to describe a language using a dependency grammar;
- learn more about parsing and generating sentences with Python.

Teamwork

- Given: a test set of English sentences
- Task: determine if the sentences use the passive voice
  - y: passive voice found
  - n: passive voice not found
- Submission on GradeScope (DDL: Dec. 2)
  - Code and linguistic data used for this task
  - Written report (including but not limited to methodology, algorithm, linguistic data, difficulties, error analysis, etc.)
- Oral presentation based on the written report (Dec. 5)

Homework

- Read/Review (Quiz 6 on Nov. 14, 2018)
  - J+M 11
  - J+M 12 (excluding 12.7)
- Practice
  - http://www.nltk.org/book/ch08.html

References:
http://www.nltk.org/howto/parse.html
http://www.nltk.org/howto/generate.html
https://www.cs.bgu.ac.il/~elhadad/nlp16/NLTK-PCFG.html

Next session

Features and Unification, Language and Complexity