Logic Programming

Gilian Henke

Universität zu Lübeck

January 5, 2016

Content

Content

History

- Idea from 1930s
- Based on automation of theorem proving and AI
- 1970s Kowalski developed SLD resolutions
- Colmerauer developed Prolog

Content

Basics

- Based on First-Order-Predicate-Logic
- Declarative language
- Divided into Logic and Control
- Most common language: Prolog
- Others: Datalog, Parlog

Content

Horn Clause

Definition (Horn Clause)

A clause, i.e. a disjunction of literals, where at most one positive literal exists.

Example: $A \vee \neg B \vee \neg C$

Different Horn Clauses

Normal name	Normal logic	Logic programming	Logic programming name
Definite clause	$A \vee \neg B_1 \vee \neg B_2 \vee ... \vee \neg B_n$	$A \leftarrow B_1 \wedge B_2 \wedge ... \wedge B_n$	Rule
Unit clause	A	$A \leftarrow$	Fact
Goal clause	$\neg G_1 \vee \neg G_2 \vee ... \vee \neg G_n$	$\leftarrow G_1 \wedge G_2 \wedge ... \wedge G_n$	Query

Hello World!

```
hw(helloworld).
?-hw(X).

?- write('Hello world!').
```

Example in Prolog

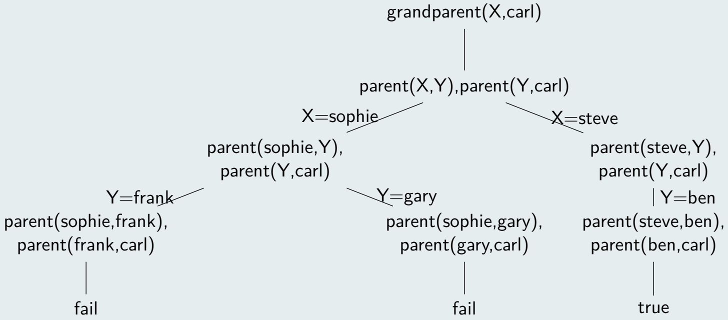
```
parent(sophie,frank).
parent(sophie,gary).
parent(steve,ben).
parent(steve,sophie).
parent(claire,ben).
parent(claire,sophie).
parent(alice,carl).
parent(ben,carl).
parent(tom,frank).
parent(tom,gary).
grandparent(X,Z):-parent(X,Y),parent(Y,Z).
ancestor(X,Y):-parent(X,Y).
ancestor(X,Y):-parent(Z,Y),ancestor(X,Z).
?-grandparent(X,gary).
?-parent(sophie,gary).
?-ancestor(steve,carl).
```

Content

Search-Tree

- How to get the solution to a query?
- Build a Search-Tree with Depth-First Traversal, where
  - the query is the root
  - nodes are subgoals
  - leaves are success or failure nodes
- If a path to a success node exist the query is successful

Example


grandparent(X,carl)
parent(X,Y),parent(Y,carl)
X=sophie
X=steve
parent(sophie,Y),
parent(Y,carl)
parent(steve,Y),
parent(Y,carl)
Y=frank
Y=gary
Y=ben
parent(sophie,frank),
parent(frank,carl)
parent(sophie,gary),
parent(gary,carl)
parent(steve,ben),
parent(ben,carl)
fail
fail
true

Substitution

Definition (Substitution)

Let X be variable and t be a term. For a given set of tuples (X,t) the substitution θ denotes that if θ is applied to a term s every X_i is replaced by its corresponding t_i.

Example: $\theta = (X, alice)$
$\theta(parent(X, Y)) = parent(alice, Y)$

Unification

Definition (Unification)

Let s and t be terms. If there exists a Substitution θ so that $s\theta = t\theta$, then θ is called the unifier of s and t. This is also called that t and s unify.

Example: $\theta = (X, alice)$
$\theta(parent(X, Y)) = parent(alice, Y)$
$\theta(parent(alice, Y)) = parent(alice, Y)$

Resolution

Definition (Resolution)

If the Rules $A \leftarrow B$ and $B \leftarrow C$ holds true then also $A \leftarrow C$ holds true.

SLD-Resolution

```
Data: Q=G_1, G_2...G_n
Result: substitution σ and failure
Resolvent = Q;
σ={};
failure=false;
while Resolvent! = {} do
    select G_i ∈ Resolvent;
    if G_i=true then
        delete G_i;
        continue;
    end
    select Rule A ← B_1...B_m where A and G_i unify with θ;
    if A does not exists then
        failure = true;
        break;
    end
    replace G_i with B_1...B_m;
    apply θ to Resolvent;
    σ = θσ;
end
return σ, failure;
```

Content

Problems

- Negation as failure
- Determinism
- Termination
- Prolog: Non-Uniformity

Conclusion

- Declarative programming language based on First Order Predicate Logic
- Useful in specialised contexts
- Non-trivial problems exist

Thank you for your attention!