Uniform Reliability of Self-Join-Free Conjunctive Queries

Antoine Amarilli\textsuperscript{1}, Benny Kimelfeld\textsuperscript{2}

March 11, 2021

\textsuperscript{1}Télécom Paris

\textsuperscript{2}Technion
Uncertain data and tuple-independent databases (TID)

- We consider data in the relational model on which we have uncertainty
- Simplest uncertainty model: tuple-independent databases (TID)

<table>
<thead>
<tr>
<th>Class</th>
<th>Room</th>
<th>Date</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>Databases 1</td>
<td>101</td>
<td>2021-03-26</td>
<td>80%</td>
</tr>
<tr>
<td>Databases 2</td>
<td>101</td>
<td>2021-04-02</td>
<td>70%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Date</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>2021-03-26</td>
<td>20%</td>
</tr>
<tr>
<td>2021-04-02</td>
<td>40%</td>
</tr>
</tbody>
</table>
Uncertain data and tuple-independent databases (TID)

- We consider data in the **relational model** on which we have **uncertainty**
- Simplest uncertainty model: **tuple-independent databases (TID)**

<table>
<thead>
<tr>
<th>Classes</th>
<th>Room</th>
<th>Date</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>Databases 1</td>
<td>101</td>
<td>2021-03-26</td>
<td>80%</td>
</tr>
<tr>
<td>Databases 2</td>
<td>101</td>
<td>2021-04-02</td>
<td>70%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Lockdowns</th>
<th>Date</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2021-03-26</td>
<td>20%</td>
</tr>
<tr>
<td></td>
<td>2021-04-02</td>
<td>40%</td>
</tr>
</tbody>
</table>

- Semantics:
  - Every tuple is annotated with a **probability**
  - We assume that all tuples are **independent**
  - A TID concisely represents a **probability distribution** over the **subinstances**
We consider data in the relational model on which we have uncertainty.

Simplest uncertainty model: tuple-independent databases (TID)

<table>
<thead>
<tr>
<th>Class</th>
<th>Room</th>
<th>Date</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>Databases 1</td>
<td>101</td>
<td>2021-03-26</td>
<td>80%</td>
</tr>
<tr>
<td>Databases 2</td>
<td>101</td>
<td>2021-04-02</td>
<td>70%</td>
</tr>
</tbody>
</table>

Semantics:
- Every tuple is annotated with a probability
- We assume that all tuples are independent
- A TID concisely represents a probability distribution over the subinstances

Warning: we only use the TID model to show theoretical results :)
Probabilistic query evaluation (PQE)

- We consider **conjunctive queries** (CQs), which we assume to be **Boolean**
  - \( Q : \exists c r d \text{ Classes}(c, r, d) \land \text{Lockdowns}(d) \)
Probabilistic query evaluation (PQE)

- We consider **conjunctive queries** (CQs), which we assume to be **Boolean**
  - \( Q : \exists c r d \text{ Classes}(c, r, d) \land \text{Lockdowns}(d) \)

- **Semantics** of \( Q \) on a TID: compute the **probability** that \( Q \) is true

\[ \text{PQE}(Q) \]

- Input:
  - a TID \( I \)
- Output:
  - the total probability of the subinstances of \( I \) where \( Q \) is true

\( \text{PQE}(Q) \) can always be solved by looking at all subinstances (exponential)

\( \rightarrow \) When can we achieve a better complexity?
Probabilistic query evaluation (PQE)

• We consider **conjunctive queries** (CQs), which we assume to be **Boolean**
  
  - $Q : \exists c r d \text{ Classes}(c, r, d) \land \text{Lockdowns}(d)$

• **Semantics** of $Q$ on a TID: compute the **probability** that $Q$ is true

• Formally, problem **PQE($Q$)** for a fixed CQ $Q$:
  
  - **Input:** a TID $I$
  - **Output:** the total probability of the subinstances of $I$ where $Q$ is true
Probabilistic query evaluation (PQE)

- We consider **conjunctive queries** (CQs), which we assume to be **Boolean**
  - \( Q : \exists c \, r \, d \, \text{Classes}(c, r, d) \land \text{Lockdowns}(d) \)

- **Semantics** of \( Q \) on a TID: compute the **probability** that \( Q \) is true

- Formally, problem \( \text{PQE}(Q) \) for a fixed CQ \( Q \):
  - **Input**: a TID \( I \)
  - **Output**: the total probability of the subinstances of \( I \) where \( Q \) is true

- We can always solve \( \text{PQE}(Q) \) by looking at all subinstances (exponential)
Probabilistic query evaluation (PQE)

- We consider **conjunctive queries** (CQs), which we assume to be **Boolean**
  - \( Q : \exists c \quad r \quad d \text{ Classes}(c, r, d) \quad \land \text{ Lockdowns}(d) \)

- **Semantics** of \( Q \) on a TID: compute the **probability** that \( Q \) is true

- Formally, problem **PQE(\( Q \))** for a fixed CQ \( Q \):
  - **Input:** a TID \( I \)
  - **Output:** the total probability of the subinstances of \( I \) where \( Q \) is true

- We can always solve **PQE(\( Q \))** by looking at all subinstances (exponential)

→ **When can we achieve a better complexity?**
Existing results

- Complexity of PQE shown in [Dalvi and Suciu, 2007] for **self-join-free CQs** (SJFCQs)
  - A CQ is **self-join-free** if no relation symbol is repeated
- Later extended to **unions of conjunctive queries** [Dalvi and Suciu, 2012]
Existing results

• Complexity of PQE shown in [Dalvi and Suciu, 2007] for **self-join-free CQs** (SJFCQs)
  • A CQ is **self-join-free** if no relation symbol is repeated
• Later extended to **unions of conjunctive queries** [Dalvi and Suciu, 2012]

In this work we stick to the result on SJFCQs:

**Theorem [Dalvi and Suciu, 2007]**

Let $Q$ be a SJFCQ. Then:

• Either $Q$ is **hierarchical** and $\text{PQE}(Q)$ is in $\text{PTIME}$
• Or $Q$ is **not hierarchical** and $\text{PQE}(Q)$ is $\#P$-hard
Existing results

- Complexity of PQE shown in [Dalvi and Suciu, 2007] for self-join-free CQs (SJFCQs)
  - A CQ is self-join-free if no relation symbol is repeated
- Later extended to unions of conjunctive queries [Dalvi and Suciu, 2012]

In this work we stick to the result on SJFCQs:

**Theorem [Dalvi and Suciu, 2007]**


definition of SJFCQ

\[
\text{Let } Q \text{ be a SJFCQ. Then:}
\]

- Either \( Q \) is hierarchical and \( \text{PQE}(Q) \) is in \( \text{PTIME} \)
- Or \( Q \) is not hierarchical and \( \text{PQE}(Q) \) is \( \#P \)-hard

What is this class of hierarchical CQs?
Hierarchical CQs

For a CQ $Q$, write $\text{atoms}(x)$ for the set of atoms where $x$ appears

- A CQ is \textit{hierarchical} if for every variables $x$ and $y$
  - Either $\text{atoms}(x)$ and $\text{atoms}(y)$ are \textit{disjoint}
  - Or one is $\textit{included}$ in the other

Exercise: Is our example CQ hierarchical?

$\exists x y \ R(x), S(x, y), T(y)$

Yes!
Hierarchical CQs

For a CQ $Q$, write $\text{atoms}(x)$ for the set of atoms where $x$ appears

- A CQ is **hierarchical** if for every variables $x$ and $y$
  - Either $\text{atoms}(x)$ and $\text{atoms}(y)$ are **disjoint**
  - Or one is **included** in the other

- A CQ is **non-hierarchical** if there are two variables $x$ and $y$ such that
  - Some atom contains both $x$ and $y$
  - Some atom contains $x$ but not $y$
  - Some atom contains $y$ but not $x$

→ Simplest example: the $R$-$S$-$T$ query: $Q_1 : \exists x \ y \ R(x), S(x, y), T(y)$

Intuition for arity-2 queries: the hierarchical CQs are unions of star-shaped patterns

Exercise: Is our example CQ hierarchical?

Yes!
Hierarchical CQs

For a CQ \( Q \), write \( \text{atoms}(x) \) for the set of atoms where \( x \) appears

- A CQ is **hierarchical** if for every variables \( x \) and \( y \)
  - Either \( \text{atoms}(x) \) and \( \text{atoms}(y) \) are **disjoint**
  - Or one is **included** in the other

- A CQ is **non-hierarchical** if there are two variables \( x \) and \( y \) such that
  - Some atom contains **both** \( x \) and \( y \)
  - Some atom contains \( x \) but not \( y \)
  - Some atom contains \( y \) but not \( x \)
  → Simplest example: the \( R-S-T \) query: \( Q_1 : \exists x y \ R(x), \ S(x, y), \ T(y) \)

- **Intuition** for arity-2 queries: the hierarchical CQs are **unions of star-shaped patterns**
Hierarchical CQs

For a CQ \( Q \), write \( \text{atoms}(x) \) for the set of atoms where \( x \) appears

- A CQ is **hierarchical** if for every variables \( x \) and \( y \)
  - Either \( \text{atoms}(x) \) and \( \text{atoms}(y) \) are **disjoint**
  - Or one is **included** in the other

- A CQ is **non-hierarchical** if there are two variables \( x \) and \( y \) such that
  - Some atom contains **both** \( x \) and \( y \)
  - Some atom contains \( x \) **but not** \( y \)
  - Some atom contains \( y \) **but not** \( x \)

→ Simplest example: the \( R\-S\-T \) query: \( Q_1 : \exists x \ y \ R(x), S(x, y), T(y) \)

- **Intuition** for arity-2 queries: the hierarchical CQs are **unions of star-shaped patterns**

**Exercise:** Is our example CQ hierarchical? \( \exists c \ r \ d \ \text{Classes}(c, r, d) \land \text{Lockdowns}(d) \)
Hierarchical CQs

For a CQ $Q$, write $\text{atoms}(x)$ for the set of atoms where $x$ appears

- A CQ is **hierarchical** if for every variables $x$ and $y$
  - Either $\text{atoms}(x)$ and $\text{atoms}(y)$ are **disjoint**
  - Or one is **included** in the other

- A CQ is **non-hierarchical** if there are two variables $x$ and $y$ such that
  - Some atom contains both $x$ and $y$
  - Some atom contains $x$ but not $y$
  - Some atom contains $y$ but not $x$

→ Simplest example: the $R$-$S$-$T$ query: $Q_1 : \exists x \ y \ R(x), S(x, y), T(y)$

- **Intuition** for arity-2 queries: the hierarchical CQs are **unions of star-shaped patterns**

**Exercise:** Is our example CQ hierarchical? $\exists c \ r \ d \ \text{Classes}(c, r, d) \land \text{Lockdowns}(d)$ ... Yes!
We study the **uniform reliability** (UR) problem, a simpler variant of PQE:

- We **fix** a CQ $Q$, and consider the problem $\text{UR}(Q)$:
  - **Input:** a database instance $I$ without probabilities
  - **Output:** how many **subinstances** of $I$ satisfy $Q$

Remark: UR is PQE but where all facts have probability $\frac{1}{\text{the number of facts}}$ for $N$ the number of facts of the TID

Our goal: What is the complexity of UR?
• We study the **uniform reliability** (UR) problem, a simpler variant of PQE:
  • We **fix** a CQ $Q$, and consider the problem $\text{UR}(Q)$:
    • **Input:** a database instance $I$ **without probabilities**
    • **Output:** how many **subinstances** of $I$ satisfy $Q$

• Remark: UR is PQE but where all facts have **probability** $1/2$
  (up to a **multiplicative factor** of $2^N$ for $N$ the number of facts of the TID)
Uniform reliability

- We study the **uniform reliability** (UR) problem, a simpler variant of PQE:
  - We fix a CQ $Q$, and consider the problem UR($Q$):
    - **Input**: a database instance $I$ **without probabilities**
    - **Output**: how many subinstances of $I$ satisfy $Q$

- Remark: UR is PQE but where all facts have **probability $1/2$**
  (up to a **multiplicative factor** of $2^N$ for $N$ the number of facts of the TID)

- Our goal: **What is the complexity of UR?**
Our results

- We only study the problem on SJFCQs (see future work)
Our results

• We only study the problem on **SJFCQs** (see future work)

• For **hierarchical SJFCQs**, UR is tractable because PQE is tractable
Our results

• We only study the problem on SJFCQs (see future work)

• For hierarchical SJFCQs, UR is tractable because PQE is tractable

• For non-hierarchical SJFCQs, the complexity of UR is unknown
  • The hardness proof of PQE (see later) crucially uses probabilities $1/2$ and $1$
Our results

- We only study the problem on SJFCQs (see future work)
- For hierarchical SJFCQs, UR is tractable because PQE is tractable
- For non-hierarchical SJFCQs, the complexity of UR is unknown
  - The hardness proof of PQE (see later) crucially uses probabilities $\frac{1}{2}$ and 1

We settle the complexity of UR for SJFCQs by showing:

**Theorem**

Let $Q$ be a non-hierarchical SJFCQ. Then $\text{UR}(Q)$ is $\#P$-hard.
Our results

• We only study the problem on SJFCQs (see future work)

• For hierarchical SJFCQs, UR is tractable because PQE is tractable

• For non-hierarchical SJFCQs, the complexity of UR is unknown
  - The hardness proof of PQE (see later) crucially uses probabilities 1/2 and 1

We settle the complexity of UR for SJFCQs by showing:

**Theorem**

Let $Q$ be a non-hierarchical SJFCQ. Then $\text{UR}(Q)$ is $\#P$-hard.

**Rest of the talk:** proof sketch of this result
Reducing to $R$-$S$-$T$-type queries

- An **$R$-$S$-$T$-type query** is a non-hierarchical SJFCQ of the form:

  $R_1(x), \ldots, R_r(x), S_1(x, y), \ldots, S_s(x, y), T_1(y), \ldots, T_t(y)$

  for some integers $r, s, t > 0$
Reducing to \( R-S-T \)-type queries

- An **\( R-S-T \)-type query** is a non-hierarchical SJFCQ of the form:

\[
R_1(x), \ldots, R_r(x), S_1(x, y), \ldots, S_s(x, y), T_1(y), \ldots, T_t(y)
\]

for some integers \( r, s, t > 0 \)

- **Lemma:** for any non-hierarchical SJFCQ \( Q \), there is an \( R-S-T \)-type query \( Q' \) such that \( UR(Q') \) reduces to \( UR(Q) \)
Reducing to \(R-S-T\)-type queries

- An \(R-S-T\)-type query is a non-hierarchical SJFCQ of the form:

\[
R_1(x), \ldots, R_r(x), S_1(x,y), \ldots, S_s(x,y), T_1(y), \ldots, T_t(y)
\]

for some integers \(r, s, t > 0\)

- **Lemma:** for any non-hierarchical SJFCQ \(Q\), there is an \(R-S-T\)-type query \(Q'\) such that \(\text{UR}(Q')\) reduces to \(\text{UR}(Q)\)

- So it suffices to show that \(\text{UR}(Q')\) is \#P-hard for the \(R-S-T\)-type queries \(Q'\)
Reducing to \textbf{R-S-T-type queries}

- An \textbf{R-S-T-type query} is a non-hierarchical SJFCQ of the form:

\[ R_1(x), \ldots, R_r(x), S_1(x, y), \ldots, S_s(x, y), T_1(y), \ldots, T_t(y) \]

for some integers \( r, s, t > 0 \)

- **Lemma:** for any non-hierarchical SJFCQ \( Q \), there is an \textbf{R-S-T-type query} \( Q' \) such that \( UR(Q') \) reduces to \( UR(Q) \)

- So it suffices to show that \( UR(Q') \) is \#P-hard for the \textbf{R-S-T-type queries} \( Q' \)

- \textbf{In this talk:} we focus for simplicity on \( Q_1 : \exists x y \ R(x), S(x, y), T(y) \)
Hard problem: counting independent sets of bipartite graphs

- Independent set of a bipartite graph: subset of its vertices that contains no edge
  - Example: \( \{u_2, v_1\} \)
Hard problem: counting independent sets of bipartite graphs

- **Independent set** of a bipartite graph: subset of its vertices that contains no edge
  - Example: \{u_2, v_1\}

- It is \#P-hard, given a bipartite graph, to count its independent sets
Hard problem: counting independent sets of bipartite graphs

- **Independent set** of a bipartite graph: subset of its vertices that contains no edge
  - Example: \{u_2, v_1\}
- It is \#P-hard, given a bipartite graph, to count its independent sets

This easily shows the \#P-hardness of PQE (but not UR!) for \(Q_1 : \exists x y \ R(x), \ S(x, y), \ T(y)\):

\[
\begin{align*}
R(u_1) & : 1/2 & \quad S & : 1 & \quad T(v_1) & : 1/2 \\
R(u_2) & : 1/2 & \quad S & : 1 & \quad T(v_2) & : 1/2 \\
R(u_3) & : 1/2 & \quad S & : 1 & \quad T(v_2) & : 1/2
\end{align*}
\]
Hard problem: counting independent sets of bipartite graphs

- **Independent set** of a bipartite graph: subset of its vertices that contains no edge
  - Example: \{u_2, v_1\}
- It is $\#P$-hard, given a bipartite graph, to count its independent sets

This easily shows the $\#P$-hardness of PQE (but not UR!) for $Q_1 : \exists x \ y \ R(x), S(x, y), T(y)$:

We will show how to reduce from counting independent sets to UR($Q_1$)
Idea: parameterizing the count

For a bipartite graph \((U, V, E)\) and a subset \(W \subseteq U \cup V\) of vertices, we write:

\[ c(W) \] the number of edges contained in \(W\)
\[ d(W) \] (resp., \(d'(W)\)) the number of edges having exactly their left (resp., right) endpoint in \(W\)
\[ e(W) \] the number of edges excluded from \(W\)

Hard problem: counting independent sets \(X = |\{W \subseteq U \cup V | c(W) = 0\}|\)

Harder problem: computing all the values: \(X_{c,d,d',e} = |\{W \subseteq U \cup V | c(W) = c\text{ and } d(W) = d\text{ and } d'(W) = d'\text{ and } e(W) = e\}|\)
For a bipartite graph \((U, V, E)\) and a subset \(W \subseteq U \cup V\) of vertices, we write:

- \(c(W)\) the number of edges contained in \(W\)
  - Here, \(c(W) = 1\)

\[
\begin{array}{c}
\text{\(u_1\)} \\
\text{\(u_2\)} \\
\text{\(u_3\)} \\
\end{array}
\quad\quad
\begin{array}{c}
\text{\(v_1\)} \\
\text{\(v_2\)} \\
\end{array}
\]

Idea: parameterizing the count
For a bipartite graph \((U, V, E)\) and a subset \(W \subseteq U \cup V\) of vertices, we write:

- \(c(W)\) the number of edges contained in \(W\)
  - Here, \(c(W) = 1\)
- \(d(W)\) (resp., \(d'(W)\)) the number of edges having exactly their left (resp., right) endpoint in \(W\)
  - Here, \(d(W) = 2\) and \(d'(W) = 1\)
Idea: parameterizing the count

For a bipartite graph \((U, V, E)\) and a subset \(W \subseteq U \cup V\) of vertices, we write:

- \(c(W)\) the number of edges contained in \(W\)
  - Here, \(c(W) = 1\)

- \(d(W)\) (resp., \(d'(W)\)) the number of edges having exactly their left (resp., right) endpoint in \(W\)
  - Here, \(d(W) = 2\) and \(d'(W) = 1\)

- \(e(W)\) the number of edges excluded from \(W\)
  - Here, \(e(W) = 1\)
For a bipartite graph \((U, V, E)\) and a subset \(W \subseteq U \cup V\) of vertices, we write:

- \(c(W)\) the number of edges **contained** in \(W\)
  - Here, \(c(W) = 1\)
- \(d(W)\) (resp., \(d'(W)\)) the number of edges having exactly their left (resp., right) endpoint in \(W\)
  - Here, \(d(W) = 2\) and \(d'(W) = 1\)
- \(e(W)\) the number of edges **excluded** from \(W\)
  - Here, \(e(W) = 1\)

- **Hard problem:** counting independent sets \(X = |\{W \subseteq U \cup V \mid c(W) = 0\}|\)
Idea: parameterizing the count

For a bipartite graph \((U, V, E)\) and a subset \(W \subseteq U \cup V\) of vertices, we write:

- \(c(W)\) the number of edges **contained** in \(W\)
  - Here, \(c(W) = 1\)
- \(d(W)\) (resp., \(d'(W)\)) the number of edges having exactly their left (resp., right) endpoint in \(W\)
  - Here, \(d(W) = 2\) and \(d'(W) = 1\)
- \(e(W)\) the number of edges **excluded from** \(W\)
  - Here, \(e(W) = 1\)

- **Hard problem:** counting independent sets \(X = |\{W \subseteq U \cup V \mid c(W) = 0\}|\)
- **Harder problem:** computing all the values:

\[
X_{c,d,d',e} = \{|W \subseteq U \cup V \mid c(W) = c \text{ and } d(W) = d \text{ and } d'(W) = d' \text{ and } e(W) = e\}|
\]
Idea: coding to several copies

- We want to design a **reduction**:
  - We reduce **from** (we want): given a bipartite graph $G$, compute the $X_{c,d,d',e}$
  - We reduce **to** (we have): given a database instance $D$, compute $\text{UR}(Q_1)$
Idea: coding to several copies

• We want to design a **reduction**:  
  • We reduce **from** (we want): given a bipartite graph $G$, compute the $X_{c,d,d',e}$  
  • We reduce **to** (we have): given a database instance $D$, compute $UR(Q_1)$  

• **Idea:** code $G$ to a **family** of instances $D_p$ **indexed** by $p > 0$
**Idea: coding to several copies**

- We want to design a **reduction**:
  - We reduce **from** (we want): given a bipartite graph $G$, compute the $X_{c,d,d',e}$
  - We reduce **to** (we have): given a database instance $D$, compute $UR(Q_1)$

- **Idea**: code $G$ to a family of instances $D_p$ indexed by $p > o$
- Fix a **box** $B_p(a, b)$ for index $p > o$: an instance with two distinguished elements $(a, b)$
- **Code $G$** for index $p > o$ to an instance by:
  - putting an $R$-fact on each $U$-vertex and a $T$-fact on each $V$-vertex
  - coding every edge $(u, v)$ by a **copy of the box** $B_p(u, v)$

![Diagram](image-url)
Getting an equation system

Take the coding of $G$ for index $p$, and compute the number $N_p$ of subinstances violating $Q_1$.

\[
\begin{align*}
N_p &= \sum_{W \subseteq V} N_{Wp} \\
N_{Wp} &= \sum_{c, d, d', e} N_{Wp} \cdot \gamma_c(\delta d p) \cdot \delta d' e p
\end{align*}
\]

where $N_{Wp}$ is the number of subinstances violating $Q_1$ when fixing the $R$-facts and $T$-facts to be precisely on $W$.

Now $N_{Wp}$ only depends on:

1. The numbers $c(W), d(W), d'(W), e(W)$ of edges contained, dangling, or excluded from $W$.
2. The numbers $\gamma_p, \delta_p, \delta'_p, \eta_p$ of subinstances of the box $B_p$ that violate $Q_1$ when fixing $R$-facts on $a$ and/or $T$-facts on $b$. 

![Diagram](image-url)
Getting an equation system

Take the coding of $G$ for index $p$, and compute the number $N_p$ of subinstances violating $Q_1$

$$N_p = \sum_{W \subseteq V} N_p^W$$

where $N_p^W$ is the number of subinstances violating $Q_1$ when fixing the $R$-facts and $T$-facts to be precisely on $W$
Getting an equation system

Take the coding of $G$ for index $p$, and compute the number $N_p$ of subinstances violating $Q_1$

![Graph with nodes and edges]

- We have:
  \[ N_p = \sum_{W \subseteq V} N_p^W \]

  where $N_p^W$ is the number of subinstances violating $Q_1$ when fixing the $R$-facts and $T$-facts to be precisely on $W$

- Now $N_p^W$ only depends on:
Getting an equation system

Take the coding of $G$ for index $p$, and compute the number $N_p$ of subinstances violating $Q_1$ when fixing the $R$-facts and $T$-facts to be precisely on $W$. Now $N_p^W$ only depends on:

- The numbers $c(W), d(W), d'(W), e(W)$ of edges contained, dangling, or excluded from $W$.

\[
N_p = \sum_{W \subseteq V} N_p^W
\]

where $N_p^W$ is the number of subinstances violating $Q_1$ when fixing the $R$-facts and $T$-facts to be precisely on $W$. We have:

\[
N_p = \sum_{W \subseteq V} N_p^W
\]
Getting an equation system

Take the coding of $G$ for index $p$, and compute the number $N_p$ of subinstances violating $Q_1$

\[
N_p = \sum_{W \subseteq V} N^W_p
\]

where $N^W_p$ is the number of subinstances violating $Q_1$ when fixing the $R$-facts and $T$-facts to be precisely on $W$

- We have:
- Now $N^W_p$ only depends on:
  - The numbers $c(W), d(W), d'(W), e(W)$ of edges contained, dangling, or excluded from $W$
  - The numbers $\gamma_p, \delta_p, \delta'_p, \eta_p$ of subinstances of the box $B_p$ that violate $Q_1$ when fixing $R$-facts on $a$ and/or $T$-facts on $b$
Getting an equation system

Take the coding of $G$ for index $p$, and compute the number $N_p$ of subinstances violating $Q_1$

\[
N_p = \sum_{W \subseteq V} N_p^W = \sum_{W \subseteq V} \gamma_p^c(W) \delta_p^d(W) (\delta'_p)^{d'(W)} \eta_p^e(W)
\]

where $N_p^W$ is the number of subinstances violating $Q_1$ when fixing the $R$-facts and $T$-facts to be precisely on $W$

• We have:

• Now $N_p^W$ only depends on:
  • The numbers $c(W), d(W), d'(W), e(W)$ of edges contained, dangling, or excluded from $W$
  • The numbers $\gamma_p, \delta_p, \delta'_p, \eta_p$ of subinstances of the box $B_p$ that violate $Q_1$ when fixing $R$-facts on $a$ and/or $T$-facts on $b$
Getting an equation system

Take the coding of $G$ for index $p$, and compute the number $N_p$ of subinstances violating $Q_1$

\[ N_p = \sum_{W \subseteq V} N_p^W = \sum_{W \subseteq V} \gamma_p^c(W) \delta_p^d(W) (\delta'_p)^{d'(W)} \eta_p^e(W) = \sum_{c,d,d',e} X_{c,d,d',e} \times \gamma_p^c \delta_p^d (\delta'_p)^{d'} \eta_p^e \]

where $N_p^W$ is the number of subinstances violating $Q_1$ when fixing the $R$-facts and $T$-facts to be precisely on $W$

- Now $N_p^W$ only depends on:
  - The numbers $c(W)$, $d(W)$, $d'(W)$, $e(W)$ of edges contained, dangling, or excluded from $W$
  - The numbers $\gamma_p$, $\delta_p$, $\delta'_p$, $\eta_p$ of subinstances of the box $B_p$ that violate $Q_1$ when fixing $R$-facts on $a$ and/or $T$-facts on $b$
We have shown the equation:

\[ N_p = \sum_{c,d,d',e} X_{c,d,d',e} \times \gamma_p^c \delta_p^d (\delta'_p)^{d'} \eta_p^e \]

where:

- The \(X_{c,d,d',e}\) are what we want (to count independent sets)
- The \(N_p\) are what we have (by solving UR(\(Q_1\)))
- The \(\gamma_p^c \delta_p^d (\delta'_p)^{d'} \eta_p^e\) are coefficients of a matrix \(M\) depending on our choice of box \(B_p\)
Equation system and conclusion

We have shown the equation:

\[ N_p = \sum_{c,d,d',e} X_{c,d,d',e} \times \gamma_p^{c \delta_p^d (\delta'_p)^{d'}} \eta_p^e \]

where:

- The \( X_{c,d,d',e} \) are what we want (to count independent sets)
- The \( N_p \) are what we have (by solving \( UR(Q_1) \))
- The \( \gamma_p^{c \delta_p^d (\delta'_p)^{d'}} \eta_p^e \) are coefficients of a matrix \( M \) depending on our choice of box \( B_p \)

In other words we have:

\[ \vec{N} = M \vec{X} \]
Equation system and conclusion

We have shown the equation:

\[ N_p = \sum_{c,d,d',e} X_{c,d,d',e} \times \gamma_p^{c d} \delta_p (\delta'_p)^{d'} \eta_p^e \]

where:

- The \( X_{c,d,d',e} \) are what we want (to count independent sets)
- The \( N_p \) are what we have (by solving \( UR(Q_1) \))
- The \( \gamma_p^{c d} \delta_p (\delta'_p)^{d'} \eta_p^e \) are coefficients of a matrix \( M \) depending on our choice of box \( B_p \)

In other words we have:

\[ \vec{N} = M \vec{X} \]

We can design a box where \( M \) is invertible, so we can recover \( \vec{X} \) from \( \vec{N} \), showing hardness
Conclusion and future work

- We have shown that uniform reliability (UR) for non-hierarchical SJFCQs is \#P-hard, so it is no easier than PQE.
- We also have preliminary results for other PQE restrictions.

Future work directions:
- Can we extend to the UCQ dichotomy, e.g., following [Kenig and Suciu, two.zero.zero]?
- What about the case of PQE with a constant probability \( \neq \) or a different constant probability per relation?
- Which connection to symmetric model counting [Beame et al., two.zero.five]?
Conclusion and future work

• We have shown that uniform reliability (UR) for non-hierarchical SJFCQs is \#P-hard, so it is no easier than PQE.

• We also have preliminary results for other PQE restrictions.

Future work directions:

• Can we extend to the UCQ dichotomy, e.g., following [Kenig and Suciu, 2020]?

• What about the case of PQE with a constant probability \( \neq 1/2 \)? or a different constant probability per relation?

• Which connection to symmetric model counting [Beame et al., 2015]?
Conclusion and future work

- We have shown that uniform reliability (UR) for non-hierarchical SJFCQs is \#P-hard, so it is no easier than PQE.
- We also have preliminary results for other PQE restrictions.

Future work directions:

- Can we extend to the UCQ dichotomy, e.g., following [Kenig and Suciu, 2020]?
- What about the case of PQE with a constant probability $\neq 1/2$? or a different constant probability per relation?
- Which connection to symmetric model counting [Beame et al., 2015]?

Thanks for your attention!

