Report 3

Logic

Nguyễn Văn Hậu –20127493

content

1. Overview

Student information:

MSSV	Họ tên
20127493	**Nguyên Văn Hậu**

completion level: 100%

a. Project information:

- Given a knowledge base(kb) and a query α,set of propositional clause in CNF.
- Provide 5 non-trial test cases, along with submibtion
- Evaluate the advantages and disadvantages of the union algorithm on propositional logic, as well as propose solutions to overcome the problem.
- **Note**
  - that terminate state is checked at the end of each loop, not after a new propositional is generated
  - literal in the input or output are arranged alphabetically

Description of the program

1.Algorithm used:

Propositional Logic Resolution (PL-Resolution) Algorithm

PL-Resolution Pesudo code

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

Figure 7.12 A simple resolution algorithm for propositional logic. The function PL-RESOLVE returns the set of all possible clauses obtained by resolving its two inputs.

- resolve each and every possible pair of propositional to generate new propositional
- New propositional which is not already contained in the KB will be added to the KB
- Terminate only if :
  - No new propositional that can be added
  - Empty propositional is generated

- **Disavantage**:

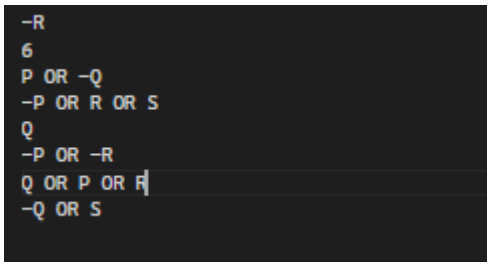
- Propositional logic has very limited expressive power(unlike natural language)
- There are many duplicate literals or redundant clauses
- Too many propositional to handle but many step are pointless

- **Avantage**:
  - easy to combine with operand such as (or..)
  - complete algorithm
  - Number of propositionals is finite so there are only finitely many distinct propopsitionals that can be constructed
- **Improvement**
  - pruning of repeating/redundant clauses before added to KB

2. test scenarios

a. test case 1

- input



- Output

```
9
-Q OR R OR S
P
-Q OR -R
P OR R
-P OR S
Q OR R OR S
S
-P
P OR R OR S
8
9
-Q OR R OR S
P
-Q OR -R
P OR R
-P OR S
Q OR R OR S
S
-P
P OR R OR S
-Q
R OR S
-P OR -Q OR S
-R
-P OR Q OR S
Q OR R
{}
P OR -Q OR S
YES
```

Log

```
[!] Finished read file input.txt
[*] KB: [['P', '-Q'], ['-P', 'R', 'S'], ['Q'], ['-P', '-R'], ['Q', 'P', 'R'], ['-Q', 'S']]
[*] NOT alpha: [['R']]
        Resolve ['P', '-Q'] with ['-P', 'R', 'S'] get ['-Q', 'R', 'S']
        Resolve ['P', '-Q'] with ['Q'] get ['P']
        Resolve ['P', '-Q'] with ['-P', '-R'] get ['-Q', '-R']
        Resolve ['P', '-Q'] with ['Q', 'P', 'R'] get ['P', 'R']
        Resolve ['-P', 'R', 'S'] with ['-P', '-R'] get ['-P', 'S']
        Resolve ['-P', 'R', 'S'] with ['Q', 'P', 'R'] get ['Q', 'R', 'S']
        Resolve ['Q'] with ['-Q', 'S'] get ['S']
        Resolve ['-P', '-R'] with ['R'] get ['-P']
        Resolve ['Q', 'P', 'R'] with ['-Q', 'S'] get ['P', 'R', 'S']
        Resolve ['P', '-Q'] with ['-P'] get ['-Q']
        Resolve ['-P', 'R', 'S'] with ['P'] get ['R', 'S']
        Resolve ['-P', 'R', 'S'] with ['-Q', '-R'] get ['-P', '-Q', 'S']
        Resolve ['Q'] with ['-Q', '-R'] get ['-R']
        Resolve ['-P', '-R'] with ['Q', 'R', 'S'] get ['-P', 'Q', 'S']
        Resolve ['Q', 'P', 'R'] with ['-P'] get ['Q', 'R']
        Resolve ['P'] with ['-P'] get {}
        Resolve ['-Q', '-R'] with ['P', 'R', 'S'] get ['P', '-Q', 'S']
[+] KB entails alpha.
[!] Finished write to output.txt.
```

b. test case 2

- Input

```
-R
3
P OR Q
-Q OR P OR -R
-P OR -R
```

- Output

```
5
P OR -R
Q OR -R
-Q OR -R
P OR -Q
-P
4
5
P OR -R
Q OR -R
-Q OR -R
P OR -Q
-P
P
Q
-R
-Q
1
4
5
P OR -R
Q OR -R
-Q OR -R
P OR -Q
-P
P
Q
-R
-Q
{}
YES
```

- Log

```
[*] KB: [['P', 'Q'], ['-Q', 'P', '-R'], ['-P', '-R']]
[*] NOT alpha: [['R']]
        Resolve ['P', 'Q'] with ['-Q', 'P', '-R'] get ['P', '-R']
        Resolve ['P', 'Q'] with ['-P', '-R'] get ['Q', '-R']
        Resolve ['-Q', 'P', '-R'] with ['-P', '-R'] get ['-Q', '-R']
        Resolve ['-Q', 'P', '-R'] with ['R'] get ['P', '-Q']
        Resolve ['-P', '-R'] with ['R'] get ['-P']
        Resolve ['P', 'Q'] with ['P', '-Q'] get ['P']
        Resolve ['P', 'Q'] with ['-P'] get ['Q']
        Resolve ['-P', '-R'] with ['P', '-R'] get ['-R']
        Resolve ['R'] with ['-Q', '-R'] get ['-Q']
        Resolve ['R'] with ['-R'] get {}
[+] KB entails alpha.
[!] Finished write to output.txt.
```

c. Test case 3

- Input

```
-A
4
-A OR B
B OR -C
A OR -B OR C
-B
```

- Output

```
3
-A
B
-C
4
3
-A
B
-C
-B OR C
A OR C
A OR -B
{}
YES
```

- Log

```
[!] Finished read file input3.txt
[*] KB: [['-A', 'B'], ['B', '-C'], ['A', '-B', 'C'], ['-B']]
[*] NOT alpha: [['A']]
        Resolve ['-A', 'B'] with ['-B'] get ['-A']
        Resolve ['-A', 'B'] with ['A'] get ['B']
        Resolve ['B', '-C'] with ['-B'] get ['-C']
        Resolve ['A', '-B', 'C'] with ['-A'] get ['-B', 'C']
        Resolve ['A', '-B', 'C'] with ['B'] get ['A', 'C']
        Resolve ['A', '-B', 'C'] with ['-C'] get ['A', '-B']
        Resolve ['-B'] with ['B'] get {}
[+] KB entails alpha.
[!] Finished write to output.txt.
```

d. Test case 4

- Input

```
U
5
P OR Q
-Q OR R
-Q OR S
-P OR U
-R OR U
```

- Output

```
6
P OR R
P OR S
Q OR U
-Q OR U
-P
-R
9
6
P OR R
P OR S
Q OR U
-Q OR U
-P
-R
P OR U
Q
R OR U
-Q
S OR U
R
```

```
P
S
U
1
9
6
P OR R
P OR S
Q OR U
-Q OR U
-P
-R
P OR U
Q
R OR U
-Q
S OR U
R
P
S
U
{}
YES
```

- Log

```
[*] KB: [['P', 'Q'], ['-Q', 'R'], ['-Q', 'S'], ['-P', 'U'], ['-R', 'U']]
[*] NOT alpha: [['-U']]
        Resolve ['P', 'Q'] with ['-Q', 'R'] get ['P', 'R']
        Resolve ['P', 'Q'] with ['-Q', 'S'] get ['P', 'S']
        Resolve ['P', 'Q'] with ['-P', 'U'] get ['Q', 'U']
        Resolve ['-Q', 'R'] with ['-R', 'U'] get ['-Q', 'U']
        Resolve ['-P', 'U'] with ['-U'] get ['-P']
        Resolve ['-R', 'U'] with ['-U'] get ['-R']
        Resolve ['P', 'Q'] with ['-Q', 'U'] get ['P', 'U']
        Resolve ['P', 'Q'] with ['-P'] get ['Q']
        Resolve ['-Q', 'R'] with ['Q', 'U'] get ['R', 'U']
        Resolve ['-Q', 'R'] with ['-R'] get ['-Q']
        Resolve ['-Q', 'S'] with ['Q', 'U'] get ['S', 'U']
        Resolve ['P', 'R'] with ['-P'] get ['R']
        Resolve ['P', 'R'] with ['-R'] get ['P']
        Resolve ['P', 'S'] with ['-P'] get ['S']
        Resolve ['Q', 'U'] with ['-Q', 'U'] get ['U']
        Resolve ['-U'] with ['U'] get {}
[+] KB entails alpha.
[!] Finished write to output.txt
```

e. Test case 5

- Input

```
-U
5
P OR Q
-Q OR R
-Q OR S
-P OR U
-R OR U
```

- Output

```
4
P OR R
P OR S
Q OR U
-Q OR U
3
4
P OR R
P OR S
Q OR U
-Q OR U
P OR U
R OR U
S OR U
0
3
4
P OR R
P OR S
Q OR U
-Q OR U
P OR U
R OR U
S OR U
NO
```

- Log

```
[*] KB: [['P', 'Q'], ['-Q', 'R'], ['-Q', 'S'], ['-P', 'U'], ['-R', 'U']]
[*] NOT alpha: [['U']]
        Resolve ['P', 'Q'] with ['-Q', 'R'] get ['P', 'R']
        Resolve ['P', 'Q'] with ['-Q', 'S'] get ['P', 'S']
        Resolve ['P', 'Q'] with ['-P', 'U'] get ['Q', 'U']
        Resolve ['-Q', 'R'] with ['-R', 'U'] get ['-Q', 'U']
        Resolve ['P', 'Q'] with ['-Q', 'U'] get ['P', 'U']
        Resolve ['-Q', 'R'] with ['Q', 'U'] get ['R', 'U']
        Resolve ['-Q', 'S'] with ['Q', 'U'] get ['S', 'U']
[-] KB does not entail alpha.
[!] Finished write to output.txt.
```

3. How to run the program

Step 1: enter the syntax python MSSV.py (20127493.py)

Step 2: enter filename

❖ **Note that:**

• The input files must exist.
• The input files can not be duplicated.

There is also a sample command stored in file ***input.txt***

```
PS C:\Users\Admin\Desktop\ai p2> py .\20127493.py
```

→view result at output.txt and step by step solve at console

4. Reference

- https://github.com/t3bol90/PL-Resolution
- http://logic.stanford.edu/intrologic/notes/chapter_05.html
- The document in the Computer Science Department at the University of Science, Vietnam National University, Ho Chi Minh City
- Book Artificial Intelligence: A Modern Approach, Third Edition, Chapter 7