VNUHCM - University of Science

The Faculty of Information Technology

PROJECT REPORT 3: LOGIC

Subject: Fundamentals of Artificial Intelligence

Supervisor :	Pham Trong Nghia
	Nguyen Thai Vu
Student's Fullname :	Thai Chi Hien
Student's ID :	20127495

HCM CITY - 2022

Table of Contents

1. Project Requirements:

Given the knowledge base KB and a query α, both represented by propositional logic and normalized to CNF form. Check if the KB entails α (KB ⊨α) by using resolution.

- Report at least 5 test scenarios to validate the program (and therefore the scenario should not be too simple).
- Briefly evaluate the advantages and disadvantages of the resolution algorithm on propositional logic, as well as suggest some solutions to overcome the problem.

- Input data specification: KB and α in CNF form are stored in file **input.txt.** The file has the following format:

- The first line contains the query α
- The second line contains an integer ***N*** – the number of clauses in KB
- The next ***N*** lines represent the clauses in the KB, one per line

Positive literals are represented by a single uppercase character (A-Z). A negative literal is a positive literal with a minus sign ('-') right before the character .

The OR keyword joins concatenated literals. There can be one or more spaces between literals and keywords.

- Output data specification: The set of clauses generated during the resolution and the conclusion is stored in the file **output.txt**. The file has the following format:

- The first line contains the integer M_1 – the number of clauses generated in the first loop. The next M_1 line represents the clauses generated in the first loop (including the empty one), one per line. Empty clauses are represented by the string "{}"
- The next loops (with M_2, M_3…, M_n clauses respectively) are represented similarly as above.
- The last line presents the conclusion, that is, answering the question “KB entails α?”. Print YES if KB entails α. In contrast, print NO.
- Ignore duplicate clauses (occurring in the same loop or original KB or previous loops).

- The main function must perform the following basic task:
  - Read input data and store in appropriate data structure
  - Call the PL-Resolution function to execute the resolution algorithm
  - Write output data to output file in valid format
- Note the meaning of true and false return values in the PL-RESOLUTION function. Don't forget that when we perform the resolution, we need the negation of the query α.
- Literals in the same clause (for both input and output data) are ordered alphabetically
- Check the inference condition at the end of each loop, that is, when all new clauses have been generated from the current KB, not after each clauses is generated
- Clauses of the form A ∨ B ∨ -B have the value True because they are equivalent to A ∨ True. Clauses like these are useless for inference and can therefore be ignored
- Input data is assumed to be in valid format, you don't need to check this

2. Project Process:

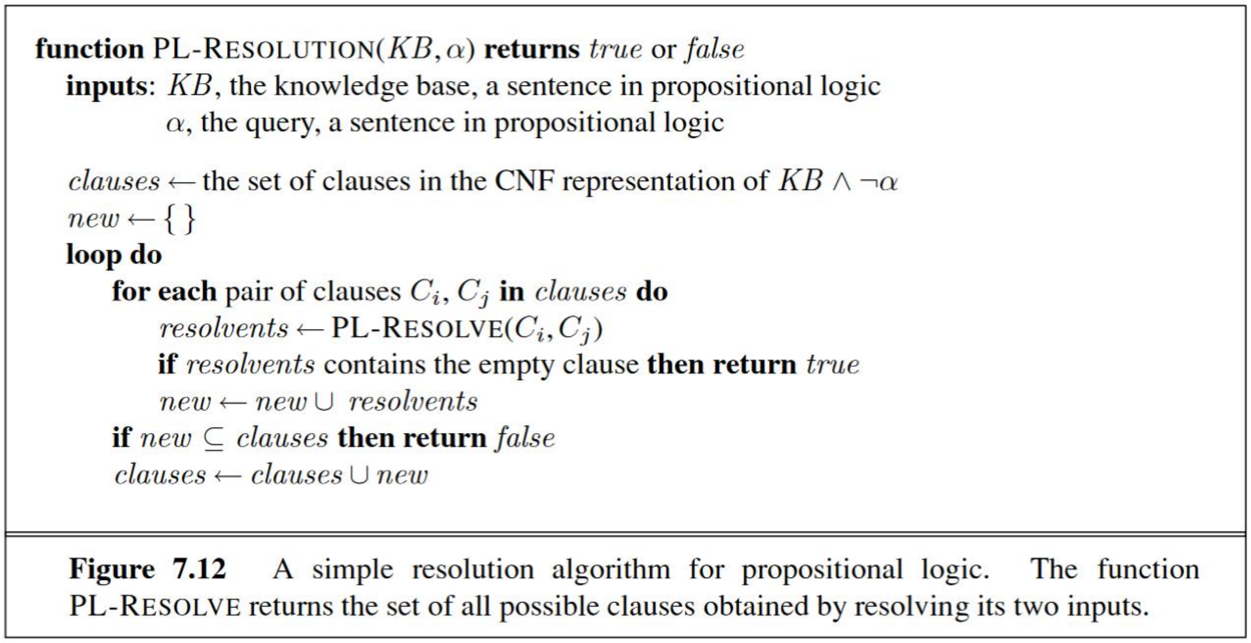
#	**Criteria**	**Progress**
1	Read input data and store it in a suitable data structure	100%
2	Implement the resolution algorithm on propositional logic	100%
3	The inference steps generate correct clauses and conclusions	100%
4	Follow strictly the format of the lab specification	100%
5	Reports on test cases and algorithm evaluation	100%

3. Resolution Algorithm:

- The Resolution Algorithm helps to check wether KB ⊨ α by using the inference rules to prove that KB ∧ ¬ α is unsatisfiable. The resolution works only on clauses in CNF form, so KB ∧ ¬ α must be converted into CNF form first. In this project, the input data is guaranteed to be in CNF form so we don't need to take care of it. The resolution is performed on each pair of clauses with complementary literals and produce a new resolved clause. These clauses will be added to the KB if it is not already present. The process goes on until two cases occur:

- Empty clauses are created by resolving two clauses. The derivation of the empty clause means that the database contains a contradiction or KB entails α.
- No new clauses are created during the resolution. This leads to the conclusion that KB does not entail α.

- Below is the pseudocode code of the resolution algorithm from the book *Artificial Intelligence: A Modern Approach (AIMA) Third Edition*, Chapter 7, Figure 7.12 :



```
function PL-RESOLUTION($KB, \alpha$) returns $true$ or $false$
  inputs: $KB$, the knowledge base, a sentence in propositional logic
          $\alpha$, 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.

4. Test cases:

Note : run the program in file **main.py*

a. Test case 1: Simple case with and two singleton clauses and two clauses which have 3 literals

Input.txt	**Output.txt**
D	5
4	-A OR -B OR D
-A OR -B OR C	-B OR C
-B OR -C OR D	-A OR C
A	-C OR D
B	-B OR -C
	6
	-A OR -B
	-B OR D
	C
	-A OR D
	-C
	-B
	3
	D
	-A
	{}
	YES

b. Test case 2: The test case has a clause with 4 literals

Input.txt	**Output.txt**
E	10
9	B
A	F
-A OR B	-A OR -C OR -D OR E
-B OR -C OR -D OR E	-B OR -D OR E
-A OR F	-B OR -C OR E
-G OR H	-B OR -C OR -D
-F OR G	-A OR G
-G OR E	-F OR H
C	-F OR E
D	-G
	17
	-C OR -D OR E
	G
	-A OR -D OR E
	-A OR -C OR E

	-A OR -C OR -D -A OR H -A OR E -F -B OR E -B OR -D -B OR -C -D OR E -C OR E -C OR -D H E -A 6 {} -A OR -D -A OR -C -B -D -C YES

c. Test case 3: Many clauses have more than one literal and few singleton clauses

Input.txt	**Output.txt**
H 12 -A OR C -B OR C -A OR D -B OR D -A OR -E OR -D OR F -E OR B -E OR C -A OR F OR -G -A OR F OR H -E OR -F OR H A E	17 C C OR -E -A OR -E OR F D -A OR -B OR -E OR F D OR -E -A OR -E OR -D OR H -E OR -D OR F -A OR -D OR F B -A OR -E OR -G OR H F OR -G -A OR -E OR H F OR H -A OR F -F OR H -E OR -F 23 -A OR -B OR -E OR H

	-B OR -E OR F -A OR -B OR F -A OR -E OR -D -A OR -G OR H -A OR -E OR -G -A OR H -E OR -D OR H -A OR -D OR -E OR H -E OR -G OR H -E OR H -E OR F -D OR F F -A OR -D OR H -F -A OR -E -A OR -B OR -E -E OR -D -A OR -D OR -E -G OR H -E OR -G H 15 -B OR -E OR H -B OR F -A OR -B OR H -B OR -E -A OR -G -D OR -E OR H -D OR H -E -D OR -E -A OR -D -A -A OR -B -D -G {} YES

d. Test case 4: The query α is not just a literal

Input.txt	**Output.txt**
-A OR E 6 -A OR -B OR C OR D	8 -A OR -B OR D OR E -A OR -B OR D OR F

-C OR E -C OR F -D OR E -D OR F B	-A OR -B OR C OR E -A OR -B OR C OR F -A OR C OR D -B OR C OR D -C -D 18 -A OR -B OR D -A OR -B OR C -A OR -B OR E -A OR -B OR E OR F -A OR D OR E -B OR D OR E -A OR -B OR F -A OR D OR F -B OR D OR F -A OR C OR E -B OR C OR E -A OR C OR F -B OR C OR F C OR D -A OR D -A OR C -B OR D -B OR C 15 -A OR E -B OR E -A OR E OR F -B OR E OR F D OR E -A OR F -B OR F D OR F C OR E C OR F D C -A OR -B -A -B 4 E E OR F F {} YES

e. Test case 5: Same as test case 2, but we change the query α to its negative

Input.txt	**Output.txt**
-E	8
9	B
A	F
-A OR B	-A OR -C OR -D OR E
-B OR -C OR -D OR E	-B OR -D OR E
-A OR F	-B OR -C OR E
-G OR H	-A OR G
-F OR G	-F OR H
-G OR E	-F OR E
C	10
D	-C OR -D OR E
	G
	-A OR -D OR E
	-A OR -C OR E
	-A OR H
	-A OR E
	-B OR E
	-D OR E
	-C OR E
	H
	0
	NO

5. Algorithm analysis and suggestions:

a. Advantages:

The Resolution Algorithm is simple and easy to understand so it's easy to implement it too. The Resolution Algorithm also guarantees to be complete and returns a result if KB entails α or not. Its completeness is explained that the set of KB when running the algorithm is finite because the clauses added when resolving will be completely different from the ones in the set. So the algorithm will terminate when it can generate a empty clause or no more new clauses can be added. The detailed explanation can be found in the book *Artificial Intelligence: A Modern Approach (AIMA) Third Edition*, Chapter 7, Page 255, **Completeness of resolution.**

b. Disadvantages:

Check and perform the resolution on each pair of clauses and then go back and execute this algorithm from the beginning with the newly added clauses is a time consuming. Moreover, the resolution also leads to many useless inferences. Some inferences are redundant in that their conclusions can be derived in other ways. Some inferences are irrelevant in that they do not lead to derivations of the desired result.

c. Suggestions for improvement:

So to improve the resolution algorithm, we must pay atttention to the ways to reduce the number of useless inferences. The following are useful ways to do that:

- *Tautology Elimination* : A tautology is a clause containing a pair of complementary literals. For example, $A \lor B \lor \neg B$ is equivalent to which is $A \lor True$ which is equivalent to *True*. Deducing that *True* is true is not very helpful. So we can ignore these tautological clause during the process of the algorithm. Due to the requirements of the project, this method has been applied to the project's algorithm.

- *Pure Literal Elimination* : A pure literal is a literal that never occurs negated in clause set. A clause with a pure literal is useless because the literal cannot be resolved anymore. Then these clauses with pure literals can be deleted.

For example:

```
A OR -B
B OR D         (clause with pure literal)
C OR D         (clause with pure literal)
-A OR -C
C
B
```

- *Keep track of the clauses that was used for resolution*: In each loop, the algorithm will choose the clause from the beginning, including the unused clauses and the clauses that have been used to create the new clause. This wastes time and also creates unnecessary clauses. Therefore, keeping track of used clauses helps to optimize the algorithm.

6. References:

- Stuart J. Russell and Peter Norvig (2010) - Chapter 7 : Logical Agents, *Artificial Intelligence, A Modern Approach, Third Edition*
- Michael Genesereth and Eric J. Kao (2016) – Chapter 6 : Resolution Proofs, *Introduction to Logic, Third Edition* - Stanford University
- Michael Genesereth and Eric J. Kao (2016) – Chapter 14 : Resolution Proofs, *Introduction to Logic, Third Edition* - Stanford University
- Java T Point - Resolution in FOL