**Programming Assignment2 - DNF Conversion**

21600108 Kim Yeong Bin

GitHub : <https://github.com/yeongbin123/hw2>

**1. Introduction**

A DNF formula is a conjunctive clause or disjunction of multiple conjunctive clauses. The problem is that when user input a formula, we should check valid form and convert formula to ***NNF form*** that then convert *NNF form* to ***DNF form***. Finally, check that formula is satisfiable. When formula is Unsatisfiable, print outs *UNSAT*. Otherwise, print outs possible combination.

**2. Approach**

First, the propositional formula entered must be converted to *NNF*. Then, the *NNF* is converted to *DNF* by applying the distribution law. I will solve this problem using recursion. Especially, I will obtain clause and judge the logical relationship between each clause whether it is satisfiable or not.

**3. Solution**

Thinking about how to express formula, I decided to express it in ***doubly linked list*** rather than ***tree***. I was not used to using the *tree* yet, and I chose it because I thought I could easily reach the desired value by using *doubly linked list*. So first I converted formula to linked list form, then removed ***not*** to convert it to *NNF*. Because the structure of linked list stored the type of each value and block, after check *not* type, remove the not node form the list and reduce the block size by the number of *not*. Then, if there is any part to be distributed, the new clause is obtained by applying the distribution law. This was complicated to use linked list, so the values were sorted separately in array and distributed using the array contents. Finally, to solve the ***SAT problem***, if or is the case, the proposition is true if there is an ***atomic variable***. If there is no atomic variable, identify the logical relationship between each clause and return the proposition to true if any of them is true. If and is the case, assume that one clause is true and judge the other clause and return whether it is true or false.

**4. Demonstration**

If you input a propositional formula, then program will return values of DNF formula and whether formula is satisfiable or unsatisfiable. If formula is unsatisfiable, program will return message that “UNSAT”.

|  |  |
| --- | --- |
| Input | Output |
| (or a1 (not (or (not (not (or a2 a3)) a4))) | 1  -2 -4  -3 -4  0  1 - 2 -3 -4 |

**5. Discussion**

The limit of this program is that distribution law cannot be applied if atomic variable is not entered or only entered in clause units. I searched various algorithms to solve this problem, but the linked list was difficult to implement. I think it would be theoretically possible to implement the vectors used in c++ directly, save each value, and pop in the order of distribution.