Lecture 26

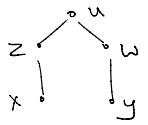
Thursday, November 19, 2020 4:05 PM

Plan for today

- How are Datalog programs evaluated?
  - Bottom-up algorithms
    - Seminaive evaluator (≃ state-of-the-art)
    - Naive evaluator: ←
  - Top-down: Prolog
    - SLD resolution

Prolog = Datalog + Mandated top down
+ cuts
+ Negation-as-failure
+ ~~functions~~

Question: How are Datalog programs evaluated?



cousins (x y):-
parent (x z) parent(z u)
parent (y w) parent(w u)

Approach 1:
- Guess the value of x y. (What do you draw them from? For all possible values of x y)
- Guess the intermediate variables z w u.
- Check if all body literals are true.
- If so, then emit the tuple cousins (x y).

---

cousins (x y):-
parent (x z) parent(z u)
parent (y w) parent(w u)

Parent	
John	Michael

parent(y w) parent(w u)

Parent	
John	Michael
Michael	Joseph

Approach 2:

- Start with _____

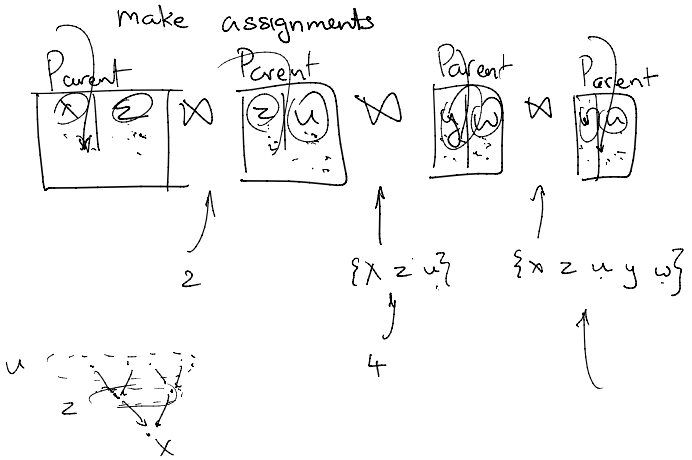
- Walk over all of the literals in the rule

  P_{xz} P_{zu} P_{yw} P_{wu}

  - Consult the corresponding relation.

  - For each tuple in the relation

    make assignments



$C_{xy} :- P_{xz}\ P_{zu}\ P_{wu}\ P_{yw}$

Literal

Selinger's algorithm.

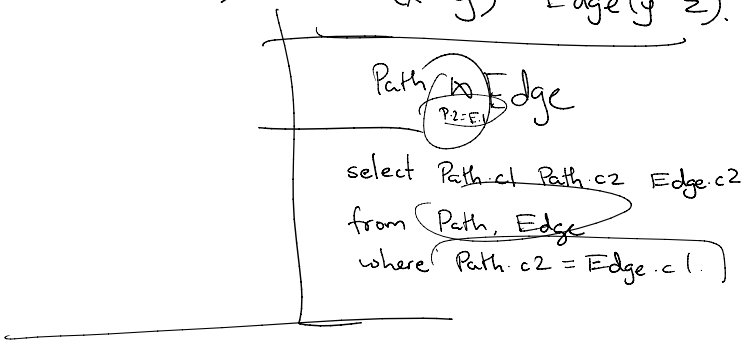
Conclusion: Evaluating a single (non-recursive) Datalog rule is not hard.

Even simpler approach: Translate rule into relational algebra.

Question 2: How to evaluate recursive Datalog rules?

Path(x z) :- Path(x y) Edge(y z).

Path(x z) :- Path(x y) Edge(y z).



R1 Path(x y) :- Edge(x y)

R2 Path(x z) :- Path(x y) Edge(y z).

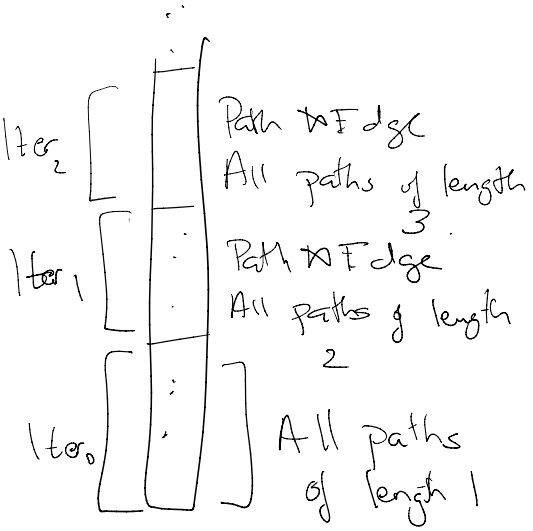
- Initialize Path = empty set.
- While old paths ≠ new paths

  Let P_1 = Rule1(Edge, Path)

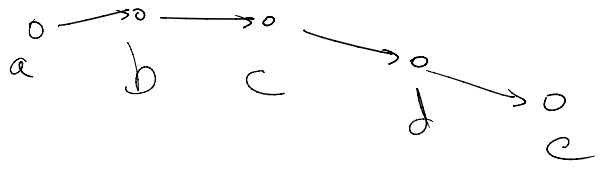
  Imagine that Rule 1 is a fn which takes the old values of Edge & Path & produces new paths

  Let P_2 = Rule2(Edge Path).

  Update Path = Path ∪ P_1 ∪ P_2.



In every iteration, we add new tuples.



Path
a b
a c
d d

→ Pxy :- Exy

Pxz :- Pxy Eyz

Why Pab?

[Because of Rule 1 & Eab] "Proof".

a e

b c

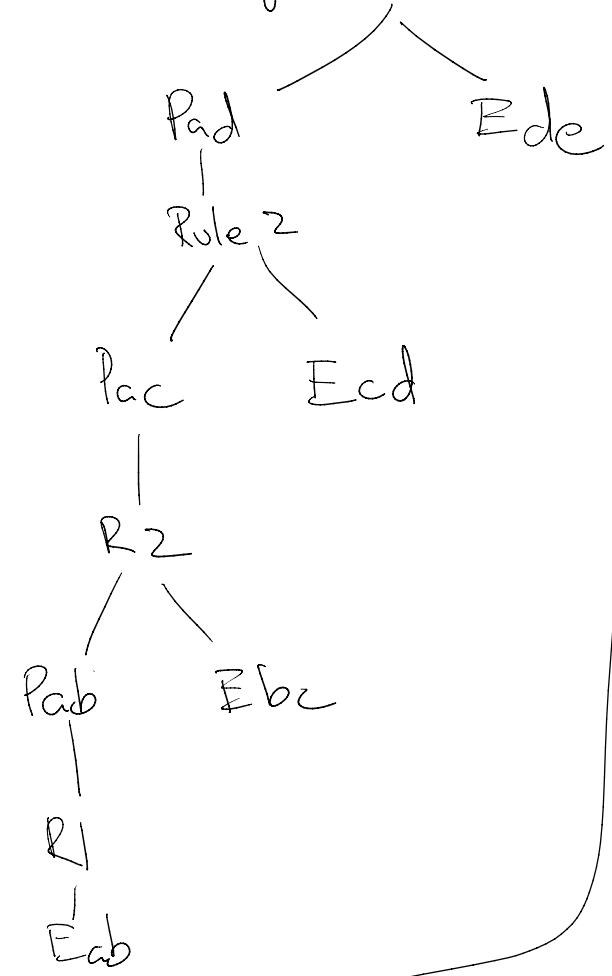
b d

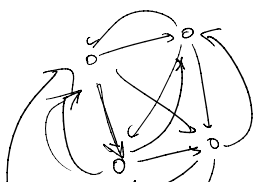
b e

:

Why Pae?

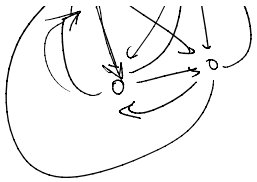
Because of Rule 2





Path

a g : The naive evaluation



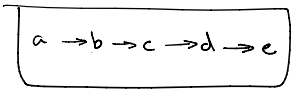
a	a	1
a	b	0
a	c	0
a	d	0
b	a	0
b	b	1
b	c	0
b	d	0
⋮		

The naive evaluation algorithm will discover all 16 tuples in just 2 iterations.

Iter 0 : P_{xy} for $x \neq y$ (Rule 1)

Iter 1 : P_{xx} for all x. (Rule 2)

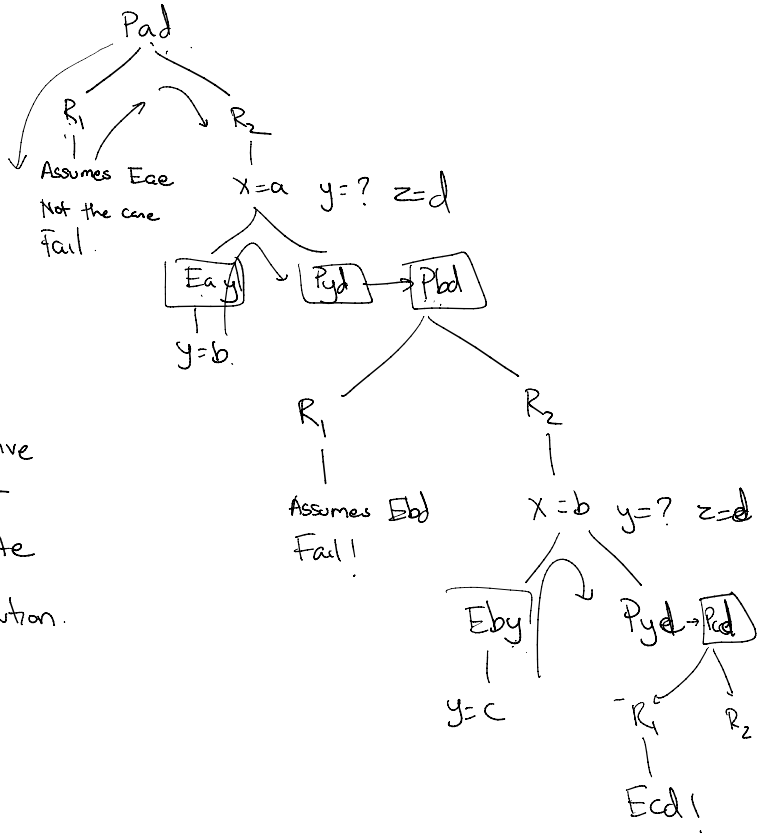
Question: What if we instead ask the solver "Is there a path from a to d?"



R_1 $P_{xy} :- E_{xy}$

R_2 $P_{xz} :- E_{xy}\ P_{yz}$

Eab
Ebc
Ecd
Ede



Selective Linear Definite Resolution.

Prolog: Datalog + Top-down SLD

+ Cuts