Concepts of Programming Languages

Lecture 16 - Logic Programming

Patrick Donnelly

Montana State University

Spring 2014

Administrivia

Assignments:

Programming #3 : due 04.14

Homework #3 : due 04.16

Reading:

Chapter 16

Q: How many legs does a dog have if you call its tail a leg?
A: Four. Calling a tail a leg doesn't make it one.

Abraham Lincoln

Introduction

Logic programming language or declarative programming language

Express programs in a form of symbolic logic

Use a logical inferencing process to produce results

Declarative rather that procedural:

Definition

Declarative languages only specify the results desired rather than a detailed procedures for producing them.

Proposition

Definition

A logical statement that may or may not be true.

- Consists of objects and relationships of objects to each other

Symbolic Logic

Definition

Logic which can be used for the basic needs of formal logic:

- Express propositions
- Express relationships between propositions
- Describe how new propositions can be inferred from other propositions

Particular form of symbolic logic used for logic programming called *predicate calculus*.

Object Representation

Objects in propositions are represented by simple terms: either constants or variables

Constant: a symbol that represents an object

Variable: a symbol that can represent different objects at different times

- Different from variables in imperative languages

Compound Terms

Atomic propositions consist of compound terms

Compound term: one element of a mathematical relation, written like a mathematical function

- Mathematical function is a mapping
- Can be written as a table

Compound Terms

Compound term composed of two parts

- Functor: function symbol that names the relationship
- Ordered list of parameters (tuple)

Example

```
student(jon)
like(seth, OSX)
like(nick, windows)
like(jim, linux)
```

Forms of a Proposition

Propositions can be stated in two forms:

- Fact: proposition is assumed to be true
- Query: truth of proposition is to be determined

Compound proposition:

- Have two or more atomic propositions
- Propositions are connected by operators

Logical Operators and Quantifiers

Name	Symbol	Example	Meaning
negation	$\neg$	$\neg$ a	not a
conjunction	$\cap$	a $\cap$ b	a and b
disjunction	$\cup$	a $\cup$ b	a or b
equivalence	$\equiv$	$a \equiv b$	a is equivalent to b
implication	$\supset$ $\subset$	a $\supset$ b a $\subset$ b	a implies b b implies a
universal	$\forall$	$\forall$ X.P	For all X, P is true
existential	$\exists$	$\exists$ X.P	There exists a value of X such that P is true

Clausal Form

Too many ways to state the same thing

Use a standard form for propositions

Clausal form example:

- $B_1 \cup B_2 \cup \ldots \cup B_n \subset A_1 \cap A_2 \cap \ldots \cap A_m$
- means if all the A's are true, then at least one B is true

Antecedent: right side

Consequent: left side

Horn Clauses

Definition

A ***Horn clause*** has a head h, which is a predicate, and a body, which is a list of predicates $p_1, p_2, \ldots, p_n$.

It is written as: $h \leftarrow p_1, p_2, \ldots, p_n$

This means, "h is true only if $p_1, p_2, \ldots,$ and p_n are simultaneously true."

Horn Clauses

Definition

A ***Horn clause*** has a head h, which is a predicate, and a body, which is a list of predicates $p_1, p_2, \ldots, p_n$.

It is written as: $h \leftarrow p_1, p_2, \ldots, p_n$

This means, "h is true only if $p_1, p_2, \ldots,$ and p_n are simultaneously true."

Example

The Horn clause:

$snowing(C) \rightarrow precipitation(C), freezing(C)$

says, "it is snowing in city C only if there is precipitation in city C and it is freezing in city C."

Horn Clauses and Predicates

Any Horn clause

$$h \leftarrow p_1, p_2, \ldots, p_n$$

can be written as a predicate:

$$p_1 \cap p2 \cap \ldots \cap p_n \supset h$$

or equivalently:

$$\neg(p_1 \cap p_2 \cap \ldots \cap p_n) \cup h$$

But not every predicate can be written as a Horn clause.

E.g., $literate(x) \supset reads(x) \cup writes(x)$

Theorem Proving

A use of propositions is to discover new theorems that can be inferred from known axioms and theorems.

Basis for logic programming

When propositions used for resolution, only restricted form can be used.

Resolution

Definition

Resolution is an inference principle that allows inferred propositions to be computed from given propositions

If h is the head of a Horn clause

$$\text{h} \leftarrow \text{terms}$$

and it matches one of the terms of another Horn clause:

$$t \leftarrow t_1, h, t_2$$

then that term can be replaced by h's terms to form:

$$t \leftarrow t_1, \mathit{terms}, t_2$$

Resolution

Definition

Unification is a pattern-matching process that determines what particular instantiations can be made to variables during a series of resolutions.

During resolution, assignment of variables to values is called instantiation.

Definition

Instantiation is the assigning temporary values to variables to allow unification to succeed

Definition

After instantiating a variable with a value, if matching fails, may need to ***backtrack*** and instantiate with a different value

Resolution Example

The two clauses:

$speaks(Mary, English)$
$talkswith(X, Y) \leftarrow speaks(X, L), speaks(Y, L), X \neq Y$

can resolve to:

$$talkswith(Mary, Y) \leftarrow speaks(Mary, English), speaks(Y, English), Mary \neq Y$$

The assignment of values *Mary* and *English* to the variables X and L is an instantiation for which this resolution can be made.

Proof by Contradiction

Definition

Hypotheses: a set of pertinent propositions

Definition

Goal: negation of theorem stated as a proposition

Theorem is proved by finding an inconsistency

Overview of Logic Programming

Declarative semantics

- There is a simple way to determine the meaning of each statement
- Simpler than the semantics of imperative languages

Programming is nonprocedural

- Programs do not state now a result is to be computed, but rather the form of the result

Example: Sorting a List

Describe the characteristics of a sorted list, not the process of rearranging a list

$$sort(old\_list, new\_list) \subset permute(old\_list, new\_list) \cap sorted(new\_list)$$

$$sorted(list) \subset \forall j \mid 1 \leq j < n, list(j) \leq list(j+1)$$

Logic Programming

Logic programming has applications in AI and databases:

- Natural language processing (NLP)
- Automated reasoning and theorem proving
- Expert systems (e.g., MYCIN)
- Database searching, as in SQL (Structured Query Language)

Prolog emerged in the 1970s.

Distinguishing features:

- Nondeterminism
- Backtracking