Creating a Treebank for CCG

- A CCG treebank consists of (sentence, CCG analysis) pairs
- The CCG analysis is likely to be a derivation, and may also contain additional information such as predicate-argument dependencies
- The treebank is useful for:
  - deriving a wide-coverage grammar (or extending an existing one)
  - inducing statistical disambiguation models
- How can we build a CCG treebank?
  - manually from scratch (or at least by correcting the output of an existing CCG parser)
  - by automatically transforming the analyses from an existing treebank (e.g. the Penn Treebank) into CCG derivations
- Manual creation of a treebank is expensive so we choose the 2nd option
• 50k sentences/1M words of WSJ text annotated with phrase-structure (PS) trees

• How might we turn this into a CCG treebank?

• What information do we need in the PS trees?
  – head information
  – argument/adjunct distinction (so we can derive the CCG categories)
  – trace information/extracted arguments so we can analyse long-range dependencies
It is difficult to understand what I want to do.

The diagram represents a parse tree of the sentence "It is difficult to understand what I want to do." Each node in the tree corresponds to a part of speech or a phrase, and the lines show the syntactic relationships between the elements of the sentence.
The Basic Translation Algorithm

- Ignoring long-range dependency_trace information, the basic algorithm is straightforward:
  - foreach tree $\tau$
    * determineConstituentTypes($\tau$)
    * makeBinary($\tau$)
    * assignCategories($\tau$)
• Constituent type is either head, complement or adjunct

• This information is not marked explicitly in the PTB, but can be inferred (using heuristic rules) based on:
  
  – *function tags* in the PTB, e.g. –\texttt{SBJ} (subject), –\texttt{TMP} (temporal modifier), –\texttt{DIR} (direction)

  – constituent label of a node and its parent (e.g. NP daughters of VPs are complements, unless they carry a function tag such as –\texttt{LOC}, –\texttt{DIR}, –\texttt{TMP} and so on)

• Appendix A of Collins’ thesis gives a list of the head rules

• See p.362 of H&S 2007 and Appendix A of CCGbank manual
• A PTB tree is not binarized, whereas a CCG derivation is
• Insert dummy nodes into the tree such that:
  – all children to the left of the head branch off in a right-branching tree
  – all children to the right of the head branch off in a left-branching tree
• Some PTB structures are very flat, e.g. compound noun phrases – in
  the compound noun case we just assume a right-branching structure
  (but see Vadas and Curran for inserting NP structure into the PTB)
• See p.362 of H&S 2007
• The root node
  – mapping from categories of root nodes of PTB trees to CCG categories, e.g. \{VP\} → S\(\backslash\)NP, \{S, SINV, SQ\} → S

• Head and complement
  – category of complement child defined by a similar mapping, e.g. 
    \{NP\} → NP, \{PP\} → PP
  – category of the head is a function which takes the category of the complement as argument and returns the category of the parent node; direction of the slash is given by the position of the complement relative to the head

• Head and adjunct
  – given a parent category \(C\), the category of an adjunct child is \(C / C\) if the adjunct child is to the left of the head child (a premodifier), or \(C \backslash C\) if it is to the right (postmodifier)
Comments on the Basic Algorithm

- Assigns a *normal-form* derivation, i.e. only uses type-raising and composition when necessary
- Sometimes modifier is allowed to compose with the head (giving a more elegant analysis – see p. 364 of H&S)
- Long-range dependencies require extensions to the basic algorithm, using type-raising and composition rules
The co-indexed trace element *T*−1 is crucial in assigning the correct categories

- used as an indication of the presence of a direct object for the verb
- used to assign the correct category to the Wh-pronoun (using a similar mechanism to GPSG’s “slash-passing”)

p.57 of the CCGbank manual has a detailed example
Properties of the resulting CCGbank

- 99.4% of the sentences in the PTB are translated into CCG derivations
- Words with the most number of category types:

<table>
<thead>
<tr>
<th>Word</th>
<th>num cats</th>
<th>Freq</th>
<th>Word</th>
<th>num cats</th>
<th>Freq</th>
</tr>
</thead>
<tbody>
<tr>
<td>as</td>
<td>130</td>
<td>4237</td>
<td>of</td>
<td>59</td>
<td>22782</td>
</tr>
<tr>
<td>is</td>
<td>109</td>
<td>6893</td>
<td>that</td>
<td>55</td>
<td>7951</td>
</tr>
<tr>
<td>to</td>
<td>98</td>
<td>22056</td>
<td>LRB</td>
<td>52</td>
<td>1140</td>
</tr>
<tr>
<td>than</td>
<td>90</td>
<td>1600</td>
<td>not</td>
<td>50</td>
<td>1288</td>
</tr>
<tr>
<td>in</td>
<td>79</td>
<td>15085</td>
<td>are</td>
<td>48</td>
<td>3662</td>
</tr>
<tr>
<td>–</td>
<td>67</td>
<td>2001</td>
<td>with</td>
<td>47</td>
<td>4214</td>
</tr>
<tr>
<td>’s</td>
<td>67</td>
<td>9249</td>
<td>so</td>
<td>47</td>
<td>620</td>
</tr>
<tr>
<td>for</td>
<td>66</td>
<td>7912</td>
<td>if</td>
<td>47</td>
<td>808</td>
</tr>
<tr>
<td>at</td>
<td>63</td>
<td>4313</td>
<td>on</td>
<td>46</td>
<td>5112</td>
</tr>
<tr>
<td>was</td>
<td>61</td>
<td>3875</td>
<td>from</td>
<td>46</td>
<td>4437</td>
</tr>
</tbody>
</table>
• Lexicon has 74,669 entries for 44,210 word types (929,552 tokens)
• Average number of lexical categories per *token* is 19.2
• 1,286 lexical category types in total
  – 439 categories occur only once
  – 556 categories occur 5 times or more
• Coverage on unseen data: lexicon contains correct categories for 94% of tokens in section 00
  – 3.8% due to unknown words
  – 2.2% known words but not with the relevant category
References


• Data and models for statistical parsing with Combinatory Categorial Grammar, Julia Hockenmaier, PhD thesis, Edinburgh, 2003

• M. Marcus, B. Santorini, and M. Marcinkiewicz, Building a large annotated corpus of English: the Penn Treebank. Computational Linguistics, 19(2), 1993

• Head-Driven Statistical Models for Natural Language Parsing, Michael Collins, PhD Thesis UPenn, 1999