NANYANG TECHNOLOGICAL UNIVERSITY
SINGAPORE

CZ3005 Artificial Intelligence
Lab Report 2

Marathe Ajinkya Avinash
U1522716K
TSP1

Table of Contents

Sympathetic Doctor

Objective:

Assume that the prolog script is a sympathetic doctor, conversing with a patient who can answer only yes or no. The doctor should be able to diagnose the patient's condition while asking question sensitively depending upon patient's pain level and mood level. You can choose 5 or more different mood considerations (calm, angry, etc.) and 5 or more levels of pain. Five or more diseases should be diagnosable, each disease characterized by 5 or more symptoms.

Overview:

The assignment has been completed with Prolog programming. The program does not have a GUI but the script interacts with the user by printing the questions on the screen in SWI-Prolog. After loading the prolog file in the memory, the diagnosis of the doctor can be initiated by typing *main/0* in the prolog console. The doctor starts by asking the level of pain that the patient is suffering from. If the patient says *no,* then the doctor asks the same question with the next level of pain. If the patient says *yes*, then the program moves on to the next symptoms such as mood, fever, bowel movements and miscellaneous in that order. After all the questions have been asked, the doctor matches the symptoms of the patient with the symptoms of the diseases from the knowledge base and prints the disease that the patient is suffering from. If no match is found then it prints that further testing is needed to ascertain the cause.

Rationale:

1. Create lists of different levels of symptoms like pain levels, mood types, extent of fever etc.
2. Create rules for different diseases which state what symptoms should be present for that disease.
3. Create predicates to read the pain levels, mood levels by traversing through the lists of symptoms by recursion.
4. Accept user input in the atom form and assert new facts when the patient affirms a particular symptom.
5. Diagnose the disease by unification of the asserted facts in the disease predicates.

Explanation of the program:

Main Rule:

The main rule is used to start the program. It consists of all the main objectives of the program. It starts with the predicate initialize () that retracts all the facts that have been asserted for the previous patient. It contains all the compound terms read Pain (), read Mood () etc.to find out the symptoms levels by asking questions. Finally, it contains the diagnose () compound term used to diagnose and print out the disease.

```
main:-initialize(),
      readPain(),
      readMood(),
      readFever(),
      readBowel(),
      readMiscellaneous(),
      diagnose().
```

Symptom Lists:

The symptom lists are used to state all the different levels of the five types of symptoms namely pain, mood, fever, bowel movements and miscellaneous. The list as the argument for the predicate pain library forms a fact in the knowledge base of the program.

```
% Symptom lists
pain_library([unbearable,strong,mild,manageable,no]).
mood_library([calm,angry,weepy,depressed,stressed]).
fever_library([very_high,high,mild,low,no]).
bowel_movements_library([hard,loose,tarry,bloody,normal]).
miscellaneous_library([itchy,giddy,hallucinating,twitches,palpitations]
```

Disease Rules:

The rules listed below are the rules for diagnosing the disease of the patient. For example, for the patient to be diagnosed with cancer, he should have experienced strong pain, depression, mild fever, bloody stools and giddiness. When we assert the symptom levels of the patients for different patients, then each of these rules is checked and the rule that evaluates to true is the diagnosed disease.

```
% Diseases rules
cancer():-pain(strong),mood(depressed),fever(mild),bowel(bloody),
          miscellaneous(giddy).
amoebiasis():-pain(unbearable),mood(weepy),fever(high),bowel(loose),
              miscellaneous(itchy).
rabies():-pain(strong),mood(angry),fever(high),bowel(normal),
          miscellaneous(hallucinating).
heart_trouble():-pain(unbearable),mood(stressed),fever(no),bowel(normal),
                 miscellaneous(palpitations).
dehydration():-pain(no),mood(calm),fever(no),bowel(hard),
               miscellaneous(giddy).
```

Rules to pass the symptom lists:

These predicates with arity 0 help in passing the lists from the libraries to the functions that ask the questions. In the code snippet, the variable X in shared by pain_library and read Pain/1, which implies that all the X belonging to pain library will be passed on to read Pain/1. By default, prolog always works in the universal quantifier mode. Here X means all X.

```
readPain():-pain_library(X),readPain(X).
readMood():-mood_library(X),readMood(X).
readFever():-fever_library(X),readFever(X).
readBowel():-bowel_movements_library(X),readBowel(X).
readMiscellaneous():-miscellaneous_library(X),readMiscellaneous(X).
```

Rules to ask questions:

```
readPain(List):-
                    [H|T]=List,
                    format("Do you experience ~w pain",[H]),
                    read(X),
                    ((test(X);list_empty(T))->assert(pain(H));
                    readPain(T)
                    ).
```

Let us look one by one at the clauses in the rule to ask questions to the patient.
Firstly, the read Pain () gets a list as a parameter. The list is the pain library.
Then it splits the list into head and tail. The head is the first element of the list while the tail is the rest of the list. The next line uses the format/1 to print the question along with the value from the list. The read(X) predicate from the next line reads the yes or no from the user and stores it in variable X.

```
((test(X);list_empty(T))->assert(pain(H));
readPain(T)
).
```

The line given above checks some conditions-

1. Test(X) decides whether the answer given by the patient was *yes* or *no*
2. The list empty predicate decides whether the tail of the list is empty
3. So, if the answer is yes or the list has no more answers left, then we assert the symptom level as the fact
4. Note that if the patient answers no for all the questions, then the last item of the list is taken as the default option.
5. If the answer is no but the list is not empty, then the tail of the list is passed on to the predicate again called inside itself (recursion)

   Here are the rules for the rest of the symptoms-

```
readMood(List):-
                  [H|T]=List,
                  format("Do you feel ~w",[H]),
                  read(X),
                  ((test(X);list_empty(T))->assert(mood(H));
                  readMood(T)
                  ).


readFever(List):-
                  [H|T]=List,
                  format("Do you have ~w fever",[H]),
                  read(X),
                  ((test(X);list_empty(T))->assert(fever(H));
                  readFever(T)
                  ).


readBowel(List):-
                  [H|T]=List,
                  format("Do you experience ~w stools",[H]),
                  read(X),
                  ((test(X);list_empty(T))->assert(bowel(H));
                  readBowel(T)
                  ).

readMiscellaneous(List):-
                  [H|T]=List,
                  format("Do you experience ~w ",[H]),
                  read(X),
                  ((test(X);list_empty(T))->assert(miscellaneous(H));
                  readMiscellaneous(T)
                  ).
```

Diagnosis:

The diagnosis rule contains all the diseases clauses to check which disease the patient is suffering. The appropriate prognosis is printed when the disease is ascertained.

```
% checks meets conditions for which disease and prints the disease
diagnose():-nl,
            cancer()->write("You may have cancer. I recommend undergoing medical tests");
            amoebiasis()->write("You have amoebiasis. I will prescribe you the antibiotics");
            rabies()->write("You have rabies. We have to admit you to the hospital");
            heart_trouble()->write("You may have heart trouble. I recommend a stress test immediately");
            dehydration()->write("Nothing to worry. Drink a lot of water with electrolytes");
            write("We need further testing to ascertain the cause").
```

Supporting predicates:

Initialize:

Retracts all the asserted facts about the last patient so that the diagnosis for the next patient can be started without re consulting the prolog script. retractall/0 in prolog is used to retract the facts. The pain, mood, fever symptoms facts are retracted, the underscore in the clauses implies that the variables used are insignificant and are not going to be used. This is done to avoid the singleton variables.

```
initialize():- retractall(pain(_)),
               retractall(mood(_)),
               retractall(fever(_)),
               retractall(bowel(_)),
               retractall(miscellaneous(_)).
```

list_empty:

The two list_empty () rules were stated to find out whether the list is empty or not. If the list is empty the it unifies with the first one to return true. If the list is not empty, then it unifies with the other one returning false. These functions are used to find out whether a list is empty.

```
list_empty([]):-true().
list_empty([_|_]):-false().
```

test(X):

The test rule is to check whether the patient replied a yes or no. If he replied yes, then the predicate test evaluated to true.

```
((test(X);list_empty(T))->assert(pain(H));
readPain(T)
).
```

Appendix A:

Prolog Script:

```
% SYMPATHETIC DOCTOR
/*
----------main rule----------
start the program
*/

main:-initialize(),
    readPain(),
    readMood(),
    readFever(),
    readBowel(),
    readMiscellaneous(),
    diagnose().


% function to test yes or no
test(X):-
  X=='yes'.

% retracting all the facts asserted for the previous patient so do not have to reconsult the
script
initialize():- retractall(pain(_)),
            retractall(mood(_)),
            retractall(fever(_)),
            retractall(bowel(_)),
            retractall(miscellaneous(_)).

% unifies with the appropriate predicate depending on whether the list is empty.
list_empty([]):-true().
list_empty([_|_]):-false().


% Symptom lists
pain_library([unbearable,strong,mild,manageable,no]).
mood_library([stressed,angry,weepy,depressed,calm]).
fever_library([very_high,high,mild,low,no]).
bowel_movements_library([hard,loose,tarry,bloody,normal]).
miscellaneous_library([itchy,giddy,hallucinating,twitches,palpitations,no_special_symptoms
]).

% Record the symptoms of the patient by asserting the appropriate facts
pain().
mood().
fever().
```

```
bowel().
miscellaneous().



% Diseases rules
% for example, a patient has cancer if pain is strong, depressed mood, mild fever, bloody stools and giddiness
cancer():-pain(strong),mood(depressed),fever(mild),bowel(bloody),
        miscellaneous(giddy).
amoebiasis():-pain(unbearable),mood(weepy),fever(high),bowel(loose),
           miscellaneous(itchy).
rabies():-pain(strong),mood(angry),fever(high),bowel(normal),
        miscellaneous(hallucinating).
heart_trouble():-pain(unbearable),mood(stressed),fever(no),bowel(normal),
             miscellaneous(palpitations).
dehydration():-pain(no),mood(calm),fever(no),bowel(hard),
           miscellaneous(giddy).



% checks meets conditions for which disease and prints the disease
diagnose():-nl,
         cancer()->write("You may have cancer. I recommend undergoing medical tests"); % if have cancer then print
         amoebiasis()->write("You have amoebiasis. I will prescribe you the antibiotics");
         rabies()->write("You have rabies. We have to admit you to the hospital");
         heart_trouble()->write("You may have heart trouble. I recommend a stress test immediately");
         dehydration()->write("Nothing to worry. Drink a lot of water with electrolytes");
         write("We need further testing to ascertain the cause").



/*
predicates  readPain/0, readMood/0, etc  to pass the appropriate lists to readPain/1, readMood/1 etc.
*/
readPain():-pain_library(X),readPain(X). % pass all X that is in pain_library to readPain
readMood():-mood_library(X),readMood(X).
readFever():-fever_library(X),readFever(X).
readBowel():-bowel_movements_library(X),readBowel(X).
readMiscellaneous():-miscellaneous_library(X),readMiscellaneous(X).

/*
predicates to readPain/1, moodLevel/1... to read pain,mood  recursively.
it splits the list into [H|L] i.e head and Tail
if the user says yes to head symptom, then true,
or else it calls the same predicate recursively passing the tail as parameter
```

```
*/
readPain(List):-
        [H|T]=List, % Split the list into head and tail
        format("Do you experience ~w pain",[H]), % print the questions along with the
head
        read(X), % read user input
        ((test(X);list_empty(T))->assert(pain(H)); % assert fact if last item in the list or
answer is yes
        readPain(T) % if answer is no and there are more items, then recursion
        ).


readMood(List):-
        [H|T]=List,
        format("Do you feel ~w",[H]),
        read(X),
        ((test(X);list_empty(T))->assert(mood(H));
        readMood(T)
        ).


readFever(List):-
        [H|T]=List,
        format("Do you have ~w fever",[H]),
        read(X),
        ((test(X);list_empty(T))->assert(fever(H));
        readFever(T)
        ).


readBowel(List):-
        [H|T]=List,
        format("Do you experience ~w stools",[H]),
        read(X),
        ((test(X);list_empty(T))->assert(bowel(H));
        readBowel(T)
        ).

readMiscellaneous(List):-
        [H|T]=List,
        format("Do you experience ~w ",[H]),
        read(X),
        ((test(X);list_empty(T))->assert(miscellaneous(H));
        readMiscellaneous(T)
        ).
```

Appendix B:
Sample Run: Cancer Diagnosis

```
Welcome to SWI-Prolog (threaded, 64 bits, version 7.6.0)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

For online help and background, visit http://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

?- ['Users/Ajinkya/Desktop/doctor.pl'].
true.

?- main.
Do you experience unbearable painno.
Do you experience strong pain|: yes.
Do you feel calm|: no.
Do you feel angry|: no.
Do you feel weepy|: no.
Do you feel depressed|: yes.
Do you have very_high fever|: no.
Do you have high fever|: no.
Do you have mild fever|: yes.
Do you experience hard stools|: no.
Do you experience loose stools|: no.
Do you experience tarry stools|: no.
Do you experience bloody stools|: yes.
Do you experience itchy |: no.
Do you experience giddy |: yes.

You may have cancer. I recommend undergoing medical tests
true.

?-
```