HW4

Mike Roylance
roylance@uw.edu

Describe and discuss your work in a write-up file

I completed this assignment using Python with nltk. My solution is organized into the following files and directories:

Location Description	Purpose
docs/	folder that contains all the documents needed for this assignment, such as feature_sentences.txt and grammar.fcfg
docs/grammar.fcfg	file that contains the feature grammar
docs/feature_sentences.txt	feature sentences to be parsed
source/	folder that contains all the source code
source/main.py	entry point script that reads in the files from the user and prints out the result to the console
source/parseResult.py	builder class that parses the feature grammar and returns its parse
source/queryUtils.py	utility file to help interrogate the parses of the sentence. these functions help determine if a given tree has a feature of a specific type, like gender or number.
source/tests.py	unit tests for each sentence. this ensures that there are correct parses.
hw4.cmd	command file used by condor. this calls hw4.sh with the parameters of docs/grammar.fcfg, docs/feature_sentence.txt and result
hw4.sh	file to handle calling the Python file main.py with specific parameters

Include problems you came across and how (or if) you were able to solve them, any insights, special features, and what you learned. Give examples if possible.

I like to solve programming problems using test driven development. Each test has a specific template:

- Arrange
- Act
- Assert

In my Arrange, I set up the sentence to be used to run against the grammar file. My Act then uses the parseResult to build the result of parsing the sentence with the grammar file. My Assert the verifies that the tree was correctly given.

I solved this iteratively, one sentence at a time. The good thing about tests is that I don't have to worry about breaking existing functionality, every time I create a new test I regressively check that all my previous cases are functioning correctly too. I find this iterative type of development both relaxing and effective. It also helps me organize better because I am focusing on smaller things at a time.

I learned a lot about feature grammars here. I like being able to specify the gender and number of a particular production, and have the chart parser correctly pick it up. This seems very powerful, especially when considering that a person can use the ?n feature to handle all different matching cases. This saves a lot of time compared to having to create new productions for each variation.

I also learned about combining particular features, for instance I have a noun phrase that looks for two nouns that match a number followed by a time (this follows the rule "five minutes"). I solve this by looking for a particular feature type (in this case, number) and then another particular type (time).

Overall I enjoyed this assignment, I find python and nltk very powerful and useful.