ICSI 409

CYK Parsing Report

Alexander Crowe, Gregory Vincent, Victor Sotannde, Thomas Lloyd-Jones, Owen Sterling & Hana Kastrati

Context for CYK Algorithm

The CYK Algorithm is used in order to determine the validity of a given string in regards to some Context Free Grammar(CFG). In order for the CYK Algorithm to work correctly, the given CFG must be in CNF - or follow a certain set of rules that simplify the CFG considerably while maintaining the same string producing capability. As such, every nonterminal must only imply two nonterminals, one terminal, or nothing denoted by an epsilon.

Implementation and Issues encountered while developing the CYK Parser

This program was developed using the basis of lexical analysis - i.e, through token objects to classify given characters of the strings representing the CFG and input string for said CFG in the input text file provided by the user, in addition to a Lexer and Parser with the ability to properly identify and label said portions of an input. “App.java” is the program with the ability to then take these strings and perform the necessary pattern matching and Carteisian operations needed to complete the operation. In creating the Parser, the main hardship was found when trying to find a proper way to develop multiplication between two different portions of the cartesian products on the middle levels of the process. So for a string like “aba,” it’s easy to develop the initial grammar of all individual characters. But figuring out a way to perform the correct cartesian multiplication proved to be a challenge. Ultimately, we were successful in performing CYK correctly using a 3D matrix, and performing the needed operation that way. Our chosen data structure was an arraylist because of the ease with which we can perform CRUD (Create, Update, Read, Remove) operations.

Testing Process and Results

In order to test this parser, we developed some strings that are and are not in the grammar of a particular CFG, and then compared our results to what the CFG should return as a proper response. We decided on using the given CFGs in the instructions of this assignment as a baseline.

Time and Space Complexity of the Parser

The CYK Parser at its slowest runs in $O(n^3)$ time and takes up $O(n^3)$ space. This is directly because of the triple for loops needed to properly index into the 3D matrix and perform the correct operations. While it is on the slower end, we feel that given the nature of the operations and the need to store, classify, search through, and perform operations on data simultaneously, that it was an appropriate measure to take.