Logic Programming

Georg Moser

Institute of Computer Science @ UIBK

Summer 2015

Summary of Last Lecture

Definition

- goals (aka formulas) are constants or compound terms
- goals are typically non-ground

Definitions (Clause)

- a clause or rule is a universally quantified logical formula of the form
$$A \leftarrow B_1, B_2, \ldots, B_n \ .$$
where A and the B_i's are goals
- A is called the head of the clause; the B_i's are called the body
- a rule of the form $A \leftarrow$ is called a fact; we write facts simply A.

Definition

a logic program is a finite set of clauses

Example (cont'd)

Tower of Hanoi in Prolog

```
% hanoi(N,X,Y,Z) <-- a tower of N disks is moved from
%                    peg X to peg Y using peg Z as storage
hanoi(0,_,_,_).
hanoi(N,X,Y,Z) :-
    N > 0, M is N-1,
    hanoi(M,X,Z,Y),
    move(N,X,Y),
    hanoi(M,Z,Y,X).

move(D,X,Y) :-
    write('move disk '), write(D),
    write(' from '), write(X),
    write(' to '), write(Y), nl.

?- hanoi(4,a,c,b).
```


Outline of the Lecture

Logic Programs

introduction, basic constructs, database and recursive programming, theory of logic programs

The Prolog Language

programming in pure prolog, arithmetic, structure inspection, meta-logical predicates, cuts, extra-logical predicates, how to program efficiently

Advanced Prolog Programming Techniques

nondeterministic programming, incomplete data structures, definite clause grammars, meta-programming, constraint logic programming

Some Examples

Example (Multiplication)

logic program

```
plus(0,X,X).
plus(s(X),Y,s(Z)) ← plus(X,Y,Z).
times(0,X,0).
times(s(X),Y,Z) ← times(X,Y,U), plus(U,Y,Z).
```

goal

`plus(s(s(0)),s(0),s(s(s(0))))`	$X \mapsto s(0),\ Y \mapsto s(0),\ Z \mapsto s(s(0))$
`plus(s(0),s(0),s(s(0)))`	$X \mapsto 0,\ Y \mapsto s(0),\ Z \mapsto s(0)$
`plus(0,s(0),s(0))`	$X \mapsto s(0)$

solved

Renaming of Rules is Needed

Example

logic program

```
plus(0,X3,X3).
plus(s(X2),Y2,s(Z2)) ← plus(X2,Y2,Z2).
times(0,X,0).
times(s(X),Y,Z) ← times(X,Y,U), plus(U,Y,Z).
```

goal

`plus(s(s(0)),s(0),X)`	$X_1 \mapsto s(0),\ Y_1 \mapsto s(0),\ X \mapsto s(Z_1)$
`plus(s(0),s(0),Z1)`	$X_2 \mapsto 0,\ Y_2 \mapsto s(0),\ Z_1 \mapsto s(Z_2)$
`plus(0,s(0),Z2)`	$X_3 \mapsto s(0),\ Z_2 \mapsto s(0)$

solution $X \mapsto s(s(s(0)))$

Example

logic program

```
plus(0,X3,X3).
plus(s(X),Y,s(Z)) ← plus(X,Y,Z).
times(0,X2,0).
times(s(X1),Y1,Z1) ← times(X1,Y1,U1), plus(U1,Y1,Z1).
```

goal

`times(X,X,Y)`	$X \mapsto s(X_1),\ Y_1 \mapsto s(X_1),\ Z_1 \mapsto Y$
`times(X1,s(X1),U1), plus(U1,s(X1),Y)`	$X_1 \mapsto 0,\ X_2 \mapsto s(0),\ U_1 \mapsto 0$
`plus(0,s(0),Y)`	$X_3 \mapsto s(0),\ Y \mapsto s(0)$

solution $X \mapsto s(0),\ Y \mapsto s(0)$

Example

logic program

```
plus(0,X2,X2).
plus(s(X),Y,s(Z)) ← plus(X,Y,Z).
times(0,X3,0).
times(s(X),Y,Z) ← times(X,Y,U), plus(U,Y,Z).
```

goal

`times(X,X,Y)`	$X \mapsto s(X_1)$
`times(X1,s(X1),U1), plus(U1,s(X1),Y)`	$U_1 \mapsto 0,\ X_2 \mapsto s(X_1),\ Y \mapsto s(X_1)$
`times(X1,s(X1),0)`	$X_1 \mapsto 0,\ X_3 \mapsto s(0)$

solution $X \mapsto s(0),\ Y \mapsto s(0)$

Three Choices

1. goal in sequence of goals
2. rule in logic program
3. substitution

Definition

composition of substitutions

$$\theta = \{X_1 \mapsto t_1, \dots, X_n \mapsto t_n\}$$

and

$$\sigma = \{Y_1 \mapsto s_1, \dots, Y_k \mapsto s_k\}$$

is substitution

$$\theta\sigma = \{X_1 \mapsto t_1\sigma, \dots, X_n \mapsto t_n\sigma\} \cup \{Y_i \mapsto s_i \mid Y_i \notin \{X_1, \dots, X_n\}\}$$

Example

$$\theta = \{X \mapsto g(Y,Z), Y \mapsto a\} \qquad \theta\sigma = \{X \mapsto g(Y, f(X)), Y \mapsto a, Z \mapsto f(X)\}$$

$$\sigma = \{X \mapsto f(Y), Z \mapsto f(X)\} \qquad \sigma\theta = \{X \mapsto f(a), Z \mapsto f(g(Y,Z)), Y \mapsto a\}$$

Definition

- substitution θ is at least as general as substitution σ if $\exists \mu\ \theta\mu = \sigma$
- unifier of set S of terms is substitution θ such that $\forall s, t \in S\ s\theta = t\theta$
- most general unifier (mgu) is at least as general as any other unifier

Example

terms $f(X, g(Y), X)$ and $f(Z, g(U), h(U))$ are unifiable:

$\{X \mapsto h(a), Y \mapsto a, Z \mapsto h(a), U \mapsto a\}$

$\{X \mapsto h(U), Y \mapsto U, Z \mapsto h(U)\}$ mgu

$\{X \mapsto h(g(U)), Y \mapsto g(U), Z \mapsto h(g(U)), U \mapsto g(U)\}$

Theorem

- *unifiable terms have mgu*
- $\exists$ *algorithm to compute mgu*

Definition

- sequence $E = u_1 \stackrel{?}{=} v_1, \dots, u_n \stackrel{?}{=} v_n$ is called an equality problem
- if $E = X_1 \stackrel{?}{=} v_1, \dots, X_n \stackrel{?}{=} v_n$, with X_i pairwise distinct and $X_i \notin \mathcal{V}\mathrm{ar}(v_j)$ for all i, j, then E is in solved form
- let $E = X_1 \stackrel{?}{=} v_1, \dots, X_n \stackrel{?}{=} v_n$ be a equality problem in solved form E induces substitution $\sigma_E = \{X_1 \mapsto v_1, \dots, X_n \mapsto v_n\}$

Unification Algorithm

$$u \stackrel{?}{=} u, E \Rightarrow E$$

$$f(s_1, \dots, s_n) \stackrel{?}{=} f(t_1, \dots, t_n), E \Rightarrow s_1 \stackrel{?}{=} t_1, \dots, s_n \stackrel{?}{=} t_n, E$$

$$f(s_1, \dots, s_n) \stackrel{?}{=} g(t_1, \dots, t_n), E \Rightarrow \bot \quad f \neq g$$

$$X \stackrel{?}{=} t, E \Rightarrow X \stackrel{?}{=} t, E\{X \mapsto t\} \quad X \in \mathcal{V}\mathrm{ar}(E), X \notin \mathcal{V}\mathrm{ar}(t)$$

$$X \stackrel{?}{=} t, E \Rightarrow \bot \quad X \neq t, X \in \mathcal{V}\mathrm{ar}(t)$$

$$t \stackrel{?}{=} X, E \Rightarrow X \stackrel{?}{=} t, E \quad t \notin \mathcal{V}$$

Theorem

1. *equality problems E is unifiable iff the unification algorithm stops with a solved form*
2. *if $E \Rightarrow^* E'$ such that E' is a solved form, then $\sigma_{E'}$ is mgu of E*

Example

$$f(X,g(Y),X) \stackrel{?}{=} f(Z,g(U),h(U)) \Rightarrow X \stackrel{?}{=} Z, g(Y) \stackrel{?}{=} g(U), X \stackrel{?}{=} h(U)$$

$$\Rightarrow X \stackrel{?}{=} Z, g(Y) \stackrel{?}{=} g(U), Z \stackrel{?}{=} h(U)$$

$$\Rightarrow X \stackrel{?}{=} Z, Y \stackrel{?}{=} U, Z \stackrel{?}{=} h(U)$$

$$\Rightarrow X \stackrel{?}{=} h(U), Y \stackrel{?}{=} U, Z \stackrel{?}{=} h(U) \quad \text{mgu}$$

Two Choices

1. goal in sequence of goals
2. rule in logic program

substitution – avoid choice by always taking mgu

Computation Model of Logic Programs

- the choice of goal is arbitrary
  if there is a successful computation for a specific order, then there is a successful computation for any other order
- the choice of rules is essential
  not every choice will lead to a successful computation; thus the computation model is nondeterministic

Exercise 1

Consider the following implementation that attempts to solve the tower of Hanoi puzzle. Is this program correct? Please explain your answer:

```
hanoi(0,_,_,_).
hanoi(N,X,Y,Z) :-
  N > 0, M is N-1,
  hanoi(M,X,Z,Y),
  move(N,X,Z),
  hanoi(M,Y,Z,X).

move(D,X,Y) :-
  write('move disk '), write(D),
  write(' from '), write(X),
  write(' to '), write(Y), nl.
```


Exercise 2

Consider lists with arbitrary entries and implement a binary predicate member(X, Xs) that checks whether X belongs to the list Xs.

Exercise 3

Consider lists with arbitrary entries and implement a ternary predicate append(Xs, Ys, Zs) that is true, if $Zs = Xs@Ys$.