Kaloyan Dragiev

PSN: 3214311

Automata & Logic software Engineering  
 Design Documentation

FHICT English Stream

# Assignment 1: Parse + Tree

For parsing the given expression, I have used a recursion which every time it takes the first character of the input and makes a different proposition, then it removes the redundant brackets and processed characters and parses again until there is no more input left. With the resulting a proposition which is a tree, I use Graphiz to represent in of the form. Every proposition type has its own unique ToString method which gives the whole tree into an infix form notation string which is displayed.

# Assignment 2: Truth table + Hash code

For the truth table I have Truth Table containing many table rows and depending on the parsed proposition and how many variables there are the table creates 2 to the power of number of variables, rows. Then each row has a calculated value property which is assigned when a proposition calls a method calculate. After that all the calculated values are transformed into a binary and then to hexadecimal code and then displayed on the form.

# Assignment 3: Simplify

For the simplify I use a method in a truth table class which takes the current table goes through the rows and row + 1 and then it checks whether there is only one difference in these rows and if so it replaces on that index with \*, and then only one of the rows is kept. After that it checks for duplicate rows and removes them if found. It simplifies 9 times.

# Assignment 4: Normalize

For the normalize I first get all the rows with a result 1 then I create a proposition for each row depending of the value of the variables and then with each row of the table I create an Or proposition using recursion and then I call a method ToString on the proposition for displaying it on the form.

# Assignment 5: Nandify

For Nandify each proposition has its own unique Nandify method which creates different types of propositions depending on the input. Then a tree proposition is created then I call a method ToString on the proposition for displaying it on the form.

# Assignment 6: Semantic tableau for propositions

The semantic tableau class takes the parsed proposition, makes it a negation proposition and then creates an Element class with a list of Propositions with the given one. Then it calls on the object Element a method GenerateNewNodes which checks for every time Alpha and Beta rules and then creates new Propositions and adds them to the list in the Element class. Then the Element is returned and checked whether is closed and then displayed on the label on the form.

# Assignment 7: Semantic tableau for predicates

Similarly, to the semantic tableau for propositions, for this assignment I use the same class for taking the proposition and negating it and making a new Element with a list of Propositions. Then it calls on the object Element a method GenerateNewNodes which checks for every time Gamma and Delta rules and creates a different proposition or propositions every time and then it checks them again until the status Closed becomes true then recursion goes back and closes the whole tree or if not after 1000 iterations it throws exeption for an infinite recursion.

# Additional Features

* Added a Button Create Random which generates a random proposition and then it calls the event Onclick for the Parse Button.

# GUI

* Two buttons Parse and Create Random. The former parses the inputted expression the textbox and the latter one creates a random expression and then it calls the event Onclick for the Parse Button.

# Testing

* I have created 2 different classes for testing the application – one for the Parser (11) and one for the general functionality (7). The parser tests check whether with a various type of input the application doesn’t crash and with the functionality tests it tests the creation of the truth table, the simplification of it, the disjunctive normal form, the simplified disjunctive normal form, Nandify and the semantic tableau.

# Conclusions and future implementations

To Conclude the overall project was very helpful and exciting for me. I really understood a lot of programming techniques and methods for solving different kinds of problems. Also, for me the whole brainstorming and approaching a certain problem has changed completely. For this project for a future implementations and improvements I can make a semantic tableau for predicates more tested and more complete because I couldn’t have finished it on time and not all the functionalities are there.