MINISTRY OF EDUCATION AND TRAINING

VIETNAM NATIONAL UNIVERSITY, HO CHI MINH CITY - VNUHCM

UNIVERSITY OF SCIENCE

FACULTY OF INFORMATION TECHNOLOGY

-----ଓ📖ଌ -----

ARTIFICIAL INTELLIGENCE

LAB 02: PL RESOLUTION

Instructor: Bùi Tiến Lên, Trần Quốc Huy, Phạm Trọng Nghĩa

Name: Phạm Phú Toàn

Identity: 21127183

class: 21CLC08

Table of content

I. Checklist

No	Work	Score
1	Read the input data and successfully store it in some data structure	10%
2	The output file strictly follows the lab specifications.	10%
3	Implement the propositional resolution algorithm.	20%
4	Provide a complete set of clauses and exact conclusion.	30%
5	Five test cases: both input and output files.	10%
6	Discussion on the algorithm's efffiency and suggestions.	20%
Total		100%

→ Complete all the requirements: 100%.

II. Describe Works

- *Input files* and *Output files*:

(Each output${number}.txt is corresponding to the file input${number}.txt, output1.txt and input1.txt is for the example in the Instruction PDF, from input2.txt to input6.txt is my own generated input tests)

- *Src/Main.py*:
    - **inputFile(), outPut()**: for reading, writing result to file.
        - **inputFile()**:
            - return Kb, query as list in python.
            - For example:
                - KB = A OR B, -C → KB = [['A', 'B'],['-C']]
    - **interact()**
        - function let user input the input file name and the specified output file name.
    - **checkIfLiteralIsOpposite(literal1, literal2)**:
        - check if two given literal is opposite.
        - For example:
            - 'A' and '-A' are opposite, and function will return true.
            - 'A' and 'B' are not opposite, and function will return false.
    - **returnOppositeQuery(query)**:
        - Return the opposite query (-α)

    - For example:
        - [‘A’] return [‘-A’]
        - [‘B or A’] return [‘-B’,’-A’]
        - [‘A or B’, ‘-C’] return [‘-A’,’-B’,’C’]
    - Note: the requirement is not really clear and there is no example when the query is more than two clauses or a clause with two literals, so I do with my own perspective.
- **checkIfSimilarClause(clause1, clause2):**
    - Return true when clause 1 is the same as clause 2.
    - For example:
        - [‘A’] and [‘B’] are not the same.
        - [‘A’, ‘C’] and [‘A’, ‘C’] are the same.
- **PLResolve(clause1, clause2)**:
    - The function receives two clauses and returns new clause, if possible, none if can’make new clause or new clause contains two complementary literals, [].
- **PLResolution(kb, query):**
    - The main function of PLResolution Algorithm, implemented according to the instruction, uses and connects all the functions above.
    - Function returns the result: ‘YES’, ‘NO’; newClause contains clause created through the algorithm.

III. Evaluate the algorithm.

- **The complexity**:based on the the KB and query, is sound and complete.
- **Obstacle**:
    - The KB and α need to be converted to CNF clauses before passing to function.
    - Every time a new clause is created, it is passed to the current clauses list and so the clauses, which need to be considered, increase dramatically after every loop, which leads to difficulty in finding the result.
- **Proposed Improvement**:
    - The algorithm should focus on processing and creating new clauses which are related to query rather than focus on all clauses.

- If possible, the clauses should be converted to Horn clauses and applied Modus ponens inference rule with forward chaining or backward chaining instead to improve the running time.

IV. Reference

Bùi Tiến Lên. (n.d.). *Slide cơ sở trí tuệ nhân tạo.* Khoa công nghệ thông tin - Đại học khoa học tự nhiên.

Python Tutorial. (n.d.). Retrieved from W3 School: https://www.w3schools.com/python/default.asp