

ACS Introduction to Syntax and Parsing

Lecture 5: PCFG Models for Statistical Parsing



UNIVERSITY OF
CAMBRIDGE

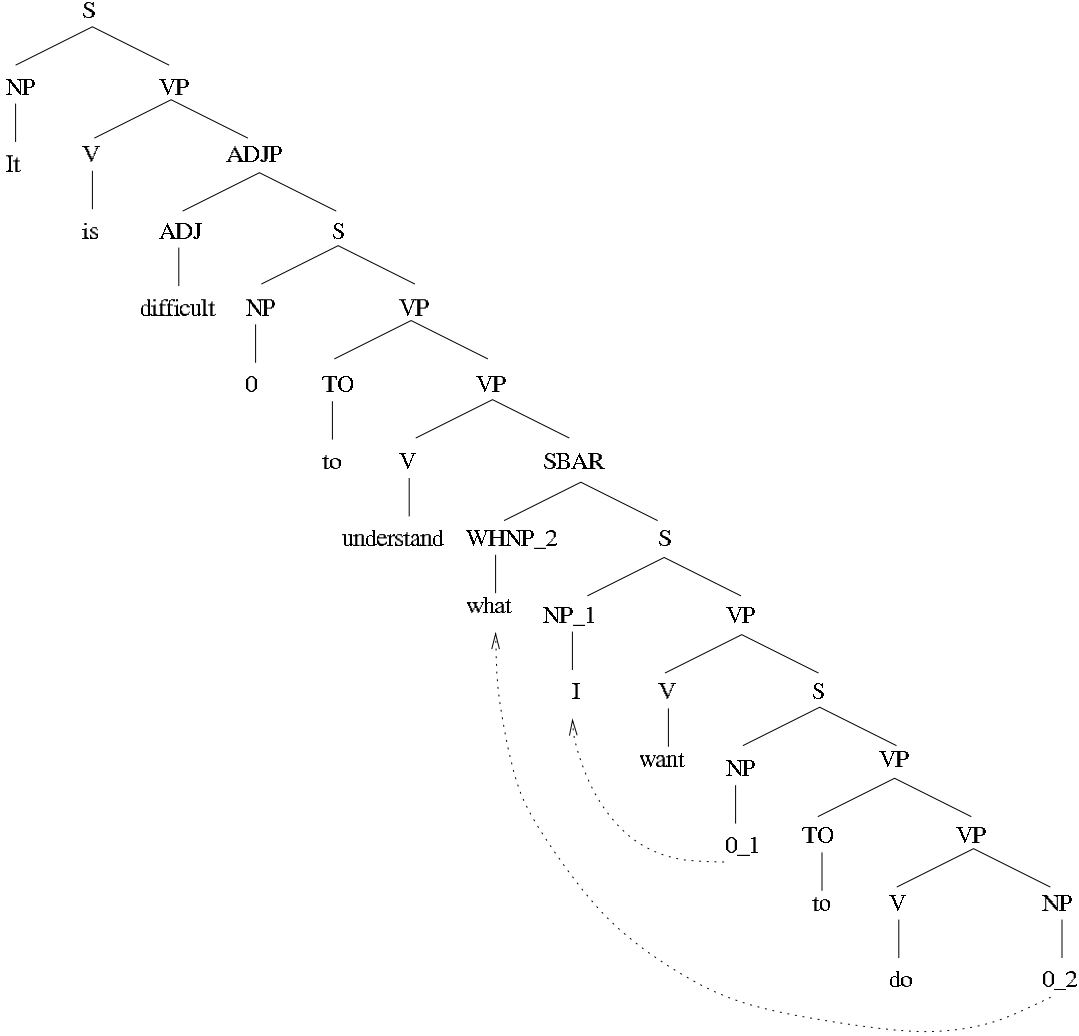
Stephen Clark

Natural Language and Information Processing (NLIP) Group

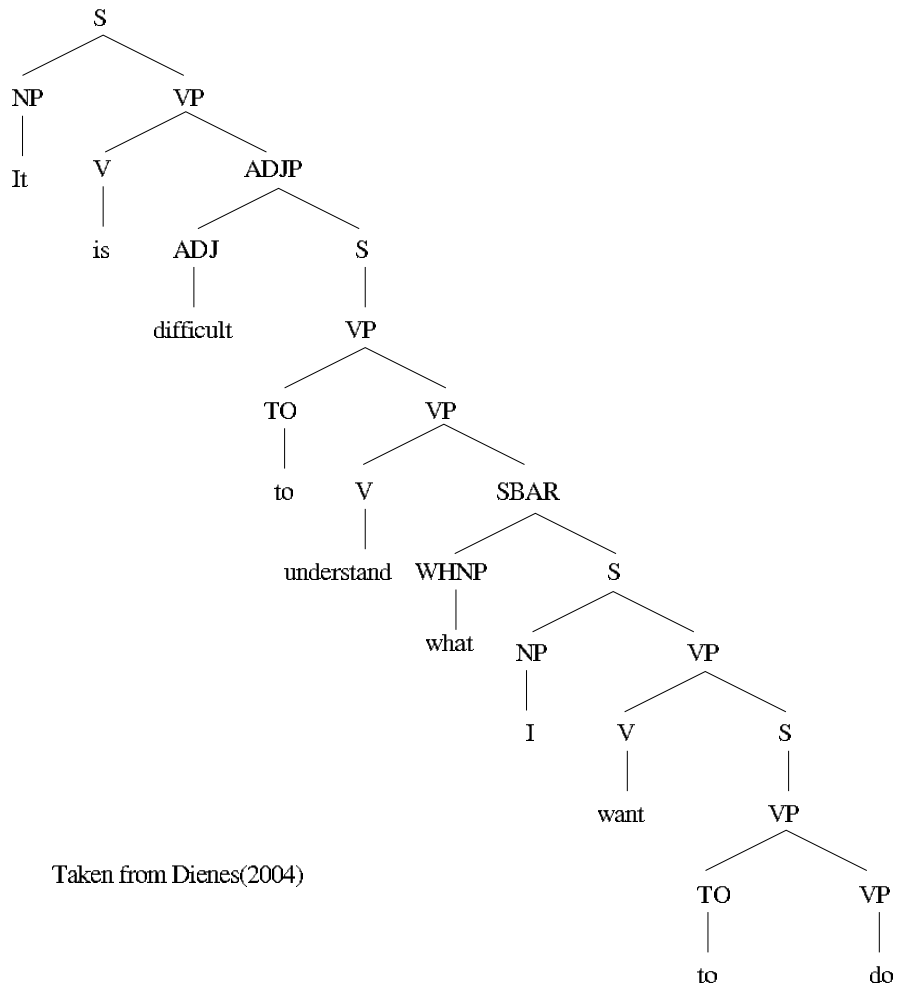
`sc609@cam.ac.uk`

-
- 40,000 WSJ newspaper sentences annotated with phrase-structure trees
 - The trees contain some predicate-argument information and traces
 - Created in the early 90s
 - Produced by automatically parsing the newspaper sentences followed by manual correction
 - Took around 3 years to create
 - Sparked a parsing “competition” which is still running today
 - leading some commentators to describe the last 15 years of NLP as the study of the WSJ

An Example Penn Treebank Tree



A Tree a typical PTB Parser would produce



Taken from Dienes(2004)

-
- What is the grammar which determines the set of legal syntactic structures for a sentence? How is that grammar obtained?
 - What is the algorithm for determining the set of legal parses for a sentence (given a grammar)?
 - What is the model for determining the plausibility of different parses for a sentence?
 - What is the algorithm, given the model and a set of possible parses, which finds the best parse?

$$T_{\text{best}} = \arg \max_T \text{Score}(T, S)$$

- Just two components:
 - the *model*: a function *Score* which assigns scores (probabilities) to tree, sentence pairs
 - the *parser*: the algorithm which implements the search for T_{best}
- Statistical parsing seen as more of a pattern recognition/Machine Learning problem plus search
 - the grammar is only implicitly defined by the training data and the method used by the parser for generating hypotheses

-
- Probabilistic approach would suggest the following *Score* function:

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

- Lots of research on different probability models for Penn Treebank trees
 - generative models, log-linear (maxent) models, perceptron, ...

$$\begin{aligned}\arg \max_T P(T|S) &= \arg \max_T \frac{P(T, S)}{P(S)} \\ &= \arg \max_T P(T, S)\end{aligned}$$

- Why model the joint probability when the sentence is given?
- Modelling a parse as a generative process allows the parse to be broken into manageable parts, for which the corresponding probabilities can be reliably estimated
- Probability estimation is easy for these sorts of models (ignoring smoothing issues)
 - maximum likelihood estimation = relative frequency estimation
- But choosing how to break up the parse is something of a black art

- A PCFG is a CFG with a set of probability distributions on the rules:

$$\sum_{\alpha} P(X \rightarrow \alpha) = 1$$

- Simple example (generating some ungrammatical sentences):

| | | | |
|--------------------------|-----|------------------------|-----|
| $S \rightarrow NP VP$ | 1.0 | $N \rightarrow man$ | 0.3 |
| $VP \rightarrow V$ | 0.1 | $N \rightarrow woman$ | 0.3 |
| $VP \rightarrow V NP$ | 0.7 | $V \rightarrow chased$ | 0.8 |
| $VP \rightarrow V NP NP$ | 0.2 | $V \rightarrow kissed$ | 0.2 |
| $NP \rightarrow Det N$ | 0.6 | | |
| $NP \rightarrow N$ | 0.4 | | |
| $Det \rightarrow the$ | 0.5 | | |
| $Det \rightarrow a$ | 0.5 | | |
| $N \rightarrow cat$ | 0.2 | | |
| $N \rightarrow dog$ | 0.2 | | |

- Joint probability of a tree T and sentence S is just the product of the probabilities of the rules used to build the tree
- For example, the probability of the tree associated with *the cat chased a dog*, using the previous grammar, is as follows:

$$\begin{aligned} &P(S) \times P(S \rightarrow NP VP|S) \times P(NP \rightarrow Det N|NP) \times P(VP \rightarrow V NP|VP) \times \\ &P(NP \rightarrow Det N|NP) \times P(Det \rightarrow the|Det) \times P(N \rightarrow cat|N) \times \\ &P(V \rightarrow chased|V) \times P(Det \rightarrow a|Det) \times P(N \rightarrow dog|N) \\ &= 1.0 \times 1.0 \times 0.6 \times 0.7 \times 0.6 \times 0.5 \times 0.2 \times 0.8 \times 0.5 \times 0.2 \end{aligned}$$

-
- Think of a random “generative process” as having generated the tree top-down, according to the rule probabilities
 - The probability above is just an application of the chain rule, plus independence assumptions (similar to the HMM for the tagging case)
 - Independence assumption is that the probability of rewriting a non-terminal in a particular way only depends on the non-terminal, and nothing else in the tree
 - Similar idea to the notion of context-freeness in the non-probabilistic grammar case

- A CFG can be read directly off the trees in the PTB
- For the tree on p.4, for example, we would get rules such as:

$S \rightarrow NP VP$

$NP \rightarrow It$

$VP \rightarrow V ADJP$

$S \rightarrow VP$

- Estimating the probabilities is easy!

$$\hat{P}(S \rightarrow NP VP | S) = freq(S \rightarrow NP VP) / freq(S)$$

- And relative frequency estimates are maximum likelihood estimates in this case (as they were for the HMM tagging model)

-
- The main problem is that a PCFG only has *structural* probabilities
 - The words only have an effect at the leaves of the tree (by which time almost all of the tree has been generated)
 - Consider trying to distinguish the parses for *John ate the pizza with a fork* and *John ate the pizza with the anchovies* using a PCFG

-
- Chapter 11 of Manning and Schuetze
 - Michael Collins (1999), Head-Driven Statistical Models for Natural Language Parsing, UPenn PhD thesis