"... the overall goal is to produce a system that can place a provably useful structure over arbitrary sentences, that is, to build a parser."

tags: chunking, grammar induction, parser
12.1.1 Parsing for disambiguation

- Probabilities for determining the sentence
- Probabilities for speedier parsing
- *Probabilities for choosing between parses*
12.1.2 Treebanks

- A collection of example parses = treebank
- Penn Treebank tree:

```
(S
 (NP (PRP I))
 (VP (VBD saw)
  (NP (DT the) (NN man))
  (PP (IN with)
   (NP (DT the) (NN telescope))))
 (. .))
```

- NP-over-NP is wrong by syntactic theories
- but captures the notion of chunks

tags: treebank, penn treebank, chunking
12.1.3 Parsing models vs. Language models

- Parsing model: evaluates the probability of trees $t$ for a sentence $s$

$$
\hat{t} = \arg \max_t P(t|s, G)
$$

- Language model: assigns a probability to all trees generated by a grammar

$$
\hat{t} = \arg \max_t P(t|s) = \arg \max_t \frac{P(t,s)}{P(s)} = \arg \max_t P(t,s)
$$

- Language models appear to provide a better foundation for modeling
12.1.4 Weakening independence assumptions

- Context and independence assumptions
  - TV vs. Bar, who, immediate prior context
- PCFGs lack lexicalization
- Probabilities dependent on structural context

<table>
<thead>
<tr>
<th>Expansion</th>
<th>% as 1st Obj</th>
<th>% as 2nd Obj</th>
</tr>
</thead>
<tbody>
<tr>
<td>NP → NNS</td>
<td>7,5%</td>
<td>0,2%</td>
</tr>
<tr>
<td>NP → PRP</td>
<td>13,4%</td>
<td>0,9%</td>
</tr>
<tr>
<td>NP → NP PP</td>
<td>12,2%</td>
<td>14,4%</td>
</tr>
<tr>
<td>NP → DT NN</td>
<td>10,4%</td>
<td>13,3%</td>
</tr>
<tr>
<td>NP → NNP</td>
<td>4,5%</td>
<td>5,9%</td>
</tr>
<tr>
<td>NP → NNP</td>
<td>3,9%</td>
<td>9,2%</td>
</tr>
<tr>
<td>NP → JJ NN</td>
<td>1,1%</td>
<td>10,4%</td>
</tr>
<tr>
<td>NP → NP SBAR</td>
<td>0,3%</td>
<td>5,1%</td>
</tr>
</tbody>
</table>

tags: priming, lexicalization
12.1.5 Tree probabilities and derivational prob.

- Canonical derivation
- History-based grammars

Tags: canonical derivation, history-based grammars
12.1.6 There's more than one way to do it

• Probabilistic left-corner grammars

comment: Initialization
Place the predicted start symbol S on top of the stack

comment: Parser
while (an action is possible) do one of the following
  actions
    [Shift] Put the next input symbol on top of the stack
    [Attach] If $\alpha\alpha$ is on top of the stack, remove both
    [Project] If $\alpha$ is on top of the stack ans $A \rightarrow \alpha\gamma$, replace $\alpha$ by $\gamma A$
  endactions
end

comment: Termination
if empty(input) ^ empty(stack)
  then
    exit success
  else
    exit failure
fi

tags: top-down parsing, left corner parsers, shifting, projecting, attaching
12.1.7 Phrase structure / dependency grammars

- [phrase structure] is not really needed to construct an understanding of sentences

\[
\begin{align*}
\text{a. phrase structure model} & \quad \text{N}^x \\
\text{n}^y & \quad \text{model} \\
\text{phrase structure} & \\
\text{b. phrase structure model} & \quad \text{N}^u \\
\text{phrase} & \quad \text{n}^v \\
\text{structure model}
\end{align*}
\]

tags: dependency grammar, head
12.1.8 Evaluation

- Objective criterion
- Tree accuracy
- Exact match
- PARSEVAL measures
- Precision
- Recall
- Crossing Brackets
12.1.9 Equivalent models

• Three ways of thinking of a PCFG model
  – as using more of the derivational history
  – as using more of the parse tree context
  – as enriching the category labels

• ”.. it is frequently easier to write a quick program to produce transformed trees than to write a new probabilistic parser“
12.1.10 Building Parsers: Search methods

- Tableau / Viterbi Algorithm
- Stack decoding algorithm
  - Uniform-cost search
  - Beam search
- A* search
  - Best-first search
  - A* search
  - Optimally efficient
- Other methods
12.1.11 Use of the geometric mean

- Multiplying probabilities -> errors accumulate
- Ad hoc scoring functions
  - Treating the symptoms not the problems
- PCFGs give higher probability to smaller trees