UNIT-IV: Declarative Programming Paradigm: Logic Programming

Faculty In-charge

Mrinmoyee Mukherjee
Assistant Professor (IT Dept.)
email: mrinmoyeemukherjee@sfit.ac.in
Mob: 9324378409

Academic Year: 2023-24

Looping a fixed number of times.....

- Most programming languages provide loops that allow a set of instructions to be executed repeatedly either a fixed number of times or until a given condition is met

- Prolog has no looping facilities.

- The same effects can be obtained (that enable a sequence of events to be evaluated repeatedly) through backtracking, recursion, built in predicates or a combination of both.

loops.pl [modified]

File Edit Browse Compile Prolog Pce Help

loops.pl [modified]

```
loop(0).
loop(N):-N>0,
    write('The value is: '),
    write(N),
    nl,
    M is N-1,
    loop(M).
```

Will display the o/p fixed number of times

SWI-Prolog (AMD64, Multi-threaded, versi

File Edit Settings Run Debug Help

```
?- loop(9).
The value is: 9
The value is: 8
The value is: 7
The value is: 6
The value is: 5
The value is: 4
The value is: 3
The value is: 2
The value is: 1
true
```

Applying recursion.....

sum.pl

File Edit Browse Compile Prolog Pce Help

loops.pl sum.pl

```
/*sum the integers from 1 to N*/
 sumto(1,1).
sumto(N,S):-N>1,
    N1 is N-1,
    sumto(N1,S1),
    S is S1+N.
```

?- sumto(5,N).
N = 15

?- sumto(1,3).
false.

?- sumto(1,1).
true

Ex5 Recursion: Loop till condition satisfied

- Recursion Example5
- **Knowledge base**

```
go :- loop(start).
loop(end).
loop(A) :- A\=end, write('The value is:'), read(Word),
        write('Input value is: '), write(Word), nl, loop(Word).
```

- **Queries**

```
?- go.
The value is:
Input value is a
The value is:
Input value is: hi
The value is:
Input value is end
yes
```

Conjunction and Backtracking

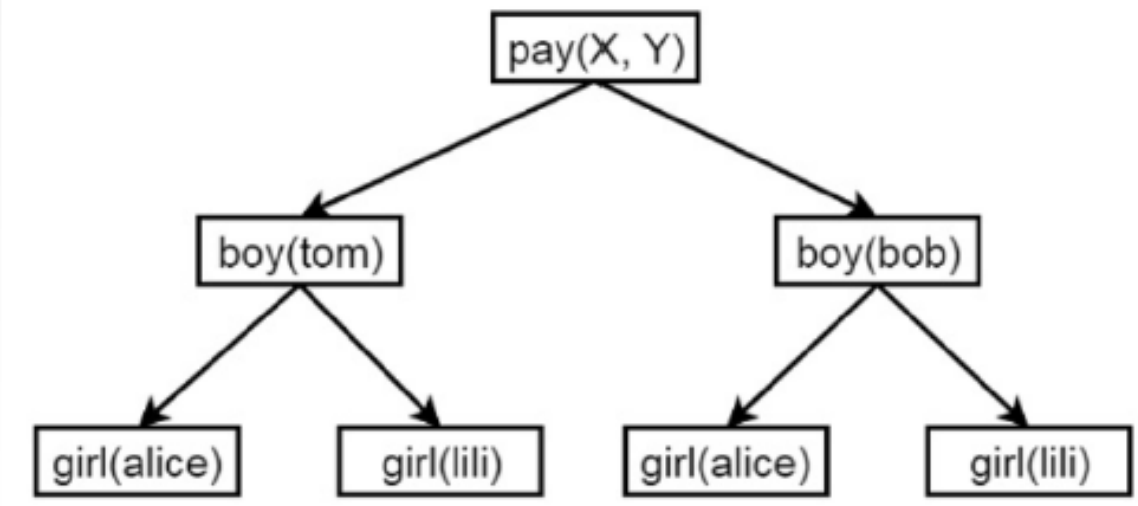
- Conjunction:
  - Conjuction means 'and' it is indicated by use of comma ',' . Ex:

  ?- likes(john, mary), likes(mary, john).

  - Here we have given two subgoals in one query using conjunction.
  - For answering the query as success Prolog has to match and satisfy each subgoals with the knowledge base (i.e. Facts and Rules).
  - When goals are given as conjunction, Prolog finds the match and satisfy goals in conjunction, in left-to-right manner.
- Backtracking:
  - Prolog repeatedly tries to find match and satisfy the goals by looking to the knowledge base in top-down manner which is nothing but backtracking.
  - To control Backtracking , Cut operator ' ! ' is used
- For above mentioned query, prolog will first match and satisfy the left most goal i.e. likes(john, mary). and then the second goal.
- If any of subgoals does not satisfy, then query will be answered as failure.

Backtracking Ex

- Consider two people X and Y can pay each other, but the condition is that a boy can pay to a girl
- **Knowledge Base:**
- boy(tom).
- boy(bob).
- girl(alice).
- girl(lili).
- pay(X,Y) :- boy(X), girl(Y).



Cut in Backtracking

- Sometimes we write the same predicates more than once when our program demands, in such cases uncontrolled backtracking may prove inefficient.
- To resolve this, we will use the Cut in Prolog.
- Ex: Consider we have three mutually exclusive rules and any given time only one of them ill be true.
- **Knowledge Base:**

```
f (X,0) :- X < 3.          % Rule 1
f (X,2) :- 3 =< X, X < 6.  % Rule 2
f (X,4) :- 6 =< X.         % Rule 3
```

```
f (X,0) :- X < 3, !.          % Rule 1
f (X,2) :- 3 =< X, X < 6, !.  % Rule 2
f (X,4) :- 6 =< X.            % Rule 3
```

- **Queries**:

?- f(1,Y), 2<Y.

- There are two subgoals to be satisfied. As per first fact X=1 so Y will be set to 0, but now second clause or query fails since Y is not >2.
- So Prolog backtracks and goes rule 2, here since X is not between 3 and 6, first query clause itself fails. Prolog backtracks and checks third rule, the goal fails here too. To avoid such backtracking cut is used.

Example-2 of Cut

- **Ex: Knowledge Base:**
- animal(dog).
- animal(cat).
- animal(elephant).
- animal(tiger).
- animal(cobra).
- animal(python).

- snake(cobra).
- snake(python).
- likes(mary, X) :- snake(X).
- likes(mary, X) :- animal(X).

- **Queries:**

?- likes(mary, X).

Use semicolon to see different X values

- **Ex: Knowledge Base: (Using cut)**
- animal(dog).
- animal(cat).
- animal(elephant).
- animal(tiger).
- animal(cobra).
- animal(python).

- snake(cobra).
- snake(python).
- likes(mary, X) :- snake(X), !.
- likes(mary, X) :- animal(X).

- **Queries:**

?- likes(mary, X).

Use semicolon to see different X values

PROLOG facilities and deficiencies

- Advantages :
  1. Easy to build database. Doesn't need a lot of programming effort.
  2. Pattern matching is easy. Search is recursion based.
  3. It has built in list handling. Makes it easier to play with any algorithm involving lists.
- Disadvantages :
  1. LISP (another logic programming language) dominates over prolog with respect to I/O features.
  2. Sometimes input and output is not easy.
- Applications :
  1. Prolog is highly used in artificial intelligence(AI).
  2. Prolog is also used for pattern matching over natural language parse trees.

Deficiencies of Prolog

Resolution Order Control:

- Prolog always matches in the same order – user can control the ordering
- Prolog allows explicit control of backtracking using cut which is actually, a goal and not an operator. It always succeeds but can not be resatisfied through backtracking.

The Closed World Assumption:

- In Prolog truths are those that can be proved using its database
- If there is insufficient information in database, to prove a query, it is not actually false, it fails.
- It relates to negation problem.

END