CSCI 561: Foundations of Artificial Intelligence
Instructor: Prof. Laurent Itti

Homework #3: Logic Inference
Due on Nov 24, 2014 at 11:59pm

Introduction

To check the spread of the infectious disease Tebola, the National Center for Epidemic Control (NCEC) is preparing a large taskforce in identifying early symptoms of the disease. After a series of meetings with the experts, the NCEC has concluded that a practical way to set forth would be to train the taskforce in using a medical knowledge base for early identification of symptoms and infected people. This would allow expert knowledge to be widely available through the knowledge base and minimize human error.

You have been hired by the NCEC to facilitate the training of the taskforce in using the knowledge base. Each trainee is presented with disease related information in form of first order logic clauses. The trainee can read the clauses in the knowledge base and provide a logical conclusion that is provable from the clauses. You are required to develop a program that can check the conclusion made by the trainee and provide a quick feedback about whether the conclusion is right or not. The trainee can thus be trained to make conclusions using the knowledge base.

Problem

You are given a knowledge base and a query sentence, and you need to determine if the query can be inferred from the information given in the knowledge base. You are required to use **backward-chaining** (AIMA 3rd edn. Figure 9.6) to solve this problem.

Input:

You will be given the knowledge base and the query in a text file called **input.txt**.
The first line of the input file contains the query. The second line contains an integer *n* specifying the number of clauses in the knowledge base. The remaining lines contain the clauses in the knowledge base, **one per line**. Each clause is written in one of the following forms:

1) as an *implication* of the form $p_1 \wedge p_2 \wedge \ldots \wedge p_n \Rightarrow q$, whose premise is a conjunction of atomic sentences and whose conclusion is a **single** atomic sentence.
2) as a *fact* with a **single** atomic sentence: q

Each atomic sentence is a predicate applied to a certain number of arguments. Note that negation is not used in this homework.

Output:

If the query sentence can be inferred from the knowledge base, your output should be TRUE, otherwise, FALSE. Your answer (TRUE/FALSE) should be made available in the file **output.txt**.

Sample:

The input file will be formatted as below:

```
Diagnosis(John,Infected)
6
HasSymptom(x,Diarrhea)=>LostWeight(x)
LostWeight(x)&Diagnosis(x,LikelyInfected)=>Diagnosis(x,Infected)
HasTraveled(x,Tiberia)&HasFever(x)=>Diagnosis(x,LikelyInfected)
HasTraveled(John,Tiberia)
HasFever(John)
HasSymptom(John,Diarrhea)
```

Note:

1. & denotes the AND operator.
2. => denotes the implication operator.
3. No other operators besides & and => are used.
4. Variables are denoted by a single lower case letter (For this homework, you can assume that only variable x will be used. No other variables are used.)
5. All predicates (such as HasFever) and constants (such as John) begin with uppercase letters.
6. You can assume that all predicates have at least one and at most two arguments.
7. You can assume that there will be no more than 10 clauses in the knowledge base.

Since the query can be inferred from the given clauses in the sample input file, the output file will mention TRUE.

Submission

1. The deadline for submission is **Nov 24, at 11:59pm** Los Angeles time.
2. Please turn in all your code as a .zip file via the USC DEN dropbox, with the title format [firstname]_[lastname]_HW3.zip (e.g., Tommy_Trojan_HW3.zip).
3. Please include a Makefile with your submission. The Makefile must contain an “agent” part to compile (or do nothing if no compilation is needed) and a “run” part to execute your program.
4. **Late submission**: you lose 20% of the homework’s grade per 24-hour period that you are late. Beware, the penalty grows very fast: grade = points * (1 – n * 0.2) where n is the

number of days late (n=0 if submitted on time, n=1 is submitted between 1 second and 24h late, etc).

Grading

1. While you are free to implement your code in any language of your choice, we require that your code be able to run from the command line on the USC aludra.usc.edu Linux server. For information on accessing aludra, please see https://itservices.usc.edu/web/hosting/students
2. If the grader is unable to compile or execute your code successfully on aludra, you will not receive any credit.
3. You are allowed to use standard libraries only, such as C++ STL. You have to implement any other function or method by yourself.
4. Please make sure you use the compiler version available on aludra (Java 1.6/ gcc 4.2.1).