Logic Programming

- Declarative programming style
- Example: Prolog
- Fundamental concept: First-order Predicate Logic
  - The execution of a Prolog program is a proof
  - A proof is an execution of a Prolog program

Horn Clauses

- Conjunctive Normal Form
  - Conjunction of clauses, clauses are disjunction of literals
  - $(A \lor B) \land (C \lor D \lor E) ...$

- Horn Clauses
  - At most one literal is positive, the others are negative

- Definite Horn Clauses
  - Exactly one positive literal
  - $A \lor \neg B \lor \neg C \lor \neg D ...$
  - $A \leftarrow B \land C \land D ...$

Facts

- Facts model a relation between elements.
- Facts are definite Horn clauses without negative literals

- Example:
  - olympics.
  - olympics(1896, athens).
  - olympics(1908, london).
  - olympics(2012, london).
  - olympics(2020, tokyo).
Rules

- Conditional expressions of the form
  - $A : - B, C, D, ... Z$
  - Semantics: $A$ becomes true if $B$ is true and $C$ is true, ...

- Example:
  - hostCity(X) :- olympics(_, X).
  - megaHostCity(City) :- olympics(_, City), population(City, Size), Size > 5000000.

- Variables need to start with capital letter or underscore.
- '_' means "don't care"
Resolution
- gradStudent(bill).
- gradStudent(sally).
- newborn(tom).
- newborn(mary).
- father(bill, mary).
- father(bill, joe).
- mother(sally, tom).
- parent(X, Y) :- father(X, Y).
- parent(X, Y) :- mother(X, Y).
- tired(X) :- gradStudent(X), parent(X, Y), newborn(Y).

SLD-Resolution

SLD-Resolution

Selective Linear Definite Clause Resolution
Lists

- A list can be described as [Head|Tail] where Head is an element and Tail is a list.
- The empty list is a list.

- We write:
  - []
  - [a,b,c]
  - this is the same as [a][b][c]]

- List membership:
  - member(X, [X|T]).
  - member(X, [Y|T]) :- member(X, T).

Appending lists:

- append([], X, X).
- append([X|L1], Y, [X|L2]) :- append(L1, Y, L2).

Failure-driven Loops

- olympics(X, athens), write('athens: '), writeln(X), fail; true.

Conditions

- olympics(X,Y), writeln(X), Y=Atlanta -> writeln(Y).

- olympics(X,Y), Y == athens -> writeln(' in greece'); writeln(' somewhere else').
The Cut operator

- `olympics(X, Y), write(Y), write(': '), writeln(X), Y==atlanta -> !, fail; true.`

Arithmetic

- `factorial(1,1).`
- `factorial(N,M) :- _n is N-1, factorial(_n, _fact), M is N * _fact.`

Equality

- `=`
  - Unification
  - `==, \\==`
  - Term equality, inequality
- `=:=, =\=`
  - Arithmetic or boolean equality, inequality
- `is/2`
  - Evaluation and unification

  - `1==1.0.`
  - `false`
  - `1=:=1.0.`
  - `true`

Negation

- `Negation as failure`
  - `not(Goal) :- Goal, !, fail.`
  - `not(Goal).`
  - Built-in predicate in SWI-Prolog but deprecated.

  - `\+`
    - `notHost(City) :- \+ hostCity(City).`
Operators

- op/3 (Directive)
- op(+Precedence, +Type, +Name)
  - Precedence is a number between 0 and 1200
  - Operators like + have ~ 200, */ have 400.
- Type is
  - xf, xy, yfx for infix
  - fx, fy for prefix
  - xf, yf for suffix
- :- op(Precedence, Type, Name).

DCG Grammars

- Parsing:
  - s -- [olympic], [games], year, [in], city.
  - year -- [Num], { number(Num) }.
  - city -- [City].
- phrase(s, ['olympic', 'games', 1984, 'in', 'losAngeles']).

Association Lists

- empty_assoc(-Assoc)
- put_assoc(+Key, +Assoc, +Value, ?NewAssoc)
- get_assoc(+Key, +Assoc, ?Value)
The (ugly) Truth About Unification

?- \( (X = f(X), X) \).
?- \( X = (X = f(X)) \).

?- unify_with_occurs_check(X = f(X), X).
false.

The Zebra Puzzle

There are five houses.
The Englishman lives in the red house.
The Spaniard owns the dog.
Coffee is drunk in the green house.
The Ukrainian drinks tea.
The green house is immediately to the right of the ivory house.
The Old Gold smoker owns snails.
Kools are smoked in the yellow house.
Milk is drunk in the middle house.
The Norwegian lives in the first house.
The man who smokes Chesterfields lives in the house next to the man with the fox.
Kools are smoked in the house next to the house where the horse is kept.
The Lucky Strike smoker drinks orange juice.
The Japanese smokes Parliaments.
The Norwegian lives next to the blue house.
Now, who drinks water? Who owns the zebra?