16/04/2022

BUI QUOC HUNG

vnuhcm – uNIVERSITY OF SCIENCE

**REPORT**

Logic Inference Project

Fundamentals of Artificial Intelligence

**Instructors**

Pham Trong Nghia

Nguyen Thai Vu

Mục lục

[Introduction 2](#_Toc101085348)

[About the author 2](#_Toc101085349)

[Problem statement 2](#_Toc101085350)

[Implementation 3](#_Toc101085351)

[Experiments 4](#_Toc101085352)

[Conclusion 14](#_Toc101085353)

[Pros 14](#_Toc101085354)

[Cons 14](#_Toc101085355)

[Potential Improvements 14](#_Toc101085356)

[Evaluation 14](#_Toc101085357)

[References 15](#_Toc101085358)

# Introduction

## About the author

Student name: Bui Quoc Hung

Student ID: 20127508

## Problem statement

Given a knowledge base (KB) with and both of them are represented by propositional logic and standized into CNF. Determine KB entails () by resolutions.

# Implementation

First, according to the problem requirements, the input are stored in input.txt, therefore we use built-in modules of Python to read line-by-line. For each statement, we preprocess them, convert them into string, then store them into several variable to keep their information.

Once we done for the preparation stage, we call PL\_Resolution to start our resolution progress. In general, the main idea of the algorithms is almost represented by fig1.

Ảnh có chứa văn bản

Mô tả được tạo tự động

*Fig1: Propositional logic resolution algorithm – via Artificial Intelligence: A Modern Approach*

First, we push into the clauses set (this source code does not support with multiple literals inside), then according to the requirements “All the sentences have the truth value is True, then we should eliminate it”, we call a redundant-filtering function to eliminate any clauses like that. After that, we consider all pair of clause in the set of clauses that we have processed, called PL\_Resolve, which performs resolution if there is any pair of negative and non-negative literal in that pair, then the function returns the results.

* If the resolvents is empty, then the function PL\_Resolution returns true, which means we can performs the query by the given KB

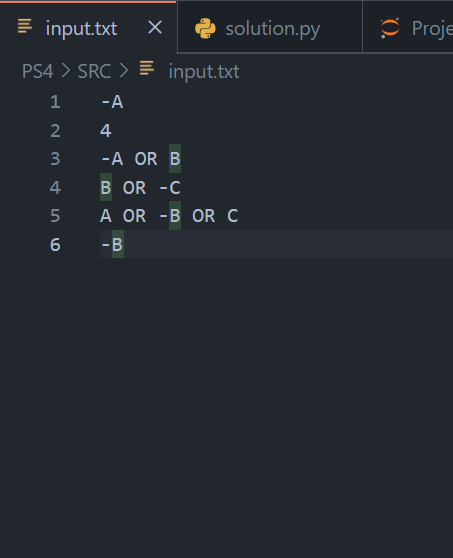
After each time we invoke PL\_Resolve, we store the resolvents into a set, called “new”. If “new” is the subset of clauses, which indicates we can’t performs from the KB, return False. If it not, merge the new storage into the set of clauses then continue to resolve these clauses.

After the PL\_Resolution function reachs its terminal state, following the format in the problem file, we write the logs storage – which save the resolvents if it does not appears in clauses storage – into the output.txt by the number of resolvents and resolvents in each loop. Finally, we display the terminal state of PL\_Resolution at the end of the file, YES for true and NO for false.

# Experiments

In the source code, I also display the 2 reason factors of each resolvent, so we can easily understand of the algorithm. According to the requirements, the algorithm does not stop when the resolvents so there are also inference even the resolvents is empty. Because I have used set – built-in data structure in my source code – the order of clauses after being pushed into set is messy since they use several kinds of stuff random stuff with unknown seed so I can’t control the order. However, my implementation is reliable, which is demonstrated in these following examples.

First example (from the problem statement)



*Fig2: Sample 1’s input*

Ảnh có chứa văn bản

Mô tả được tạo tự động

*Fig3: Sample 1’s output*

Second example (from the problem statement)

Ảnh có chứa văn bản

Mô tả được tạo tự động

*Fig4: Sample 2’s input*

Ảnh có chứa văn bản, ảnh chụp màn hình, màn hình, đen

Mô tả được tạo tự động

*Fig5: Sample 2’s output*

Third example

Ảnh có chứa văn bản

Mô tả được tạo tự động

*Fig6: Sample 3’s input*

Ảnh có chứa văn bản

Mô tả được tạo tự động

*Fig7: Sample 3’s output*

Fourth example

Ảnh có chứa văn bản

Mô tả được tạo tự động

*Fig8: Sample 4’s input*

Ảnh có chứa văn bản

Mô tả được tạo tự động

*Fig9: Sample 4’s output*

Fifth example

Ảnh có chứa văn bản, màn hình, đen, ảnh chụp màn hình

Mô tả được tạo tự động

*Fig10: Sample 5‘s input*

Ảnh có chứa văn bản

Mô tả được tạo tự động

*Fig11: Sample 5‘s output*

# Conclusion

## Pros

* If the knowledge base is enough, the algorithm can find the suitable solution.
* The algorithm is simple and easy to understand and implementation

## Cons

* There are so many redundant steps that this algorithm performs if the KB’s clauses set are not focus on the query
* In general, proportional logic has limited expressive power, for example, all, some,… so the algorithm need various clauses and literals to represent the problem and solve it.
* Takes much time in the resolution function since we need to consider all pair of clauses and all pair of literals within them.
* It is hard to extend the KB to solve a very different query compare to the original KB
* Inference takes time since it must iterates all the KB’s clauses
* Difficult to represent KB If the KB is changed overtime

## Potential Improvements

* According to the problem, the input follows CNF, so there are not any kind of inputs C1, C2 that need to be resolved, between both complex clauses, for example: and , and , and there is not any clause like that in CNF standard, so just let C1 is always longer than C2 (length of literal of C2 is always 1, as I mentioned above), then just construct each longer clauses into a binary search tree (Red BLack Tree, AVL Tree, etc.), then search its negative literal efficiently.
* Faster inference progress

## Evaluation

|  |  |  |
| --- | --- | --- |
| No. | Criteria | Percentage of completeness |
| 1 | Read the input and store those data in suitable data structure | 100% |
| 2 | Implement resolution for propositional logic | 100% |
| 3 | Each step of inference progress generate completely clauses and correct conclusion | 100% |
| 4 | Follow the description of format | 100% |
| 5 | Report contains test case and evaluation | 100% |

# References

[1] Artificial Intelligence: A Modern Approach, Fourth Edition, Chapter 7