1 Assignment

Download the Penn WSJ corpus (training set, all trees on one line) from the course website. Calculate the word frequency distribution, the subject vs. object NP length distribution, and the phrasal rule frequency distribution. Use the unix tools `grep`, `sed`, `gawk` and regular expressions to obtain the frequency counts you need, and familiarize yourself with these tools.

Getting started Download the file 'treebanks.zip' from `http://www.illc.uva.nl/laco/clas/ull14/`

Open a unix-shell (applications → system tools → terminal). Create a folder for your work (`mkdir ull14`) and move the corpus to that folder (e.g., `mv ~/Downloads/treebanks.zip ~/ull14`). Then unzip it with `unzip treebanks.zip`. Watch for the password on the whiteboard. If you don't know grep, sed, awk and other unix tools, find a unix cheat sheet online and start with the tutorial on regular expressions (t1-regexp-ull12.pdf).

Word frequency distributions A very useful combination of the unix commands grep, sed, sort and uniq will give you word frequency distributions in one go. First convince yourself that the following regular expression matches all lexical items: `[^ ]\+)`

The following string of commands will then give you a word frequency distribution:

```
cat penn-wsj-line.txt | sed 's/)/)\n/g' | grep -o '[^ )]\+)' | sed
   's/)//' | sort | uniq -c | sort -g -r -k 1 | sed 's/^ *//' > wordfreqdistro
```

Make sure you understand all parts of this expression. You can plot the distribution with `gnuplot`, using the commands:

```
set logscale xy
plot 'wordfreqdistro' u ($0):($1)
```

Subject vs. object NP length distributions The key to solving this assignment is the fact that subject NP's tend to have an S as parent; object NP tend to have VP as parent. You may assume that all subject NP are the first daughter of an S, and that all NP that aren't the first daughter of an S or a PP are object-NP's. First remove the existing NP-subcategorizations as they are not consistent (i.e., NP-SBJ etc), and then add the parent label to all leftmost daughters:

```
cat penn-wsj-line.txt | sed 's/(NP[^ ]* /(NP /g;s/(\([A-Z]\+\) (NP/(NP-\1 /g' > tmp0
```

Regular expression cannot count brackets, but awk can. You can use these commands to put spaces around brackets and then add a depth counter to every bracket:

```
cat tmp0 | sed 's/(/( /g;s/)/ )/g;s/ \+/ /g' | sed 's/%/PERCENT/g' >tmp1
cat tmp1 | awk '{j=0; for (i=1;i<=NF;i++) {printf " " $i; if ($i=="(")
   printf ++j; else if ($i==")") printf j--; } printf "\n";}' > tmp2
```

Then, if you have labeled all subject-NPs as 'NP-S the following grep command can find their complete span (using "perl-compatible" syntax and the perl construct `*?` that matches the shortest span rather than the longest span, as is grep's default):

```
cat tmp2 | grep -oP '\(([0-9]+) NP-S.*?\)\1'
```

The final step is to replace all lexical elements (terminals) with a dummy symbol (e.g., x) and remove all other annotations.

Phrasal rule frequency distribution Extracting phrasal rules with regular expressions is a bit of a pain. To make life easier, the program PCFG_extractor.jar is provided:

```
java -jar PCFG_extractor.jar penn-wsj-line.txt outputfile
```