CS370

Symbolic Programming

Declarative Programming

LECTURE 2: Introduction to Prolog

Jong C. Park
park@cs.kaist.ac.kr
Computer Science Department
Korea Advanced Institute of Science and Technology
http://nlp.kaist.ac.kr/~cs370

Overview

- Primer

- Introduction to Prolog

Primer

⦿ Symbolic Programming

- cf. numeric computation
- There are well-known examples of symbolic computation whose implementation in other standard languages took tens of pages of indigestible code.
- When the same algorithms were implemented in Prolog, the result was a crystal-clear program easily fitting on one page.

Primer

Declarative Programming

- cf. procedural programming
- Many believe that every student of computer science should learn something about Prolog at some point because Prolog enforces a different problem-solving paradigm complementary to other programming languages.

Primer

⦿ Logic Programming

- cf. functional programming
- Prolog stands for *programming in logic*, emerging from the idea of using logic as a programming language.
- But Prolog is a general programming language and any algorithm can be programmed in it.

Primer

Prerequisites

- none
- No particular programming experience is required.
- In fact, plentiful experience and devotion to procedural programming - for example in C or Pascal - might even be an impediment to the fresh way of thinking Prolog requires.

Introduction to Prolog

- Defining relations by facts
- Defining relations by rules
- Recursive rules
- How Prolog answers questions
- Declarative and procedural meaning of programs

Defining relations by facts

Example Sentence

- Tom is a parent of Bob.

Example Representation

- parent(tom,bob).

Are there any other ways?

-
-
-
-

What are the pros and cons?

-

Defining relations by facts

⦿ Questions

- Who is Tom a parent of?

```
?- parent(tom,X).
X = bob
yes
```

- Who else is Tom a parent of?

```
parent(tom,bob).
parent(tom,liz).
?- parent(tom,X).
X = bob;
X = liz;
no
```

Defining relations by facts

⦿ Question

- Who is a grandparent of Jim?

```
parent(tom,bob).
parent(tom,liz).
parent(bob,jim).

?- parent(X,Y), parent(Y,jim).
X = tom, Y = bob
yes
```

- Any other possibilities?

Defining relations by rules

⦿Encoding gender information

- female(liz).
- Are there any other ways?
- What are the pros and cons?

Defining relations by rules

⦿ Other relations

- the offspring relation
  - Method 1: `offspring(liz,tom).`
  - Method 2: `offspring(Y,X) :- parent(X,Y).`
- the mother relation
  - `mother(X,Y) :- parent(X,Y), female(X).`
- the grandparent relation

  -
    ```
    grandparent(X,Z) :- parent(X,Y),
                        parent(Y,Z).
    ```
- Are there any other ways?

Defining relations by rules

Defining the sister relation

- sister(X,Y) :- parent(Z,X), parent(Z,Y), female(X).
- Any problems?
- One possible solution

Recursive rules

- What is recursion?
  -
- Why do we need recursion?
  -
- Example: the **predecessor** relation

```
predecessor(X,Z) :- parent(X,Z).
predecessor(X,Z) :- parent(X,Y),
                    predecessor(Y,Z).
```

How Prolog answers questions

Terminologies

- predicate, argument, clause, procedure
- fact, rule, head and body, goal, question

Sample Interaction

- Axioms

```
fallible(X) :- man(X).        % All men are fallible.
man(socrates).                % Socrates is a man.
```

- Is this a theorem?

```
?- fallible(socrates).        % Is Socrates fallible?
yes
```

How Prolog answers questions

⦿ Another sample interaction

- ?- predecessor(tom,pat).

```
parent(pam,bob). parent(tom,bob). parent(tom,liz).
parent(bob,ann).  parent(bob,pat).  parent(pat,jim).
female(pam).      male(tom).        male(bob).
female(liz).      female(ann).      female(pat).
male(jim).
offspring(Y,X) :- parent(X,Y).
mother(X,Y) :- … .      sister(X,Y) :- … .
grandparent(X,Z) :- parent(X,Y), parent(Y,Z).
predecessor(X,Z) :- parent(X,Z).
predecessor(X,Z) :- parent(X,Y), predecessor(Y,Z).
```

Meaning of programs

Declarative meaning

- concerned with the relations defined by the program
- determines what will be the output of the program

Procedural meaning

- determines how this output is obtained (or, how the relations are actually evaluated by the Prolog system)

Summary

- Prolog programming consists of defining relations and querying about relations.
- A program consists of clauses. These are of three types: facts, rules and questions.
- A relation can be specified by facts, or by stating rules about the relation.
- A procedure is a set of clauses about the same relation.