Logic Programming - 1

Datalog

Prolog facts → Relational Database

- One Predicate per row; One fact per row

Prolog Rules → Relational Views

- Intentional definition; materialised when needed

Prolog Queries → Relational Algebra

Prolog allows us to use declarative problem specifications as an executable solution generator. This means formalising a problem amounts to solving a problem. Prolog clauses can be viewed both declaratively (as denoting a logical formula) and also procedurally (as a way of unfolding a query).

Prolog:

```
teenager(X) :- (Male(X) ; Female(X)), age(X, Y), Y > 12, Y < 20.
```

Declarative:

$$\forall X \forall Y (teenager(X) \leftarrow (Male(X) \wedge female(X)) \wedge age(X, Y) \wedge Y > 12 \wedge Y < 20$$

Prolog's logically specified definitions can be queried in ways that go beyond their original purpose. For example, if we can specify how to concatenate two lists then the inverse is splitting the resultant list. This can break the correspondence between declarative and procedural semantics and limit the ways in which definitions can be queried.

Arithmetic operators require grounded arguments.

Logic programming is not classical logic, prolog's core syntax restricts first-order logic (to definite clausal normal form). Prolog's core semantics extend first-order logic (to minimal model constructors)

Prolog assumes false atoms with no reason to suggest that they may be true (An atom is a single data item without inherent meaning).

- This allows prolog to compute relations that are not classically definable via transitive closure.

Variables: X Y - denote arbitrary objects (in some implicit domain)

Constants: Oliver Peter - denote specific objects

Functions: mother/1 father/1 denote mappings between objects

Propositions: p q - represent unstructured assertions

Predicates: happy/1 loves/2 represent object properties and relations

Functions and predicates have "arities" which is the number of arguments they can take. Constants and propositions have zero arity.

For example:

If x(y(z)) is a well-formed ground formula of first-order classical logic where x, y and z are symbols.

X must be a predicate since if it was a function the expression is no longer a formula. y must be a function and z must be a constant.

A term is a constant c, variable X or a function f of arity n applied to an n-tuple of terms.

An atom is a proposition p or predicate r of arity n applied to an n-tuple of terms.

A formula is an atom; a logical constant; a negation of a formula; a conjunction; a disjunction or a conditional.

Clausal Form:

The clausal form is the conjunctive normal form (CNF) with the prefix omitted and the matrix written as a set of sets of literals or a set of clauses of the form

$\leftarrow$

where the head is an (often implicit) disjunction of literals and the body is a conjunction of literals. Prolog is a variation of clausal form where exactly one atom must be used in the head of each clause but nested conjunctions, disjunctions and negations of atoms may be used in the body.

To facilitate the storage of logical formulae in memory and simplify inference procedures it's convenient to transform formulae into some restricted subsets of First-Order Logic (FOL) known as normal forms.