New Results for the Complexity of Resilience for Binary Conjunctive Queries with Self-Joins

PODS – June 2020

Cibele Freire, Wolfgang Gatterbauer, Neil Immerman, Alexandra Meliou

Resilience

$q :- R(x, y), P(x, z)$ ~~TRUE~~ FALSE

R

	X	Y
~~r_1~~	~~1~~	~~3~~
r_2	4	2

P

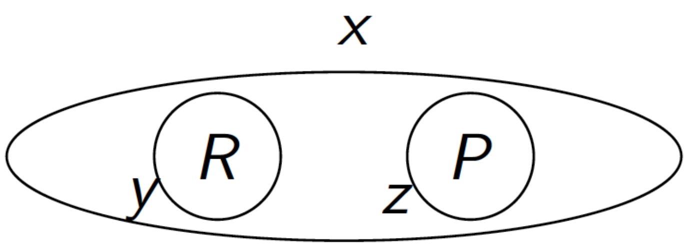
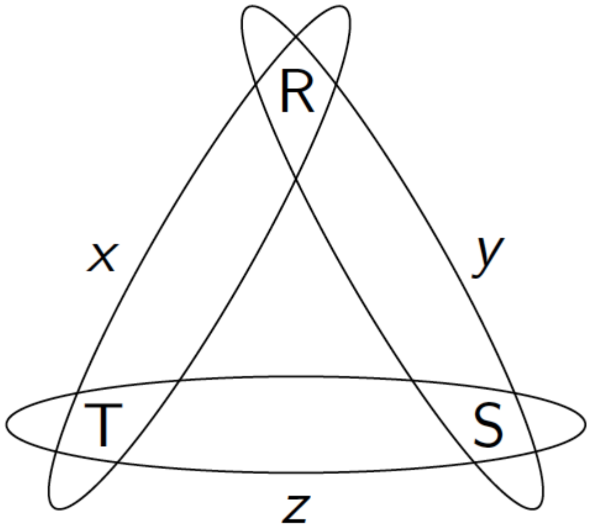
	X	Z
~~s_1~~	~~4~~	~~2~~
s_2	1	3
s_3	1	2

Closely related to *Deletion propagation with source-side effects*.

<u>*Data complexity*</u> of Resilience for conjunctive queries.

Self-join-free CQ

(prior results [PVLDB 2015])

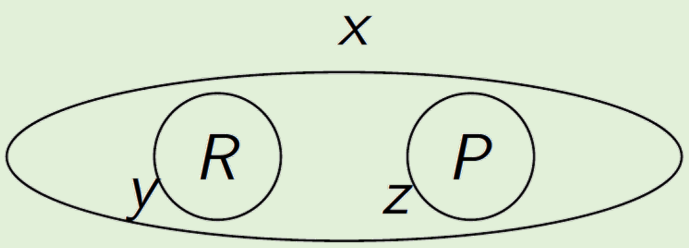
No triad	Triad
$q :- R(x, y), P(x, z)$ 	$q_{\triangle} :- R(x, y), S(y, z), T(z, x)$ 

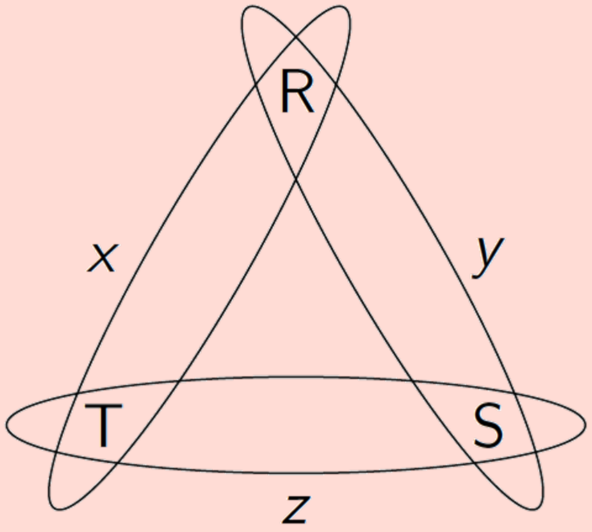
Definition (triad): A set of three atoms, $\{S_0; S_1; S_2\}$ such that for every pair i,j , there is a path from S_i to S_j that uses no variable occurring in the other atom.

Self-join-free CQ

(prior results [PVLDB 2015])

No triad	Triad
$q :- R(x, y), P(x, z)$	$q_\triangle :- R(x, y), S(y, z), T(z, x)$



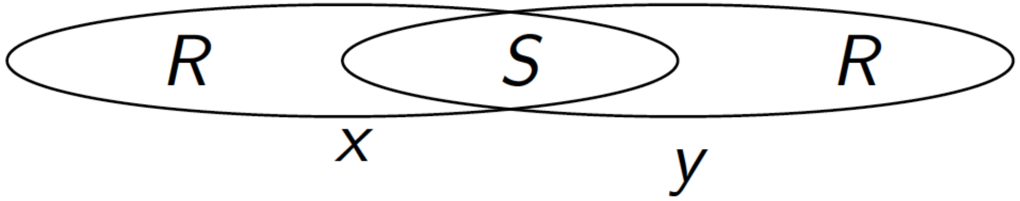


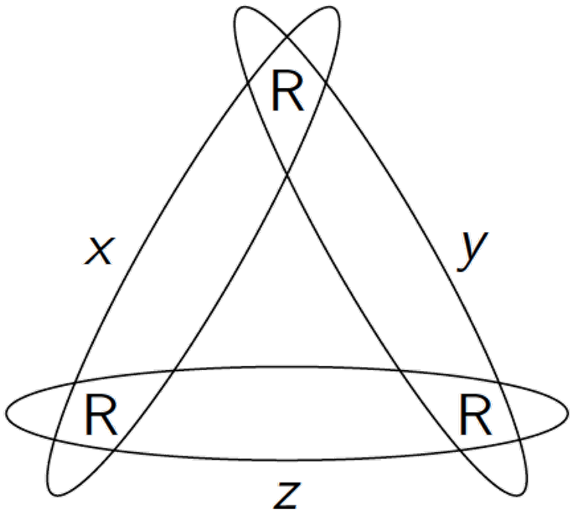
easy ← Dichotomy → hard

~~Self-join-free~~ CQ

Now with self-joins

No triad	Triad
$q_{\text{vc}} :- R(x), S(x,y), R(y)$	$q^{sj}_{\triangle} :- R(x,y), R(y,z), R(z,x)$





Lemma: RES(q_{vc}) is NP-complete. ✗

If q has a triad, then RES(q) is NP-complete. ✓

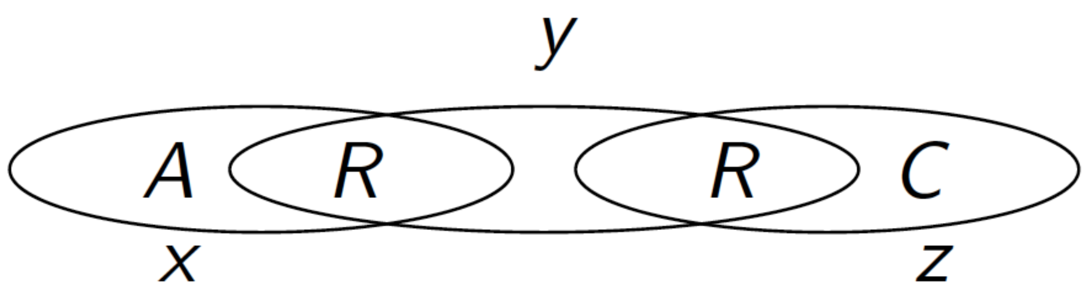
Overview for self-join case

- Triads still imply resilience is hard

- If no triads, then
  - Path
  - Chain
  - Confluence
  - Permutation

For a subclass of queries with self-joins

No triads



$q_{\text{conf}} \coloneq A(x), R(x,y), R(z,y), C(z)$

$q^{\text{ac}}_{\text{chain}} \coloneq A(x), R(x,y), R(y,z), C(z)$

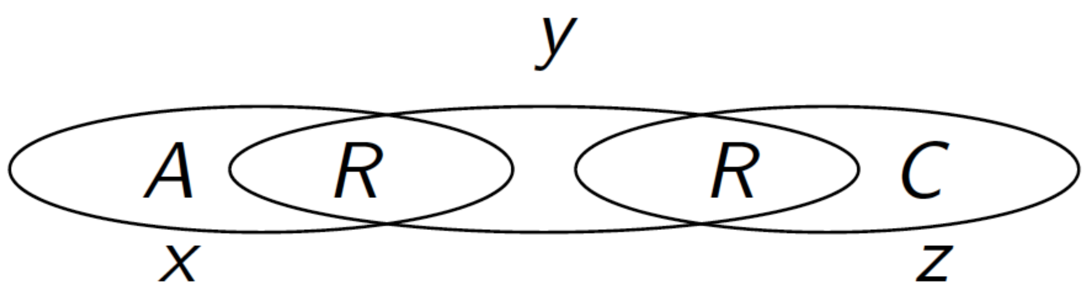
For self-join queries, dual hypergraph is not as useful.

Binary single-self-join queries (ssj-CQ)

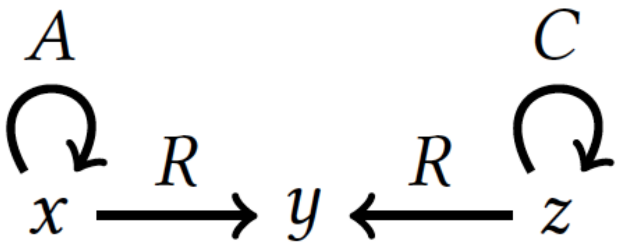
Moving forward we restrict the set of queries to have

- unary and binary relations; and
- only one relation can be part of a self-join, usually denoted as R.

Binary directed graphs

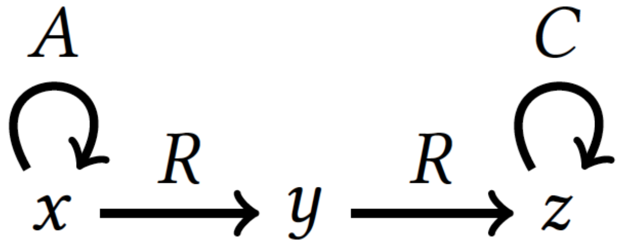


$$q_{\text{conf}} \text{ :- } A(x), R(x, y), R(z, y), C(z)$$



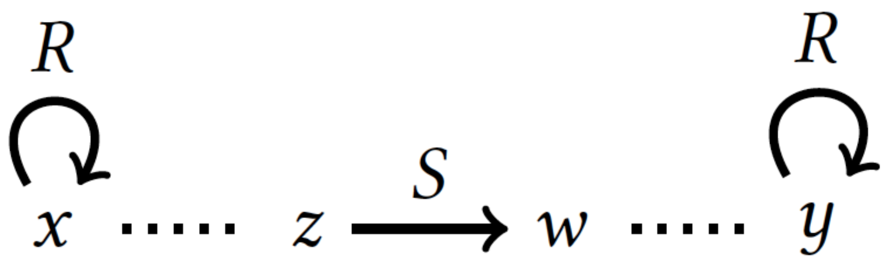
RES(q_{conf}) is in P

$$q^{\text{ac}}_{\text{chain}} \text{ :- } A(x), R(x, y), R(y, z), C(z)$$



RES(q_{chain}) is NP-complete

Paths



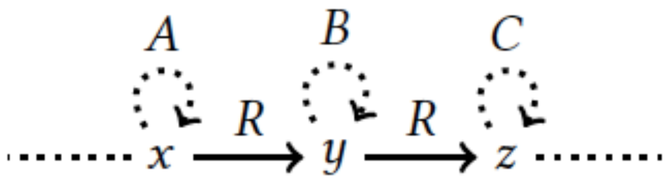
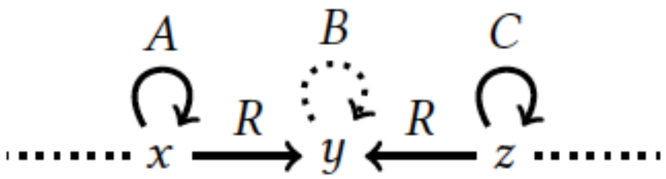
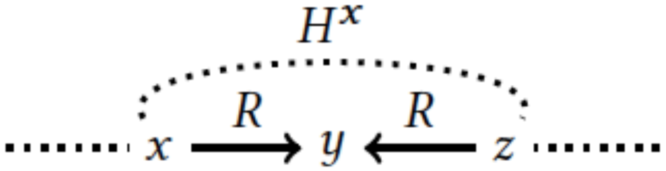
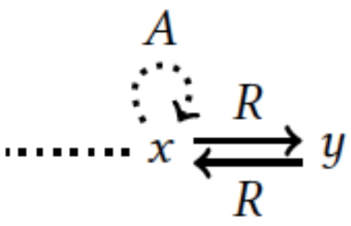
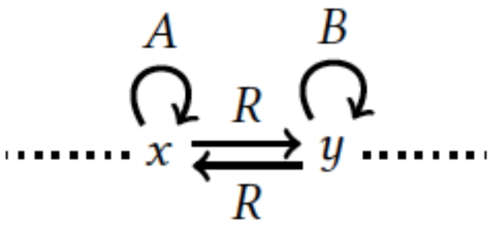
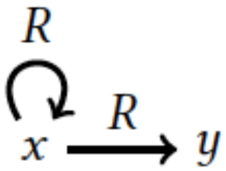
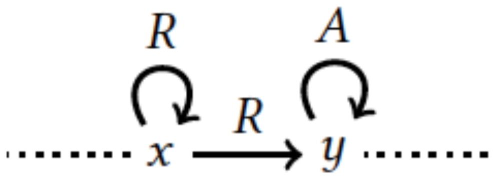
Unary Path

Binary Path

Theorem: Let q be a single-self-join query. If q has a unary or binary path, then RES(q) is NP-complete.

Proof: Reduction from RES(q_{vc}).

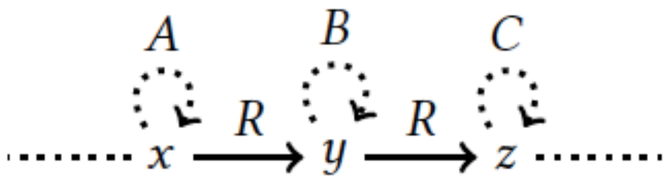
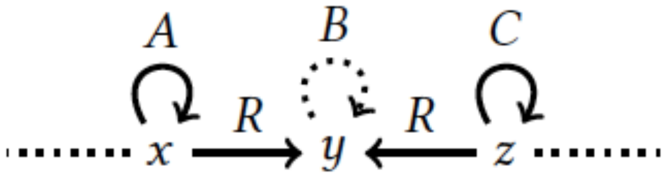
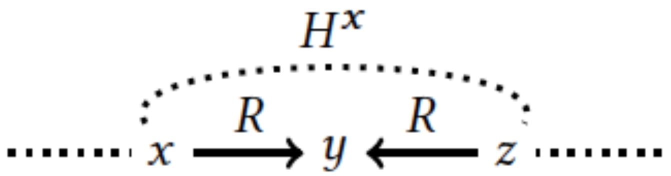
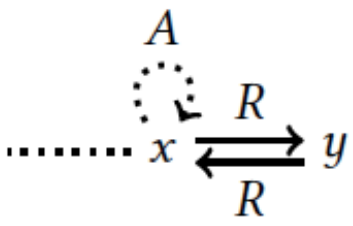
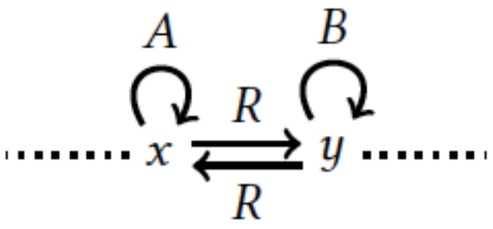
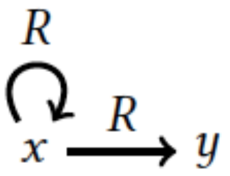
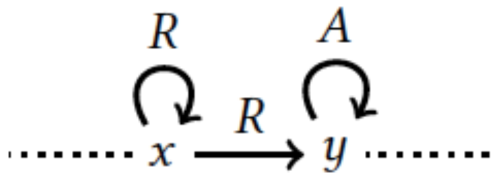
No triads, no paths

Binary Graph	PTIME cases	NP-hard cases
$x \xrightarrow{R} y \xrightarrow{R} z$	No PTIME case	
$x \xrightarrow{R} y \xleftarrow{R} z$		
$x \underset{R}{\overset{R}{\rightleftarrows}} y$		
		No NP-hard case

2-chain

Binary Graph	PTIME cases	NP-hard cases
$x \xrightarrow{R} y \xrightarrow{R} z$	No PTIME case	$\cdots\; \underset{x}{\overset{A}{\circlearrowright}} \xrightarrow{R} \underset{y}{\overset{B}{\circlearrowright}} \xrightarrow{R} \underset{z}{\overset{C}{\circlearrowright}} \;\cdots$
$x \xrightarrow{R} y \xleftarrow{R} z$	$\cdots\; \underset{x}{\overset{A}{\circlearrowright}} \xrightarrow{R} \underset{y}{\overset{B}{\circlearrowright}} \xleftarrow{R} \underset{z}{\overset{C}{\circlearrowright}} \;\cdots$	$\cdots\; \overbrace{x \xrightarrow{R} y \xleftarrow{R} z}^{H^x} \;\cdots$
$x \underset{R}{\overset{R}{\rightleftarrows}} y$	$\cdots\; \underset{x}{\overset{A}{\circlearrowright}} \underset{R}{\overset{R}{\rightleftarrows}} y$	$\cdots\; \underset{x}{\overset{A}{\circlearrowright}} \underset{R}{\overset{R}{\rightleftarrows}} \underset{y}{\overset{B}{\circlearrowright}} \;\cdots$
$\underset{x}{\overset{R}{\circlearrowright}} \xrightarrow{R} y$	$\cdots\; \underset{x}{\overset{R}{\circlearrowright}} \xrightarrow{R} \underset{y}{\overset{A}{\circlearrowright}} \;\cdots$	No NP-hard case

2-confluence

Binary Graph	PTIME cases	NP-hard cases
$x \xrightarrow{R} y \xrightarrow{R} z$	No PTIME case	
$x \xrightarrow{R} y \xleftarrow{R} z$		
$x \underset{R}{\overset{R}{\rightleftarrows}} y$		
		No NP-hard case

2-permutation

Binary Graph	PTIME cases	NP-hard cases
$x \xrightarrow{R} y \xrightarrow{R} z$	No PTIME case	$\cdots\, \overset{\overset{A}{\circlearrowright}}{x} \xrightarrow{R} \overset{\overset{B}{\circlearrowright}}{y} \xrightarrow{R} \overset{\overset{C}{\circlearrowright}}{z} \,\cdots$
$x \xrightarrow{R} y \xleftarrow{R} z$	$\cdots\, \overset{\overset{A}{\circlearrowright}}{x} \xrightarrow{R} \overset{\overset{B}{\circlearrowright}}{y} \xleftarrow{R} \overset{\overset{C}{\circlearrowright}}{z} \,\cdots$	$\cdots\, \overbrace{x \xrightarrow{R} y \xleftarrow{R} z}^{H^x} \,\cdots$
$x \underset{R}{\overset{R}{\rightleftarrows}} y$	$\cdots\, \overset{\overset{A}{\circlearrowright}}{x} \underset{R}{\overset{R}{\rightleftarrows}} y$	$\cdots\, \overset{\overset{A}{\circlearrowright}}{x} \underset{R}{\overset{R}{\rightleftarrows}} \overset{\overset{B}{\circlearrowright}}{y} \,\cdots$
$\overset{\overset{R}{\circlearrowright}}{x} \xrightarrow{R} y$	$\cdots\, \overset{\overset{R}{\circlearrowright}}{x} \xrightarrow{R} \overset{\overset{A}{\circlearrowright}}{y} \,\cdots$	No NP-hard case

Variable repetition

Binary Graph	PTIME cases	NP-hard cases
$x \xrightarrow{R} y \xrightarrow{R} z$	Using network flow! No PTIME case	$\begin{tikzcd} x \arrow[loop above, dotted, "A"] \arrow[r, "R"] & y \arrow[loop above, dotted, "B"] \arrow[r, "R"] & z \arrow[loop above, dotted, "C"] \end{tikzcd}$
$x \xrightarrow{R} y \xleftarrow{R} z$	$\begin{tikzcd} x \arrow[loop above, "A"] \arrow[r, "R"] & y \arrow[loop above, dotted, "B"] & z \arrow[loop above, "C"] \arrow[l, "R"'] \end{tikzcd}$	$\begin{tikzcd} x \arrow[r, "R"] \arrow[rr, bend left, dotted, dash, "H^x"] & y & z \arrow[l, "R"'] \end{tikzcd}$
$\begin{tikzcd} x \arrow[r, shift left, "R"] & y \arrow[l, shift left, "R"] \end{tikzcd}$	$\begin{tikzcd} x \arrow[loop above, dotted, "A"] \arrow[r, shift left, "R"] & y \arrow[l, shift left, "R"] \end{tikzcd}$	$\begin{tikzcd} x \arrow[loop above, "A"] \arrow[r, shift left, "R"] & y \arrow[loop above, "B"] \arrow[l, shift left, "R"] \end{tikzcd}$
$\begin{tikzcd} x \arrow[loop above, "R"] \arrow[r, "R"] & y \end{tikzcd}$	$\begin{tikzcd} x \arrow[loop above, "R"] \arrow[r, "R"] & y \arrow[loop above, "A"] \end{tikzcd}$	No NP-hard case

Dichotomy for ssj-CQ with 2 R-atoms

Theorem: Consider q an ssj-CQ, with at most two occurrences of the self-join relation. If q has any of the following

- triad
- path
- chain
- confluence with exogenous path
- bounded permutation

then RES(q) is NP-complete. Otherwise, RES(q) is in P.

Unifying hardness criterion

Many cases to consider even in very restrictive settings. However:

- all polynomial cases are solved with a reduction to network flow
- there are common patterns in the different reductions we defined

Independent Join Paths: property of a database with relation to a query. If a query admits such database, we conjecture that RES(q) is NP-complete.

Thanks!
Questions?