Regular Queries on Graph Databases

Juan Reutter
PUC Chile

Miguel Romero
Univ. of Chile

Moshe Y. Vardi
Rice Univ.

March 23, ICDT 2015, Brussels, Belgium
Graph Databases

Applications:

• Semantic Web, linked data
• Biological Networks
• Social Network Analysis
• …

Graph db implementations:

• Neo4j
• Sparskee
• …
Graph Databases

Graph Model:

- Finite directed graph
- Set of labels $\Sigma$
- Labeling of the edges
Graph Databases: example

\[ \Sigma = \{ \text{is\_friend, knows, helps} \} \]
Querying Graph Databases

We are interested in **navigational** queries

Basic query mechanism: **regular path queries** (RPQs) *(first studied in Cruz, Mendelzon, Wood 1987)*

- Set of labels $\Sigma$
- $\text{RPQ} = \text{regular expression} \text{ over } \Sigma$
- $\text{RPQ} \ E \text{ over graph db } G \text{ selects all the pairs of nodes connected by a path labeled in } E$
Potential friends:  $(\text{is\_friend} + \text{knows})^*$
Potential friends: \((\text{is\_friend} + \text{knows})^*\)
RPQ: examples

Potential friends: \((\text{is\_friend} + \text{knows})^*\)
Two-way RPQs (2RPQs)

(Calvanese, De Giacomo, Lenzerini, Vardi 2000)

Collaborators: help the same person

Indirect collaborators: \((\text{helps} \cdot \text{helps}^-)^*\)
Two-way RPQs (2RPQs)

(Calvanese, De Giacomo, Lenzerini, Vardi 2000)

Collaborators: help the same person

Indirect collaborators: \((\text{helps} \cdot \text{helps}^\rightarrow)^*\)
C2RPQs: Conjunctive 2RPQs

\[ C2RPQs = \textit{joins} \text{ of } 2\text{RPQs} + \textit{projection} \]
C2RPQs: Conjunctive 2RPQs

Persons who have a potential friend that is a collaborator:

\[
\text{Ans}(x) \leftarrow (\text{is\_friend} + \text{knows})^*(x, y), \text{helps}(x, z), \text{helps}(y, z).
\]
C2RPQs: Conjunctive 2RPQs

Persons who have a potential friend that is a collaborator:

$$Ans(x) \leftarrow (\text{is\_friend} + \text{knows})^*(x, y), \text{helps}(x, z), \text{helps}(y, z).$$
\[
\text{Ans}(x) \leftarrow (\text{is\_friend} + \text{knows})^*(x, y), \text{helps}(x, z), \text{helps}(y, z).
\]
\[
\text{Ans}(x) \leftarrow \text{helps}(x, y), \text{helps}^-(x, y).
\]
Beyond 2RPQs: nested 2RPQs (N2RPQs)

\[
\text{nested 2RPQs} = \underbrace{\text{2RPQs extended with an }}_{\text{existential test operator}} \langle E \rangle
\]

Arenas, Gutierrez, Pérez 2010
Bienvenu, Calvanese, Ortiz, Simkus 2014
nested 2RPQs: examples

Potential friends only via persons who help somebody:

\[((\text{is\_friend} + \text{knows}) \cdot \langle\text{help}\rangle)^* (\text{is\_friend} + \text{knows})\]
nested 2RPQs: examples

*Potential friends only via persons who help somebody:*

\[
((\text{is	extunderscore friend} + \text{knows}) \cdot \langle \text{help} \rangle)^*(\text{is	extunderscore friend} + \text{knows})
\]
UCN2RPQs: Unions of conjunctive N2RPQs

**UCN2RPQ:** contains the basic desirable features of graph languages

**Complexity of UCN2RPQs:**

- Combined complexity: **NP-complete**
- Data complexity: **NLOGSPACE-complete**
- Query containment: **EXPSPACE-complete**

(Calvanese, Di Giacomo, Lenzerini, Vardi 2000)
Expressiveness of UCN2RPQs

UCN2RPQs cannot express the following patterns:
Expressiveness of UCN2RPQs

UCN2RPQs cannot express the following patterns:

\[ u \]

\[ a \rightarrow a \rightarrow b \]

\[ b \rightarrow a \rightarrow b \]

\[ a \rightarrow c \rightarrow a \]

\[ a \rightarrow c \rightarrow a \]

\[ v \]
More extensions of UC(N)2RPQs

• **Nested monadically defined queries** (NEMODEQs) 
  (*Krotzsch, Rudolph 2013*)
• **Extended CRPQs**: path variables 
  (*Barceló, Libkin, Lin, Wood 2012*)
• **Context-free path queries** 
  (*Hellings 2014*)
• **Algebraic languages** 
  (*Fletcher et al. 2011*)
More extensions of UC(N)2RPQs

Query containment for all these extensions is either **undecidable** or **non-elementary**

**Question**
Is there an expressive extension of UCN2RPQs with an **elementary** containment problem?
Contributions

We propose the class of **Regular queries**

- It subsumes **UCN2RPQs**
- Same combined/data complexity as **UCN2RPQs**
- Containment for regular queries is **elementary**
Regular queries: motivation

**Key property:** closure under *query nesting*

- *Relational algebra:*
  closure of standard relational operators
- *Unions of conjunctive queries:*
  closure of project, join, union
- *2RPQs:*
  closure of concatenation, union, and transitive closure
- *C2RPQs:*
  the **transitive closure** of a binary C2RPQ **is not** a C2RPQ

Idea

We close UC2RPQs under **transitive closure**
Regular queries: examples

Set of labels $\Sigma = \{a, b, c\}$

$I(x, y) \leftarrow a^+ b(x, z), c(a + b)(z, y), bac^+(y, x)$.

$I(x, y) \leftarrow ac^+ bc^+(x, y)$.

$R(x, y) \leftarrow a^+(x, y)$.

$R(x, y) \leftarrow (a + I^+)(x, y), (bcI)^+(y, z)$.

$S(x, y) \leftarrow I^+ aI^+(x, y)$.

$Ans(x, y) \leftarrow b^+(x, y), RaR^+ cS^+(y, x)$.

Intentional labels/predicates $\mathcal{I} = \{I, R, S, Ans\}$
Regular queries: definition

Set of **basic** labels $\Sigma$  
Set of **intentional** labels $\mathcal{I}$

**Basic rules:** binary C2RPQs over $\Sigma \cup \mathcal{I}$

$$I(x, y) \leftarrow E_1(y_1, y_1'), \ldots, E_m(y_m, y_m').$$

$\in \mathcal{I}$  
2RPQs over $\Sigma \cup \mathcal{I}$

**Selection rules:**

$$Ans(x_1, \ldots, x_n) \leftarrow E_1(y_1, y_1'), \ldots, E_m(y_m, y_m').$$

2RPQs over $\Sigma \cup \mathcal{I}$

All selection rules have the same arity $n \geq 0$
Regular queries: definition

**nonrecursive program:**
no intentional predicate depends on itself
Regular queries: examples

\[
I(x, y) \leftarrow a(x, y), b(x, y).
\]
\[
Ans(x, y) \leftarrow I^+(x, y).
\]
Regular queries: examples

\[
I(x, y) \leftarrow a(x, y), b(x, y).
\]
\[
J(x, y) \leftarrow c(x, y), a^+(x, z), I^+(y, z).
\]
\[
\text{Ans}(x, y) \leftarrow J^+(x, y).
\]
Previous work

Regular queries have already been considered (Bourhis, Krötzsch, Rudolph 2014)

The class of regular queries is equivalent to positive FO with unary transitive closure without parameters

We can only use $\text{TC}[\varphi(x, y)](u, v)$

Previous work: focus on expressiveness
This work: focus on complexity, specially, on the containment problem
Basic results: evaluation of regular queries

Proposition:
Query evaluation for regular queries is **NP-complete** in combined complexity and **NLOGSPACE-complete** in data complexity.

Consequence:
Regular queries are not harder to evaluate than UCN2RPQs.
Main result: containment of regular queries

Theorem:

Query containment for regular queries is $2\text{EXPSPACE}$-complete
Nested UC2RPQs

We first consider a non-succint version of regular queries

\[
\text{nested UC2RPQ} = \text{a regular query where each defined intensional predicate can be reused only once}
\]
Nested UC2RPQs

\[
\begin{align*}
I(x, y) & \leftarrow \ldots \ldots \\
R(x, y) & \leftarrow \ldots \ldots I \ldots \ldots \\
S(x, y) & \leftarrow \ldots \ldots \ldots \ldots \\
\end{align*}
\]

Only once
Nested UC2RPQs

\[ I(x, y) \leftarrow \ldots \]

\[ R(x, y) \leftarrow \ldots \quad I \quad \ldots \quad I \]

\[ S(x, y) \leftarrow \ldots \quad \ldots \quad \ldots \]
Nested UC2RPQs

\[ I(x, y) \leftarrow \ldots \]

\[ R(x, y) \leftarrow \ldots I \ldots \]

\[ S(x, y) \leftarrow \ldots I \ldots \]
Nested UC2RPQs: examples

\[ I(x, y) \leftarrow a^+ b(x, z), c(a + b)(z, y), bac^+(y, x). \]
\[ I(x, y) \leftarrow ac^+ bc^+ (x, y). \]
\[ R(x, y) \leftarrow a^+(x, y). \]
\[ R(x, y) \leftarrow (a + I^+)(x, y), (bcI)^+(y, z). \]
\[ S(x, y) \leftarrow I^+ aI^+(x, y). \]
\[ Ans(x, y) \leftarrow b^+(x, y), RaR^+ cS^+(y, x). \]

this is not a nested UC2RPQ…

…but it can be transformed into one
Nested UC2RPQs: examples

\[ I(x, y) \leftarrow a^+ b(x, z), c(a + b)(z, y), bac^+(y, x). \]
\[ I(x, y) \leftarrow ac^+ bc^+(x, y). \]
\[ R(x, y) \leftarrow a^+(x, y). \]
\[ R(x, y) \leftarrow (a + I^+)(x, y), (bcI)^+(y, z). \]
\[ S(x, y) \leftarrow I^+ aI^+(x, y). \]
\[ Ans(x, y) \leftarrow b^+(x, y), R_1 a R_2^+ c S^+(y, x). \]
Nested UC2RPQs: examples

\[ I(x, y) \leftarrow a^+ b(x, z), c(a + b)(z, y), bac^+(y, x). \]
\[ I(x, y) \leftarrow ac^+ bc^+(x, y). \]
\[ R(x, y) \leftarrow a^+(x, y). \]
\[ R(x, y) \leftarrow (a + I^+)(x, y), (bcI)^+(y, z). \]

\[ S(x, y) \leftarrow I^+ aI^+(x, y). \]
\[ Ans(x, y) \leftarrow b^+(x, y), R_1 aR_2^+ cS^+(y, x). \]
Nested UC2RPQs: examples

\[ I(x, y) \leftarrow a^+ b(x, z), c(a + b)(z, y), bac^+(y, x). \]
\[ I(x, y) \leftarrow ac^+bc^+(x, y). \]

\[ R_1(x, y) \leftarrow a^+(x, y). \]
\[ R_1(x, y) \leftarrow (a + I^+)(x, y), (bcI)^+(y, z). \]
\[ R_2(x, y) \leftarrow a^+(x, y). \]
\[ R_2(x, y) \leftarrow (a + I^+)(x, y), (bcI)^+(y, z). \]

\[ S(x, y) \leftarrow I^+ aI^+(x, y). \]

\[ Ans(x, y) \leftarrow b^+(x, y), R_1 a R_2^+ cS^+(y, x). \]
Nested UC2RPQs: examples

\[ I(x, y) \leftarrow a^+ b(x, z), c(a + b)(z, y), bac^+(y, x). \]
\[ I(x, y) \leftarrow ac^+ bc^+(x, y). \]

\[ R_1(x, y) \leftarrow a^+(x, y). \]
\[ R_1(x, y) \leftarrow (a + I_1^+)(x, y), (bcI_2)^+(y, z). \]
\[ R_2(x, y) \leftarrow a^+(x, y). \]
\[ R_2(x, y) \leftarrow (a + I_3^+)(x, y), (bcI_4)^+(y, z). \]

\[ S(x, y) \leftarrow I_5^+ aI_6^+(x, y). \]

\[ Ans(x, y) \leftarrow b^+(x, y), R_1 a R_2^+ cS^+(y, x). \]
Nested UC2RPQs: examples

\[ I_1(x, y) \leftarrow a^+b(x, z), c(a + b)(z, y), bac^+(y, x) . \]
\[ I_1(x, y) \leftarrow ac^+bc^+(x, y) . \]
\[ I_2(x, y) \leftarrow a^+b(x, z), c(a + b)(z, y), bac^+(y, x) . \]
\[ I_2(x, y) \leftarrow ac^+bc^+(x, y) . \]
\[ I_3(x, y) \leftarrow a^+b(x, z), c(a + b)(z, y), bac^+(y, x) . \]
\[ I_3(x, y) \leftarrow ac^+bc^+(x, y) . \]
\[ I_4(x, y) \leftarrow a^+b(x, z), c(a + b)(z, y), bac^+(y, x) . \]
\[ I_4(x, y) \leftarrow ac^+bc^+(x, y) . \]
\[ I_5(x, y) \leftarrow a^+b(x, z), c(a + b)(z, y), bac^+(y, x) . \]
\[ I_5(x, y) \leftarrow ac^+bc^+(x, y) . \]
\[ I_6(x, y) \leftarrow a^+b(x, z), c(a + b)(z, y), bac^+(y, x) . \]
\[ I_6(x, y) \leftarrow ac^+bc^+(x, y) . \]

\[ R_1(x, y) \leftarrow a^+(x, y) . \]
\[ R_1(x, y) \leftarrow (a + I_1^+)(x, y), (bcI_2)^+(y, z) . \]
\[ R_2(x, y) \leftarrow a^+(x, y) . \]
\[ R_2(x, y) \leftarrow (a + I_3^+)(x, y), (bcI_4)^+(y, z) . \]

\[ S(x, y) \leftarrow I_5^+aI_6^+(x, y) . \]

\[ Ans(x, y) \leftarrow b^+(x, y), R_1aR_2^+cS^+(y, x) . \]
Nested UC2RPQs

Each regular query can be unfolded into an equivalent nested UC2RPQ…

…but we could have an exponential blow-up in size.
Theorem:
Query containment for nested UC2RPQs is \textit{EXPSPACE}-complete

Consequence:
Query containment for regular queries is in \textit{2EXPSPACE}
Proof idea: EXPSPACE algorithm

Step 1: Serialization of $\Gamma$ and $\Gamma'$

Reduce the problem to containment of a 2RPQ in a nested UC2RPQ

$2\text{RPQ} \quad \tilde{E} \quad \text{nested UC2RPQ} \quad \tilde{\Gamma}$

$\Gamma \subseteq \Gamma' \iff \tilde{E} \subseteq \tilde{\Gamma}$

Main challenge: construction of $\tilde{\Gamma}$
Proof idea: EXPSPACE algorithm

\[
\begin{align*}
2\text{RPQ} & \quad \tilde{E} & \quad \text{nested UC2RPQ} & \quad \tilde{\Gamma} \\
\end{align*}
\]

**Step 2:** Applying **automata** techniques

\[\mathcal{A}_{\tilde{E},\tilde{\Gamma}} : \text{accepts } \textbf{counterexamples} \text{ of } \tilde{E} \subseteq \tilde{\Gamma}\]

\[\tilde{E} \subseteq \tilde{\Gamma} \iff \mathcal{L}(\mathcal{A}_{\tilde{E},\tilde{\Gamma}}) = \emptyset\]

**Main challenge:** construction of \(\mathcal{A}_{\tilde{E},\tilde{\Gamma}}\) of at most **doubly exponential** size in \(|\tilde{E}|, |\tilde{\Gamma}|\)
Further results

Observation:
Let \( d \geq 0 \). Then each regular query of depth at most \( d \) can be unfolded into a polynomial-sized nested UC2RPQ.

Consequence:
Query containment for regular queries of bounded depth is EXPSPACE-complete.
Conclusions

Regular queries achieve a good balance between expressiveness and complexity

- subsume UCN2RPQs
- evaluation is not harder than evaluation of UCN2RPQs
- containment is elementary: $2\text{EXPSPACE}$-complete
- containment is $\text{EXPSPACE}$-complete
  (as hard as that of UCN2RPQs) in two interesting cases:
  - nested UC2RPQs
  - queries of bounded depth
Open questions

- Can we extend the class of regular queries and still obtain an **elementary** containment problem?

  Containment is **non-elementary** even for a slight extension of regular queries: **positive FO with unary TC (with parameters)** *(Bourhis, Krötzsch, Rudolph 2014)*

- What is the complexity of containment of a **Datalog program** in a **regular query**? Is it elementary?

  Thanks!