[C.] Proiecte - Reprezentarea grafică a circuitului logic al unei funcții booleene

Ilieș Andrei-Cristian (10LF222)
Macovei Alexandru-Fabian (10LF222)

Descrierea proiectului

În acest proiect ne-am propus să realizăm o aplicație grafică în Python, în care utilizatorul poate alege o funcție booleană pe care o va scrie într-un fișier de intrare sub formă de tabel de valori sau cu valorile funcției pe o singură linie. Această funcție va fi simplificată și în final se va afișa într-o interfață grafică schema circuitului logic al formei simplificate ale funcției. Funcțiile pot fi de maxim 6 variabile, limită sugerată de doamna profesor coordonator Vasilescu Anca, dar aplicația poate fi cu ușurință extinsă pentru un număr mai mare de necunoscute.

Legătura directă cu Arhitectura Sistemelor de Calcul

Ne-am dorit să realizăm un proiect ce are ca temă Unitatea 2 din cadrul cursului de Arhitectura Sistemelor de Calcul, și anume Circuitele logice. Am încercat să ne gândim dacă este posibil să construim scheme ale circuitelor logice utilizând limbajul Python. Ulterior am găsit o metodă ce simplifică funcțiile în modulul sympy, cea mai complexă parte a acestui proiect fiind realizarea schemei circuitului logic într-o interfață grafică.

Astfel, proiectul are legătură directă cu Arhitectura Sistemelor de Calcul datorită subiectului pe care l-am abordat și care face trimitere la cea de-a doua unitate din cadrul acestui curs.

Instructiuni de utilizare

Pentru executarea programului este nevoie ca modulele `math`, `sympy` și `tkinter` să fie instalate. Datele de intrare se introduc într-unul dintre fișierele cu extensia `.txt` astfel:

1. Dacă doriți ca citirea să se realizeze din fișierul `table.txt`, atunci valorile funcției vor fi introduse în ordine pe ultima coloană a tabelului,

celelalte coloane fiind folosite pentru a reprezenta combinația de intrare corespunzătoare. Coloanele trebuie despărțite prin câte un spațiu. Este necesar ca tabelul sa aibă pe linii toate combinațiile variabilelor de intrare. **Atenție! Prima linie a fișierului este doar pentru o organizare vizuală mai bună a datelor; aceasta va fi ignorată de algoritm, alcătuirea funcției începând de la linia a doua.** De asemenea, la apariția meniului de selecție se va alege opțiunea `1`.

2. Dacă doriți ca citirea să se realizeze din fișierul `line.txt`, atunci valorile funcției vor fi introduse pe prima linie a fișierului, separate prin câte un spatiu, în ordinea crescatoare a combinațiilor variabilelor binare de intrare pe care le valorizează. Deoarece numărul de variabile nu este citit, ci calculat pe baza numărului de valori ale funcției, este necesar ca numărul de valori introduse să fie o putere de-a lui 2 pentru buna funcționare a algoritmului. De asemena, la apariția meniului de selecție se va alege opțiunea `2`.

După introducerea datelor de intrare se va lansa în execuție fișierul `main.py`, unde veți întâlni mai întâi meniul. Dupa selecția corespunzătoare a opțiunii, programul își va începe execuția. În noua fereastră "Circuit" creată va apărea schema grafică a circuitului, porțile logice fiind numerotate începând cu 1. În consolă pe primul rând se va afișa expresia funcției booleene după simplificarea acesteia. În plus, tot în consolă se vor afișa pentru fiecare poartă logică urmatoarele informații:

- Numărul porții, pentru a o identifica pe schema grafică.
- "Nivelul" (coloana din reprezentarea grafică) pe care se află poarta.
- Expresia logică a porții, formată din membrul stâng, unul dintre cuvintele `AND` si `OR`, respectiv membrul drept.

Elemente utilizate din limbajul Python

Module utilizate:

- sympy - utilizat pentru funcția `simplify`, care aduce o funcție logică oarecare la una din formele ei minimale
- math - utilizat pentru funcția logaritmică
- tkinter - utilizat pentru crearea interfeței grafice

Elemente specifice de limbaj:

- Tăierea listelor folosind operatorul `[:]` (exemplu: `expr = expr[3:]`)
- Întoarcerea a mai multor valori folosind o singura instrucțiune `return` (exemplu: `return expr, var_count`)

- Asocierea de valori mai multor variabile intr-o singură instrucțiune (exemplu: `WIDTH, HEIGHT = 1600, 800`)
- Utilizarea comprehensiunii de liste pentru a crea alte liste (exemplu: `func_values = [int(x) for x in lines[i].split(" ")]`)

Utilitate

Aplicația propusă poate fi utilizată în mai multe moduri, în funcție de nevoile utilizatorilor. Iată câteva exemple:

- Învățarea conceptelor logicii booleene
- Automatizarea procesului de realizare a schemei grafice a circuitelor logice

Originalitate

Implementarea acestei aplicații a constat în împărțirea în mai multe etape distincte, fiecare parte având scopul său specific. Aceste etape au fost concepute pentru a aborda diferite aspecte ale funcțiilor booleene și ale realizării circuitului logic.

Am căutat pe internet elementele de care am avut nevoie pentru a realiza toți pașii, am construit fiecare bucată separat, am combinat corespunzător secvențele de cod și astfel am obținut rezultatul dorit. Nu am utilizat deloc secvențe de cod deja existente, ci ne-am ales libertatea de a alcătui o implementare proprie în cadrul acestui proiect.

Autoevaluare

Suntem foarte mulțumiți de rezultatul final al aplicației. Programul funcționează pentru orice date de intrare valide (așezate în fișier corepunzător nevoilor algoritmilor de citire) în limita spațiului de afișaj grafic.