Prolog assignments

Overview

Talking box

In perspective

Note: Talking Box main prolog script is given at the end of this file for your reference

Lists (basic_interactive1.pl)

- romantic([gifts,wine,dinner,candlelight,rains,tea,concert,night,poetry,music,movie,dating,magic,novels,stories,roses,bouquets,courtship,chickflicks,cruise,breeze,diaries,painting]).
- outdoorsy([picnic,**trekking**,soccer,sports,jogging,kayaking,parks,event,woods,mountains,beaches,cricket,action_movies,dogs,lakes,fairs,swimming,breeze,fitness,water]).
- social([coffee,picnic,friends,party,beer,music,concert,event,movie,soccer,dinner,gifts,gardens, roses, flowers,bouquet,cricket,board_games,cafe,netflix,dogs,networking,exhibitions,fairs,debates]).
- health_freak([sports,tea,fruitjuice,smoothie,trekking,training,jogging,soccer,sweating,sleeping,swimming,lakes,exercise,burpees,fitness,cats,water, music]).
- loner([books,coffee,woods,candlelight,sleeping,training,jogging,night,tea, beaches,poetry,tea,rains,fiction, novels,board_games,netflix,cats,music,painting,sketching,writing,diaries]).

Rules

- ask(X,Y):- like(Y), related(X,Y).
- ask(X,Y):- random(X).

If the user likes Y, ask user about X that is related to Y
Otherwise, ask user about random X

- ask (trekking, 0) → Sequence of queries starts by giving ask(0)

Query goes in this sequence

- related(X,Y):- romantic(L),member(X,L),member(Y,L).
- related(X,Y):- outdoorsy(L),member(X,L),member(Y,L).
- related(X,Y):- social(L),member(X,L),member(Y,L).
- related(X,Y):- health_freak(L),member(X,L),member(Y,L).
- related(X,Y):- loner(L),member(X,L),member(Y,L).

Load the list romantic in L. X is related to Y if both X and Y are members of L

Rules (contd.)

- random(X):- romantic(L),member(X,L).
- random(X):- outdoorsy(L),member(X,L).
- random(X):- social(L),member(X,L).
- random(X):- health_freak(L),member(X,L).
- random(X):- loner(L),member(X,L).

Load romantic in L.
X is a member of L

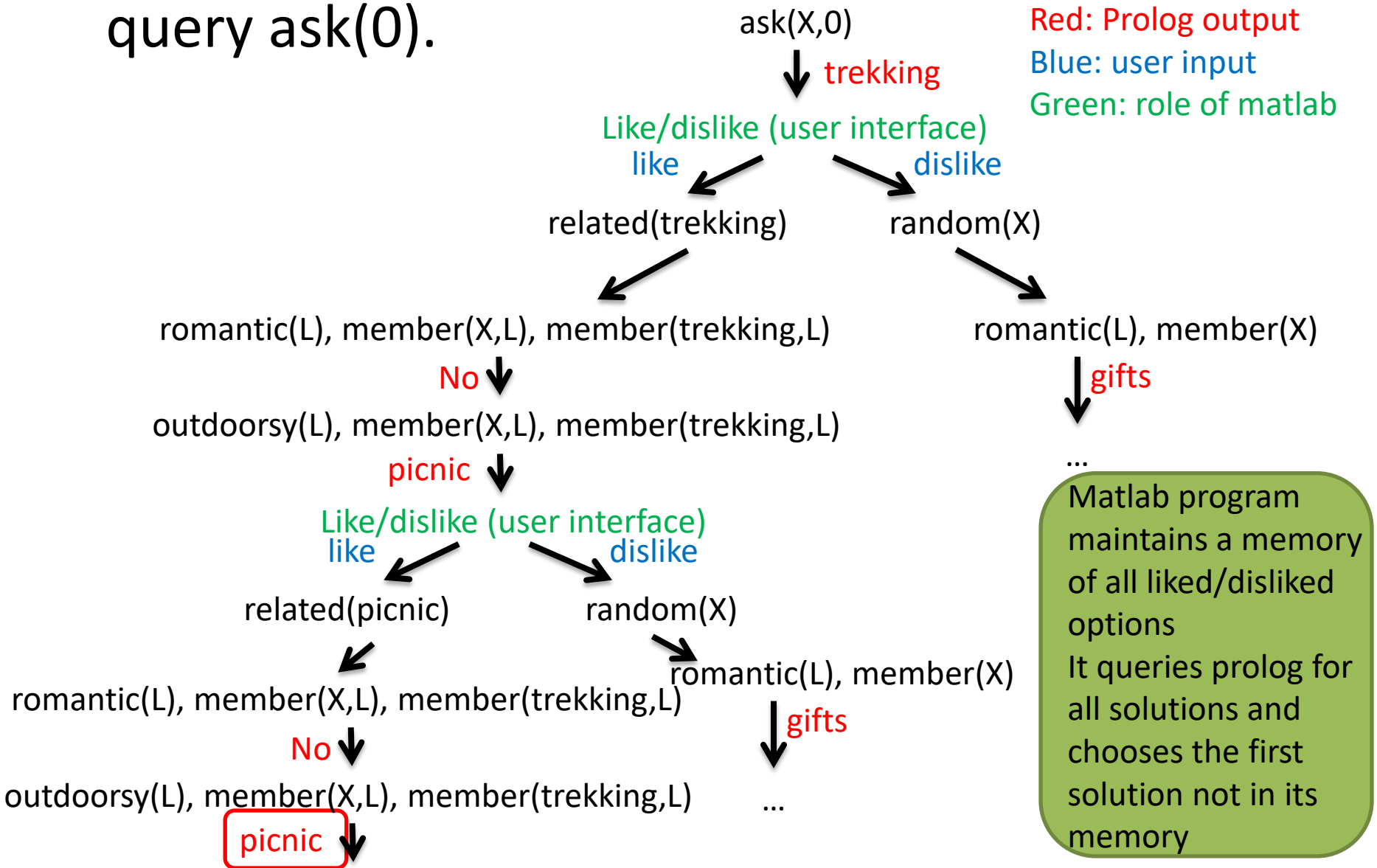
- member(X,[X|_]).
- member(X,[_|R]) :- member(X,R).

- like(nothing)
- dislike(nothing)

Definition for useless facts allows for the prolog to know that there can be queries related to like

Role of matlab??

- For this, consider what happens when we query ask(0).



Hints for the assignment

- Crude hints have been given for the four assignment
- Each student is required to complete only one assignment
- !!! Assignment 4 is the most challenging

Assignment 1

10 questions

Lists and counters

- tennis([court, singles, doubles, outdoor, ball, racket, <others>]).
- diving([pool, singles, jump, <others>]).

- <others games>

- counter (a counter variable, initialized as zero)

Google ‘prolog’ counters in scripts

Queries and rules

- <chosen game>(L). → The list related to the chosen game is now available in L
- selected(<chosen game>) → Assigns the selected game as a fact.

You can use assert in command prompt or you can code at the end of the prolog script.

Ex. tennis(L). selected(tennis)

- all_options(X) → Flatten all the lists to a super list containing unique options like [singles, doubles, pool, <others>]. Google ‘prolog in-built flatten’

User should be able to see what are the options he/she can ask. You can find commands to display all the options . Google ‘prolog in-built write’

Queries and rules

- has(X) ⟶ If counter is more than 10, write ‘Guess the game’
  Otherwise
  Write ‘Yes’ if X is in the list L. Also, increments counter by 1.
  Otherwise
  Write ‘No’, if it is a valid option and not in L. Also, increments counter by 1
  Otherwise
  Write ‘Invalid options’

- is(<guessed game>) ⟶ Writes ‘successful guess’ if guessed game is the selected game)

Assignment 2

Kid’s day at school

Lists

- play([slides, sandbox, toys, trains, cars, playmat, build, bears, soft_toys, alphabets, numbers,<others>]).
- eat([cake, toffee, candy, sandwich, pizza, cheerios, veggies, fries,<others>]).
- do([build, veggies, trains, <others>, draw]).
- see([cake, trains, alphabets, draw]).
- <others>

Rules and queries

- Analogous to Talking Box

Assignment 3

Subway sandwich interactor

Lists

- breads([parmesan, honeywheat,<others>]).
- main([chicken, tuna, <others>]).
- veggies([lettuce, tomato, <others>]).
- sauce([mustard, chipotle, <others>]).
- sides([soup, soda, <others>])
- <others>

Rules and queries

- options(<list name>). Displays the options for the list name. Ex. options(bread) should display all bread options

- selected(<selected option>,<selected list>)

  Ex. selected(parmesan,bread). assigns parmesan as the selected bread

  If selected list can have multiple options, for examples veggies and sauces, then write('do you want to choose more')

  If selected(0), then go to next list.

- done(X)

  If X=1, then display all selected options

Assignment 4

Patient with a sympathetic doctor

Remember: patient says only yes or no

Lists

- pain_library([unbearable_pain, lot_of_pain, manageable_pain, mild_pain, no_pain,<others>]).
- mood_library([calm, angry, weepy, stressed, <others>]).

- polite_gesture([look_concerned,mellow_voice,light_touch, faint_smile,<others>]).
- calming_gesture([greet, look_composed, look attentive,<others>]).
- normal_gesture([broad_smile, joke, beaming_voice,<others>]).
- <other response behaviours>

- fever([temperature, sweat, ache, weepy, <others>]
- cold([sneeze, cough, temperature,<others>])
- injury([blood, lot_of_pain, weepy, angry, <others>])
- <other diagnostic possibility>

Ideas for what prolog script should do

Identify pain level → Similarly, identify mood level → Define rules to select the appropriate gesture list and store it in L

Choose one member from pain library at a time and display it. write 'yes/no'. Take input from user. <Google prolog script user input >

If the patient (example your tutor) says yes, assign pain(<displayed pain>) as a fact and exit.

If the patient says no, choose next member.

OR

Assume you(doctor) assessed the pain and mood level and just specify:
assert(pain(<pain option>)).
assert(mood(<mood option>)).

Initialize counter for each diagnosis

Perform actions similar to Talking box, but with two important changes.

(1) Every time ask(X,Y) is called, select a random member from list L and write it to the screen

(2) Every time, the patient says yes (like in Talking box), increment the counter of the lists which have the queried option.

prolog script, select a random member from a list

Appendix: Talking Box prolog script

```
ask(trekking,0).

ask(X,Y):-
                like(Y), related(X,Y).
ask(X,Y):-
                random(X).

/* member of a list*/
member(X,[X|_]).
member(X,[_|R]) :- member(X,R).
/* takeout a member from a list*/
takeout(X,[X|R],R).
takeout(X,[F|R],[F|S]) :- takeout(X,R,S).
/* append a member to a list*/
append([A | B], C, [A | D]) :- append(B, C, D).
                append([], A, A).
related(X,Y):- romantic(L),member(X,L),member(Y,L).
/*,succ(Count_romantic,Count_romantic).*/
related(X,Y):- outdoorsy(L),member(X,L),member(Y,L).
/*,succ(Count_outdoorsy,Count_outdoorsy).*/
related(X,Y):- social(L),member(X,L),member(Y,L).
/*,succ(Count_social,Count_social).*/
related(X,Y):- health_freak(L),member(X,L),member(Y,L).
/*,succ(Count_health_freak,Count_health_freak).*/
related(X,Y):- loner(L),member(X,L),member(Y,L).
```

Script contd..

```
random(X):- romantic(L),member(X,L).
/*,succ(Count_romantic,Count_romantic).*/
random(X):- outdoorsy(L),member(X,L).
/*,succ(Count_outdoorsy,Count_outdoorsy).*/
random(X):- social(L),member(X,L).
/*,succ(Count_social,Count_social).*/
random(X):- health_freak(L),member(X,L).
/*,succ(Count_health_freak,Count_health_freak).*/
random(X):- loner(L),member(X,L).
/*,succ(Count_loner,Count_loner).*/

romantic([gifts,wine,dinner,candlelight,rains,tea,concert,night,poetry,music,movie,dating,magic,novels,stories,roses,bouquets,courtship,chick flicks,cruise,breeze,diaries,painting]).
outdoorsy([picnic,trekking,soccer,sports,jogging,kayaking,parks,event,woods,mountains,beaches,cricket,action_movies,dogs,lakes,fairs,swim ming,breeze,fitness,water]).
social([coffee,picnic,friends,party,beer,music,concert,event,movie,soccer,dinner,gifts,gardens, roses,
flowers,bouquet,cricket,board_games,cafe,netflix,dogs,networking,exhibitions,fairs,debates]).
health_freak([sports,tea,fruitjuice,smoothie,trekking,training,jogging,soccer,sweating,sleeping,swimming,lakes,exercise,burpees,fitness,cats, water,music]).
loner([books,coffee,woods,candlelight,sleeping,training,jogging,night,tea,beaches,poetry,tea,rains,fiction,
novels,board_games,netflix,cats,music,painting,sketching,writing,diaries]).

a.
like(nothing).
dislike(nothing).
a.
```

Sample run of the Talking Box script

```
?- ask(X,0).
X = trekking ,

?- like(trekking).
false.

?- assert(like(trekking)).
true.

?- like(trekking).
true.

?- ask(X,trekking).
X = picnic ,

?- assert(dislike(picnic)).
true.

?- ask(X,picnic).
X = gifts ,

?- assert(like(picnic)).
true.

?- ask(X,picnic).
X = picnic ;
X = trekking ;
X = soccer ,

?- assert(dislike(soccer)).
true.

?- ask(X,soccer).
X = gifts ;
X = wine ;
X = dinner ;
X = candlelight ,

?-
```