Artificial Intelligence

Lecture 20

- **Prolog Programming for AI**

Prepared by:

Md. Mijanur Rahman, Prof. Dr.

Dept. of CSE, Jatiya Kabi Kazi Nazrul Islam University

Email: mijanjkkniu@gmail.com

Prolog Programming for AI

- **Outlines:**
  - Operators and Expressions
  - Looping
  - Manu-base Program
  - …

Operators

- **Arithmetic Operators:**

SYMBOL	OPERATION
+	addition
-	subtraction
*	multiplication
/	real division
//	integer division
mod	modulus
**	power

- **Comparison Operators:**

Expressions	Meanings
X>Y	
X<Y	
X>=Y	
X=<Y	
X =:= Y	Equal
X =\= Y	Not Equal

Operators

- **Logical Operators:**

SYMBOL	OPERATION
not	negation
\+	not provable
,	logical conjunction
;	logical disjunction
:-	logical implication
->	if-then-else

- The **logical operators** are used in the definition of rules.

a :- b. % a if b

a :- b,c. % a if b and c.

a :- b;c. % a if b or c.

a :- \++ b. % a if b is not provable

a :- not b. % a if b fails

a :- b -> c;d. % a if (if b then c else d)

Expressions

- Arithmetic expressions are evaluated with the built in predicate '**is**' which is used as an infix operator in the following form.:

  variable **is** *expression*

- For example,

  ?- X is 3*4.

  X = 12

Operator Notation

- Consider the mathematical expression, like

  2*a + b*c

- Where + and * are operators, and a, b, c, 2 are arguments. + and * are infix operators. Such expression can be represented as trees, and can be written as Prolog terms with + and * as functors:

  +(*(2, a), *(b, c))

- A programmer can define his or her own operators. For example, we can define the atoms '**has**' and '**supports**' as infix operators:
  - Peter has apples.
  - Floor supports table.
- These facts are equivalent to:
  - Has(peter, apples).
  - Supports(floor, table).

Type Predicates

PREDICATE	CHECKS IF
var(V)	V is a variable
nonvar(NV)	NV is not a variable
atom(A)	A is an atom
integer(I)	I is an integer
real(R)	R is a floating point number
number(N)	N is an integer or real
atomic(A)	A is an atom or a number
functor(T,F,A)	T is a term with functor F and arity A
T =..L	T is a term, L is a list
clause(H,T)	H :- T is a rule in the program

LOOPING in PROLOG

- **Looping until a Condition is Satisfied:**
- **Example-15:** Use of recursion to read terms entered by the user from the keyboard and output them to the screen, until a word **end** is encountered, using 'disjunctive goal'(**word=end**).

```
test :- write('Type the word : '), read(word),
write('Input was: '), write(word), nl, (word=end; test).
```

- `?- test.`

  Type the word : Hello.
  Input was: Hello
  Type the word : ITS.
  Input was: ITS
  Type the word : end.
  Input was: end.
  yes

LOOPING in PROLOG

- **Example-16:** Looping a fixed number of times.

```
testloop(0).
testloop(N):- N>0, write('Number : '), write(N),
nl, M is N-1, testloop(M).
```

- The `testloop` predicate is defined as **'loop from N, write the value of N, then subtract one to give to M, then loop from M'**. **'And by the first clause, is defined as 'if the argument is zero, do nothing (stop!)'**.

- Test of the program :
- `?- testloop(3).`

Number : 3
Number :2
Number :1
yes

LOOPING in PROLOG

- **Example-17:** Reading the value N and printing the numbers starting from 1 to N.

```
start:- write('N= ?'), read(N), loop(0,N).
loop(N0, N):- N1 is N0+1, N1=< 10, write(N1), loop(N1, N).
?- start.
    N=?
    5
    1 2 3 4 5
```

- **Example-18:** Reading the value N and printing the numbers starting from N to 1.

```
start:- write('N= ?'), read(N), loop(N).
loop(N):- write(N), N1 is N-1, N1>0, loop(N1).
?- start.
    N=?
    5
    5 4 3 2 1
```

Infinite Loop

- **Danger of Infinite Loop:**

- Consider the following clause:

  P :- P.

- This says that P is true if P is true. This is declaratively perfectly correct, but procedurally is quite useless. In fact, such a clause cause problems to Prolog.

- Consider the query: **?- P.**
- Here, the goal P is replaced by the same goal P; this will be in turn replaced by P, etc. In such a case Prolog will enter an infinite loop, not noticing that no process is being made.

Infinite Loop

- What happens if we ask:

  X :- f(X).

- Should this request for matching succeed or fail? According to the definition of unification in logic this should fails, but what should happens according to our definition of matching? This will answer:

  X = f(f(f(f(f(f(f(f(f(f(f(f(…

Menu-base Program

- **Example-19:** Manu-base Prolog program.

```
start:- nl,
    write('**MENU**'),nl,
    write('1. Personal Info'),nl,
    write('2. Family Info'),nl,
    write('3. Academic Info'),nl,
    write('0. Exit'),nl,nl,
    write('Choice = ? '),read(X),
    (   X = 1,
        write('Personal Info:'),nl,
        write('Name: '),read(Name),
        write('Mobile: '),read(Mob),
        write('Email: '),read(Em),
        write('Name: '),write(Name),nl,
        write('Mobile: '),write(Mob),nl,
        write('Email: '),write(Em),nl,
        start;
        X = 2,
```

```
write('Family Info:'),nl,
write('Father name: '),read(Fnam),
write('Mother name: '),read(Mnam),
write('Home Address: '),read(Home),
write('Father name: '),write(Fnam),nl,
write('Mother name: '),write(Mnam),nl,
write('Home Address: '),write(Home),nl,
start;
X = 3,
write('Academic Info:'),nl,
write('Dept.: '),read(Dept),
write('Roll No.: '),read(Roll),
write('CGPA: '),read(Cgpa),
write('Dept.: '),write(Dept),nl,
write('Roll No.: '),write(Roll),nl,
write('CGPA: '),write(Cgpa),nl,
start;
X = 0,
write('Thanks')).
```

Prolog Programming for AI

TO BE CONTINUED…