

Announcements

• Individual Assignment #1:

– Due Friday.

• This week's schedule:

– Lectures

- Mon, 3:10pm
- **Wed 9am (discussion slot)**
- Wed 3:10pm

– Discussion??

- **Fri 3:10pm (lecture slot)**

ECS-165B

1

More on Datalog Evaluation

⑥ $e(a,b) \cdot e(b,c) \cdot e(c,d)$ % FACTS (EDB-relation: $e/2$)

④ $tc(X,Y) :- e(X,Y)$ % RULES (IDB-relation: $tc/2$)

⑤ $tc(X,Y) :- e(X,Z), tc(Z,Y)$

$\{Z=Y\}$

Plan:
1. exec r_1 (completely)
2. repeat
 exec r_2
 until no new tc atoms

n I_n

0 \emptyset

1 $\{e(a,b), e(b,c), e(c,d)\} \leftarrow \text{facts (EDB)} - \text{rule 0}$

2 $\{e(a,b), e(b,c), e(c,d)\} \cup \{tc(a,b), tc(b,c), tc(c,d)\} \leftarrow \text{② copy } e/2 \text{ into } tc/2$

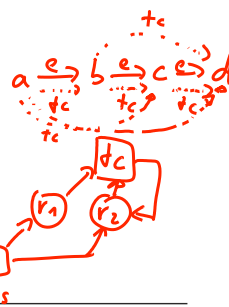
3 $\{e(a,b), e(b,c), e(c,d)\} \cup \{tc(a,b), tc(b,c), tc(c,d), tc(a,c), tc(b,d)\} \leftarrow \text{②}$

4 $\{e(a,b), e(b,c), e(c,d)\} \cup \{tc(a,b), tc(b,c), tc(c,d), tc(a,c), tc(b,d), tc(a,d)\}$

5 $\{...\} = \uparrow \text{no change!}$

\Rightarrow if the longest path in $e/2$ has length n , then $O(n)$ rounds are needed!

(Exercise: how about the following rule? $tc(X,Y) \leftarrow tc(X,Z), tc(Z,Y)$.)



ECS-165B

2

Three transitive closure variants

```
% EDB
e(1,2).
e(2,3).
e(3,4).

% IDB
|
% left-recursive (linear)
tc_lr(X,Y) :- e(X,Y).
tc_lr(X,Y) :- tc_lr(X,Z), e(Z,Y).

% right-recursive (linear)
tc_rr(X,Y) :- e(X,Y).
tc_rr(X,Y) :- e(X,Z), tc_rr(Z,Y).

% double-recursive (non-linear)
tc_dr(X,Y) :- e(X,Y).
tc_dr(X,Y) :- tc_dr(X,Z), tc_dr(Z,Y).
```



Round	tc_rr
1	{ (1,2), (2,3), (3,4) }
2	- - \cup { (1,3), (2,4) }
3	- - \cup { (1,4) }
4	no change! \Rightarrow fix point

ECS-165B

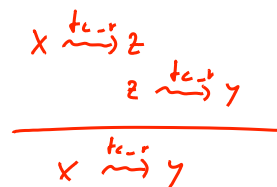
3

$$v_1: tc_dr(x,y) \leftarrow e(x,y).$$

$$v_2: tc_dr(x,y) \leftarrow tc_dr(x,z), tc_dr(z,y).$$



Round	tc_dr
1	{ (1,2), (2,3), (3,4) } // v_1
2	{ (1,3), (2,4) } \cup { (1,4) } // v_2
3	{ (1,4) } \cup { (1,3), (2,4) } // v_2
4	no change



ECS-165B

4

Recursion vs Aggregation, Negation

• Negation before or after recursion: OK

- (1) $E(X,Y) :- A(X,Y), \text{ not } B(X,Y).$ // $E := A \setminus B$ | $\text{node}(x) \in E(x, -).$
 (2) $TC(X,Y) :- E(X,Y).$ } $TC := E^+$ | $\text{node}(x) \in E(-, x)$
 (3) $TC(X,Y) \Leftarrow E(X,Z), TC(Z,Y).$
 (4) $nTC(X,Y) :- \text{node}(X), \text{node}(Y), \text{ not } TC(X,Y).$ (0)

↑ complement of $TC (= E^+)$ → one cannot reach Y from X via E^+

"Wrong schedule":

- (1)
 (4)
 {(2), (3)} until
 no change

"correct schedule":

- (0)
 (1)
 (2)
 repeat (3) until no change
 (4)

ECS-165B

5

Lowest Common Ancestor

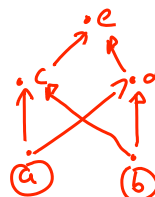
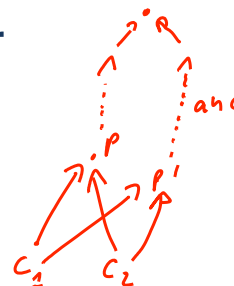
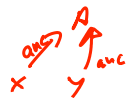
% anc(Node, Ancestor)
 $\text{anc}(C,P) :- \text{parent}(C,P).$ anc := parent⁺
 $\text{anc}(X,Y) :- \text{anc}(X,Z), \text{anc}(Z,Y).$

% ca(NodeX, NodeY, CommonAncestor)
 $\text{ca}(X,Y,A) :-$

$\text{anc}(X,A),$
 $\text{anc}(Y,A),$
 $X \neq Y.$

$\text{not_lca}(X,Y,P) :-$
 $\text{pc}(P,C),$
 $\text{ca}(X,Y,P),$
 $\text{ca}(X,Y,C).$

$\text{lca}(X,Y,L) :-$
 $\text{ca}(X,Y,L),$
 $\text{not not_lca}(X,Y,L).$



$\text{cn}(a,b) = \{c,d,e\}$
 $\text{L}(a,b) = \{c,d\}$

ECS-165B

6

Lowest Common Ancestor

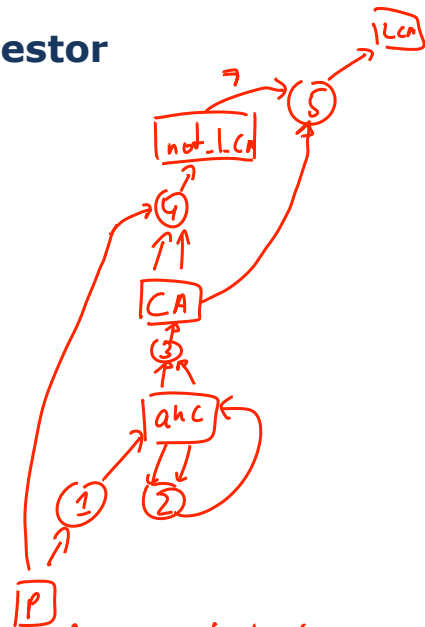
```
% anc(Node, Ancestor)
1 anc(C,P) :- parent(C,P).
2 anc(X,Y) :- anc(X,Z), anc(Z,Y).
```

```
% ca(NodeX,NodeY, CommonAncestor)
3 ca(X,Y,A) :-
    anc(X,A),
    anc(Y,A).
```

```
4 not_lca(X,Y,P) :-
    parent pc(P,C),
    ca(X,Y,P),
    ca(X,Y,C).
```

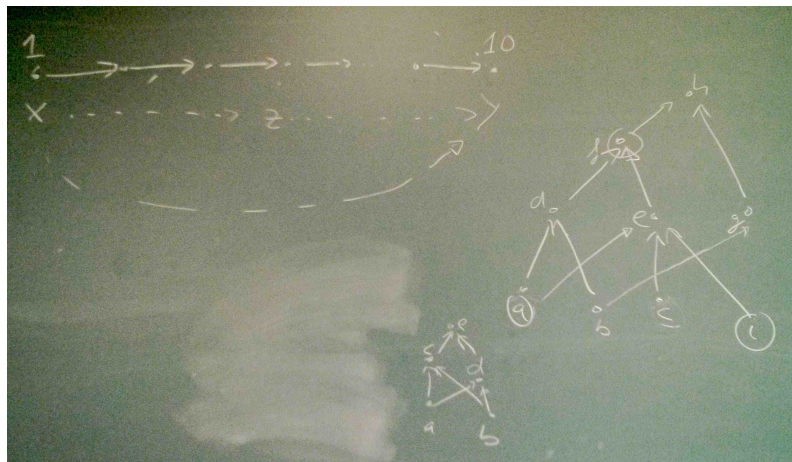
```
5 lca(X,Y,L) :-
    ca(X,Y,L),
    not not_lca(X,Y,L).
```

↑ need to compute not_lca completely before applying this rule ⇒ stratification



Recursion Examples

- TC variants
- LCA: on trees vs on DAGs



ECS-165B

8

LCA (ambiguous case on DAGs)

```

new Open Recent Print Copy Search Preferences Help
tc-prov.dlv 7 *shell* 8 ambiguous.dlv 1 lca.dlv

parent(a, c).
parent(a, d).
parent(b, c).
parent(b, d).
parent(c, e).
parent(d, e).

% anc(Node, Ancestor)
anc(C,P) :- parent(C,P).
anc(X,Y) :- anc(X,Z), anc(Z,Y).

% ca(NodeX,NodeY, CommonAncestor)
ca(X,Y,A) :-
    anc(X,A),
    anc(Y,A).

not_lca(X,Y,P) :-
    parent(C,P),
    ca(X,Y,P),
    ca(X,Y,C).

lca(X,Y,L) :-
    ca(X,Y,L),
    X \= Y,
    not not_lca(X,Y,L).

% are there multiple LCAs for a pair (X,Y)
ans(X,Y, L1, L2) :-
    lca(X,Y,L1),
    lca(X,Y,L2),
    X<Y,
    L1 < L2.

-:--- ambiguous.dlv All (7,0) (Prolog) -:--- lca.dlv Bot (2,10) (Prolog)
DLV [build BEN/Dec 21 2011 gcc 4.2.1 (Apple Inc. build 5666) (dot 3)]

parent(a,c), parent(a,d), parent(c,e), parent(d,e), parent(b,c), parent(b,d), anc(a,c), anc(a,d), anc(a,e), anc(c,e), anc(d,e), anc(b,c), anc(b,d), anc(b,e), ca(a,a,c), ca(a,a,d), ca(a,a,e), ca(a,b,c), ca(a,b,d), ca(a,b,e), ca(c,c,e), ca(c,c,d), ca(c,c,e), ca(c,b,e), ca(d,a,e), ca(d,c,e), ca(d,d,e), ca(d,b,e), ca(b,a,c), ca(b,a,d), ca(b,a,e), ca(b,c,e), ca(b,d,e), ca(b,b,c), ca(b,b,d), ca(b,b,e), not_lca(a,a,e), not_lca(a,b,e), not_lca(b,a,e), not_lca(b,b,e), lca(a,c,e), lca(a,d,e), lca(a,b,c), lca(a,b,d), lca(c,a,e), lca(c,d,e), lca(c,b,e), lca(d,a,e), lca(d,c,e), lca(d,b,e), lca(b,a,c), lca(b,a,d), lca(b,c,e), lca(b,d,e), ans(a,b,c,d)
-:*** *shell* 91% (21,668) (Shell:run)

```

Counting rule firings (right-recursive vs double-recursive)

```

e.dlv 1 ambiguous.dlv 1 lca.dlv 2 TC 3 tc-prov.dlv

% EDB
e(1,2).
e(2,3).
e(3,4).
e(4,5).
e(5,6).
e(6,7).
e(7,8).
e(8,9).
e(9,10).

% right-recursive (linear)
tc_rr(X,Y) :-
    e(X,Y).
fire_rr( f(X,Z,Y) ) :-
    e(X,Z),
    tc_rr(Z,Y).
tc_rr(X,Y) :-
    fire_rr( f(X,_,Y) ).

% double-recursive (non-linear)
tc_dr(X,Y) :-
    e(X,Y).
fire_dr( f(X,Z,Y) ) :-
    tc_dr(X,Z), tc_dr(Z,Y).
tc_dr(X,Y) :-
    fire_dr( f(X,_,Y) ).

% output
o(lr, N) :- #count{ F : fire_lr(F) } = N.
o(rr, N) :- #count{ F : fire_rr(F) } = N.
o(dr, N) :- #count{ F : fire_dr(F) } = N.

-:--- e.dlv All (1,0) (Prolog) -:--- tc-prov.dlv Bot (20,14) (Prolog)
*Completions*
fire_dr(f(2,7,8)), fire_dr(f(2,8,10)), fire_dr(f(2,8,9)), fire_dr(f(2,9,10)), fire_dr(f(3,8,10)), fire_dr(f(3,8,9)), fire_dr(f(3,9,10)), fire_dr(f(4,9,10)), fire_dr(f(1,2,7)), fire_dr(f(1,2,8)), fire_dr(f(1,2,9)), fire_dr(f(1,2,10)), fire_dr(f(2,3,8)), fire_dr(f(1,3,8)), fire_dr(f(2,3,9)), fire_dr(f(1,3,9)), fire_dr(f(2,3,10)), fire_dr(f(1,3,10)), fire_dr(f(3,4,9)), fire_dr(f(1,4,9)), fire_dr(f(2,4,9)), fire_dr(f(3,4,10)), fire_dr(f(1,4,10)), fire_dr(f(2,4,10)), fire_dr(f(3,5,10)), fire_dr(f(1,5,10)), fire_dr(f(2,5,10)), fire_dr(f(4,5,10)), o(lr,36), o(rr,36), o(dr,120)
[TC :]:5
-:*** *shell* Bot (77,5449) (Shell:run)

```

Counting derived facts and "rounds" (right recursive)

Counting derived facts and "rounds" (right recursive)

```

% EDB
e(1,2).
e(2,3).
e(3,4).
e(4,5).
e(5,6).
e(6,7).
e(7,8).
e(8,9).
e(9,10).

% maxInt=20
% right-recursive (linear)
tc_rr(C1, X, Y) :-
    e(X,Y),
    tc_rr(N1, X, Y) :-
        e(X,Z),
        tc_rr(N, Z, Y),
        N1 = N + 1.

% refly (N,X,Y) into an object f(N,X,Y)
d(f(N,X,Y)) :- tc_rr(N,X,Y).

% output
facts(C) :- %count{ X : d(X) } = C.

%----- edbview All (4,7) (Prolog)
%----- tc_rr_rounds.dlv All (11,28) (Prolog)

% shell* % Completions*
5,50, b1g1(6,9), b1g1(6,7,9), b1g1(6,8,9), d(f(1,4,5)), d(f(1,5,6)), d(f(1,6,7)), d(f(1,7,8)), d(f(1,8,9)), d(f(1,9,10)), b1g1(9,5,10), b1g1(9,4,10), b1g1(9,5,10), b1g1(9,6,10), b1g1(9,7,10), b1g1(9,8,10), b1g1(9,9,10),
d(f(1,1,2)), d(f(1,2,3)), d(f(1,3,4)), d(f(1,4,5)), d(f(1,5,6)), d(f(1,6,7)), d(f(1,7,8)), d(f(1,8,9)), d(f(1,9,10)), d(f(2,1,3)), d(f(2,2,4)), d(f(2,3,5)), d(f(2,4,6)),
d(f(2,5,7)), d(f(2,6,8)), d(f(2,7,9)), d(f(2,8,10)), d(f(3,1,4)), d(f(3,1,5)), d(f(3,2,6)), d(f(3,3,7)), d(f(3,4,8)), d(f(3,5,9)), d(f(3,6,10)), d(f(3,7,11)), d(f(4,1,5)), d(f(4,1,6)), d(f(4,1,7)), d(f(4,1,8)), d(f(4,1,9)), d(f(4,2,6)), d(f(4,2,7)), d(f(4,2,8)), d(f(4,2,9)), d(f(4,3,10)), d(f(4,3,7)), d(f(4,3,8)), d(f(4,3,9)), d(f(4,3,10)), d(f(4,4,8)), d(f(4,4,9)), d(f(4,5,9)), d(f(4,5,10)), d(f(4,6,10)), d(f(5,1,6)), d(f(5,1,7)), d(f(5,1,8)), d(f(5,1,9)), d(f(5,1,10)), d(f(5,2,7)), d(f(5,2,8)), d(f(5,2,9)), d(f(5,2,10)), d(f(5,3,8)), d(f(5,3,9)), d(f(5,4,9)), d(f(5,4,10)), d(f(5,5,11)), d(f(5,6,11)), d(f(6,1,8)), d(f(6,1,9)), d(f(6,1,10)), d(f(6,2,9)), d(f(6,2,10)), d(f(6,3,10)), d(f(6,4,10)), d(f(6,5,10)), d(f(6,6,10)), d(f(6,7,10)), d(f(6,8,10)), d(f(6,9,10)), d(f(7,1,9)), d(f(7,1,10)), d(f(7,2,10)), d(f(7,3,10)), d(f(7,4,10)), d(f(7,5,10)), d(f(7,6,10)), d(f(7,7,10)), d(f(7,8,10)), d(f(7,9,10)), d(f(7,10,10)), d(f(8,1,9)), d(f(8,2,10)), d(f(8,3,10)), d(f(8,4,10)), d(f(8,5,10)), d(f(8,6,10)), d(f(8,7,10)), d(f(8,8,10)), d(f(8,9,10)), d(f(8,10,10)), d(f(9,1,10)), d(f(9,2,10)), d(f(9,3,10)), d(f(9,4,10)), d(f(9,5,10)), d(f(9,6,10)), d(f(9,7,10)), d(f(9,8,10)), d(f(9,9,10)), d(f(9,10,10)), d(f(10,1,10)), d(f(10,2,10)), d(f(10,3,10)), d(f(10,4,10)), d(f(10,5,10)), d(f(10,6,10)), d(f(10,7,10)), d(f(10,8,10)), d(f(10,9,10)), d(f(10,10,10)), d(f(11,1,10)), d(f(11,2,10)), d(f(11,3,10)), d(f(11,4,10)), d(f(11,5,10)), d(f(11,6,10)), d(f(11,7,10)), d(f(11,8,10)), d(f(11,9,10)), d(f(11,10,10)), d(f(12,1,10)), d(f(12,2,10)), d(f(12,3,10)), d(f(12,4,10)), d(f(12,5,10)), d(f(12,6,10)), d(f(12,7,10)), d(f(12,8,10)), d(f(12,9,10)), d(f(12,10,10)), d(f(13,1,10)), d(f(13,2,10)), d(f(13,3,10)), d(f(13,4,10)), d(f(13,5,10)), d(f(13,6,10)), d(f(13,7,10)), d(f(13,8,10)), d(f(13,9,10)), d(f(13,10,10)), d(f(14,1,10)), d(f(14,2,10)), d(f(14,3,10)), d(f(14,4,10)), d(f(14,5,10)), d(f(14,6,10)), d(f(14,7,10)), d(f(14,8,10)), d(f(14,9,10)), d(f(14,10,10)), d(f(15,1,10)), d(f(15,2,10)), d(f(15,3,10)), d(f(15,4,10)), d(f(15,5,10)), d(f(15,6,10)), d(f(15,7,10)), d(f(15,8,10)), d(f(15,9,10)), d(f(15,10,10)), d(f(16,1,10)), d(f(16,2,10)), d(f(16,3,10)), d(f(16,4,10)), d(f(16,5,10)), d(f(16,6,10)), d(f(16,7,10)), d(f(16,8,10)), d(f(16,9,10)), d(f(16,10,10)), d(f(17,1,10)), d(f(17,2,10)), d(f(17,3,10)), d(f(17,4,10)), d(f(17,5,10)), d(f(17,6,10)), d(f(17,7,10)), d(f(17,8,10)), d(f(17,9,10)), d(f(17,10,10)), d(f(18,1,10)), d(f(18,2,10)), d(f(18,3,10)), d(f(18,4,10)), d(f(18,5,10)), d(f(18,6,10)), d(f(18,7,10)), d(f(18,8,10)), d(f(18,9,10)), d(f(18,10,10)), d(f(19,1,10)), d(f(19,2,10)), d(f(19,3,10)), d(f(19,4,10)), d(f(19,5,10)), d(f(19,6,10)), d(f(19,7,10)), d(f(19,8,10)), d(f(19,9,10)), d(f(19,10,10)), d(f(20,1,10)), d(f(20,2,10)), d(f(20,3,10)), d(f(20,4,10)), d(f(20,5,10)), d(f(20,6,10)), d(f(20,7,10)), d(f(20,8,10)), d(f(20,9,10)), d(f(20,10,10)), d(f(21,1,10)), d(f(21,2,10)), d(f(21,3,10)), d(f(21,4,10)), d(f(21,5,10)), d(f(21,6,10)), d(f(21,7,10)), d(f(21,8,10)), d(f(21,9,10)), d(f(21,10,10)), d(f(22,1,10)), d(f(22,2,10)), d(f(22,3,10)), d(f(22,4,10)), d(f(22,5,10)), d(f(22,6,10)), d(f(22,7,10)), d(f(22,8,10)), d(f(22,9,10)), d(f(22,10,10)), d(f(23,1,10)), d(f(23,2,10)), d(f(23,3,10)), d(f(23,4,10)), d(f(23,5,10)), d(f(23,6,10)), d(f(23,7,10)), d(f(23,8,10)), d(f(23,9,10)), d(f(23,10,10)), d(f(24,1,10)), d(f(24,2,10)), d(f(24,3,10)), d(f(24,4,10)), d(f(24,5,10)), d(f(24,6,10)), d(f(24,7,10)), d(f(24,8,10)), d(f(24,9,10)), d(f(24,10,10)), d(f(25,1,10)), d(f(25,2,10)), d(f(25,3,10)), d(f(25,4,10)), d(f(25,5,10)), d(f(25,6,10)), d(f(25,7,10)), d(f(25,8,10)), d(f(25,9,10)), d(f(25,10,10)), d(f(26,1,10)), d(f(26,2,10)), d(f(26,3,10)), d(f(26,4,10)), d(f(26,5,10)), d(f(26,6,10)), d(f(26,7,10)), d(f(26,8,10)), d(f(26,9,10)), d(f(26,10,10)), d(f(27,1,10)), d(f(27,2,10)), d(f(27,3,10)), d(f(27,4,10)), d(f(27,5,10)), d(f(27,6,10)), d(f(27,7,10)), d(f(27,8,10)), d(f(27,9,10)), d(f(27,10,10)), d(f(28,1,10)), d(f(28,2,10)), d(f(28,3,10)), d(f(28,4,10)), d(f(28,5,10)), d(f(28,6,10)), d(f(28,7,10)), d(f(28,8,10)), d(f(28,9,10)), d(f(28,10,10)), d(f(29,1,10)), d(f(29,2,10)), d(f(29,3,10)), d(f(29,4,10)), d(f(29,5,10)), d(f(29,6,10)), d(f(29,7,10)), d(f(29,8,10)), d(f(29,9,10)), d(f(29,10,10)), d(f(30,1,10)), d(f(30,2,10)), d(f(30,3,10)), d(f(30,4,10)), d(f(30,5,10)), d(f(30,6,10)), d(f(30,7,10)), d(f(30,8,10)), d(f(30,9,10)), d(f(30,10,10)), d(f(31,1,10)), d(f(31,2,10)), d(f(31,3,10)), d(f(31,4,10)), d(f(31,5,10)), d(f(31,6,10)), d(f(31,7,10)), d(f(31,8,10)), d(f(31,9,10)), d(f(31,10,10)), d(f(32,1,10)), d(f(32,2,10)), d(f(32,3,10)), d(f(32,4,10)), d(f(32,5,10)), d(f(32,6,10)), d(f(32,7,10)), d(f(32,8,10)), d(f(32,9,10)), d(f(32,10,10)), d(f(33,1,10)), d(f(33,2,10)), d(f(33,3,10)), d(f(33,4,10)), d(f(33,5,10)), d(f(33,6,10)),
```