

## Probabilistic Context-Free Grammars

### Informatics 2A: Lecture 18

Bonnie Webber and Frank Keller

School of Informatics  
University of Edinburgh  
bonnie@inf.ed.ac.uk

5 November 2009

### Motivation

Four things motivate the use of probabilities in grammars and parsing:

- 1 Ambiguity – ie, the same thing motivating chart parsing, LL(1) parsing, etc.
- 2 Coverage – Issues in developing a grammar for a language
- 3 Zipf's Law
- 4 Empirical evidence from studies of human language processing

### Reading:

*J&M 2<sup>nd</sup> edition, ch. 14 (Section 14.2–14.6.1)*

### Motivation 1: Ambiguity

- Language is highly ambiguous: The amount of ambiguity – both lexical and structural – increases with sentence length.
- Real sentences, even in newspapers or email, are fairly long (avg. sentence length in the *Wall Street Journal* is 25 words).  
*A second provision passed by the Senate and House would eliminate a rule allowing companies that post losses resulting from LBO debt to receive refunds of taxes paid over the previous three years.* [wsj.1822] (33 words)
- The amount of (unexpected!) ambiguity increases rapidly with sentence length. This poses a problem for parsers, even chart parsers, that keep track of all possible analyses.
- We could cut down the amount of work if we could ignore improbable analyses.

## Motivation 2: Coverage

- It is actually very difficult to write a grammar that covers all the constructions used in ordinary text or speech (e.g., in a newspaper).
- Typically hundreds of rules are required in order to capture both all the different linguistic patterns and all the different possible analyses of the same pattern. (Recall from lecture 13, the grammar rules we had to add to cover three different analyses of *You made her duck*.)
- Ideally, one wants to **induce** (learn) a grammar from a corpus.
- Grammar induction requires probabilities.

## Motivation 4: Human Sentence Processing

While almost every sentence is ambiguous in some way (even this one!), we (as people) rarely notice it.

Instead, we seem to see only one interpretation – although we may see different interpretations in different contexts.

Probabilities in the grammar or parser or both seem a good way to model this.

More about this in Lectures 28–30.

## Motivation 3: Zipf's Law (Again)

As with words and parts-of-speech, the distribution of grammar constructions is also Zipfian, but the likelihood of a particular construction can vary, depending on:

- register** (formal vs. informal): eg, *greenish*, *alot*, subject-drop (*Want a beer?*) are all more probable in informal than formal register;
- genre** (newspapers, essays, mystery stories, jokes, ads, etc.): Clear from the difference in PoS-taggers trained on different genres in the Brown Corpus.
- domain** (biology, patent law, football, etc.).

Probabilistic grammars and parsers can reflect these kinds of distributions.

## Example: Improbable parse

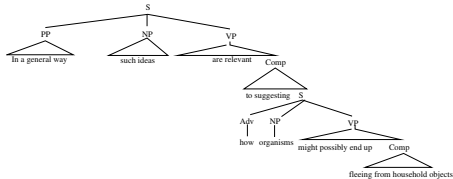
Let's compare an unexpected (and improbable) but grammatical parse for a 22-word sentence with a more probable parse.

- In a general way, such ideas are relevant to suggesting how organisms we know might possibly end up fleeing from household objects.



What's odd about this? Why is it improbable?

## Example: Probable parse



Both parses **and many more** would be produced by a parser that had to compute all grammatical analyses.

What's the alternative?

## Probabilistic Context-Free Grammars

We can try associating the likelihood of an analysis with the likelihood of its grammar rules.

Given a grammar  $G = (N, \Sigma, P, S)$ , a PCFG augments each rule in  $P$  with a **conditional probability**  $p$ .

This  $p$  represents the probability that non-terminal  $A$  will expand to the sequence  $\beta$ , which we can write as

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

or

$$P(A \rightarrow \beta | A) = p$$

or

$$P(A \rightarrow \beta) = p$$

## Probabilistic Context-Free Grammars

If we consider all the rules for a non-terminal  $A$ :

$$A \rightarrow \beta_1 [p_1]$$

...

$$A \rightarrow \beta_k [p_k]$$

then the sum of their probabilities ( $p_1 + \dots + p_k$ ) must be 1.

This ensures the probabilities form a valid **probability distribution**.

## Example

Suppose there's only one rule for the non-terminal  $S$  in the grammar:

$$S \rightarrow NP VP$$

What is  $P(S \rightarrow NP VP)$ ?

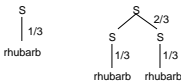
A PCFG is said to be **consistent** if the sum of the probabilities of **all sentences** in the language equals 1.

**Note:** Recursive rules can cause a grammar to be inconsistent. Let's see why.

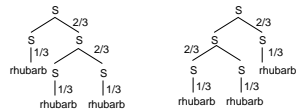
## Example (from nlp.stanford.edu)

Consider the very simple grammar  $G_{rhubarb}$ :

$$\begin{aligned} S &\rightarrow rhubarb \left[\frac{1}{3}\right] \\ S &\rightarrow S S \left[\frac{2}{3}\right] \end{aligned}$$



$$\begin{aligned} P(rhubarb) &= \frac{1}{3} \\ P(rhubarb rhubarb) &= \frac{2}{3} \times \frac{1}{3} \times \frac{1}{3} = \frac{2}{27} \end{aligned}$$



$$P(rhubarb rhubarb rhubarb) = \left(\frac{2}{3}\right)^2 \times \left(\frac{1}{3}\right)^3 \times 2 = \frac{8}{243}$$

...

$$\Sigma P(L_{rhubarb}) = \frac{1}{3} + \frac{2}{27} + \frac{8}{243} + \dots = \frac{1}{2}$$

So the grammar  $G_{rhubarb}$  is **inconsistent**.

## Questions about PCFGs

Four questions are of interest regarding PCFGs:

- Applications:** What can we use PCFGs for?
- Estimation:** Given a corpus and a grammar, how can we induce the rule probabilities?
- Parsing:** Given a string and a PCFG, how can we efficiently compute the most probable parse?
- Grammar induction:** Given a corpus, how can we induce both the grammar and the rule probabilities?

In this lecture, we will deal with question 1. The next lecture will deal with questions 2 and 3. Question 4 is addressed in the 3<sup>rd</sup>-year course *Introduction to Cognitive Science* (Semester 1) and the 4<sup>th</sup>-year courses in *Cognitive Modelling* and in *Machine Translation* (both Semester 2).

## Application 1: Disambiguation

The probability that a PCFG assigns to a parse tree can be used to **disambiguate sentences** that have more than one parse.

**Assumption:** The most probable parse is the intended parse.

The probability of a parse  $T$  for a sentence  $S$  is defined as the **product** of the probability of each rule  $r$  used to expand a node  $n$  in the parse tree.

$$P(T, S) = \prod_{n \in T} p(r(n))$$

Since a sentence  $S$  corresponds to the **yield** of the parse tree  $T$ ,  $P(S|T) = 1$ , hence:

$$P(T) = \frac{P(T, S)}{P(S|T)} = \frac{P(T, S)}{1} = P(T, S) = \prod_{n \in T} p(r(n))$$

## Application 1: Disambiguation

Example grammar:

R1	$S \rightarrow NP VP$	(0.85)	R9	$VP \rightarrow TV NP NP$	(0.05)
R2	$S \rightarrow Aux NP VP$	(0.15)	R10	$VP \rightarrow TV NP$	(0.4)
R3	$NP \rightarrow PRO$	(0.4)	R11	$VP \rightarrow IV$	(0.55)
R4	$NP \rightarrow NOM$	(0.05)	R12	$Aux \rightarrow can$	(0.4)
R5	$NP \rightarrow NPR$	(0.35)	R13	$N \rightarrow flights$	(0.5)
R6	$NP \rightarrow NPR NOM$	(0.2)	R14	$PRO \rightarrow you$	(0.4)
R7	$NOM \rightarrow N$	(0.75)	R15	$TV \rightarrow book$	(0.3)
R8	$NOM \rightarrow N PP$	(0.25)	R16	$NPR \rightarrow TWA$	(0.4)

$$P(R1) + P(R2) = ?$$

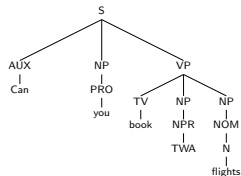
$$P(R3) + P(R4) + P(R5) + P(R6) = ?$$

...

What does this imply about the lexical rules given here?

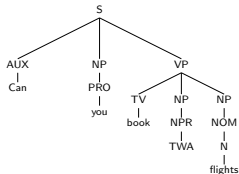
## Example: Can you book TWA flights?

Reading 1: "Can you book flights on behalf of TWA?"



## Example: Can you book TWA flights?

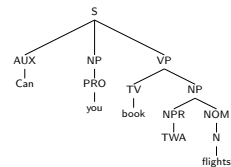
Reading 1: "Can you book flights on behalf of TWA?"



$$\begin{aligned}
 P(T1) &= P(R2)P(R12)P(R3)P(R14)P(R9)P(R15)P(R5)P(R16) \\
 &\quad \cdot P(R4)P(R7)P(R13) \\
 &= 0.15 \cdot 0.4 \cdot 0.4 \cdot 0.4 \cdot 0.05 \cdot 0.3 \cdot 0.35 \cdot 0.4 \cdot 0.05 \cdot 0.75 \cdot 0.5 \\
 &= 3.78 \cdot 10^{-7}
 \end{aligned}$$

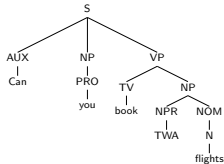
## Example

Reading 2: "Can you book flights associated with TWA?"



## Example

Reading 2: "Can you book flights associated with TWA?"



$$\begin{aligned}
 P(T_2) &= P(R_2)P(R_{12})P(R_3)P(R_{14})P(R_{10})P(R_{15})P(R_6) \\
 &\quad \cdot P(R_{16})P(R_7)P(R_{13}) \\
 &= 0.15 \cdot 0.4 \cdot 0.4 \cdot 0.4 \cdot 0.3 \cdot 0.2 \cdot 0.4 \cdot 0.75 \cdot 0.5 \\
 &= 3.46 \cdot 10^{-5}
 \end{aligned}$$

## Example

Since

$$P(T_2) = 3.46 \cdot 10^{-5} > P(T_1) = 3.78 \cdot 10^{-7}$$

we can conclude that  $T_2$  is the more likely to be the correct parse.

Note that we can simplify the computation by **ignoring** those rules used in deriving **both**  $T_1$  and  $T_2$ . Thus:

$$\begin{aligned}
 P(T_1) &\sim P(R_9)P(R_5)P(R_4) = 0.05 \cdot 0.35 \cdot 0.05 = 0.000875 \\
 P(T_2) &\sim P(R_{10})P(R_6) = 0.4 \cdot 0.2 = 0.08
 \end{aligned}$$

As expected,  $P(T_2) = 0.08 > P(T_1) = 0.000875$ , which confirms that  $T_2$  is more likely parse.

## Formalization

Recall the concept of arg max: In bigram PoS-tagging (lecture 13), we choose the tag  $t_i$  for word  $w_i$  that maximizes the probability of  $t_i$  given the tag of the previous word  $t_{i-1}$  and  $w_i$ .

$$t_i = \arg \max_j P(t_j | t_{i-1}, w_i)$$

Here we use arg max for specifying the most probable parse tree, given a sentence  $S$  and the set of its parse trees  $\tau(S)$ :

$$\hat{T}(S) = \arg \max_{T \in \tau(S)} P(T|S)$$

## Formalization

By definition,  $P(T|S) = \frac{P(T,S)}{P(S)}$ . Therefore:

$$\hat{T}(S) = \arg \max_{T \in \tau(S)} \frac{P(T,S)}{P(S)}$$

All parse trees are for the same  $S$ , so  $P(S)$  is the same for all of them:

$$\hat{T}(S) = \arg \max_{T \in \tau(S)} P(T, S)$$

We already know that  $P(T, S) = P(T)$ , therefore:

$$\hat{T}(S) = \arg \max_{T \in \tau(S)} P(T)$$

## Application 2: Language Modeling

A **language model** is a probabilistic model that assigns probabilities to strings. This is useful in a number of applications:

- **speech recognition**: most likely string for a speech signal;
- **spelling correction**: most likely string for an input with spelling mistakes;
- **text completion**, in texting and augmentative communication: most likely string for an initial string or otherwise underspecified input.

PCFGs can be used for language modeling. We will look at speech recognition as an example.

## Application 2: Language Modeling

Speech recognition is the task of finding the most probable sequence of words  $W = w_1, \dots, w_n$  for a speech signal, which is a sequence of acoustic observations  $O = o_1, \dots, o_t$ :

$$\hat{W} = \arg \max_W P(W|O)$$

Using Bayes' rule, we can write this as:

$$\hat{W} = \arg \max_W \frac{P(O|W)P(W)}{P(O)} = \arg \max_W P(O|W)P(W)$$

Here,  $P(W)$  is the **language model** and  $P(O|W)$  is the **acoustic model**.

How do we get  $P(W)$ ?

## Application 2: Language Modeling

Using a PCFG, we can compute the probability of any sentence  $S$  (ie, word string) by summing over all its possible parses:

$$P(S) = \sum_{T \in \tau(S)} P(T)$$

This task differs from **disambiguation** in that we want the most likely word string, independent of its parse trees.

If we assume that the input to the speech recognizer is a sentence, then we have  $P(W) = P(S) = \sum_{T \in \tau(S)} P(T)$ .

Such a model is referred to as a **structured language model**, in contrast to an  **$n$ -gram language model**, which computes  $P(W) = P(w_1)P(w_2|w_1) \dots P(w_n|w_{n-1})$  (for  $n = 2$ ).

## Summary

- A PCFG is a CFG with each rule annotated with a conditional probability;
- the sum of the probabilities of all rules that expand the same non-terminal must be 1;
- the probability of a parse tree is the product of the probabilities of all the rules used in this parse;
- the probability of a sentence is the sum of the probabilities of all its parses;
- applications for PCFGs:
  - disambiguation (selecting the most probable parse);
  - language modeling (selecting the most probable string).