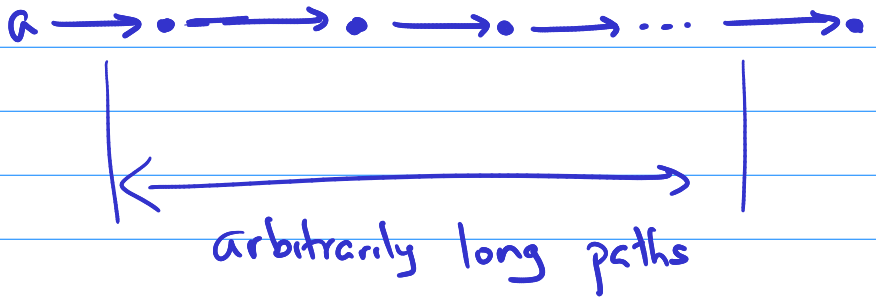
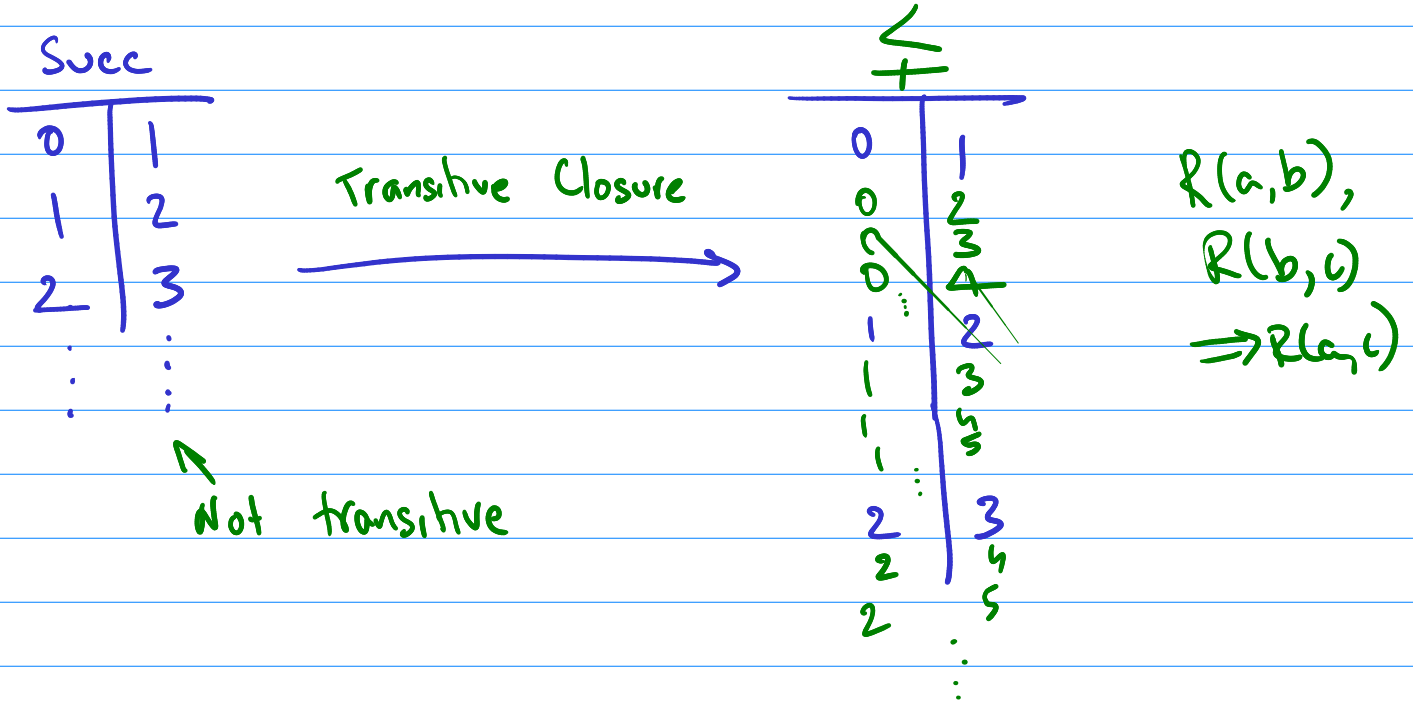
Graph Query Languages

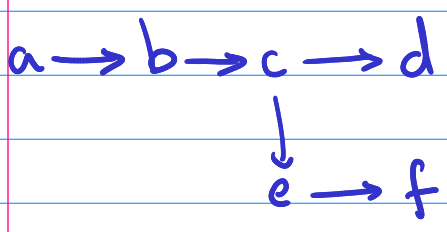
Recursion.

"All nodes which you can reach from a"





If you confine yourself to relational algebra, the best you can do is "bounded reachability".



Edge	
a	b
b	c
c	d
c	e
e	f

Edge⁺	
a	b
b	c
c	d
c	e
e	f
a	c
a	d
a	e
a	f
⋮	

---

① If $\exists\ a \longrightarrow b$ then $a \rightsquigarrow b$

If there is an edge from a to b then we can "reach" b from a

② If there is an edge from a to b & a path from b to c, then we can reach c from a.

Repeat these rules until nothing more can be derived.

Least Fixpoint

Edge(a, b)

Reachable(a, b)

Edge(a, b) Reachable(b, c)

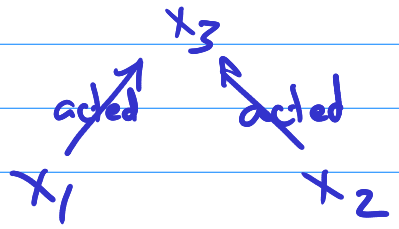
Reachable(a, c)

Repeat

until nothing changes

```
bool changed := true
while (changed) {
    changed := false
    ...
    if (——) changed := true
}
```

Task 1 : Find all x_1 x_2 x_3 s.t

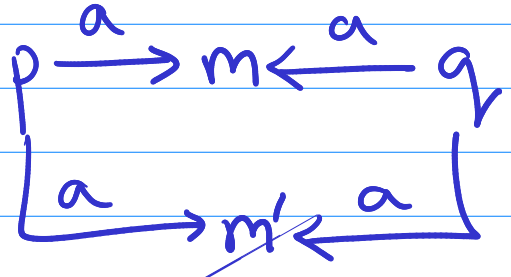


Homomorphism
SPARQL

Task 2 : Find all distinct x_1 x_2 x_3 s.t

Isomorphism
Neo4j

Complexity

Data complexity	Query complexity
	NP-complete
Polynomial time	[Isomorphism-based semantics]
	Subgraph isomorphism