Python Programming and Practice

Boolean Function Simplifier Program

Proposal

Date : 2023/11/03
Name : kimsanha
ID :230382

1. Introduction

1) Background

In my logic circuits course, I was working on a problem to simplify a Boolean function, and when my answer and the answer sheet were incorrect, I had a hard time figuring out where I went wrong. So to make it easier for me and my future students, I wrote a program to draw a k-map of the input Boolean function and simplify it using the sum of products (SOP) and the product of sums (POS).

2) Project goal

Write a program that draws a k-map for a given Boolean function and simplifies it using the sum of products (SOP) and product of sums (POS).

3) Differences from existing programs

Existing programs that simplify Boolean functions often don't know what rules they use or just give you the answer, but the program I'm going to build will draw and use K-maps to simplify, making it easy to understand visually.

2. Functional Requirement

1) Draw a K-map

- Description (Ability to represent an input Boolean function as a K-map)

(1) Draw a table based on the number of variables

- Description (If the number of variables entered is two, draw a table of the form 2*2, three 4*2, and four 4*4.)

(2) Populating a table 1

- Description (In each column of the table, write down the term that the column represents. (m0=x'y')

(3) Populating a table 2

- Description (Display the values of the Boolean function you entered as 0s and 1s in the table you created in Detailed function 2.)

2) Make a Boolean function

- Description (Output the result of simplifying the input Boolean function to (sum of products) sop and (sum of products) pos using K-maps, respectively.)

(1) Make the enclosed result a Boolean function

- Description (A set of 1's is represented as a sum of products by expressing one set as a product and adding them together, and a set of 0's is represented as a sum of products by expressing one set as a sum and adding them together, and (1) output with the K-map of Detailed function 3.)

3. Implementation

(1) Draw a K-map

- Input/Output: Output a K-map with the number of variables in the Boolean function, a K-map with the number of variables given the Boolean function, and a K-map showing the input Boolean function in the K-map.

- Description: Output a K-map of the number of variables in a Boolean function and a K-map of the parts of the Boolean function that are in the input K-map.

- Applying what you learned: Loops, conditionals, functions, and classes,

-Code Screenshots

```python
class KarnaughMap:
    def __init__(self, num_of_variables):
        self.map = self.make_karnaugh_map(num_of_variables)

    @staticmethod
    def make_karnaugh_map(num_of_variables):
        if num_of_variables == 2:
            return [[0, 1], [2, 3]]
        elif num_of_variables == 3:
            return [[0, 1, 3, 2], [4, 5, 7, 6]]
        elif num_of_variables == 4:
            return [[0, 1, 3, 2], [4, 5, 7, 6], [12, 13, 15, 14], [8, 9, 11, 10]]
        else:
            print("이 함수는 2부터 4까지의 변수에 대한 카노맵을 지원합니다.")
            return None

    def display(self):
        for row in self.map:
            print(row)

    def mark_positions(self, positions):
        for position in positions:
            for i, row in enumerate(self.map):
                if position in row:
                    j = row.index(position)
                    self.map[i][j] = 'X'
        return self.map
```

(2) Make a Boolean function

- Input/Output: Converts and outputs the input Boolean function into two forms.

- Description: Converts the input Boolean function to SOP and POS and outputs it.

- Applied lessons: Loops, conditionals, functions, and classes,

- Code screenshot

```python
class BooleanFunction:
    @staticmethod
    def simplify_sop(karnaugh_map):
        rows = len(karnaugh_map)
        cols = len(karnaugh_map[0])
        simplified_terms = []

        for i in range(rows):
            for j in range(cols):
                if karnaugh_map[i][j] == 'X':
                    term = []
                    if rows == 4:
                        term.append('A' if i < 2 else 'A\'')
                    if rows >= 2:
                        term.append('B' if i % 2 == 0 else 'B\'')
                    if cols == 4:
                        term.append('C' if j < 2 else 'C\'')
                    if cols >= 2:
                        term.append('D' if j % 2 == 0 else 'D\'')
                    simplified_terms.append(''.join(term))

        return ' + '.join(simplified_terms)

    @staticmethod
    def simplify_pos(karnaugh_map):
        rows = len(karnaugh_map)
        cols = len(karnaugh_map[0])
        simplified_terms = []

        for i in range(rows):
            for j in range(cols):
                if karnaugh_map[i][j] != 'X':
                    term = []
                    if rows == 4:
                        term.append('A' if i >= 2 else 'A\'')
                    if rows >= 2:
                        term.append('B' if i % 2 != 0 else 'B\'')
```

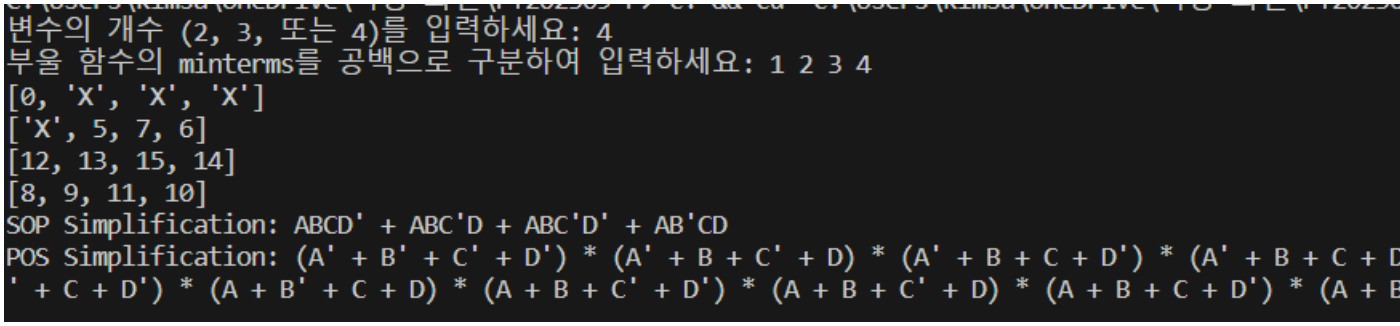
```python
                    if cols == 4:
                        term.append('C' if j >= 2 else 'C\'')
                    if cols >= 2:
                        term.append('D' if j % 2 != 0 else 'D\'')
                    simplified_terms.append('(' + ' + '.join(term) + ')')

        return ' * '.join(simplified_terms)
```

4. Test Result

(1) Full Test

- Description: Since the code to draw the K-map and the code to optimize the Boolean function work in parallel, the entire code was tested at once.

- Screenshot of the test results



5. Changes in Comparison to the Plan

1) Simplifying Boolean functions in SOPs and POS formats

- Previous: The goal was to simplify a Boolean function using SOPs and POS.

- After: Creates a Boolean function using SOP and POS, but does not simplify it.

- Rationale: Since this program is for students learning k-map and Boolean algebra for the first time, we thought it would be more intuitive to not simplify.

6. Lessons Learned & Feedback

- Simplifying Boolean functions using methods such as SOP and POS was not too difficult or complicated when I learned it in class, but writing the code was too difficult, so I changed my plan. Through this, I realized that my coding skills are very weak, and I need to study hard during my vacation.