ACS Syntax and Semantics of Natural Language

Lecture 6: Creating a Treebank for CCG

Stephen Clark

Natural Language and Information Processing (NLIP) Group

sc609@cam.ac.uk

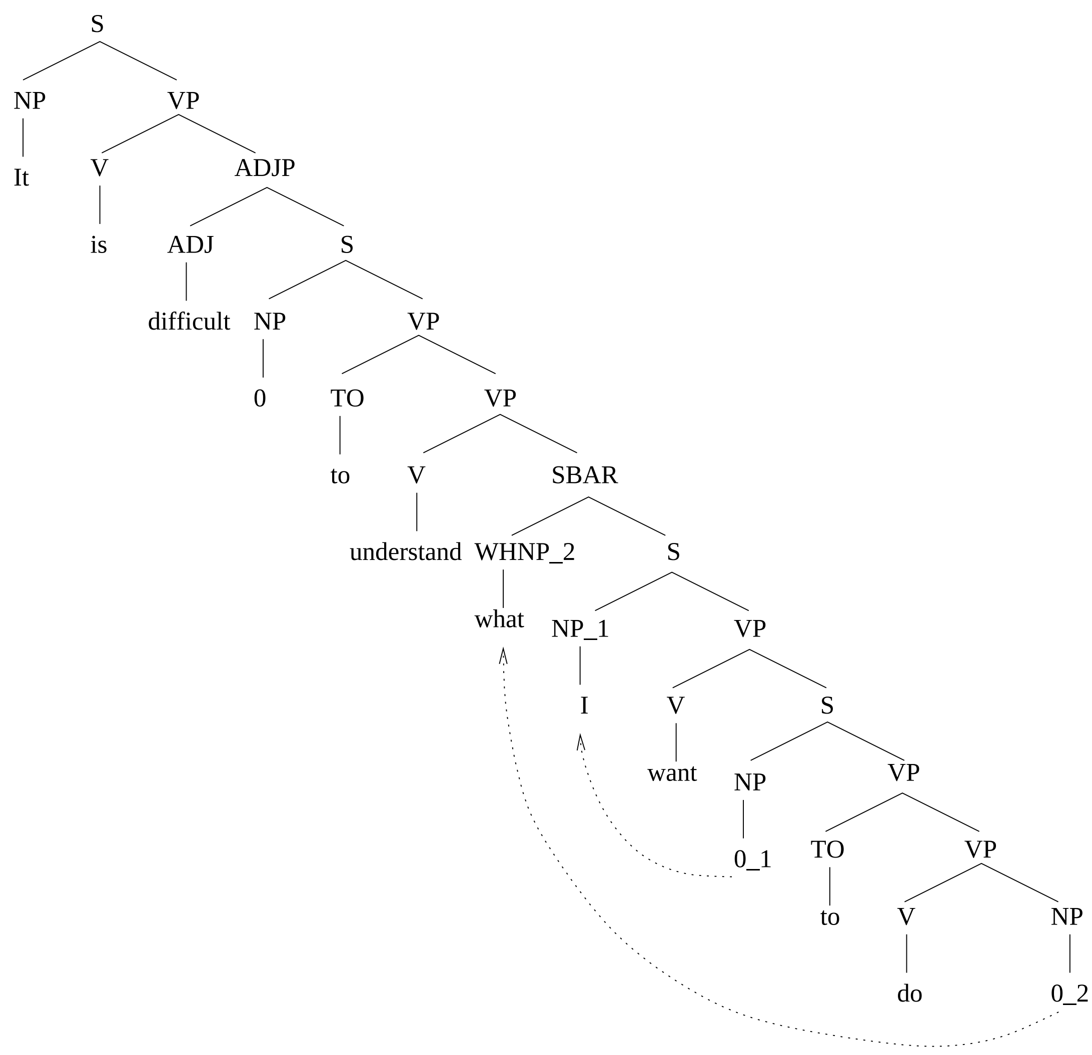
Creating a Treebank for CCG

- 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

The Penn Treebank

- 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

Example Penn Treebank Tree (with traces)



The Basic Translation Algorithm

- Ignoring long-range dependency/trace information, the basic algorithm is straightforward:
  - foreach tree τ
    - determineConstituentTypes(τ)
    - makeBinary(τ)
    - assignCategories(τ)

Determining Constituent Type

- 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. `-SBJ` (subject), `-TMP` (temporal modifier), `-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 `-LOC`, `-DIR`, `-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

Binarizing the Tree

- 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

Assigning Categories (p.363 of H&S 2007)

- The root node
  - mapping from categories of root nodes of PTB trees to CCG categories, e.g. $\{VP\} \rightarrow S\backslash NP$, $\{S, SINV, SQ\} \rightarrow S$
- Head and complement
  - category of complement child defined by a similar mapping, e.g. $\{NP\} \rightarrow NP$, $\{PP\} \rightarrow PP$
  - 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

Long-Range Dependencies Through Extraction


```
(NP-SBJ (NP Brooks Brothers))
        (, ,)
        (SBAR (WHNP-1 (WDT which))
              (S (NP-SBJ NNP Marks))
                 (VP (VBD bought)
                     (NP (-NONE- *T*-1))
                     (NP-TMP last year)))))
```

- 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:

Word	num cats	Freq	Word	num cats	Freq
as	130	4237	*of*	59	22782
is	109	6893	*that*	55	7951
to	98	22056	*LRB*	52	1140
than	90	1600	*not*	50	1288
in	79	15085	*are*	48	3662
–	67	2001	*with*	47	4214
's	67	9249	*so*	47	620
for	66	7912	*if*	47	808
at	63	4313	*on*	46	5112
was	61	3875	*from*	46	4437

More Statistics (Sections 02-21)

- 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 uneen 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

- CCGbank: A Corpus of CCG Derivations and Dependency Structures Extracted from the Penn Treebank. Julia Hockenamier and Mark Steedman. Computational Linguistics. 2007
- 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
- David Vadas and James R. Curran (2007). Adding Noun Phrase Structure to the Penn Treebank. In Proceedings of ACL-07.