Linguistics 165, Homework 5
due 2 December

23 November 2010

Turn in both code and textual description in the answers to this exercise.

1. Find all the parses you can for the first sentence in the sentence pair below.

   Time flies like an arrow. Fruit flies like a banana.

   (For full credit, you should find at least three parses, though there are more than three.) Give a parse tree for each of the parses, together a paraphrase for each parse that clarifies the intended meaning.

   Which are the most plausible parses? What effect does the second sentence have on the preferred interpretation of the first sentence? Why might this be humorous?

   Rank your parses in order of what you consider to be their plausibility out of context (that is, without the second sentence juxtaposed). Then write a PCFG that can parse sentence 2, and that also generates these parses of sentence 1 and assigns probabilities consistent with this ranking. Does your PCFG also assign the intuitively preferred parse of sentence 2? If so, why? If not, why not?

2. How often in the fragment of the Penn Treebank that we’ve been using do you find prepositional phrases (PPs) whose parent category is adjective phrase (ADJP)? Show some examples. Give examples of how PPs that should attach into ADJPs can have attachment ambiguity. Would the parsers we’ve been developing do a good job in resolving this attachment ambiguity? Why? How might you improve their performance in resolving ambiguities like these?

3. One of the problems of the Penn Treebank annotation scheme for PCFG induction is that the constraints on the internal structure of nested VPs are inadequately represented. For example, the daughter VP in the rule VP $\rightarrow$ TO VP should always be a finite verb (category VB; e.g., he wants to leave), whereas the daughter VP in a VP headed by the verb have should be participial (category VBN; he has left) and the daughter VP in a VP headed by the verb be should be gerund (category VBG; he is leaving). This insufficient annotation leads to bad misparses sometimes; for example, the sentence
is badly misparsed as

(S
  (NP (DT This))
  (VP (VBZ is)
    (VP (VB panic)
      (VP (VBG buying))))
  (. .))

in a vanilla PCFG.

To fix this problem, write one or more Tsurgeon expressions to add annotation to the trees ptb-train.txt that will cause a PCFG learned from this dataset to avoid incorrect VP parsing of this sort. Learn a PCFG from your newly annotated trees and use them to parse ptb-test.txt with PCFGparse.sh, find parse trees write Tsurgeon expressions to transform the trees back into traditional Penn Treebank notation, and then use evalb with ptb-test-noempties.txt as the gold-standard tree file to evaluate your results. Did your annotations improve parsing over vanilla PCFG performance (see the file vanillaPCFGperformance.txt)?