CSCI561 – Introduction to Artificial Intelligence
Instructor: Dr. K. Narayanaswamy
Assignment 4 – Propositional Logic Inference Algorithm
Forward chaining and Resolution
Due: 5/2/2014 11:59:59pm

1. Introduction

In this project you are to write a program to implement various inference algorithms for propositional logic (PL) . The aim of logical inference is to decide whether KB $\models \alpha$ for some sentence α. You will implement 3 specific inference algorithms

1) PL Forward chaining.

2) PL Backward chaining.

3) PL Resolution.

The KB is given in the implication form of Horn clauses. Thus, it is necessary to convert KB into CNF form for task 3.

2. Tasks

In this assignment, you will write a program to implement the following algorithms

1. PL Forward chaining.

2. PL Backward chaining.

3. PL Resolution.

3. Input

There are three inputs for your programs;

3.1 Which task to perform: there are three possible values

1 ➔ PL Forward Chaining Algorithm

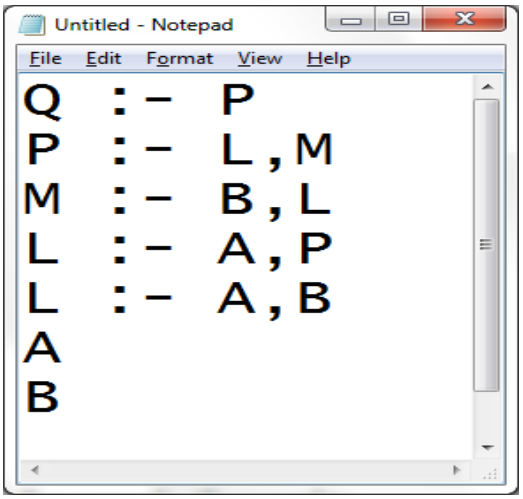
2 ➔ PL Backward Chaining Algorithm

3 ➔ PL Resolution algorithm

3.2 The KB file:

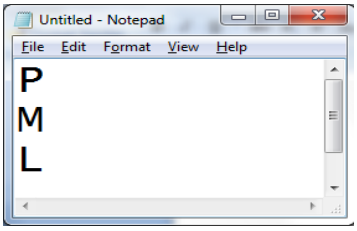
The KB file contains one or more Horn clauses (one per line) written using Prolog syntax. Each Horn clause may consist of literals, logical symbols. Literals are a letter from 'A' to 'Z' (upper or lower case) and the allowed logical symbols are only implication and conjunction operator. The implication operator is represented by " **:-** " without quotation mark. head **:-** body means body => head. The conjunction operation is represented by "**,**" without quotation mark. The figure below shows an example of KB file.

$$P \Rightarrow Q$$
$$L \wedge M \Rightarrow P$$
$$B \wedge L \Rightarrow M$$
$$A \wedge P \Rightarrow L$$
$$A \wedge B \Rightarrow L$$
$$A$$
$$B$$



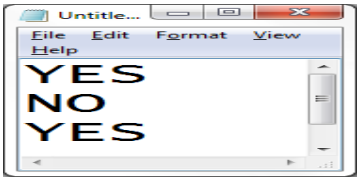
3.3 The query file:

The query file contain the literals whose entailment you need to determine. For example, if we need to test whether P, M and L are entailed by the KB or not, the query file is shown below.



4. Output

4.1 Entailment result: Show whether the literals are entailed by the KB or not.:



4.2 Log file: See output1_log_t1.txt, output2_log_t1.txt, output1_log_t2.txt, output2_log_t2.txt, output1_log_t3.txt, output2_log_t3.txt for more details.

4.2.1 Log file for task1

Show known/deduced facts, rule which are fired and newly entailed facts in order. There are 3 columns in this file:

1. Known/deduced facts

2. Rule which are fired

3. Newly entailed facts

Each column is separated by "#" (without quotation mark). The example is shown below.

<Known/Deduced Facts> # Rules Fires # Newly Entailed Facts

A,B # L :- A,B # L

A,B,L # M :- B,L # M

A,B,L,M # P :- L,M # P

4.2.1 Log file for task2:

Show queue of goals, relevant rule/fact and new goal introduced in order. There are 3 columns in this file:

1. Queue of goals

2. Relevant rule/fact

3. New goal introduced

Each column is separated by "#" (without quotation mark). The example is shown below.

Queue of Goals # Relevant Rule/Fact # New Goal Introduced

Q # Q :- P # P

P # P :- L,M # L,M

M # M :- B,L # L,B

B # B # L

4.2.3 Log file for task3:

Show resolving clause 1, resolving clause 2and added clause in order. There are 3 columns in this file:

1. Resolving clause 1

2. Resolving clause 2

3. Added clause

Each column is separated by "#". Disjunction is represented by "OR" and Negation is represented by "-" (without quotation mark). The example is shown below.

Resolving clause 1 # Resolving clause 2 # Added clause

-P OR Q # -L OR -M OR P # Q OR -L OR -M

-L OR -M OR P # -B OR -L OR M # -L OR P OR -B OR -L

-B OR -L OR M # -A OR -P OR L # -B OR M OR -A OR -P

5 Program Specifications

5.1 Your program must be written in either Java or C++.

5.2 Your program MUST compile and run on aludra.usc.edu

5.3 Write your own code. Files will be compared and any cheating will be reported.

5.4 Your program name must be "pl" (without quotation mark).

6. Execution Details

Your program will be tested on aludra.usc.edu on unseen input files that meet the input file specifications. Your program will receive 5 arguments, 3 for inputs and 2 for outputs.

6.1 C/C++

The grader will execute this command:

pl -t <task> -kb <kb_input_file> -q <query_input_file> -oe <output_entail > -ol < output_log>

Example:

pl -t 1 -kb kb1.txt -q q1.txt -oe output1.txt -ol log1.txt

6.2 JAVA

The grader will execute this command:

pl -t <task> -kb <kb_input_file> -q <query_input_file> -oe <output_entail > -ol < output_log>

Example:

pl -t 1 -kb kb1.txt -q q1.txt -oe output1.txt -ol log1.txt

6.3 Arguments:

6.3.1 <task> : there are 3 possible values 1, 2 and 3.

6.3.2 < kb_input_file >: location of an kb file.

6.3.4 < query_input_file >: location of an query file.

6.3.5 <output_entail >: location of an entailment output.

6.3.6 <output_log>: location of an log output.

Thus, you should interpret the example:

pl -t 1 -kb kb1.txt -q q1.txt -oe output1.txt -ol log1.txt

The first task is chosen. The location of the kb input file is same as your program and its name is kb1.txt. The location of the query input file is same as your program and its name is q1.txt.

The location of entailment output file is same as your program and its name is output1.txt. Finally, the location of log output file is same as your program and its name is log.txt.

7.1 Programming (90 pts):

Your program must implement all the three inference algorithms.

7.1.1 Correct outputs for task 1: 25 points

7.1.2 Correct outputs for task 2: 25 points

7.1.3 Correct outputs for task 3: 30 points

7.1.4 Your analysis of similarities/differences result between task1, task2 and task3. Your explanation must be included as part of readme.txt: 10 points

7.2 Readme.txt (10 pts):

7.2.1 A brief description of the program structure and any other issues you think will be helpful for the grader to understand your program. (5 pts)

7.2.2 Instructions on how to compile and execute your code. (5 pts)

7.2.3 Please include your name, student ID and email address on the top.

7.2.4 You must submit a program in order to get any credit for the Readme.txt. In short, if you submit ONLY a Readme.txt file you will get 0.

7.2.5 Remember to also include the explanation of outputs (Part 7.1.4)

8. Submission Guidelines

Your program files will all be submitted via blackboard. You MUST follow the guidelines below. Failure to do so will incur a -25 point penalty.

8.1 Compress and zip ALL your homework files (this includes the Readme.txt and all source files) into one .zip file. Note that only .zip file extensions are allowed. Other compression extensions such as .tar, rar, or 7z will NOT be accepted.

8.2 Name your zip file as follows: firstname_lastname.zip. For example, John_Smith.zip would be a correct file name.

8.3 To submit your assignment, simply select the appropriate assignment link from the Assignments subsection of the course blackboard website. Upload your zip file and click submit (clicking send is not enough).

Please make sure ALL source files are included in your zip file when submitted. Errors in submission will be assessed –25 points. A program that does not compile as submitted will be given 0 points. Only your FINAL submission will be graded. **NO LATE SUBMISSIONS WILL BE ALLOWED UNDER ANY CIRCUMSTANCES.**