CS-481 — AI: Natural Language Understanding

Homework Assignment — Parsing CFGs

This assignment may be done in pairs.

You are to experiment with and enhance a simple Earley's Algorithm Parser using a small grammar ("simple.gr"), supplied in the attached ZIP file. Such a parser and grammars are supplied (written in Java); if you prefer, you may write your own parser (make sure to document your code). Also provided is a sample of sentences parsed. Please see the documentation files for details on code structure and running the parser.

1. Evaluate the parser in its current form:
   - What sorts of sentences can it parse?
   - What types of sentences will it fail to parse?
   - What improvements are required for the failed sentences?
2. Look at the grammar. What changes to the grammar alone could improve the parsing (in particular, consider coordinated conjunctions)? Modify the grammar somewhat and evaluate the improvements. What new sentences may be parsed?
3. Modify the code to output the parse tree(s) computed, rather than just the dynamic programming table and whether the sentence parses. You will need to add appropriate back-links into the DP table and write a routine to extract trees from the table. Show the tree(s) by printing the input sentence in a bracketed form, for example:
   ```
   S[ NP[ Det[ the ] N[ man ] ] VP[ V[ saw ] NP[ PN[ her ] ] ] ]
   ```
4. **Extra Credit:**
   Modify the code to implement a version of probabilistic parsing for the Earley algorithm using a similar method as that discussed in class for the CYK algorithm (each dotted rule will have a probability that is the product of the rule's probability with the probabilities of all the completed children it covers so far). Enhance the grammar with reasonable probabilities for the rules and test the system on ambiguous sentences to see if the highest probability parses are indeed the most "reasonable" parses.

Handing in:

For the handing in, please submit the full system as given with the following additions:

1. A file "evaluation.txt" all about part 1, your initial evaluation of the system.
2. A file "grammarImprovements.txt" detailing the improvements you have made to the grammar, and what functionality has been added.
3. A file "codeImprovements.txt" detailing the improvements you have made to the code itself for part 3 above.

   Format:
   ```
   Class.iAddedThis
   param p_input this is an input
   ...
   returns some stuff I needed
   This is a short but informative comment about why I added this function.
   ```

```
For code modifications:
Class.Method
modified params...
modified return...
This is a short but informative comment about what I added and why.
```

The end of the file contains the evaluation of improvements due to the above modifications, and a discussion of what other improvements could be made.

These files should be in the base parser directory alongside the readme.txt and design.txt . Code must be commented as well!