<Homework #2>

(21800511 이민재 Yi Min Jae)

1. DNF Converter

1 Summary

In this problem, we created a program that converts the normal formula "(operator operand operand ..)" into DNF formula and checks whether the solution exists and print out the solution if it exists. We used binary tree structure and recursive functions to solve the problem.

2 Problem analysis

2.1

```
typedef struct node {
    int data;
    struct node *left_child;
    struct node *right_child;
    bool operatorCheck;
} tree_node;
```

Binary tree has four elements. It contains left_child and right_child address, data, and operatorCheck. If the operatorCheck is 1 the data refers to the operator type. Each data has the following operator when the operatorCheck is 1. NOT : 0, OR : -1, AND : -2. If the operatorCheck is 0 the data refers to the propositional value, for example, if the data is 2 it refers to a2.

2.2

After defining the tree structure build tree structure with makeTree(). If the parent node is an operator, the child is define as an operand of the parent. In makeTree(), we recursively call the function whenever we read left parenthesis to make the subsequent part to be the child of current node.

2.3

In NNF formula NOT operator can only be placed in front of the operand. To exclude NOT operator from other operators adjust De Morgan's Laws. Start from the root node and move to its child node as recursion. If the parent node is NOT then every child node of the parent should be opposite of itself, and if this process executed in even there is no change since NOT NOT a1 is a1.

2.4

DNF formula should be a disjunction of one or more clauses, where a clause is a conjunction of operand. To implement DNF formula, we must apply distribution. For instance, in P ^ Q form, P must have the form P1 v P2 v … v Pm, whereas Q must have the form Q1 v Q2 v … v Qn. After that P ^ Q becomes (P1 v P2 v … v Pm) ^ (Q1 v Q2 v … v Qn). Then we can change it into

(P1 ^ Q1) v (P1 ^ Q2) v … v (P1 ^ Qn)
v (P2 ^ Q1) v (P2 ^ Q2) v … v (P2 ^ Qn)
…
v (Pm ^ Q1) v (Pm ^ Q2) v … v (Pm ^ Qn).

2.5

DNF has no solution if there is an opposite proposition in the same parenthesis, for example, (a1 ^ NOT a1). To check this condition use '+' operation, if the result of adding to propositional is '0' it has no solution. As the previous example (a1 ^ NOT a1) it will return 0 since it is '1 + (-1) = 0'.

3 Demonstration

```
input : (or a1 (not (or (not (or a2 a3)) a4)))
output: 1
          2 -4
        3 -4
          0
        1 2 3 4
input : (and a1 (not a1))
output: 1 -1
          0
          UNSAT
```

4 Discussion

It was easy to make CNF program into DNF since there is no big difference between CNF and DNF except the condition of distributing. To check the satisfiable of the solution I used plus operation, because the proposition was converted into Integer and '¬' into '-'. If ¬a1 and a1 are in the same parenthesis, the sum of 1 and -1 will return 0.