Logic and Relational Databases

bioactivity

mol	ACE	SSRI
m1	0	0
m2	1	0
m3	0	1
⋮	⋮	⋮

atoms

mol	name	type	X	Y	Z
m1	a1	c	2.1	-1.3	3.4
m1	a2	o	3.0	-1.0	3.0
⋮	⋮	⋮	⋮	⋮	⋮

bonds

mol	atom1	atom2	type
m1	a1	a2	2
m1	a1	a3	1
⋮	⋮	⋮	⋮

```
bioactivity(m1, 0, 0).
bioactivity(m2, 1, 0).
bioactivity(m3, 0, 1).
    ⋮
```

```
atom(m1, a1, c, 2.1, -1.3, 3.4).
atom(m1, a2, o, 3.0, -1.0, 3.0).
    ⋮
```

```
bond(m1, a1, a2, 2).
bond(m1, a1, a3, 1).
    ⋮
```

"Extensional Database"

Datalog

Definite program containing no non-nullary function symbols.

grandparent(X, Z) ← parent(X, Y), parent(Y, Z).
parent(X, Y) ← father(X, Y).
parent(X, Y) ← mother(X, Y).

father(adam, bill).
father(adam, beth).
father(bill, cathy).
father(donald, eric).

mother(anne, bill).
mother(anne, beth).
mother(cathy, donald).
mother(diana, eric).

mother and father are defined <u>extensionally</u>.
grandparent and parent are defined <u>intensionally</u>.

Some Interesting Notes

- With function symbols, even the "simplest" logic program is undecidable — more specifically there exist choices of literals L_1, L_2, & L_3 such that it is undecidable

  if $\begin{array}{l} L_1. \\ L_2 \leftarrow L_3. \end{array} \models L_4$

  for some literals L_4.

- Moreover, there exist 2-literal clauses

- Moreover, there exist 2-literal clauses such that the other 2-literal clauses entailing/entailed by them are undecidable.

- But Datalog <u>is</u> <u>decidable</u>. (Not using SLD-resolution but using other approaches such as Magic Sets – Ramakrishnan, Ch. 15.)

What are we still missing?

- Field names.
  Syntactic sugar — will live without.

- Types for fields.
  Just unary predicates:
  male(tom).   person(X) ← male(X).

- Query language.
  Coming next... will show that Datalog with negation as finite failure has at least the computational power of relational algebra.

Primitive Operations of Relational Algebra

- Union
- Set Difference
- Cartesion (cross-) Product
- Projection
- Selection

Union

$$R_1 \cup R_2 = \{ \langle x_1, \ldots, x_n \rangle \mid \langle x_1, \ldots, x_n \rangle \in R_1 \lor \langle x_1, \ldots, x_n \rangle \in R_2 \}$$

In Datalog, use predicate symbols r_1/n & r_2/n for R_1 & R_2, respectively. Use r/n for their union:

$$r(X_1, \ldots, X_n) \leftarrow r_1(X_1, \ldots, X_n).$$
$$r(X_1, \ldots, X_n) \leftarrow r_2(X_1, \ldots, X_n).$$

Example: parent(X, Y) ← father(X, Y).
parent(X, Y) ← mother(X, Y).

Difference

$$R_1 \setminus R_2 = \{\langle x_1, \ldots, x_n \rangle \mid \langle x_1, \ldots, x_n \rangle \in R_1 \wedge \langle x_1, \ldots, x_n \rangle \notin R_2\}$$

In Datalog:

$$r(X_1, \ldots, X_n) \leftarrow r_1(X_1, \ldots, X_n),\ not\ r_2(X_1, \ldots, X_n).$$

Example:

father(X, Y) ← parent(X, Y), not mother(X, Y).

Cartesian Product

$$R_1 \times R_2 = \{\langle x_1, \ldots, x_m, y_1, \ldots, y_n \rangle \mid \langle x_1, \ldots, x_m \rangle \in R_1 \wedge \langle y_1, \ldots, y_n \rangle \in R_2\}$$

In Datalog:

$$r(X_1, \ldots, X_m, Y_1, \ldots, Y_n) \leftarrow r_1(X_1, \ldots, X_m),\ r_2(Y_1, \ldots, Y_n).$$

Example:

mfcouple(X, Y) ← male(X), female(Y).

Projection

$$\pi_{i_1,\ldots,i_m} R = \{ \langle x_{i_1},\ldots,x_{i_m} \rangle \mid \langle x_1,\ldots,x_n \rangle \in R, \; 1 \le i_1,\ldots,i_m \le n \}$$

(Projection deletes and/or rearranges one or more columns from R.)

In Datalog:

$$pr(X_{i_1},\ldots,X_{i_m}) \leftarrow r(X_1,\ldots,X_n).$$

Example: father(X) ← father(X, Y).

Selection

$$\sigma_F(R) = \{ \langle x_1,\ldots,x_n \rangle \mid \langle x_1,\ldots,x_n \rangle \in R \text{ and formula } F \text{ is true of } \langle x_1,\ldots,x_n \rangle \}$$

F is allowed to contain comparisons (=, ≥, etc.) among variables and constants, joined by ∧, ∨, ¬.

field names

values

Datalog translation depends on F. Where D(F) is the Datalog representation of F:

$\sigma_F R$ is represented as

$$rs(X_1,\ldots,X_n) \leftarrow r(X_1,\ldots,X_n),\ D(F).$$

Example: $\sigma_{Y \ge 1{,}000{,}000}$ INCOME(X, Y) is

millionaire(X, Y) ← income(X, Y), Y ≥ 1,000,000.

Notes on Selection

- Replace field names in F by corresponding variable names during translation.
- Using ; for $\vee$ and not for $\neg$, translate F directly.

R

f_1	f_2	f_3	f_4	f_5
⋮	⋮	⋮	⋮	⋮

$\sigma_{(f_2 = f_3) \vee (f_4 > f_5 \wedge f_5 \neq 10)} R$ becomes

```
sr(X1,...,X5) <- r(X1,...,X5),
                 ((X2 == X3) ;
                  (X4 > X5 , X5 \= 10)).
```

Can build other relational operators from those relational algebra primitives.

Example: Natural Join

$R \bowtie S = \pi_A \, \sigma_{R.T_1 = S.T_1 \wedge \ldots \wedge R.T_k = S.T_k} (R \times S)$

where $T_1, \ldots, T_k$ are the attributes (field names) appearing in both R and S, and A is the list of all attributes except $S.T_1, \ldots, S.T_k$.

father(X,Y) $\bowtie$ parent(Y,Z):

```
cross(X,Y1,Y2,Z) <- father(X,Y1), parent(Y2,Z).
sel (X,Y1,Y2,Z) <- cross(X,Y1,Y2,Z), Y1=Y2.
proj (X,Y1,Z) <- sel(X,Y1,Z).
```

Could do in one step:

```
gf(X,Y,Z) <- father(X,Y), parent(Y,Z).
```

Missing Values

f1	f2	f3
a	1	?
b	?	g
c	3	f

Like existential variables: we know some value must go there, but we don't know what.

Datalog: no existential quantifiers. Use skolem constants. Every missing value is represented by a distinct skolem constant.

p(a, 1, sk-1).
p(b, sk-2, g).
p(c, 3, f).