SFU NatLangLab

Natural Language Processing

Anoop Sarkar
anoopsarkar.github.io/nlp-class

Simon Fraser University

October 20, 2017

Natural Language Processing

Anoop Sarkar
anoopsarkar.github.io/nlp-class

Simon Fraser University

Part 1: Ambiguity

Context Free Grammars and Ambiguity

$$
\begin{array}{ccc}
S & \rightarrow & NP\ VP \\
VP & \rightarrow & V\ NP \\
VP & \rightarrow & VP\ PP \\
PP & \rightarrow & P\ NP \\
NP & \rightarrow & NP\ PP \\
NP & \rightarrow & Calvin \\
NP & \rightarrow & monsters \\
NP & \rightarrow & school \\
V & \rightarrow & imagined \\
P & \rightarrow & in
\end{array}
$$

What is the analysis using the above grammar for:
Calvin imagined monsters in school

Context Free Grammars and Ambiguity

Calvin imagined monsters in school

```
(S (NP Calvin)
   (VP (V imagined)
       (NP (NP monsters)
           (PP (P in)
               (NP school)))))
```

```
(S (NP Calvin)
   (VP (VP (V imagined)
           (NP monsters))
       (PP (P in)
           (NP school))))
```

Which one is more plausible?

Ambiguity Kills (your parser)

natural language learning course
(run demos/parsing-ambiguity.py)

```
((natural language) (learning course))
(((natural language) learning) course)
((natural (language learning)) course)
(natural (language (learning course)))
(natural ((language learning) course))
```

- Some difficult issues:
  - Which one is more plausible?
  - How many analyses for a given input?
  - Computational complexity of parsing language

Treebanks

- What is the CFG that can be extracted from this single tree:

```
(S   (NP (Det the) (NP man))
     (VP (VP (V played)
                 (NP (Det a) (NP game)))
           (PP (P with)
                 (NP (Det the) (NP dog)))))
```

PCFG

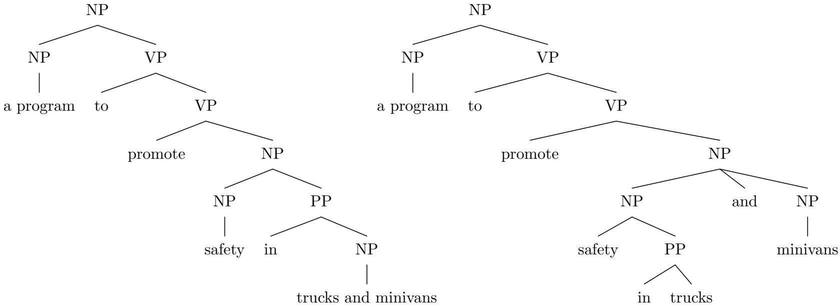
$$
\begin{array}{rcll}
S & \rightarrow & NP\ VP & c = 1 \\
NP & \rightarrow & Det\ NP & c = 3 \\
NP & \rightarrow & man & c = 1 \\
NP & \rightarrow & game & c = 1 \\
NP & \rightarrow & dog & c = 1 \\
VP & \rightarrow & VP\ PP & c = 1 \\
VP & \rightarrow & V\ NP & c = 1 \\
PP & \rightarrow & P\ NP & c = 1 \\
Det & \rightarrow & the & c = 2 \\
Det & \rightarrow & a & c = 1 \\
V & \rightarrow & played & c = 1 \\
P & \rightarrow & with & c = 1
\end{array}
$$

- We can do this with multiple trees. Simply count occurrences of CFG rules over all the trees.
- A repository of such trees labelled by a human is called a TreeBank.

Ambiguity

- Part of Speech ambiguity
  `saw` → noun
  `saw` → verb
- Structural ambiguity: Prepositional Phrases
  `I saw (the man) with the telescope`
  `I saw (the man with the telescope)`
- Structural ambiguity: Coordination
  `a program to promote safety in ((trucks) and (minivans))`
  `a program to promote ((safety in trucks) and (minivans))`
  `((a program to promote safety in trucks) and (minivans))`

Ambiguity ← attachment choice in alternative parses



Ambiguity in Prepositional Phrases

- noun attach: *I bought the shirt with pockets*
- verb attach: *I washed the shirt with soap*
- As in the case of other attachment decisions in parsing: it depends on the meaning of the entire sentence – needs world knowledge, etc.
- Maybe there is a simpler solution: we can attempt to solve it using heuristics or associations between words

Structure Based Ambiguity Resolution

- Right association: a constituent (NP or PP) tends to attach to another constituent immediately to its right (Kimball 1973)
- Minimal attachment: a constituent tends to attach to an existing non-terminal using the fewest additional syntactic nodes (Frazier 1978)
- These two principles make opposite predictions for prepositional phrase attachment
- Consider the grammar:

$$VP \rightarrow V\ NP\ PP \qquad (1)$$

$$NP \rightarrow NP\ PP \qquad (2)$$

for input: *I [$_{VP}$ saw [$_{NP}$ the man . . . [$_{PP}$ with the telescope]*, RA predicts that the PP attaches to the NP, i.e. use rule (2), and MA predicts V attachment, i.e. use rule (1)

Structure Based Ambiguity Resolution

- Garden-paths look structural:
  The emergency crews hate most is domestic violence
- Neither MA or RA account for more than 55% of the cases in real text
- Psycholinguistic experiments using eyetracking show that humans resolve ambiguities as soon as possible in the left to right sequence using the words to disambiguate
- Garden-paths are caused by a combination of lexical and structural effects:
  The flowers delivered for the patient arrived

Ambiguity Resolution: Prepositional Phrases in English

- Learning Prepositional Phrase Attachment: Annotated Data

v	n1	p	n2	Attachment
join	board	as	director	V
is	chairman	of	N.V.	N
using	crocidolite	in	filters	V
bring	attention	to	problem	V
is	asbestos	in	products	N
making	paper	for	filters	N
including	three	with	cancer	N
⋮	⋮	⋮	⋮	⋮

Prepositional Phrase Attachment

Method	Accuracy
Always noun attachment	59.0
Most likely for each preposition	72.2
Average Human (4 head words only)	88.2
Average Human (whole sentence)	93.2

Some other studies

- **Toutanova, Manning, and Ng, 2004**: 87.54% using some external knowledge (word classes)
- **Merlo, Crocker and Berthouzoz, 1997**: test on multiple PPs
  - generalize disambiguation of 1 PP to 2-3 PPs
  - 14 structures possible for 3PPs assuming a single verb
  - all 14 are attested in the Penn WSJ Treebank
  - 1PP: 84.3%   2PP: 69.6%   3PP: 43.6%
  - **This experiment is still only part of the real problem faced in parsing English**
  - Other sources of ambiguity in other languages

Natural Language Processing

Anoop Sarkar
anoopsarkar.github.io/nlp-class

Simon Fraser University

Part 2: Context Free Grammars

Parts of Speech

- We have seen that individual words can be classified into groups or classes that we call **parts of speech**
  - Determiners: *a, the*
  - Verbs: *arrive, attracts, love, sit*
  - Prepositions: *of, by, in, outside, on*
  - Nouns: *he, she, it, San, Diego*
- But these individual words can group together to form larger groups which possess meaning when put together, e.g. *San Diego*, *the man outside the building*

Constituents

- Let's consider the grouping of words into **noun phrases**
  - *three parties from Brooklyn*
  - *a high class spot such as Mindy's*
  - *they*
  - *Harry the Horse*
  - *the fact that he came into the Hot Box*
  - *swimming on a hot day*

Chunking Noun Phrases: Not as easy as it seems

- Finding noun phrases can be treated as finding a sequence of words that is a noun phrase (the **chunking** approach). Finding chunks is not trivial:
  - *(NNP San) (NNP Diego)*
  - *(NNPS Wednesdays)*
  - *(DT the) (NN company) (POS 's) (VBN refocused) (NN direction)*
  - *(DT the) (NN government) (VBZ 's) (VBG dawdling)*
  - $*$ *(DT The) (NNP Dow) (NNP Jones) (VBZ is) (VBG swimming) (IN in) (NN tech) (NNS stocks)*

Recursion in Regular Languages

- Let's attempt to provide a regular expression grammar for a subset of all the possible noun phrases
- Consider the noun phrases: *the man in the park*, *the person with the big head in the park*, *the unicorn in the garden inside the dream with a strange mark on the head*, ...
- These are simple noun phrases that have prepositional phrases (PP, for short) modifying nouns. PPs are another example of a constituent, but now we need to combine them with NPs

Recursion in Regular Languages

- Consider the noun phrases: *the man in the park*, *the person with the big head in the park*, *the unicorn in the garden inside the dream with a strange mark on the head*, ...
- (NP) (PP)* $\rightarrow$ (Det N) (PP)$*$ $\rightarrow$ (Det N) (P NP)*
- (Det N) (P (Det N)) PP* $\rightarrow$ (Det N) (P (Det N))*
- So, it's possible, but it gets ugly fast, let's widen our view of what can occur inside NPs.

Recursion in Regular Languages

- Let's call (Det N) a **basal** NP and now consider that (Det N) is not the only base NP that is possible: (N) or (A N) or (A^+ N) or even:
  (D A^* N POS N) *the short man 's dream* ...
- So this means that we can now have (P (N)) or (P (A N)) or (P (A^+ N)) or ...
- Each former type of NP can be modified by each latter type of PP
- What is the only way to rescue the regular expression approach?
  combinatorial explosion of combinations

Context-Free Languages

- Here is a simple Context Free Grammar that does word morphology. The CFG is more *elegant* and smaller than the equivalent regular grammar (consider $*$joyable, $*$richment):

$$\begin{array}{rcl} V & \rightarrow & X \\ A & \rightarrow & X \textit{ -able} \mid X \textit{ -ment} \\ X & \rightarrow & \textit{en- } NA \\ NA & \rightarrow & \textit{joy} \mid \textit{rich} \end{array}$$

- This is an engineering argument. However, it is related to the problem of describing the human learning process. Certain aspects of language are learned all at once not individually for each case.
  e.g., learning *enjoyment* automatically if *enrichment* was learnt

Context-Free Grammars

- Recall the trinity of regular expressions, finite state automata and regular languages
- Now we generalize to context free grammars, pushdown automata and context-free languages
- Just like before, certain closure properties hold, the union of two CFLs is also a CFL, etc.
  except for one property that is true in RLs but not in CFLs
- CFLs are not closed under Intersection

Context-Free Grammars

- Determinization is also not always possible for pushdown automata
  surprising fact about CFGs is that you can construct one that is *inherently* ambiguous
- Particular relevance for natural languages, compare with artificial grammars that we use routinely when we use a programming language (what happens in cases of ambiguity in finite state automata?)
- Deterministic vs. non-deterministic parsing (more on this later)

Context-Free Grammars

- A CFG is a 4-tuple: (N, T, P, S), where
  - N is a set of non-terminal symbols,
  - T is a set of terminal symbols which can include the empty string ϵ. T is analogous to Σ the alphabet in FSAs.
  - P is a set of rules of the form $A \rightarrow \alpha$, where $A \in N$ and $\alpha \in \{N \cup T\}^*$
  - S is a set of start symbols, $S \in N$

Context-Free Grammars

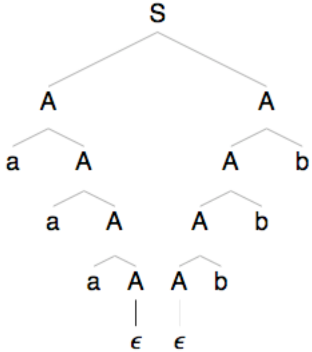
- Here's an example of a CFG, let's call this one G:
    1. $S \rightarrow a\ S\ b$
    2. $S \rightarrow \epsilon$
- What is the language of this grammar, which we will call $L(G)$, the set of strings *generated* by this grammar How? Notice that there cannot be any FSA that corresponds exactly to this set of strings $L(G)$ Why?
- What is the *tree set* or derivations produced by this grammar?

Context-Free Grammars

- This notion of generating both the strings and the trees is an important one for Computational Linguistics
- Consider the trees for the grammar G':
  $P = \{S \rightarrow A\ A, A \rightarrow aA, A \rightarrow A\ b, A \rightarrow \epsilon\}$,
  $\Sigma = \{a, b\}, N = \{S, A\}, T = \{a, b, \epsilon\}, S = \{S\}$
- Why is it called *context-free* grammar?

Context-Free Grammars

- Can the grammar G' produce only trees of the kind shown below?



Context-Free Grammars

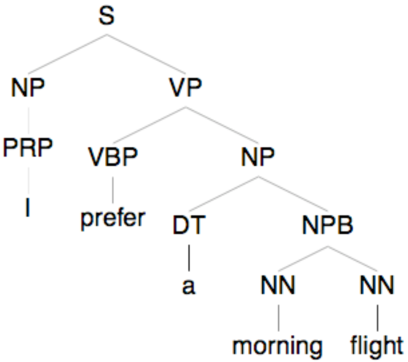
- We will come back to this issue when we try to figure out whether human languages are more powerful than CFLs.
- The distinction between strings and the trees (or any kind of structural description) is called *weak* vs. *strong* generative capacity.

Parse Trees

Consider the grammar with rules:

$$
\begin{aligned}
S &\rightarrow NP\ VP \\
NP &\rightarrow PRP \\
NP &\rightarrow DT\ NPB \\
VP &\rightarrow VBP\ NP \\
NPB &\rightarrow NN\ NN \\
PRP &\rightarrow I \\
VBP &\rightarrow prefer \\
DT &\rightarrow a \\
NN &\rightarrow morning \\
NN &\rightarrow flight
\end{aligned}
$$

Parse Trees



Parse Trees: Equivalent Representations

- (S (NP (PRP I)) (VP (VBP prefer) (NP (DT a) (NPB (NN morning) (NN flight)))))
- $[_S$ $[_{NP}$ $[_{PRP}$ I]] $[_{VP}$ $[_{VBP}$ prefer] $[_{NP}$ $[_{DT}$ a] $[_{NPB}$ $[_{NN}$ morning] $[_{NN}$ flight]]]]]

Ambiguous Grammars

- $S \rightarrow S\ S$
- $S \rightarrow a$
- Given the above rules, consider the input *aaa*, what are the valid parse trees?
- Now consider the input *aaaa*

Natural Language Processing

Anoop Sarkar
anoopsarkar.github.io/nlp-class

Simon Fraser University

Part 3: Probabilistic Context Free Grammars

Probabilistic CFG (PCFG)

$$
\begin{array}{ccccc}
S & \rightarrow & NP\ VP & 1 \\
VP & \rightarrow & V\ NP & 0.9 \\
VP & \rightarrow & VP\ PP & 0.1 \\
PP & \rightarrow & P\ NP & 1 \\
NP & \rightarrow & NP\ PP & 0.25 \\
NP & \rightarrow & Calvin & 0.25 \\
NP & \rightarrow & monsters & 0.25 \\
NP & \rightarrow & school & 0.25 \\
V & \rightarrow & imagined & 1 \\
P & \rightarrow & in & 1
\end{array}
$$

$$P(input) = \sum_{tree} P(tree \mid input)$$

$$P(Calvin\ imagined\ monsters\ in\ school) = ?$$

$$\text{Notice that } P(VP \rightarrow V\ NP) + P(VP \rightarrow VP\ PP) = 1.0$$

Probabilistic CFG (PCFG)

$$P(\textit{Calvin imagined monsters in school}) = ?$$

```
(S (NP Calvin)
   (VP (V imagined)
       (NP (NP monsters)
           (PP (P in)
               (NP school)))))

(S (NP Calvin)
   (VP (VP (V imagined)
           (NP monsters))
       (PP (P in)
           (NP school))))
```

Probabilistic CFG (PCFG)

```
(S (NP Calvin)
   (VP (V imagined)
       (NP (NP monsters)
           (PP (P in)
               (NP school)))))
```

$$
\begin{aligned}
P(\mathit{tree}_1) &= P(S \rightarrow NP\ VP) \times P(NP \rightarrow \mathit{Calvin}) \times P(VP \rightarrow V\ NP) \times \\
&\quad P(V \rightarrow \mathit{imagined}) \times P(NP \rightarrow NP\ PP) \times P(NP \rightarrow \mathit{monsters}) \times \\
&\quad P(PP \rightarrow P\ NP) \times P(P \rightarrow \mathit{in}) \times P(NP \rightarrow \mathit{school}) \\
&= 1 \times 0.25 \times 0.9 \times 1 \times 0.25 \times 0.25 \times 1 \times 1 \times 0.25 = .003515625
\end{aligned}
$$

Probabilistic CFG (PCFG)

```
(S (NP Calvin)
   (VP (VP (V imagined)
           (NP monsters))
       (PP (P in)
           (NP school))))
```

$$
\begin{aligned}
P(\mathit{tree}_2) &= P(S \rightarrow NP\ VP) \times P(NP \rightarrow \mathit{Calvin}) \times P(VP \rightarrow VP\ PP) \times \\
&\quad P(VP \rightarrow V\ NP) \times P(V \rightarrow \mathit{imagined}) \times P(NP \rightarrow \mathit{monsters}) \times \\
&\quad P(PP \rightarrow P\ NP) \times P(P \rightarrow \mathit{in}) \times P(NP \rightarrow \mathit{school}) \\
&= 1 \times 0.25 \times 0.1 \times 0.9 \times 1 \times 0.25 \times 1 \times 1 \times 0.25 = .00140625
\end{aligned}
$$

Probabilistic CFG (PCFG)

$$\begin{aligned}
P(\textit{Calvin imagined monsters in school}) &= P(\textit{tree}_1) + P(\textit{tree}_2) \\
&= .003515625 + .00140625 \\
&= .004921875 \\
\text{Most likely tree is } \textit{tree}_1 &= \underset{\textit{tree}}{\arg\max}\ P(\textit{tree} \mid \textit{input})
\end{aligned}$$

```
(S (NP Calvin)
   (VP (V imagined)
       (NP (NP monsters)
           (PP (P in)
               (NP school)))))
```

```
(S (NP Calvin)
   (VP (VP (V imagined)
           (NP monsters))
       (PP (P in)
           (NP school))))
```

PCFG

- Central condition: $\sum_{\alpha} P(A \rightarrow \alpha) = 1$
- Called a *proper* PCFG if this condition holds
- Note that this means $P(A \rightarrow \alpha) = P(\alpha \mid A) = \frac{f(A,\alpha)}{f(A)}$
- $P(T \mid S) = \frac{P(T,S)}{P(S)} = P(T,S) = \prod_i P(RHS_i \mid LHS_i)$

PCFG

- What is the PCFG that can be extracted from this single tree:

```
(S   (NP (Det the) (NP man))
     (VP (VP (V played)
             (NP (Det a) (NP game)))
         (PP (P with)
             (NP (Det the) (NP dog)))))
```

- How many different rhs α exist for $A \rightarrow \alpha$ where A can be S, NP, VP, PP, Det, N, V, P

PCFG

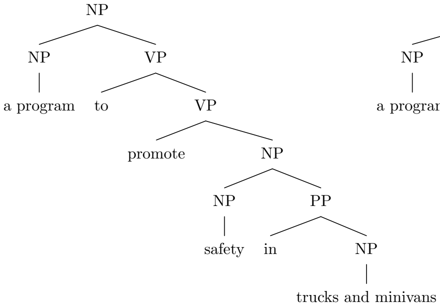
$$
\begin{array}{lcccll}
S & \rightarrow & NP\ VP & c = 1 & p = 1/1 & = 1.0 \\
NP & \rightarrow & Det\ NP & c = 3 & p = 3/6 & = 0.5 \\
NP & \rightarrow & man & c = 1 & p = 1/6 & = 0.1667 \\
NP & \rightarrow & game & c = 1 & p = 1/6 & = 0.1667 \\
NP & \rightarrow & dog & c = 1 & p = 1/6 & = 0.1667 \\
VP & \rightarrow & VP\ PP & c = 1 & p = 1/2 & = 0.5 \\
VP & \rightarrow & V\ NP & c = 1 & p = 1/2 & = 0.5 \\
PP & \rightarrow & P\ NP & c = 1 & p = 1/1 & = 1.0 \\
Det & \rightarrow & the & c = 2 & p = 2/3 & = 0.67 \\
Det & \rightarrow & a & c = 1 & p = 1/3 & = 0.33 \\
V & \rightarrow & played & c = 1 & p = 1/1 & = 1.0 \\
P & \rightarrow & with & c = 1 & p = 1/1 & = 1.0
\end{array}
$$

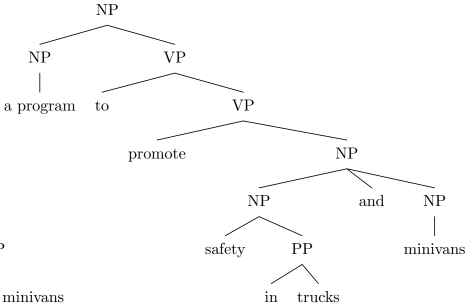
- We can do this with multiple trees. Simply count occurrences of CFG rules over all the trees.
- A repository of such trees labelled by a human is called a TreeBank.

Ambiguity

- Part of Speech ambiguity
  `saw` → noun
  `saw` → verb
- Structural ambiguity: Prepositional Phrases
  `I saw (the man) with the telescope`
  `I saw (the man with the telescope)`
- Structural ambiguity: Coordination
  `a program to promote safety in ((trucks) and (minivans))`
  `a program to promote ((safety in trucks) and (minivans))`
  `((a program to promote safety in trucks) and (minivans))`

Ambiguity ← attachment choice in alternative parses

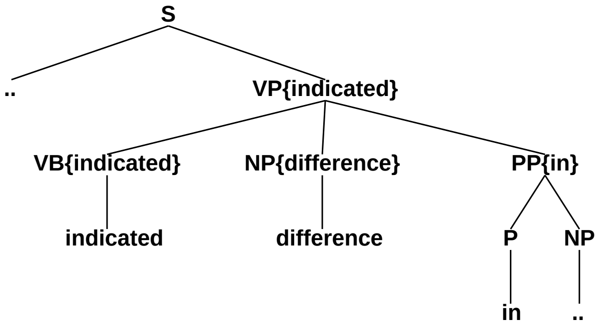




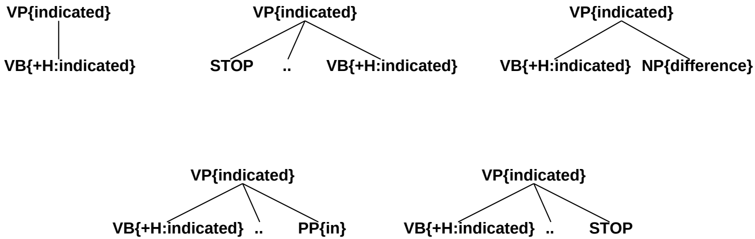
Parsing as a machine learning problem

- S = a sentence
  T = a parse tree
  A statistical parsing model defines $P(T \mid S)$
- Find best parse: $\underset{T}{\arg\max}\ P(T \mid S)$
- $P(T \mid S) = \frac{P(T,S)}{P(S)} = P(T,S)$
- Best parse: $\underset{T}{\arg\max}\ P(T,S)$
- e.g. for PCFGs: $P(T,S) = \prod_{i=1 \dots n} P(\mathrm{RHS}_i \mid \mathrm{LHS}_i)$

Adding Lexical Information to PCFG



Adding Lexical Information to PCFG (Collins 99, Charniak 00)



$$
\begin{aligned}
&P_h(\texttt{VB} \mid \texttt{VP}, \texttt{indicated}) \times P_l(\texttt{STOP} \mid \texttt{VP}, \texttt{VB}, \texttt{indicated}) \times \\
&P_r(\texttt{NP(difference)} \mid \texttt{VP}, \texttt{VB}, \texttt{indicated}) \times \\
&P_r(\texttt{PP(in)} \mid \texttt{VP}, \texttt{VB}, \texttt{indicated}) \times \\
&P_r(\texttt{STOP} \mid \texttt{VP}, \texttt{VB}, \texttt{indicated})
\end{aligned}
$$

Evaluation of Parsing

- Consider a candidate parse to be evaluated against the truth (or gold-standard parse):

```
candidate: (S (A (P this) (Q is)) (A (R a) (T test)))
gold:      (S (A (P this)) (B (Q is) (A (R a) (T test))))
```

- In order to evaluate this, we list all the constituents

Candidate	Gold
(0,4,S)	(0,4,S)
(0,2,A)	(0,1,A)
(2,4,A)	(1,4,B)
	(2,4,A)

- Skip spans of length 1 which would be equivalent to part of speech tagging accuracy.
- Precision is defined as $\frac{\#correct}{\#proposed} = \frac{2}{3}$ and recall as $\frac{\#correct}{\#in\ gold} = \frac{2}{4}$.
- Another measure: crossing brackets,

```
candidate: [ an [incredibly expensive] coat ] (1 CB)
gold:      [ an [incredibly [expensive coat]]
```

Evaluation of Parsing

$$
\begin{aligned}
\text{Bracketing recall } R &= \frac{\text{num of correct constituents}}{\text{num of constituents in the goldfile}} \\
\text{Bracketing precision } P &= \frac{\text{num of correct constituents}}{\text{num of constituents in the parsed file}} \\
\text{Complete match} &= \text{\% of sents where recall \& precision are both 100\%} \\
\text{Average crossing} &= \frac{\text{num of constituents crossing a goldfile constituent}}{\text{num of sents}} \\
\text{No crossing} &= \text{\% of sents which have 0 crossing brackets} \\
\text{2 or less crossing} &= \text{\% of sents which have} \leq 2 \text{ crossing brackets}
\end{aligned}
$$

Statistical Parsing Results

$$\text{F1-score} = 2\frac{\mathit{precision} \cdot \mathit{recall}}{\mathit{precision} + \mathit{recall}}$$

System	$\leq 100\mathit{wds}$ F1-score
Shift-Reduce (Magerman, 1995)	84.14
PCFG with Lexical Features (Collins, 1999)	88.19
PCFG with Lexical Features (Charniak, 1999)	89.54
n-best Re-ranking (Collins, 2000)	89.74
Unlexicalized Berkeley parser (Petrov et al, 2007)	90.10
n-best Re-ranking (Charniak and Johnson, 2005)	91.02
Tree-insertion grammars (Carreras, Collins, Koo, 2008)	91.10
Ensemble *n*-best Re-ranking (Johnson and Ural, 2010)	91.49
Forest Re-ranking (Huang, 2010)	91.70
Unlabeled Data with Self-Training (McCloskey et al, 2006)	92.10

Practical Issues: Beam Thresholding and Priors

- Probability of nonterminal X spanning $j \dots k$: $N[X, j, k]$
- Beam Thresholding compares $N[X, j, k]$ with every other Y where $N[Y, j, k]$
- But what should be compared?
- Just the *inside probability*: $P(X \stackrel{*}{\Rightarrow} t_j \dots t_k)$? written as $\beta(X, j, k)$
- Perhaps $\beta(\texttt{FRAG}, 0, 3) > \beta(\texttt{NP}, 0, 3)$, but NPs are much more likely than FRAGs in general

Practical Issues: Beam Thresholding and Priors

- The correct estimate is the *outside probability*:

$$P(S \stackrel{*}{\Rightarrow} t_1 \dots t_{j-1}\ X\ t_{k+1} \dots t_n)$$

  written as $\alpha(X, j, k)$
- Unfortunately, you can only compute $\alpha(X, j, k)$ efficiently after you finish parsing and reach $(S, 0, n)$

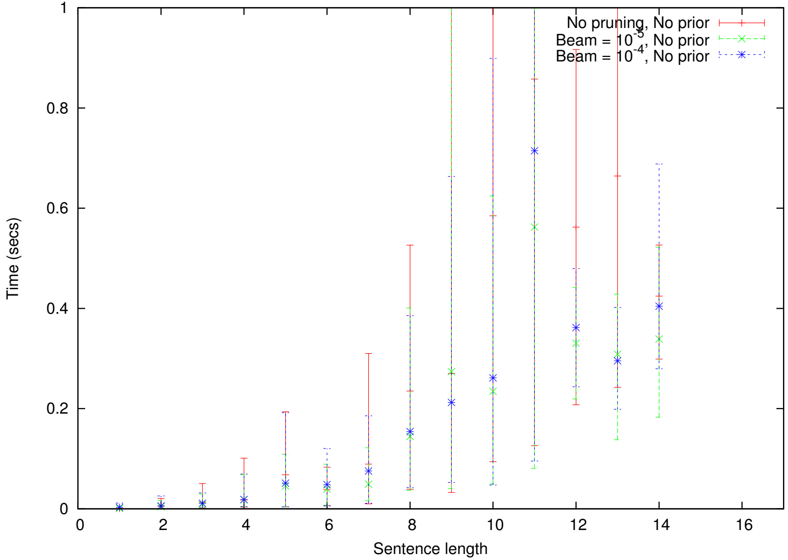
Practical Issues: Beam Thresholding and Priors

- To make things easier we multiply the prior probability $P(X)$ with the inside probability
- In beam Thresholding we compare every new insertion of X for span j, k as follows:
  Compare $P(X) \cdot \beta(X, j, k)$ with the most probable Y $P(Y) \cdot \beta(Y, j, k)$
- Assume Y is the most probable entry in j, k, then we compare

$$\text{beam} \cdot P(Y) \cdot \beta(Y, j, k) \tag{3}$$
$$P(X) \cdot \beta(X, j, k) \tag{4}$$

- If $(4) < (3)$ then we prune X for this span j, k
- beam is set to a small value, say 0.001 or even 0.01.
- As the beam value increases, the parser speed increases (since more entries are pruned).
- A simpler (but not as effective) alternative to using the beam is to keep only the top K entries for each span j, k

Experiments with Beam Thresholding



Experiments with Beam Thresholding