ĐẠI HỌC QUỐC GIA TP HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

BÁO CÁO ĐỒ ÁN 2 LOGIC MỆNH ĐỀ

MÔN: CƠ SỞ TRÍ TUỆ NHÂN TẠO

Tp Hồ Chí Minh 2022

ĐẠI HỌC QUỐC GIA TP HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

CƠ SỞ TRÍ TUỆ NHÂN TẠO

BÁO CÁO ĐỒ ÁN 2 LOGIC MỆNH ĐỀ

Sinh viên thực hiện:

20120619 – Nguyễn Mạnh Tường

GIÁO VIÊN HƯỚNG DẪN

Thầy Nguyễn Duy Khánh

Tp Hồ Chí Minh 2022

Mục lục

I. Giải thích sơ về code

1. Mã giả của thuật toán.

- Thuật toán được viết dựa theo mã giả sau

function PL-RESOLUTION(KB, α) **returns** *true* or *false*
  inputs: KB, the knowledge base, a sentence in propositional logic
    α, the query, a sentence in propositional logic

  clauses ← the set of clauses in the CNF representation of $KB \wedge \neg\alpha$
  new ← { }
  while *true* **do**
    for each pair of clauses C_i, C_j **in** *clauses* **do**
      resolvents ← PL-RESOLVE(C_i, C_j)
      if *resolvents* contains the empty clause **then return** *true*
      new ← *new* ∪ *resolvents*
    if *new* ⊆ *clauses* **then return** *false*
    clauses ← *clauses* ∪ *new*

2. Mô tả cách xử lý dự liệu từ đầu vào.

- Thực hiện biến đổi một câu thành mảng các literal thay thế cho phép OR.
- Ví dụ:
  - A → [A]
  - A OR B → [A, B]
  - -A OR B → [-A,B]

```
## Đọc file trả về câu alpha và KB
def readFile(filename : str):
    f = open(filename, "r")
    alpha = removeWhitespace(f.readline()).split('OR')
    KB = []
    nKB = int(f.readline())
    for i in range(nKB):
        i = removeWhitespace(f.readline()).split('OR')
        KB.append(i)

    return alpha, KB
```

3. Mô tả hàm plResolve

- Hàm trả về tập hợp tất cả các mệnh đề có thể thu được khi hợp giải từ 2 mệnh đề đầu vào.

- Ví dụ
    - (-A OR B) hợp giải -B = -A
    - Mô tả dưới dạng code: **plResolve**([-A,B], [-B]) = [[-A]]

```
def plResovle(clause_i, clause_j):
    clauses = []
    for l_i in clause_i:
        for l_j in clause_j:
            if negationOfLiteral(l_i) == l_j or negationOfLiteral(l_j) == l_i:
                clause_i_re = [n for n in clause_i if n != l_i]
                clause_j_re = [n for n in clause_j if n != l_j]
                clauses.append(standardForm(clause_i_re + clause_j_re))
    return clauses
```

4. Mô tả hàm plResolution

- Bước 1: Tạo danh sách mệnh đề **clauses**. Thêm vào danh sách mệnh đề các câu trong cơ sở trị thức và phủ định của câu alpha.
    - Mô tả như sau: [GT1, GT2,…,-KL]

```
def plResolution(alpha, KB):
    #clauses = KB
    clauses = list(map(standardForm, KB))

    #phủ định lại câu alpha đưa vào KB
    for literal in alpha:
        #phủ định câu alpha
        clause = [negationOfLiteral(literal)]
        #thêm vào danh sách mệnh đề
        clauses.append(clause)

    #khởi tạo ds mới bằng mảng rỗng
    newList = []
```

- Bước 2: Tiến hành hợp giải tất cả cả cặp mệnh đề trong danh sách mệnh đề **clauses.** Nếu kết quả hợp giải từ một cặp có chứa mảng rỗng **thì vấn đề được giải quyết xong**.
    - VD **plResolve**([A],[-A]) = [[]] => **solution** = True

```
solution = False
while True:
    n = len(clauses)
    pairs = [(clauses[i], clauses[j]) for i in range(n) for j in range(i + 1, n)]
    rs.append([])
    for (clause_i, clause_j) in pairs:
        #chạy vòng lặp xét 2 cặp mệnh đề trong clause
        resolvents = plResovle(clause_i, clause_j)
        if [] in resolvents:
            solution = True
```

- Bước 3: Xây dựng một danh sách mệnh đề mới bằng cách thêm các mệnh đề từ kết quả hợp giải của 2 cặp mệnh đề ở bước 2. Bỏ qua các mệnh đề không hợp lí và mệnh đề trùng.

```
for tempCR in resolvents:
    if orContainTautology(tempCR):
        break
    if not tempCR in newList:
        newList.append(tempCR)
```

- Bước 4: Khi xét hết các cặp ở danh sách mệnh đề **clauses**. Nếu danh sách mới là danh sách con của danh sách **clauses** (tức là không phát sinh mệnh đề mới) **thì vấn đề không được giải quyết.**

```
if isSublistOf(newList, clauses):
    solution = False
```

- Bước 5: Thêm vào danh sách mệnh đề **clauses** các mệnh đề trong danh sách mệnh đề mới. Bỏ các mệnh đề đã tồn tại. Quay lại Bước 2.

```
for c in newList:
    if not c in clauses:
        clauses.append(c)
```

II. Kịch bản kiểm thử

1. Test case 1.

Input0.txt	Output0.txt	Ghi chú
D	4	
4	B	A hợp giải (-A OR B)
A	-A OR -C OR D	(-A OR B) hợp giải (-B OR -C OR D)
-A OR B	-B OR D	(-B OR -C OR D) hợp giải C
-B OR -C OR D	-B OR -C	(-B OR -C OR D) hợp giải -D
C	6	
	-C OR D	(-A OR B) hợp giải (-B OR D)
	-A OR D	(-A OR B) hợp giải (-B OR D)
	-A OR -C	(-A OR B) hợp giải (-B OR -C)
	-B	C hợp giải (-B OR -C)
	D	B hợp giải (-B OR D)
	-C	B hợp giải (-B OR -C)
	2	
	-A	C hợp giải (-A OR -C)
	{}	C hợp giải -C
	YES	KB entails alpha

2. Test case 2

Input1.txt	Output1.txt	Ghi chú
-A	5	
4	D	-C hợp giải (C OR D)
-C	-A OR -B OR D	(C OR D) hợp giải (-A OR -B OR -C)
C OR D	B OR D	(C OR D) hợp giải (B OR -C)
-A OR -B OR -C	-A OR -C	(-A OR -B OR -C) hợp giải (B OR -C)
B OR -C	-B OR -C	(-A OR -B OR -C) hợp giải A
	4	
	-A OR D	(C OR D) hợp giải (-A OR -C)
	-B OR D	(C OR D) hợp giải (-B OR -C)
	-A OR -C OR D	(-A OR -B OR -C) hợp giải (B OR D)
	-C OR D	(B OR D) hợp giải (-B OR -C)
	0	
	NO	KB không entails alpha

3. Test case 3

Input2.txt	Output2.txt	Ghi chú
D	7	
5	-A OR -B OR D	(-A OR -B OR C) hợp giải (-B OR -C OR D)
-A OR -B OR C	-B OR C OR D	(-A OR -B OR C) hợp giải (A OR D)
-B OR -C OR D	-A OR C	(-A OR -B OR C) hợp giải B
A OR D	-C OR D	(-B OR -C OR D) hợp giải B
B	-B OR D	(-B OR -C OR D) hợp giải C
C	-B OR -C	(-B OR -C OR D) hợp giải -D
	A	(A OR D) hợp giải -D
	7	
	-A OR -B	(-A OR -B OR C) hợp giải (-B OR -C)
	-B OR C	(-A OR -B OR C) hợp giải A
	C OR D	(A OR D) hợp giải (-A OR C)
	-A OR D	B hợp giải (-A OR -B OR D)
	D	B hợp giải (-B OR D)
	-C	B hợp giải (-B OR -C)
	-B	C hợp giải (-B OR -C)
	2	
	-A	B hợp giải (-A OR -B)
	{}	B hợp giải -B
	YES	KB entails alpha

4. Test case 4

Input3.txt	Output3.txt	Ghi chú
-A OR C OR -F	7	
5	D OR F	(C OR F) hợp giải (-C OR D)
C OR F	-A OR -B OR F	(C OR F) hợp giải (B OR -C OR -C)
E	B OR -D OR F	(C OR F) hợp giải (B OR -C OR -D)
-C OR D	F	(C OR F) hợp giải -C
-A OR -B OR -C	B OR -C	(-C OR D) hợp giải (B OR -C OR -D)
B OR -C OR -D	-A OR -C OR -D	(-A OR -B OR -C) hợp giải (B OR -C OR -D)
	-B OR -C	(-A OR -B OR -C) hợp giải A
	9	
	B OR F	(C OR F) hợp giải (B OR -C)
	-A OR -D OR F	(C OR F) hợp giải (-A OR -C OR -D)
	-B OR F	(C OR F) hợp giải (-B OR -C)
	B OR -C OR F	(-C OR D) hợp giải (B OR -D OR F)
	-A OR -C	(-C OR D) hợp giải (-A OR -C OR -D)
	-A OR -C OR -D OR F	(-A OR -B OR -C) hợp giải (B OR -D OR F)
	-C OR D	(B OR -C OR -D) hợp giải (-B OR -C)
	-A OR -C OR -F	(D OR F) hợp giải (-A OR -C OR -D)
	-C OR -D OR F	(B OR -D OR F) hợp giải (-B OR -C)
	3	
	-A OR F	(C OR F) hợp giải (-A OR -C OR F)
	-D OR F	(C OR F) hợp giải (-C OR -D OR F)
	-C OR F	(-C OR D) hợp giải (-C OR -D OR F)
	0	
	NO	KB không entails alpha

5. Test case 5

Input4.txt	Ouput4.txt	Ghi Chú
D OR E	6	
5	-A OR C	(-A OR B OR C) hợp giải (-B OR C)
-A OR B OR C	-A OR B OR E	(-A OR B OR C) hợp giải (-C OR E)
-B OR C	B OR C	(-A OR B OR C) hợp giải A
-C OR E	-B OR E	(-B OR C) hợp giải (-C OR E)
-D OR -E	-C OR -D	(-C OR E) hợp giải (-D OR -E)
A	-C	(-C OR E) hợp giải -E
	13	
	-A OR C OR E	(-A OR B OR C) hợp giải (-B OR E)
	-A OR B OR -D	(-A OR B OR C) hợp giải (-C OR -D)
	-A OR B	(-A OR B OR C) hợp giải -C
	C	(-B OR C) hợp giải (B OR C)
	-B OR -D	(-B OR C) hợp giải (-C OR -D)
	-A OR E	(-C OR E) hợp giải (-A OR C)
	B OR E	(-C OR E) hợp giải (B OR C)
	-A OR -D	(-A OR C) hợp giải (-C OR -D)
	-A	(-A OR C) hợp giải -C
	C OR E	(B OR C) hợp giải (-B OR E)
	B OR -D	(B OR C) hợp giải (-C OR -D)
	B	(B OR C) hợp giải -C
	6	
	-A OR C OR -D	(-A OR B OR C) hợp giải (-B OR -D)
	C OR -D	(-B OR C) hợp giải (B OR -D)
	E	(-C OR E) OR C
	{}	-A OR A
	-A OR -D OR E	(-A OR B OR E) hợp giải (-B OR -D)
	-D OR E	(-B OR E) hợp giải (B OR -D)
	YES	KB entails alpha

III. Đánh giá về giải thuật

1. Ưu điểm

- Thuật toán đơn giản, dễ hiểu
- Thuật toán chạy chính xác, thực hiện đầy đủ các bước từ mã giả đưa ra.
- Về cơ bản thuật toán đã giải quyết được những yêu cầu bài toán đặt ra.

2. Nhược điểm

- Thuật toán đang xét hợp giải cho tất cả các cặp mệnh đề có trong danh sách clauses bao gồm cả các cặp mệnh đề cũ đã xét hợp giải ở vòng lặp trước → Việc hợp giải cho các cặp mệnh cũ ở vòng lặp trước là không cần thiết.
- Thuật toán giải theo hướng dữ liệu, thiếu cơ chế định hướng: các mệnh đề phát sinh có thể không liên quan đến mệnh đề cần phủ định.
- Thuật toán đang giải theo đầu vào có KB và alpha tuân theo dạng chuẩn CNF

3. Cải tiến

- Không xét hợp giải cho các cặp mệnh đã được hợp giải ở vòng lặp trước
- Ưu tiên hợp giải các mệnh đề có ít literal hơn giúp tăng khả năng thu được mệnh đề sau hợp giải ngắn hơn.
- Nếu đầu vào chưa tuân theo dạng CNF. Viết thêm các hàm áp dụng các luật biến đổi mệnh để đưa về dạng chuẩn CNF.

IV. Đánh giá dựa theo tiêu chí

STT	Tiêu chí	Mức độ hoàn thành
1	Đọc dữ liệu vào và lưu trong cấu trúc dữ liệu phù hợp	100%
2	Cài đặt giải thuật hợp giải logic mệnh đệ	100%
3	Các bước suy diễn phát sinh đủ mệnh đề và kết luận đúng	100%
4	Tuân thủ mô tả định dạng của đề bài	100%
5	Báo các test case và đánh giá	100%