Datalog

P.J. McBrien

Imperial College London

Data is held as extensional predicates

account(100, 'current', 'McBrien, P.', _, 67).
account(101, 'deposit', 'McBrien, P.', 5.25, 67).
account(103, 'current', 'Boyd, M.', _, 34).
account(107, 'current', 'Poulovassilis, A.', _, 56).
account(119, 'deposit', 'Poulovassilis, A.', 5.50, 56).
account(125, 'current', 'Bailey, J.', _, 56).

branch(56, 'Wimbledon', 94340.45).
branch(34, 'Goodge St', 8900.67).
branch(67, 'Strand', 34005.00).

movement(1000, 100, 2300.00, 5/1/1999).
movement(1001, 101, 4000.00, 5/1/1999).
movement(1002, 100, −223.45, 8/1/1999).
movement(1004, 107, −100.00, 11/1/1999).
movement(1005, 103, 145.50, 12/1/1999).
movement(1006, 100, 10.23, 15/1/1999).
movement(1007, 107, 345.56, 15/1/1999).
movement(1008, 101, 1230.00, 15/1/1999).
movement(1009, 119, 5600.00, 18/1/1999).

Rules defined as intentional predicates

```
current_account(No, Name, Sortcode) :-
        account(No, 'current', Name, _, Sortcode).
deposit_account(No, Name, Rate, Sortcode) :-
        account(No, 'deposit', Name, Rate, Sortcode).
active_customers(CName, BName) :-
        branch(Sortcode, BName, _),
        account(No, _, CName, _, Sortcode),
        movement(_, No, _, _).
```

Datalog Rules

Datalog rules take the form
Head :- Body.

- Logical semantics:
  if Body the Head
- Head may be a single predicate
- Body may be any conjunction of predicates.

Naming of predicates and variables

- You cannot use the same name for an intentional and extentional predicate
- Convention is the start predicate name with small letter
- Variables start with a capital letter
- A variable that only appears once can be replaced by '_'

Quiz 1: Valid Datalog Knowledgebase

Which Datalog Knowledgebase is invalid?

A

```
single_male('Peter').
married_to('Paul', 'Jane').
male(X) :- married_to(X, _).
male(X) :- single_male(X).
female(X) :- married_to(_, X).
female(X) :- single_female(X).
```

B

```
male('Peter').
married_to('Paul', 'Jane').
male(X) :- married_to(X, _).
female(X) :- married_to(_, X).
```

C

```
male('Peter').
male('Paul').
female('Jane').
married_to('Paul', 'Jane').
```

D

```
married_to('Peter', _).
married_to('Paul', 'Jane').
```

Model-Theoretic Interpretation

```
deposit_account(No, Name, Rate, Sortcode) :-
        account(No, 'deposit', Name, Rate, Sortcode).
account(100, 'current', 'McBrien, P.', _, 67).
account(101, 'deposit', 'McBrien, P.', 5.25, 67).
account(103, 'current', 'Boyd, M.', _, 34).
account(107, 'current', 'Poulovassilis, A.', _, 56).
account(119, 'deposit', 'Poulovassilis, A.', 5.50, 56).
account(125, 'current', 'Bailey, J.', _, 56).
```

Minimal Model

If we can assign any combination of values to the variables, what is the minimum set of predicates that must be true.

Minimal Model

deposit_account(101, 'McBrien, P.', 5.25, 67).
Is not a model, since it implies deposit_account(119, 'Poulovassilis, A.', 5.50, 56) is false, but deposit_account(119, 'Poulovassilis, A.', 5.50, 56) is true due to the rule for deposit_account.

Model-Theoretic Interpretation

```
deposit_account(No, Name, Rate, Sortcode) :-
        account(No, 'deposit', Name, Rate, Sortcode).
account(100, 'current', 'McBrien, P.', _, 67).
account(101, 'deposit', 'McBrien, P.', 5.25, 67).
account(103, 'current', 'Boyd, M.', _, 34).
account(107, 'current', 'Poulovassilis, A.', _, 56).
account(119, 'deposit', 'Poulovassilis, A.', 5.50, 56).
account(125, 'current', 'Bailey, J.', _, 56).
```

Minimal Model

If we can assign any combination of values to the variables, what is the minimum set of predicates that must be true.

Minimal Model

deposit_account(101, 'McBrien, P.', 5.25, 67).
deposit_account(119, 'Poulovassilis, A.', 5.50, 56).
deposit_account(127, 'Poulovassilis, A.', 4.50, 56).
Is not a minimal model, since deposit_account(127, 'Poulovassilis, A.', 4.50, 56) could be made false, and the model still be consistent.

Model-Theoretic Interpretation

```
deposit_account(No, Name, Rate, Sortcode) :-
        account(No, 'deposit', Name, Rate, Sortcode).
account(100, 'current', 'McBrien, P.', _, 67).
account(101, 'deposit', 'McBrien, P.', 5.25, 67).
account(103, 'current', 'Boyd, M.', _, 34).
account(107, 'current', 'Poulovassilis, A.', _, 56).
account(119, 'deposit', 'Poulovassilis, A.', 5.50, 56).
account(125, 'current', 'Bailey, J.', _, 56).
```

Minimal Model

If we can assign any combination of values to the variables, what is the minimum set of predicates that must be true.

Minimal Model

```
deposit_account(101, 'McBrien, P.', 5.25, 67).
deposit_account(119, 'Poulovassilis, A.', 5.50, 56).
```

Is a minimal model

Quiz 2: Datalog Queries

```
active_current_account(No) :-
        account(No, 'current', Name, _, Sortcode),
        movement(_, No, _, _).
```

What is the minimum model?

A

active_current_account(100).
active_current_account(101).
active_current_account(103).
active_current_account(107).
active_current_account(119).
active_current_account(125).

B

active_current_account(100).
active_current_account(101).
active_current_account(103).
active_current_account(107).
active_current_account(119).

C

active_current_account(100).
active_current_account(103).
active_current_account(107).
active_current_account(125).

D

active_current_account(100).
active_current_account(103).
active_current_account(107).

Datalog$^\neg$: Datalog with Negation

Safe Nagation

Use $\neg$ infront of a predicate to mean that it must not hold.
Any variable that appears in a negated predicate must have previously appeared in a non-negated predicate.

Find accounts without any movements

```
dormant_account(No) :-
        account(No, _, _, _, Sortcode)
        ¬movement(_, No, _, _).
```

Minimum Model

dormant_account(125).

Quiz 3: Datalog$^\neg$ Queries

branch_without_recent_debit(BName) :-
    branch(Sortcode, BName, _),
    account(No, _, _, _, Sortcode),
    ¬account_with_recent_debit(No).
account_with_recent_debit(No) :-
    movement(_, No, Value, TDate),
    Value $>$ 0,
    TDate $>$ 15/1/1999.

What is the minimum model?

A

B

branch_without_recent_debit('Wimbledon').

C

branch_without_recent_debit('Goodge St').
branch_without_recent_debit('Strand').

D

branch_without_recent_debit('Wimbledon').
branch_without_recent_debit('Goodge St').
branch_without_recent_debit('Strand').

Projection

π

RA projection is performed by only using a subset of rule body variables in the head of a rule.

$\pi_{sortcode}\, account$

```
account_sortcode(Sortcode) :-
        account(_, _, _, _, Sortcode).
```

Minimum Model

```
account_sortcode(34).
account_sortcode(56).
account_sortcode(67).
```

Selection

σ

RA selection is performed by naming a variable more than once, or by putting a data value in the rule body.

$\sigma_{\text{amount}>1000}$ account

```
big_debit(Mid, No, Amount, Date) :-
        movement(Mid, No, Amount, Date),
        Amount > 1000.
```

Minimum Model

big_debit(1000, 100, 2300.00, 5/1/1999).
big_debit(1001, 101, 4000.00, 5/1/1999).
big_debit(1008, 101, 1230.00, 15/1/1999).
big_debit(1009, 119, 5600.00, 18/1/1999).

Product

$\times$

RA product is performed by naming two predicates in the rule body.

$\pi_{\text{bname,cname}}\, \sigma_{\text{branch.sortcode=account.sortcode}}(\text{branch} \times \text{account})$

```
branch_customers(BName, CName) :-
        branch(BSortcode, BName, _),
        account(_, _, CName, _, ASortcode),
        BSortcode = ASortcode.
≡
branch_customers(BName, CName) :-
        branch(Sortcode, BName, _),
        account(_, _, CName, _, Sortcode).
```

Minimum Model

```
branch_customers('Wimbledon', 'Poulovassilis, A.').
branch_customers('Wimbledon', 'Bailey, J.').
branch_customers('Goodge St', 'Boyd, M.').
branch_customers('Strand', 'McBrien, P.').
```

Quiz 4: Translating RA to Datalog

$\pi_{\text{bname}}\, \sigma_{\text{account.sortcode}=\text{branch.sortcode} \wedge \text{type}=\text{'deposit'}} (\text{account} \times \text{branch})$

Which datalog rule for query is *not* equivalent to the above RA query?

A

```
query(BName) :-
        account(_, 'deposit', _, _, Sortcode),
        branch(Sortcode, BName, _).
```

B

```
query(BName) :-
        branch(Sortcode1, BName, _),
        account(_, 'deposit', _, _, Sortcode2),
        Sortcode1 = Sortcode2.
```

C

```
query(BName) :-
        branch(_, BName, _).
query(BName) :-
        branch(Sortcode, BName, _),
        account(_, 'deposit', _, _, Sortcode).
```

D

```
query(BName) :-
        branch(Sortcode, BName, _),
        deposit_branch(Sortcode).
deposit_branch(Sortcode) :-
        account(_, 'deposit', _, _, Sortcode).
```

Union

$\cup$

RA union is performed by having more than one rule definition for an intentional predicate.

$\sigma_{\text{amount}>1000}$ movement $\cup$ $\sigma_{\text{amount}<100}$ movement

```
big_movement(Mid, No, Amount, Date) :-
        movement(Mid, No, Amount, Date),
        Amount > 1000.
big_movement(Mid, No, Amount, Date) :-
        movement(Mid, No, Amount, Date),
        Amount < 100.
```

Minimum Model

big_movement(1000, 100, 2300.00, 5/1/1999).
big_movement(1001, 101, 4000.00, 5/1/1999).
big_movement(1002, 100, −223.45, 8/1/1999).
big_movement(1008, 101, 1230.00, 15/1/1999).
big_movement(1009, 119, 5600.00, 18/1/1999).

Difference

—

RA difference is performed using a negation on the predicate being substracted: need Datalog$^{\neg}$.

π_{no} account $-$ π_{no} movement

dormant_account(No) :-
account(No, _, _, _, _),
¬movement(_, No, _, _).

Minimum Model

dormant_account(125).