**VIETNAM UNIVERSITY – HO CHI MINH CITY**

**UNIVERSITY OF SCIENCE**

**FACULTY OF INFORMATION TECHNOLOGY**

A blue and white logo

Description automatically generated­­­­­🙢🕮🙠 ­­­­­

**ARTIFICIAL INTELLIGENCE   
LAB 02: PL RESOLUTION**

**Lecturer**: Bùi Tiến Lên, Trần Quốc Huy, Phạm Trọng Nghĩa

**Name**: Ngô Quốc Quý

**ID**: 21127679

**Class**: 21CLC08

**TABLE OF CONTENTS**

[**I.** **Checklist:** 3](#_Toc143117597)

[**II.** **Brief description:** 3](#_Toc143117598)

[**a.** **handle\_input:** 3](#_Toc143117599)

[**b.** **handle\_output:** 3](#_Toc143117600)

[**c.** **negate\_literal:** 4](#_Toc143117601)

[**d.** **check\_negation\_literal:** 4](#_Toc143117602)

[**e.** **CNF:** 4](#_Toc143117603)

[**f.** **PL\_resolve:** 4](#_Toc143117604)

[**g.** **PL\_resolution:** 4](#_Toc143117605)

[**h.** **main:** 5](#_Toc143117606)

[**III.** **Evaluation:** 5](#_Toc143117607)

[**IV.** **References:** 5](#_Toc143117608)

1. **Checklist:**

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Specifications** | **Complete** | **Finish** |
| 1 | Read the input data and successfully store it in some data structures. | 10% | Done |
| 2 | The output file strictly follows the lab specifications. | 10% | Done |
| 3 | Implement the propositional resolution algorithm. | 20% | Done |
| 4 | Provide a complete set of clauses and exact conclusion. | 30% | Done |
| 5 | Five test cases: both input and output files. | 10% | Done |
| 6 | Discussion on the algorithm’s efficiency and suggestions. | 20% | Done |
| **Total** | | **100%** |  |

1. **Brief description:**

The lab02 follow the PL-resolution algorithm to implement if a given knowledge base (KB) entails a query (alpha). There are 5 files input test cases (input\_1.txt, input\_2.txt,…) will solve and write 5 files output corresponding to input files which are followed the output requirements of the lab.

* 1. **handle\_input:**

Input: filename (string)

Output: query (list) , KB (list)

Description:

+ Function handle\_input gets a string value names filename then reads the file with the path corresponding to filename which are transmit in and follows the rules of input requirement.

+ Get the query and KB by splitting the input file, then return

* 1. **handle\_output:**

Input: is\_success (bool), clauses (list)

Output: None

Description:

+ Check if the is\_success is true or false to get YES or NO string to write to output file.

+ Get the values in clauses (list) then write it to output file.

* 1. **negate\_literal:**

Input: clause (list)

Output: new\_clause (list)

Description:

**+** Iterate over each element in clause (list) using two for loops, check if it has “-’’ then negate it by delete the “-’’ of that string.

**+** Return a new list contains the literals after negation.

* 1. **check\_negation\_literal:**

Input: l1 (list), l2 (list)

Output: True or False

Description:

**+** Comparing two literal l1 and l2, then check if they’re opposite then return True, otherwise if they’re equal then return False.

**+** The function get the boolean value to check opposite two literal which can be delete when PL-resolve.

* 1. **CNF:**

Input: query (list), KB (list)

Output: KB + negation of query (list)

Description:

**+** Use negate\_literal function to negate the query.

**+** After negating the query, merge it with the KB then return.

* 1. **PL\_resolve:**

Input: c1 (list), c2 (list)

Output: resolvents (list or None or [])

Description:

**+** Use two for loops to iterate clause 1 and clause 2, use check\_negation\_literal to check if clause 1 and clause 2 has opposite value then save that elements into 2 temp list.

**+** Remove the elements in temp list then merge two clause into resolvents.

**+** If the resolvents is none then return None, otherwise, if the resolvents has the values, I sort it alphabetical order then return.

* 1. **PL\_resolution:**

Input: query (list), KB (list)

Output: True/False, new\_clause (list)

Description:

+ Apply the PL-resolution algorithm by using while loops.

+ Create a temp new clause to store the new after using PL-resolve to get resolvents from the pair C1 and C2 , check if the resolvents are [] then append to main new\_clause and return true, new\_clause.

+ Check if the main clauses not contains the resolvents then append the resolvents into temp new clause.

+ Check if the temp new clause is [] then return False and main new clause.

+ In every loops while will append the temp new clause to main new clause, and similar to main clauses.

* 1. **main:**

Input: None

Output: None

Description:

**+** Call handle\_input to handle the input file.

**+** Get is\_success and clauses when implement the PL-resolution

+ Call handle\_output to write the output into output file.

1. **Evaluation:**

The ways to apply the algorithm maybe get lots of mistakes and not optimization.

Dealing with negative clauses is cumbersome and requires many steps, cause the algorithm is not handling in an optimization.

Use too much loops then may be make the algorithm becomes lower and quite waste of times.

The code style hasn’t clean yet.

1. **References:**

**Neso Academy:** [(1) The Resolution Principle (Preliminaries) - YouTube](https://www.youtube.com/watch?v=SjEQNOV5FMk&t=245s)

**kieuconghau’s github:** [kieuconghau/pl-resolution: HCMUS - Artificial Intelligence - Lab 2: Popositional Logic - Resolution (github.com)](https://github.com/kieuconghau/pl-resolution)

**HCMUS – moodle, slides, videos.**