CS585 Natural Language Processing - HW2: Parsing CFGs

Group Members: Cristobal Sarome (A20416473), Akhil Suryadevara(A20391322)

Code Improvements

Note: to easy identification, the modifications done to the code for this homework have the label “CS585\_HW2” in the comments.

Backtrack Links

To be able to retrieve the parsed sentence, we added backtrack links to the production in a form of an ArrayList of productions. In this way, when we reach the complete phase of the Earley’s algorithm and the completed rule is added to the column, we also add a link in this rule to the rules that led to its completion. This is done in the attach method of the Grammar class.

Thanks to this addition, when the parse method returns the final complete production (named result), we can traverse the complete production tree accessing to the backtrack ArrayList components.

Print bracketed form of the parse tree

A recursive method called bracketedForm was implemented in the Production class. This method performs a pre-order traversal over the parse tree and prints the terminal and non terminal symbols with brackets indicating the parse structure.

Print rules used for parsed

The recursivePrint method in the class productions traverse the parse tree and print all the rules used for parsing. This was not requested in the assignment but is useful for clear diagnosis and visualization of the algorithms behaviour.

Log Probability Calculation

In the attach method of the Grammar class (complete phase), we add to the rule added the combined log probability of the rules used to complete this one. In this way the attribute logProb of every Production object reflect the total probability of this structure being parsed in this way given the grammar. In this way, when we obtain the result (Production object) of a sentence successfully parsed, the logProb attribute of this object is the sum of all the log probabilities of the rules used to parse this tree and therefore, the log probability of the sentence given the grammar.

Handling ambiguity

In order to deal with ambiguity we have to allow adding two equal productions with different probabilities to the column being processed by the algorithm in the method addProd of the Grammar class. Later, with the method deleteDuplicated we examine the column and keep only the rule with the maximum probability, in this way we ensure that the parsing returned will be the one with maximum probability.

Examples of ambiguous sentences

We test the probabilistic parsing with the following sentences that can be parsed in more that one way

i saw a man with a telescope

i shot an elephant in my pajamas

And we obtained the following results using the grammar file prob-simple2.gr

ROOT[ s[ np[ fpron [ i ] ] vp[ vp[ fv [ saw ] np[ fdet [ a ] fnoun [ man ] ] ] pp[ fprep [ with ] np[ fdet [ a ] fnoun [ telescope ] ] ] ] ] ]

Sentence Log Probability= -20.869448093804756

ROOT[ s[ np[ fpron [ i ] ] vp[ vp[ fv [ shot ] np[ fdet [ an ] fnoun [ elephant ] ] ] pp[ fprep [ in ] np[ fdet [ my ] fnoun [ pajamas ] ] ] ] ] ]

Sentence Log Probability= -22.94888963548459

To demonstrate the alternative parses for these sentences, we created the grammar file prob-simple2-altered.gr in which we artificially altered the grammar by invalidating the rule vp->vp pp that makes possible the parsings shown previously. In this way we force the algorithm to show the alternative parsing posibility

ROOT[ s[ np[ fpron [ i ] ] vp[ vbar[ vbar[ fv [ saw ] ] np[ fdet [ a ] fnoun [ man ] ] pp[ fprep [ with ] np[ fdet [ a ] fnoun [ telescope ] ] ] ] ] ] ]

Sentence Log Probability= -22.543424502541157

ROOT[ s[ np[ fpron [ i ] ] vp[ vbar[ vbar[ fv [ shot ] ] np[ fdet [ an ] fnoun [ elephant ] ] pp[ fprep [ in ] np[ fdet [ my ] fnoun [ pajamas ] ] ] ] ] ] ]

Sentence Log Probability= -24.62286604422099

We notice that the sentences parsed in this way intuitively make less sense than the previous parsing and also the probabilities are lower. So we confirm that the Earley’s parser is returning the parses that have the highest probabilty among all possibilities.