A Dichotomy Structure Theorem for the Resilience Problem

Cibele Freire

College of Information and Computer Sciences
University of Massachusetts Amherst

joint with work Wolfgang Gatterbauer & Neil Immerman & Alexandra Meliou
Imagine a world where

We could selectively delete memories from the brain and by doing that we could change one’s opinions or beliefs. Would you like to know how difficult it would be to choose what memories to delete?

This is what we investigate!

Although, in a simpler, safer and more ethical scenario...

Brains = Databases
Opinions, beliefs = Query answer
Delete memories = Delete tuples from the database
Imagine a world where..

We could selectively delete memories from the brain and by doing that we could change one’s opinions or beliefs.
Imagine a world where..

We could selectively delete memories from the brain and by doing that we could change one’s opinions or beliefs.

Would you like to know how difficult it would be to choose what memories to delete?
Imagine a world where...

We could selectively delete memories from the brain and by doing that we could change one’s opinions or beliefs.

Would you like to know how difficult it would be to choose what memories to delete?

This is what we investigate!
Imagine a world where..

We could selectively delete memories from the brain and by doing that we could change one’s opinions or beliefs.

Would you like to know how difficult it would be to choose what memories to delete?

This is what we investigate! Although, in a simpler, safer and more ethical scenario...
Imagine a world where..

We could selectively delete memories from the brain and by doing that we could change one’s opinions or beliefs.

Would you like to know how difficult it would be to choose what memories to delete?

This is what we investigate! Although, in a simpler, safer and more ethical scenario...

- Brains = Databases
- Opinions, beliefs = Query answer
- Delete memories = Delete tuples from the database
Resilience problem

Definition (Resilience)

Given a query $q$ and database $D$, we say that $(D, k) \in \text{RES}(q)$ if and only if $D \models q$ and there exists some $\Gamma \subseteq D$ such that $D - \Gamma \not\models q$ and $|\Gamma| \leq k$. 

What is the complexity of $\text{RES}(q_{vc})$?

This is exactly vertex cover.

Lemma $\text{RES}(q_{vc})$ is NP-complete.

Conjunctive queries without self-joins

Deleted tuples

Contingency set $\Gamma$

Endogenous (changeable) vs exogenous (non changeable) tuples
Resilience problem

Definition (Resilience)

Given a query $q$ and database $D$, we say that $(D, k) \in \text{RES}(q)$ if and only if $D \models q$ and there exists some $\Gamma \subseteq D$ such that $D - \Gamma \not\models q$ and $|\Gamma| \leq k$.

$$q_{vc} \leftarrow V(x), E(x, y), V(y)$$
Resilience problem

Definition (Resilience)

Given a query $q$ and database $D$, we say that $(D, k) \in \text{RES}(q)$ if and only if $D \models q$ and there exists some $\Gamma \subseteq D$ such that $D - \Gamma \not\models q$ and $|\Gamma| \leq k$.

$q_{vc} : \leftarrow V(x), E(x, y), V(y)$

What is the complexity of $\text{RES}(q_{vc})$?
Resilience problem

Definition (Resilience)

Given a query $q$ and database $D$, we say that $(D, k) \in \text{RES}(q)$ if and only if $D \models q$ and there exists some $\Gamma \subseteq D$ such that $D - \Gamma \not\models q$ and $|\Gamma| \leq k$.

$q_{vc} :\leftarrow V(x), E(x, y), V(y)$

What is the complexity of $\text{RES}(q_{vc})$? This is exactly vertex cover.
**Resilience problem**

**Definition (Resilience)**

Given a query $q$ and database $D$, we say that $(D, k) \in RES(q)$ if and only if $D \models q$ and there exists some $\Gamma \subseteq D$ such that $D - \Gamma \not\models q$ and $|\Gamma| \leq k$.

$$q_{vc} :\leftarrow V(x), E(x, y), V(y)$$

What is the complexity of $RES(q_{vc})$? This is exactly vertex cover.

**Lemma**

$RES(q_{vc})$ is NP-complete.
Resilience problem

Definition (Resilience)

Given a query $q$ and database $D$, we say that $(D, k) \in \text{RES}(q)$ if and only if $D \models q$ and there exists some $\Gamma \subseteq D$ such that $D - \Gamma \not\models q$ and $|\Gamma| \leq k$.

$q_{vc} : \leftarrow V(x), E(x, y), V(y)$

What is the complexity of $\text{RES}(q_{vc})$? This is exactly vertex cover.

Lemma

$\text{RES}(q_{vc})$ is NP-complete.

- Conjunctive queries without self-joins
- Deleted tuples $\rightarrow$ Contingency set $\Gamma$
- Endogenous (changeable) vs exogenous (non changeable) tuples
Dual hypergraph

\[ q_\triangle \leftarrow R(x, y), S(y, z), T(z, x) \]
\[ q_T \leftarrow A(x), B(y), C(z), W(x, y, z) \]
Dual hypergraph

\[ q_\triangle : \neg R(x, y), S(y, z), T(z, x) \]

\[ q_T : \neg A(x), B(y), C(z), W(x, y, z) \]

Triangle query

Tripod query
Dual hypergraph

$q_\triangle := R(x, y), S(y, z), T(z, x)$  

$q_T := A(x), B(y), C(z), W(x, y, z)$

Triangle query

Tripod query

What is the complexity of resilience for those queries?
Dual hypergraph

\[ q_\triangle := R(x, y), S(y, z), T(z, x) \]

\[ q_T := A(x), B(y), C(z), W(x, y, z) \]

Triangle query

Tripod query

Lemma

\( \text{RES}(q_\triangle) \) and \( \text{RES}(q_T) \) are NP-complete.
RES($q_\triangle$) is NP-complete.

3SAT $\leq$ RES($q_\triangle$). Let $\psi = C_1 \land \cdots \land C_m$ be a 3-CNF formula, var($\psi$) = \{v_1, \ldots, v_n\}

Map $\psi \mapsto (D_\psi, k_\psi)$ s.t. $\psi \in 3\text{SAT} \iff (D_\psi, k_\psi) \in \text{RES}(q_\triangle)$
RES($q_\triangle$) is NP-complete.

3SAT $\leq$ RES($q_\triangle$). Let $\psi = C_1 \land \cdots \land C_m$ be a 3-CNF formula, $\text{var}(\psi) = \{v_1, \ldots, v_n\}$

Map $\psi \mapsto (D_\psi, k_\psi)$ s.t. $\psi \in 3\text{SAT} \iff (D_\psi, k_\psi) \in \text{RES}(q_\triangle)$

$q_\triangle := R(x, y), S(y, z), T(z, x)$

$(D_\psi, k_\psi) \in \text{RES}(q_\triangle) \iff \exists \Gamma (|\Gamma| = k_\psi) \land (D_\psi - \Gamma)$ has no
RES($q_{\Delta}$) is NP-complete.

3SAT $\leq$ RES($q_{\Delta}$). Let $\psi = C_1 \land \cdots \land C_m$ be a 3-CNF formula, $\text{var}(\psi) = \{v_1, \ldots, v_n\}$

Map $\psi \mapsto (D_\psi, k_\psi)$ s.t. $\psi \in 3\text{SAT} \iff (D_\psi, k_\psi) \in \text{RES}(q_{\Delta})$

$$q_{\Delta} := R(x, y), S(y, z), T(z, x)$$

$$(D_\psi, k_\psi) \in \text{RES}(q_{\Delta}) \iff \exists \Gamma (|\Gamma| = k_\psi) \land (D_\psi - \Gamma) \text{ has no}$$

$D_\psi$ has one circular gadget $G_i$ for each variable $v_i$. 
For each clause, e.g., $C_j = (v_1 \lor \overline{v}_2 \lor v_3)$, pick the $j$th occurrences of $v_1 \in G_1$, $\overline{v}_2 \in G_2$ and $v_3 \in G_3$. Identify head of $v_1$ with tail of $\overline{v}_2$, head of $\overline{v}_2$ with tail of $v_3$, head of $v_3$ with tail of $v_1$
For each clause, e.g., $C_j = (v_1 \lor \overline{v_2} \lor v_3)$, pick the $j$th occurrences of $v_1 \in G_1$, $\overline{v_2} \in G_2$ and $v_3 \in G_3$. Identify head of $v_1$ with tail of $\overline{v_2}$, head of $\overline{v_2}$ with tail of $v_3$, head of $v_3$ with tail of $v_1$.

This new RGB triangle is automatically removed iff one of the literals in $C_j$ is chosen true.
\( \text{RES}(q_T) \) is \( \text{NP} \)-complete.

\[
q_T : \neg A(x), B(y), C(z), W(x, y, z)
\]
\( \text{RES}(q_T) \) is NP-complete.

\[
q_T :\ A(x), B(y), C(z), W(x, y, z)
\]

\[\text{var}(A) \subseteq \text{var}(W).\]
RES\( (q_T) \) is \( \text{NP} \)-complete.

\[ q_T := A(x), B(y), C(z), W(x, y, z) \]

\( \text{var}(A) \subseteq \text{var}(W) \).

\( A \) dominates \( W \).
\( \text{RES}(q_T) \) is NP-complete.

\[
q_T := A(x), B(y), C(z), W(x, y, z)
\]

\( \text{var}(A) \subseteq \text{var}(W). \)

\( A \) dominates \( W. \)

**Proposition**

If \( A \) dominates \( W \), then we can assume that \( W \) is exogenous, i.e., rewrite as \( W^x \), tuples from \( W^x \) are never chosen.
\( \text{RES}(q_T) \) is NP-complete.

\[
q_T := A(x), B(y), C(z), W(x, y, z)
\]

\[\text{var}(A) \subseteq \text{var}(W).\]

\( A \) dominates \( W \).

**Proposition**

If \( A \) dominates \( W \), then we can assume that \( W \) is exogenous, i.e., rewrite as \( W^x \), tuples from \( W^x \) are never chosen.

\[
q_T := A(x), B(y), C(z), W^x(x, y, z)
\]
RES\( (q_T) \) is NP-complete.

\[
q_\triangle \ := \ R(x, y), \ S(y, z), \ T(z, x) \\
q_T \ := \ A(x), \ B(y), \ C(z), \ W^x(x, y, z)
\]

Show \( \text{RES}(q_\triangle) \leq \text{RES}(q_T) \)
RES($q_T$) is NP-complete.

$$q_\triangle := R(x, y), S(y, z), T(z, x)$$
$$q_T := A(x), B(y), C(z), W^x(x, y, z)$$

Show $\text{RES}(q_\triangle) \leq \text{RES}(q_T)$

Let $(D, k)$ be an instance of $\text{RES}(q_\triangle)$.

$(D, k) \mapsto (D', k)$  \quad $D' \overset{\text{def}}{=} (A, B, C, W^x)$
RES($q_T$) is NP-complete.

$$q_\triangle := R(x, y), S(y, z), T(z, x)$$
$$q_T := A(x), B(y), C(z), W^x(x, y, z)$$

Show $RES(q_\triangle) \leq RES(q_T)$

Let $(D, k)$ be an instance of $RES(q_\triangle)$.

$$(D, k) \mapsto (D', k) \quad D' \overset{\text{def}}{=} (A, B, C, W^x)$$

$$A = \{\langle ab \rangle \mid R(a, b) \in D\}$$
$$B = \{\langle bc \rangle \mid S(b, c) \in D\}$$
$$C = \{\langle ca \rangle \mid T(c, a) \in D\}$$
$$W^x = \{((\langle ab \rangle, \langle bc \rangle, \langle ca \rangle) \mid a, b, c \in \text{dom}(D)\}$$
RES\((q_T)\) is NP-complete.

\[
q_\triangle \ := \ R(x, y), S(y, z), T(z, x)
\]

\[
q_T \ := \ A(x), B(y), C(z), W^x(x, y, z)
\]

Show \(RES(q_\triangle) \leq RES(q_T)\)

Let \((D, k)\) be an instance of \(RES(q_\triangle)\).

\((D, k) \mapsto (D', k) \quad D' \overset{def}{=} (A, B, C, W^x)\)

\[
A = \{ \langle ab \rangle \mid R(a, b) \in D \}
\]

\[
B = \{ \langle bc \rangle \mid S(b, c) \in D \}
\]

\[
C = \{ \langle ca \rangle \mid T(c, a) \in D \}
\]

\[
W^x = \{ (\langle ab \rangle, \langle bc \rangle, \langle ca \rangle) \mid a, b, c \in \text{dom}(D) \}
\]

Claim

\((D, k) \in RES(q_\triangle) \iff (D', k) \in RES(q_T)\).\]
What do the triangle query and the tripod query have in common?
Triads

What do the triangle query and the tripod query have in common?

Definition (triad)

A **triad** is a set of three *endogenous atoms*, $\mathcal{T} = \{S_0, S_1, S_2\}$ 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 of $\mathcal{T}$. 
What do the triangle query and the tripod query have in common?

**Definition (triad)**

A **triad** is a set of three *endogenous atoms*, \( \mathcal{T} = \{S_0, S_1, S_2\} \) 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 of \( \mathcal{T} \).
What do the triangle query and the tripod query have in common?

Definition (triad)

A **triad** is a set of three *endogenous atoms*, $\mathcal{T} = \{S_0, S_1, S_2\}$ 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 of $\mathcal{T}$.

Where is the triad?
What do the triangle query and the tripod query have in common?

**Definition (triad)**

A **triad** is a set of three *endogenous atoms*, $\mathcal{T} = \{S_0, S_1, S_2\}$ 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 of $\mathcal{T}$.

Where is the triad? Path from $R$ to $S$. 

Cibele Freire (UMass Amherst)
What do the triangle query and the tripod query have in common?

Definition (triad)

A **triad** is a set of three *endogenous atoms*, \( T = \{ S_0, S_1, S_2 \} \) 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 of \( T \).
What do the triangle query and the tripod query have in common?

Definition (triad)

A **triad** is a set of three *endogenous atoms*, \( T = \{ S_0, S_1, S_2 \} \) 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 of \( T \).
Triads

What do the triangle query and the tripod query have in common?

Definition (triad)

A **triad** is a set of three *endogenous atoms*, $\mathcal{T} = \{S_0, S_1, S_2\}$ 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 of $\mathcal{T}$.

Where is the triad?

$\mathcal{T} = \{R, S, T\}$
Triads

What do the triangle query and the tripod query have in common?

Definition (triad)

A **triad** is a set of three *endogenous atoms*, \( T = \{ S_0, S_1, S_2 \} \) 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 of \( T \).

Where is the triad?
What do the triangle query and the tripod query have in common?

Definition (triad)

A triad is a set of three endogenous atoms, $\mathcal{T} = \{S_0, S_1, S_2\}$ 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 of $\mathcal{T}$.

Where is the triad?
Path from $A$ to $B$. 
Triads

What do the triangle query and the tripod query have in common?

Definition (triad)

A **triad** is a set of three *endogenous atoms*, \( \mathcal{T} = \{S_0, S_1, S_2\} \) 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 of \( \mathcal{T} \).

Where is the triad?
Path from \( A \) to \( C \).
Triads

What do the triangle query and the tripod query have in common?

Definition (triad)

A **triad** is a set of three *endogenous atoms*, $\mathcal{T} = \{S_0, S_1, S_2\}$ 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 of $\mathcal{T}$.

Where is the triad?
Path from $B$ to $C$. 
Triads

What do the triangle query and the tripod query have in common?

Definition (triad)

A **triad** is a set of three *endogenous atoms*, $\mathcal{T} = \{S_0, S_1, S_2\}$ 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 of $\mathcal{T}$.

Where is the triad?
$\mathcal{T} = \{A, B, C\}$
Lemma

If $q$ has a triad, then $\text{RES}(q)$ is $\text{NP}$-complete.
Lemma

If $q$ has a triad, then $\text{RES}(q)$ is NP-complete.

Important remark

It is easy to check if a query has a triad!
Triads $\rightarrow$ hardness

Lemma

*If* $q$ *has a triad, then* $\text{RES}(q)$ *is NP-complete.*

**Important remark**

It is easy to check if a query has a triad!

What if a query does not have a triad?
Definition
A query \( q \) is linear if its atoms may be arranged in a linear order such that each variable occurs in a contiguous sequence of atoms.

Fact [Meliou et.al., VLDB10]
For any linear sJ-free CQ \( q \), \( \text{RES} (q) \) is in \( \mathcal{P} \). (Reduction to network flow.)
A query $q$ is **linear** if its atoms may be arranged in a linear order such that each variable occurs in a contiguous sequence of atoms.

$\begin{align*}
q &:= A(x), R(x, y), S(y)
\end{align*}$
**Linear queries**

A query $q$ is **linear** if its atoms may be arranged in a linear order such that each variable occurs in a contiguous sequence of atoms.

**Fact [Meliou et.al., VLDB10]**

For any linear sj-free CQ $q$, $\text{RES}(q)$ is in $\mathcal{P}$. (Reduction to network flow.)
Is there a triad in the following query?

\[ q_{\text{rats}} : \neg A(x), R(x, y), S(y, z), T(z, x) \]
Is there a triad in the following query?

\[ q_{\text{rats}} : \neg A(x), R(x, y), S(y, z), T(z, x) \]

\[ \text{RES}(q_{\text{rats}}) \text{ is in P!} \]
Domination

$T$ and $R$ are dominated by $A$ in $q_{\text{rats}}$. This guarantees we don’t need tuples from $R$, $T$ in a minimum contingency set.
$T$ and $R$ are dominated by $A$ in $q_{rats}$. This guarantees we don’t need tuples from $R, T$ in a minimum contingency set.

$q_{rats} := A(x), R^x(x, y), S(y, z), T^x(z, x)$
Lemma

If $q$ is a query in normal form with no triads, we can transform it into a linear query $q'$, such that $\text{RES}(q) \leq \text{RES}(q')$. Therefore $\text{RES}(q)$ is in P.
Dichotomy for Resilience - sj-free CQ

Theorem

Let $q$ be an sj-free CQ and $\text{nf}(q)$ its normal form.

- If $\text{nf}(q)$ has a triad, then $\text{RES}(q)$ is NP-complete
- If $\text{nf}(q)$ does not have a triad, then $\text{RES}(q)$ is in P.
Adding functional dependencies

$q_T : \neg A(x), B(y), C(z), W(x, y, z)$
Adding functional dependencies

\[ q_T : A(x), B(y), C(z), W(x, y, z), \text{fd} = W : x \rightarrow y \]

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>W</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>X</td>
<td>Y</td>
<td>Z</td>
</tr>
<tr>
<td>a_1</td>
<td></td>
<td>1</td>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td>a_2</td>
<td></td>
<td>3</td>
<td>4</td>
<td>6</td>
</tr>
<tr>
<td>b_1</td>
<td>1</td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>b_2</td>
<td>2</td>
<td></td>
<td></td>
<td>2</td>
</tr>
<tr>
<td>c_1</td>
<td>3</td>
<td></td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>c_2</td>
<td>4</td>
<td></td>
<td></td>
<td>4</td>
</tr>
<tr>
<td>w_1</td>
<td>1</td>
<td>2</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>w_2</td>
<td>1</td>
<td>2</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>w_3</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>w_4</td>
<td>1</td>
<td>4</td>
<td>5</td>
<td></td>
</tr>
</tbody>
</table>
Adding functional dependencies

\[ q_T \models \neg A(x), B(y), C(z), W(x, y, z), \text{fd} = W : x \rightarrow y \]

- FDs constrain the databases we can consider.
Adding functional dependencies

\[ q_T :\neg A(x) \land B(y) \land C(z) \land W(x, y, z) \land \text{fd} = W : x \rightarrow y \]

- FDs constrain the databases we can consider
- FDs can reduce the complexity of resilience
Adding functional dependencies

Transform the query based on FDs
Adding functional dependencies

Transform the query based on FDs \(\rightarrow\) induced rewrites
Adding functional dependencies

Transform the query based on FDs → induced rewrites

\[ q_T = A(x), B(y), C(z), W(x, y, z), W : x \rightarrow y \]

\[ q^*_T = A'(x, y), B(y), C(z), W(x, y, z), W : x \rightarrow y \]

\[ q^*_T = A'(x, y), B(y), C(z), W(x, y, z) \]
Adding functional dependencies

Transform the query based on FDs → induced rewrites

\[ q_T = A(x), B(y), C(z), W(x, y, z), W : x \rightarrow y \]

\[ q^*_T = A'(x, y), B(y), C(z), W(x, y, z), W : x \rightarrow y \]

\[ q^*_T = A'(x, y), B(y), C(z), W(x, y, z) \]

\[ \text{RES}(q_T, \varphi) \text{ is in } \mathbb{P}. \]
Adding functional dependencies

**Induced rewrites**

Let \( q \) be a query and \( \overline{v} \rightarrow u \in \Phi \) be a functional dependency. We write \((q; \Phi) \rightsquigarrow (q'; \Phi)\) to mean that \( q' \) is the result of adding the dependent variable \( u \) to some relation that contains all the determinant variables \( \overline{v} \). After applying all possible rewrites, we obtain query \( q^* \), which we call **closed query**.
Adding functional dependencies

Induced rewrites

Let $q$ be a query and $\overline{v} \rightarrow u \in \Phi$ be a functional dependency. We write $(q; \Phi) \leadsto (q'; \Phi)$ to mean that $q'$ is the result of adding the dependent variable $u$ to some relation that contains all the determinant variables $\overline{v}$. After applying all possible rewrites, we obtain query $q^*$, which we call closed query.

Lemma

Let $q^*$ be $q$ after all possible induced rewrites have been applied. Then $\text{RES}(q; \Phi) \equiv \text{RES}(q^*; \Phi) \equiv \text{RES}(q^*)$. 
Theorem

Let \((q; \Phi)\) be a sj-free CQ with functional dependencies. Let \((q^*, \Phi)\) be its closure under induced rewrites, and such that all dominated atoms of \(q^*\) are exogenous.

- If \(q^*\) has a triad then \(\text{RES}(q; \Phi)\) is \(\text{NP}\)-complete.
- If \(q^*\) does not have a triad, then \(\text{RES}(q; \Phi)\) is in \(\text{P}\).
Future Directions

- Resilience for CQ with self-joins - $(q_{vc})$
- Deletion propagation: view side-effects for CQ with self-joins
  - Dichotomy results for CQ without self-joins [Kimelfeld et.al., PODS11]
  - Extended to functional dependencies [Kimelfeld, PODS12]
- Characterize the complexity of the parts of the problem that are in $P$, cf. [Allender, et. al.]