CS2136: Paradigms of Computation

Class 03: Logic Programming
Prolog

Copyright 2001, 2002, 2003 Michael J. Ciaraldi and David Finkel
Logic Programming
Logic Programming: Why?

- Another way to use computers.
- Useful for mathematical proofs.
- Useful in AI.
- This could be done in almost any language, but Prolog is designed for it.
Things to Remember

z Logic deals with domains of discourse.
  y Mathematics
  y Geography
  y Company organization

z Logic can only deal with a model of the real world.
What You Have &
What You Can Do

z You have statements which you assert are true.

z You have rules which govern how to derive conclusions from statements.

z You try to prove other statements either true or false.
Proving Something True

“Proving something true” in logic really just means you can derive that statement from what you already claim you know.

It does not guarantee it is true in the real world.

Some of the statements you based it on may be false.

Some of the rules may be incorrect.

Some things are true but cannot be proven.

- Russell & Whitehead Principia Mathematica
- Godel’s Incompleteness Theorem
“Proving something false” in logic may really mean that you cannot derive that statement from what you already claim you know.

It does not guarantee it is false in the real world.

Must really distinguish between:

- Really proving something false.
- Not being able to prove that something is true.
“Closed World Assumption”

- If you cannot prove something is true, it is considered false.
- “Negation as Failure”
First-Order Predicate Calculus

- Deals with logical statements
  - i.e. statements which can be either true or false.
- Deals with objects and their relationships.
- Deals with rules for these relationships.
FOPC Example (in English)

Socrates is a human.
All humans are mortal.
Therefore, Socrates is mortal.

Note: This was proven true in the real world.
More on FOPC later.
Prolog
Prolog

Concerned with objects and their relationships.

Not “objects” in the sense of “object-oriented”.

Examples (in English, not Prolog):

- John owns the book.
- John likes Mary.
Programming Prolog

z Consists of:

y Declaring facts about objects and their relationships.

y Defining rules about objects and their relationships.

y Asking questions about objects and their relationships.
Representing Facts

- Meaning:
  - John likes Mary.

- In Prolog
  - likes (john, mary).

- Looks like Lisp!

- In this example:
  - The first argument is the liker.
  - The second argument is the likee (??).
Representing Facts II

z likes (john, mary).

z Note:

y Relationship name (a.k.a. “predicate”) is lower-case and comes first.

y Object names are lower-case.

y Period (a.k.a. “full stop”) at the end.

y #arguments > 2 is OK.

y Order is arbitrary, but...

  x You must decide.

  x You must be consistent.
Representing Facts III

Remember:

- Prolog knows absolutely nothing about the real world.
- The meaning is entirely arbitrary.
- These would work just as well:
  - likes (john, mary).
  - likes (dirt, anger).
  - afgh (xyyxx, yhww).
More Examples of Facts

- female(jane).
- play(john, mary, football).
- owns(john, gold).

Is “gold”

- A particular object?
- A chemical elements?
- An attribute of objects?

All the facts in a Prolog program form a “database”, stored in the order entered.
Objects can be more than single words with no upper-case letters; use single quotation marks.

loves(john, ‘WPI’).

loves(john, ‘Worcester Polytechnic Institute’).

If you enter an identical fact twice, Prolog saves both copies in the database.

What do you do with these facts?
The CCC Unix machines have SWI-Prolog loaded.

You can download it for:
- Windows
  - window and command-line versions
- Unix
- Linux

Many other Prolog implementations are available.
Running Prolog II

- Click on a menu, click an icon, or type:
  - `pl`

- Get the prompt:
  - `?-`

- Load a program using:
  - `consult(filename).`
    - Don’t forget the period!
    - Suffix assumed to be “.pl”.
      - Can conflict with Perl.
    - Example: `consult(‘likes.pl’).`
Running Prolog III

z The Clocksin & Mellish book assumes you can just enter everything from the keyboard.

z SWI-Prolog is a compiler, so:
  y Programs must ordinarily be read from a file.
  y You can ask questions from the keyboard.

z Prolog has no built-in graphics, but can be integrated with graphics packages.
Listing the Database

?- consult('likes.pl'). [Me]
% likes.pl compiled 0.11 sec, 40 bytes
Yes
?- listing(likes). [Me]
likes(john,mary).
likes(john,gold).
likes(mary,john).
likes(john,football).
likes(john,mary).
Some Commands

\[ \text{z listing.} \]
\[ \text{y Lists everything.} \]
\[ \text{z listing}(\text{predicate}). \]
\[ \text{y Lists just facts and rules with that predicate.} \]
\[ \text{z halt.} \]
\[ \text{y Exits Prolog.} \]

Arity = number of arguments.
Asking Questions

- No need to type in the “?-” that shows in the book.
- Can ask specific or more general questions.
- Specific:
  - \( y \) likes(john, mary).
Questions & Answers

- Prolog answers yes or no.
- “Yes” just means it can prove what you asked is true.
  - i.e. it found a match in the database.
- “No” means “can’t prove it by me!”
  - i.e. no match.
  - This does not necessarily mean “false”.

...
Example

File ‘likes.pl’:
likes(john,mary).
likes(john,gold).
likes(mary,john).
likes(john,football).
likes(john,mary).

Questions & Answers:
?- likes(john,mary).
Yes
?- likes(john,jane).
No
Variables

- They can stand for objects.
- A variable is
  - “Instantiated” if it currently stands for an object.
  - “Not instantiated” if it doesn’t.
- Start with an upper-case letter.
Questions with Variables

z Prolog looks for a match.

z likes(john,X).

y Logically means: “Is there anything John likes?”

y Really means: “Is there an X such that this is true?”
Next Time

- More Prolog:
  - Variables
  - Goals
  - Backtracking