Logic programming

Horn logic and resolution

Joey W. Coleman, Stefan Hallerstede

AARHUS UNIVERSITY
DEPARTMENT OF ENGINEERING

11 April 2014

Horn clauses

Refutation

Search trees

Horn clauses

Refutation

Search trees

Horn clauses

- Logic programs are expressed in terms of ***Horn clauses***:
  $$g^1(s^1_1, \ldots, s^1_{m_1}) \wedge \ldots \wedge g^k(s^k_1, \ldots, s^k_{m_k}) \rightarrow h(t_1, \ldots, t_n) \quad (1)$$
  and
  $$h(t_1, \ldots, t_n) \quad (2)$$
- These formulas are implicitly quantified over all free variables that occur in them.
  $$\forall X, Y \bullet g^1(s^1_1, \ldots, s^1_{m_1}) \wedge \ldots \wedge g^k(s^k_1, \ldots, s^k_{m_k}) \rightarrow h(t_1, \ldots, t_n) \quad (3)$$
  and
  $$\forall X \bullet h(t_1, \ldots, t_n)$$
  where X are the variables occurring free in (2) and Y all variables occurring free in (1) different from X
- Formula (3) can be rewritten using the laws of predicate logic:
  $$\forall X \bullet (\exists Y \bullet g^1(s^1_1, \ldots, s^1_{m_1}) \wedge \ldots \wedge g^k(s^k_1, \ldots, s^k_{m_k})) \rightarrow h(t_1, \ldots, t_n)$$
- Variables Y are therefore sometimes referred to as existential
- Formulas of the shape (1) are usually written:
  $$h(t_1, \ldots, t_n) \leftarrow g^1(s^1_1, \ldots, s^1_{m_1}) \wedge \ldots \wedge g^k(s^k_1, \ldots, s^k_{m_k})$$

Variants

- Because all clauses are implicitly quantified we can rename variables in a clause arbitrarily
- A clause with renamed variables is called a ***variant*** of that clause
- For example,
  $h(t_1, \ldots, t_n)\{Y = Z\}$ is a variant of $h(t_1, \ldots, t_n)$
- Variants permit to introduce fresh variables in each deduction step

Derivation

Let G be $\leftarrow A_1, \ldots, A_m, \ldots, A_k$ a query and C be $A \leftarrow B_1, \ldots, B_n$.
Then G' is ***derived*** from G and C using θ if the following hold:

- A_m is an atom,
- the substitution θ is a ***unifier*** of A_m and A,
- G' is the query $\leftarrow (A_1, \ldots, B_1, \ldots, B_n, \ldots, A_k)\theta$

(where θ is a unifier of atoms A and B if $A\theta = B\theta$.)

Remark.
If the list $B_1, \ldots, B_n$ is empty, the atom A_m is simply removed.
This may continue until we arrive at an empty clause denoted by $\Box$.

Remark.
In fact, the substitution θ must be the most general one, that is,
if σ also unifies A_m and A,
then there is a substitution τ such that $\sigma = \theta\tau$.

Horn clauses

Refutation

Search trees

Refutation

Let P be a program and G a query.
A ***derivation*** of $P \cup \{G\}$ consists of

- a (finite or infinite) sequence $G = G_0, G_1, G_2, \ldots$ of queries,
- a sequence $C_1, C_2, \ldots$ of variants of clauses of P and
- a sequence $\theta_1, \theta_2, \ldots$ of unifiers

such that G_{i+1} is derived from G_i and C_{i+1} using θ_{i+1}.

A ***refutation*** of $P \cup \{G\}$ is a finite derivation of $P \cup \{G\}$
which has the empty clause $\Box$ as the last query in the derivation.

Unification

Algorithm for computing the unifier of two terms u and v

Input: a pair of terms $u \approx v$
Output: a substitution θ such that $s\theta = t\theta$ (in solved form) or ***failure***
$\Theta := \{u \approx v\}$
repeat
  select an arbitrary $s \approx t$ from Θ;
  if $s = f(s_1, \ldots, s_n)$ ***then***
    if $t = X$ ***then*** replace $s \approx t$ by $t \approx s$
    elsif $t = f(t_1, \ldots, t_n)$ ***then*** replace $s \approx t$ by $s_1 \approx t_1, \ldots, s_n \approx t_n$
    else return failure
  elsif $s = X$ ***then***
    if $t = X$ ***then*** remove $s \approx t$
    elsif X occurs in t ***then return failure***
    else replace all other occurrences of X in Θ by t
until Θ remains unchanged for any $s \approx t$ from Θ;
$\theta := \{s = t \mid s \approx t \in \Theta\}$;
return θ

Unification example 1

Unify $f(X, g(Y))$ and $f(g(Z), Z)$

$$\{f(X, g(Y)) \approx f(g(Z), Z)\}$$

$$\rightarrow \{X \approx g(Z), g(Y) \approx Z\}$$

$$\rightarrow \{X \approx g(Z), Z \approx g(Y)\}$$

$$\rightarrow \{X \approx g(g(Y)), Z \approx g(Y)\}$$

$\rightarrow$ Substitution $\{X = g(g(Y)), Z = g(Y)\}$

Unification example 2

Unify $f(X, g(X), b)$ and $f(a, g(Z), Z)$

$$\{f(X, g(X), b) \approx f(a, g(Z), Z)\}$$

$$\rightarrow \{X \approx a, g(X) \approx g(Z), b \approx Z\}$$

$$\rightarrow \{X \approx a, a \approx Z, b \approx Z\}$$

$$\rightarrow \{X \approx a, Z \approx a, b \approx Z\}$$

$$\rightarrow \{X \approx a, Z \approx a, b \approx a\}$$

$\rightarrow$ ***failure***

Unification example 3

Unify $f(X, g(X))$ and $f(Z, Z)$

$$\{f(X, g(X)) \approx f(Z, Z)\}$$

$$\rightarrow \{X \approx Z, g(X) \approx Z\}$$

$$\rightarrow \{X \approx Z, g(Z) \approx Z\}$$

$$\rightarrow \{X \approx Z, Z \approx g(Z)\}$$

$\rightarrow$ ***failure***

Derivation example

Program:

$\mathit{grandfather}(X, Z) \leftarrow \mathit{father}(X, Y), \mathit{parent}(Y, Z).$
$\mathit{parent}(X, Y) \leftarrow \mathit{father}(X, Y).$
$\mathit{parent}(X, Y) \leftarrow \mathit{mother}(X, Y).$
$\mathit{father}(\mathit{tom}, \mathit{eve}).$
$\mathit{mother}(\mathit{eve}, \mathit{tim}).$

Query: $\leftarrow \mathit{grandfather}(\mathit{tom}, X).$

Refutation example

$$\leftarrow \mathit{grandfather}(\mathit{tom}, X).$$

$$\mathit{grandfather}(X_0, Z_0) \leftarrow \mathit{father}(X_0, Y_0), \mathit{parent}(Y_0, Z_0).$$

$$\leftarrow \mathit{father}(\mathit{tom}, Y_0), \mathit{parent}(Y_0, X).$$

$$\mathit{father}(\mathit{tom}, \mathit{eve}).$$

$$\leftarrow \mathit{parent}(\mathit{eve}, X).$$

$$\mathit{parent}(X_2, Y_2) \leftarrow \mathit{mother}(X_2, Y_2).$$

$$\leftarrow \mathit{mother}(\mathit{eve}, X).$$

$$\mathit{mother}(\mathit{eve}, \mathit{tim}).$$

$$\square$$

What are the unifiers?

Failed refutation example

$$\leftarrow \mathit{grandfather}(\mathit{tom}, X).$$

$$\mathit{grandfather}(X_0, Z_0) \leftarrow \mathit{father}(X_0, Y_0), \mathit{parent}(Y_0, Z_0).$$

$$\leftarrow \mathit{father}(\mathit{tom}, Y_0), \mathit{parent}(Y_0, X).$$

$$\mathit{father}(\mathit{tom}, \mathit{eve}).$$

$$\leftarrow \mathit{parent}(\mathit{eve}, X).$$

$$\mathit{parent}(X_2, Y_2) \leftarrow \mathit{father}(X_2, Y_2).$$

$$\leftarrow \mathit{father}(\mathit{eve}, X).$$

$\mathit{grandfather}(X, Z) \leftarrow \mathit{father}(X, Y), \mathit{parent}(Y, Z).$
$\mathit{parent}(X, Y) \leftarrow \mathit{father}(X, Y).$
$\mathit{parent}(X, Y) \leftarrow \mathit{mother}(X, Y).$
$\mathit{father}(\mathit{tom}, \mathit{eve}).$
$\mathit{mother}(\mathit{eve}, \mathit{tim}).$

What are the unifiers?

Horn clauses

Refutation

Search trees

Motivation

- When evaluating a logic program . . .
- . . . some attempted refutations succeed, some fail.
- We want to find ***all succeeding refutations***.
- (Actually, our real interest is in the unifiers that accompany them.)
- We have to analyse the ***search tree*** spanned . . .
- . . . by the program and the query.
- Combining all refutations (failing and succeeding) . . .
- . . . we get such a tree

Search tree for a logic program

$\mathrm{grandfather}(X, Z) \leftarrow \mathrm{father}(X, Y), \mathrm{parent}(Y, Z).$
$\mathrm{parent}(X, Y) \leftarrow \mathrm{father}(X, Y).$
$\mathrm{parent}(X, Y) \leftarrow \mathrm{mother}(X, Y).$
$\mathrm{father}(\mathrm{tom}, \mathrm{eve}).$
$\mathrm{mother}(\mathrm{eve}, \mathrm{tim}).$
$\leftarrow \mathrm{grandfather}(\mathrm{tom}, X).$