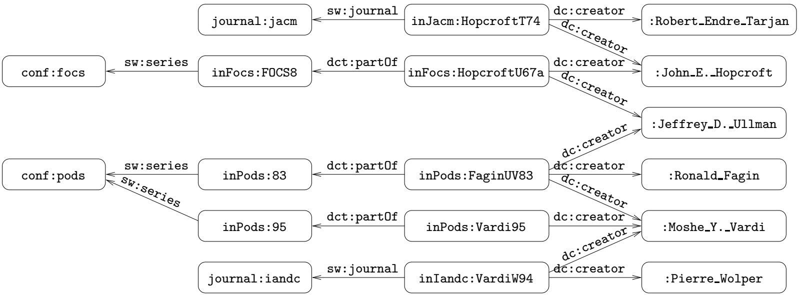
Schema Mappings and Data Exchange for Graph Databases

Pablo Barceló   Jorge Pérez   Juan Reutter

Universidad de Chile, PUC Chile

Graph structured data is now everywhere



RDF Linked Data representation of DBLP (real data!)

- DBpedia (RDF representation of Wikipedia)
- Bio2RDF, GeoNames, FreeBase, FOAF, ...
- Facebook, Twitter, ...

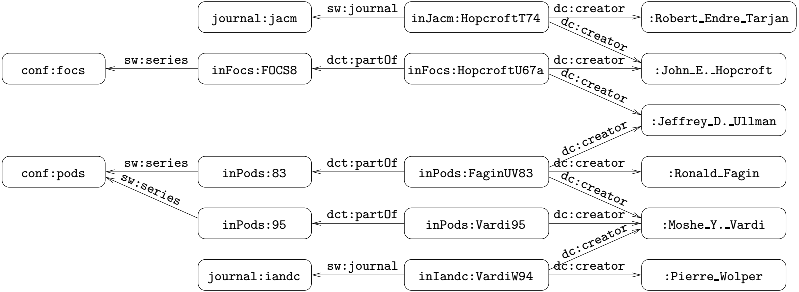
Formalisms to exchange graph databases

First define a *graph mapping language*, then

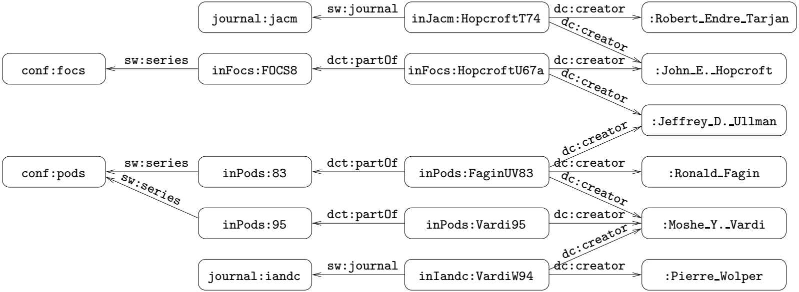
- Exchanging graph databases
- Computing solutions and answering target queries
- Advanced schema mapping operations
  - composition
  - inversion
  - ...

Outline

Graph query languages

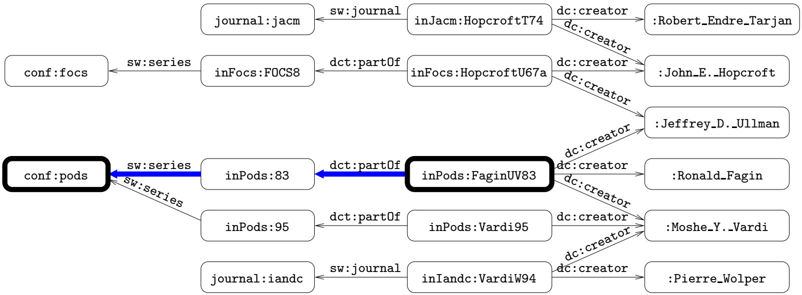


Graph query languages



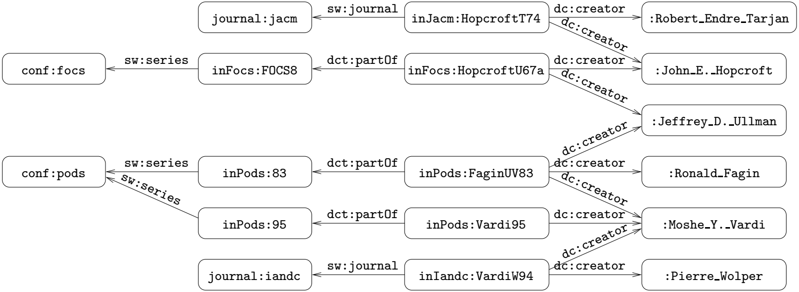
RPQ: $\texttt{partOf} \cdot \texttt{series}$

Graph query languages



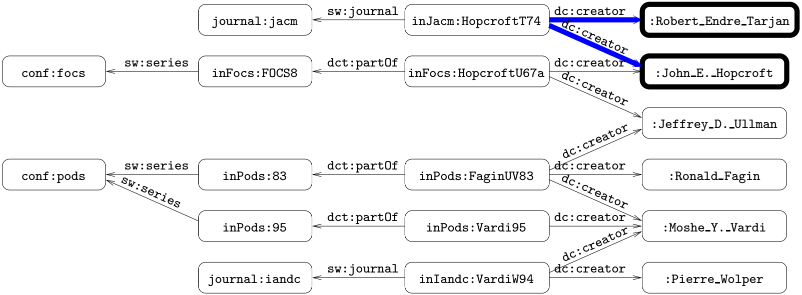
RPQ: $\texttt{partOf} \cdot \texttt{series}$

Graph query languages



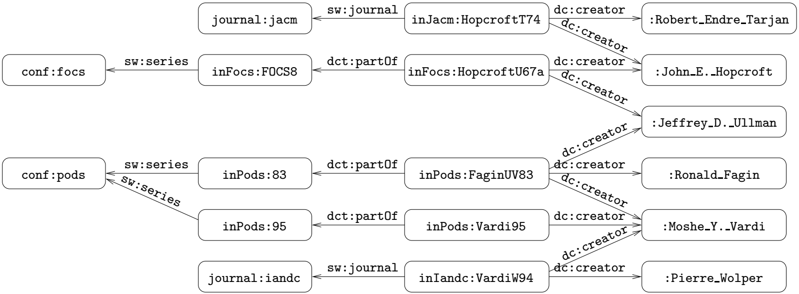
2RPQ: $\texttt{creator}^{-} \cdot \texttt{creator}$

Graph query languages



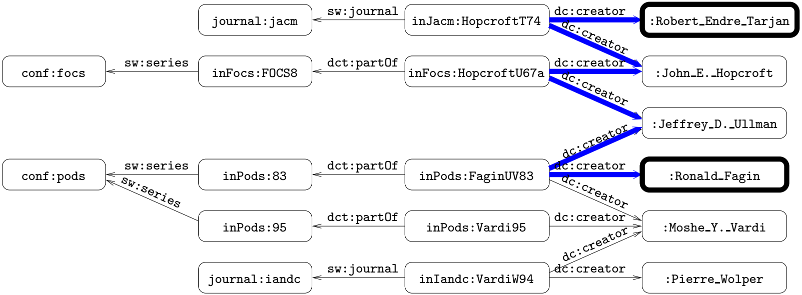
2RPQ: $\texttt{creator}^- \cdot \texttt{creator}$

Graph query languages



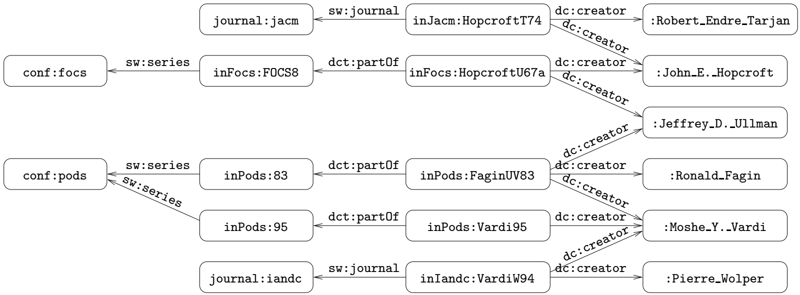
2RPQ: $(\texttt{creator}^{-} \cdot \texttt{creator})^{*}$

Graph query languages



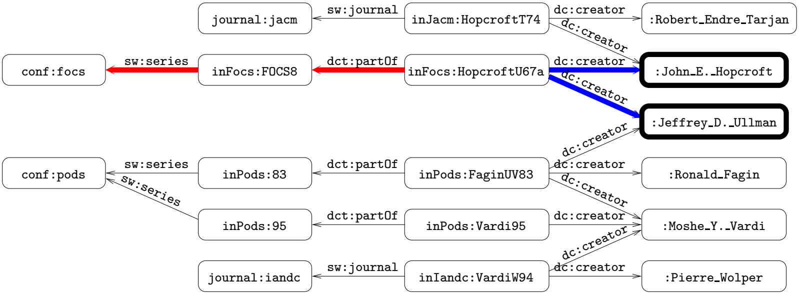
2RPQ: $(\texttt{creator}^{-} \cdot \texttt{creator})^{*}$

Graph query languages



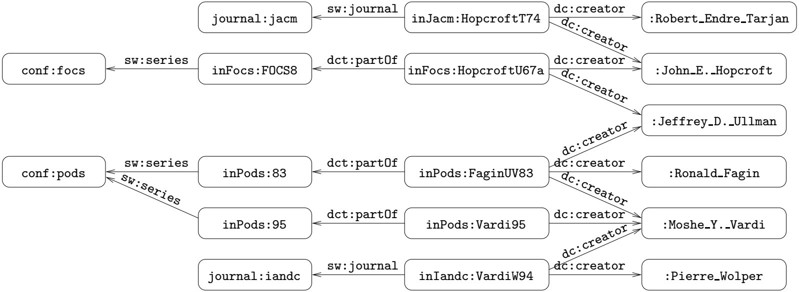
NRE: $\texttt{creator}^{-} \cdot [\, \texttt{partOf} \cdot \texttt{series} \,] \cdot \texttt{creator}$

Graph query languages



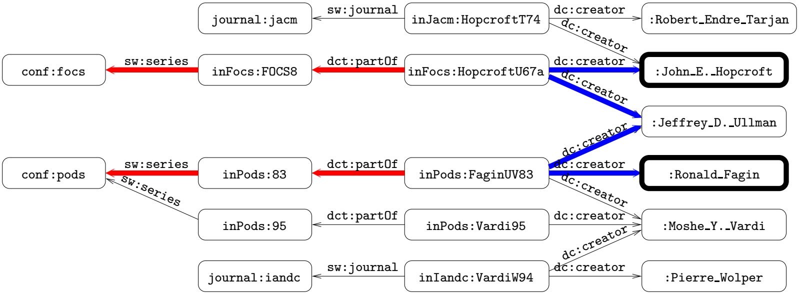
NRE: $\texttt{creator}^{-} \cdot [\, \texttt{partOf} \cdot \texttt{series} \,] \cdot \texttt{creator}$

Graph query languages



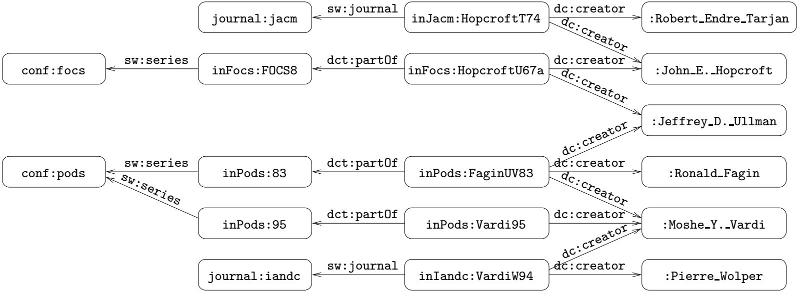
NRE: $(\texttt{creator}^{-} \cdot [\, \texttt{partOf} \cdot \texttt{series} \,] \cdot \texttt{creator})^{+}$

Graph query languages



NRE: $(\texttt{creator}^{-} \cdot [\,\texttt{partOf} \cdot \texttt{series}\,] \cdot \texttt{creator})^{+}$

Graph query languages

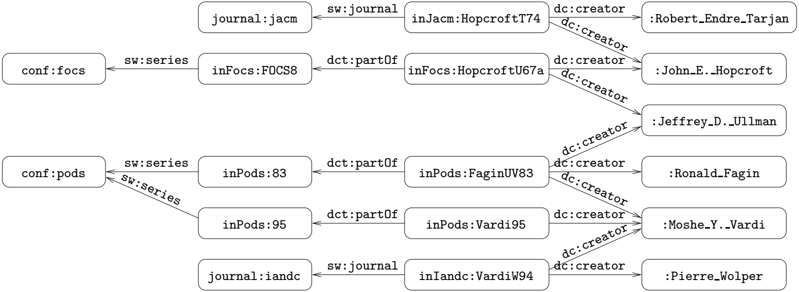


Conjunctions over RPQs, 2RPQs, and NREs

$$\exists \bar{y}\big(\ (u_1, r_1, u_1') \wedge \cdots \wedge (u_k, r_k, u_k')\ \big)$$

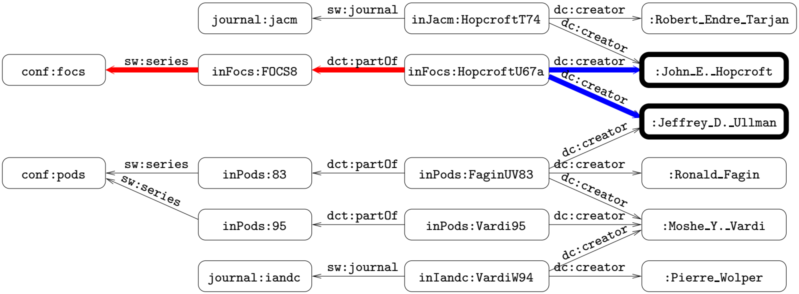
CRPQs, C2RPQs, CNREs

Graph query languages



$$\exists u \exists v \big((x, \mathtt{creator}^-, u) \wedge (u, \mathtt{partOf} \cdot \mathtt{series}, v) \wedge (u, \mathtt{creator}, y) \big)$$

Graph query languages



$$\exists u \exists v \big((x, \texttt{creator}^-, u) \wedge (u, \texttt{partOf} \cdot \texttt{series}, v) \wedge (u, \texttt{creator}, y) \big)$$

Review on expressiveness

$$\begin{array}{rcl} \text{NREs} & \not\subseteq & \text{C2RPQs} \\ \text{(binary) CRPQs} & \not\subseteq & \text{NREs} \end{array}$$

Review on expressiveness

$$\begin{aligned} \text{NREs} &\not\subseteq \text{C2RPQs} \\ \text{(binary) CRPQs} &\not\subseteq \text{NREs} \end{aligned}$$

Example

$$(\texttt{creator}^{-} \cdot [\, \texttt{partOf} \cdot \texttt{series} \,] \cdot \texttt{creator})^{+}$$

cannot be expressed as a C2RPQ

Review on expressiveness

$$\text{NREs} \not\subseteq \text{C2RPQs}$$
$$\text{(binary) CRPQs} \not\subseteq \text{NREs}$$

Example

$$(\texttt{creator}^{-} \cdot [\,\texttt{partOf} \cdot \texttt{series}\,] \cdot \texttt{creator})^{+}$$

cannot be expressed as a C2RPQ

tree-shaped binary C2RPQs $\equiv$ $(\ )^{*}$-$[\ ]$ *alternation-free* NREs

Review on complexity

Evaluation problem for NREs can be solved in $O(|G| \times |expr|)$

- via a PDL-like recursive labeling procedure

NREs properly extends a linear-time fragment of C2RPQs maintaining the complexity of evaluation

Review on complexity

Evaluation problem for NREs can be solved in $O(|G| \times |expr|)$

- via a PDL-like recursive labeling procedure

NREs properly extends a linear-time fragment of C2RPQs maintaining the complexity of evaluation

Evaluation problem for CRPQs is NP-complete

- it is in NP for CNREs

Graph mapping language

Consider two (disjoint) graph alphabets $\Sigma_{\mathbf{S}}$ and $\Sigma_{\mathbf{T}}$

- *Graph mapping:* $\mathcal{M} = (\Sigma_{\mathbf{S}}, \Sigma_{\mathbf{T}}, \mathcal{T})$ s.t. $\mathcal{T}$ contains rules

$$\varphi_{\mathbf{S}}(\bar{x}) \longrightarrow \psi_{\mathbf{T}}(\bar{x})$$

$\varphi_{\mathbf{S}}$ and $\psi_{\mathbf{T}}$ are *CNREs* over $\Sigma_{\mathbf{S}}$ and $\Sigma_{\mathbf{T}}$, resp.

Graph mapping language

Consider two (disjoint) graph alphabets $\Sigma_{\mathbf{S}}$ and $\Sigma_{\mathbf{T}}$

- *Graph mapping:* $\mathcal{M} = (\Sigma_{\mathbf{S}}, \Sigma_{\mathbf{T}}, \mathcal{T})$ s.t. $\mathcal{T}$ contains rules

$$\varphi_{\mathbf{S}}(\bar{x}) \longrightarrow \psi_{\mathbf{T}}(\bar{x})$$

  $\varphi_{\mathbf{S}}$ and $\psi_{\mathbf{T}}$ are *CNREs* over $\Sigma_{\mathbf{S}}$ and $\Sigma_{\mathbf{T}}$, resp.

- *L_1-to-L_2 mapping:* $\varphi_{\mathbf{S}} \in L_1$ and $\psi_{\mathbf{T}} \in L_2$

Graph mapping language

Consider two (disjoint) graph alphabets $\Sigma_{\mathbf{S}}$ and $\Sigma_{\mathbf{T}}$

- *Graph mapping:* $\mathcal{M} = (\Sigma_{\mathbf{S}}, \Sigma_{\mathbf{T}}, \mathcal{T})$ s.t. $\mathcal{T}$ contains rules

$$\varphi_{\mathbf{S}}(\bar{x}) \longrightarrow \psi_{\mathbf{T}}(\bar{x})$$

  $\varphi_{\mathbf{S}}$ and $\psi_{\mathbf{T}}$ are *CNREs* over $\Sigma_{\mathbf{S}}$ and $\Sigma_{\mathbf{T}}$, resp.

- *L_1-to-L_2 mapping:* $\varphi_{\mathbf{S}} \in L_1$ and $\psi_{\mathbf{T}} \in L_2$
- *L-GAV mapping:* $\varphi_{\mathbf{S}} \in L$ and $\psi_{\mathbf{T}}$ is (x, a, y) with $a \in \Sigma_{\mathbf{T}}$

Graph mapping language: example

2RPQ-GAV:

$$(x, (\texttt{creator}^- \cdot \texttt{creator})^+, y) \longrightarrow (x, \texttt{connected}, y)$$

Graph mapping language: example

2RPQ-GAV:

$$(\texttt{creator}^{-} \cdot \texttt{creator})^{+} \quad \longrightarrow \quad \texttt{connected}$$

Graph mapping language: example

2RPQ-GAV:

$$(\texttt{creator}^{-} \cdot \texttt{creator})^{+} \quad \longrightarrow \quad \texttt{connected}$$

C2RPQ-to-CRPQ:

$$(y, \texttt{creator}^{-}, x) \wedge (x, \texttt{partOf} \cdot \texttt{series}, w) \longrightarrow (y, \texttt{makes}, x) \wedge (x, \texttt{inConf}, w)$$

Graph mapping language: example

2RPQ-GAV:

$$(\mathtt{creator}^{-} \cdot \mathtt{creator})^{+} \quad \longrightarrow \quad \mathtt{connected}$$

C2RPQ-to-CRPQ:

$$(y, \mathtt{creator}^{-}, x) \wedge (x, \mathtt{partOf} \cdot \mathtt{series}, w) \longrightarrow (y, \mathtt{makes}, x) \wedge (x, \mathtt{inConf}, w)$$

NRE-GAV:

$$(x, (\mathtt{creator}^{-} \cdot [\,\mathtt{partOf} \cdot \mathtt{series}\,] \cdot \mathtt{creator})^{+}, y) \longrightarrow (x, \mathtt{confConn}, y)$$

Graph mapping language: example

2RPQ-GAV:

$$(\texttt{creator}^- \cdot \texttt{creator})^+ \quad \longrightarrow \quad \texttt{connected}$$

C2RPQ-to-CRPQ:

$$(y, \texttt{creator}^-, x) \wedge (x, \texttt{partOf} \cdot \texttt{series}, w) \longrightarrow (y, \texttt{makes}, x) \wedge (x, \texttt{inConf}, w)$$

NRE-GAV:

$$(\texttt{creator}^- \cdot [\,\texttt{partOf} \cdot \texttt{series}\,] \cdot \texttt{creator})^+ \quad \longrightarrow \quad \texttt{confConn}$$

Solutions in graph data exchange

- Let $\mathcal{M} = (\Sigma_{\mathbf{S}}, \Sigma_{\mathbf{T}}, \mathcal{T})$ be a graph mapping
- Let $G_{\mathbf{S}}$ be a source graph database
- $G_{\mathbf{T}}$ is a *solution* for $G_{\mathbf{S}}$ under $\mathcal{M}$ if
  - for every $\varphi_{\mathbf{S}}(\bar{x}) \to \psi_{\mathbf{T}}(\bar{x})$ in $\mathcal{T}$ and
  - for every tuple $\bar{a}$ of values in $G_{\mathbf{S}}$, we have that

if $\bar{a}$ is in the evaluation of $\varphi_{\mathbf{S}}$ over $G_{\mathbf{S}}$, then
$\bar{a}$ is in the evaluation of $\psi_{\mathbf{T}}$ over $G_{\mathbf{T}}$.

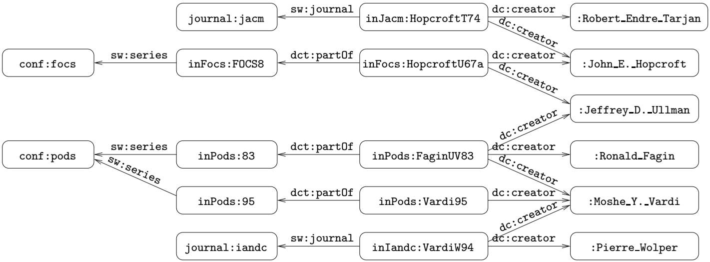
Solutions in graph data exchange

- Let $\mathcal{M} = (\Sigma_{\mathbf{S}}, \Sigma_{\mathbf{T}}, \mathcal{T})$ be a graph mapping
- Let $G_{\mathbf{S}}$ be a source graph database
- $G_{\mathbf{T}}$ is a *solution* for $G_{\mathbf{S}}$ under $\mathcal{M}$ if
  - for every $\varphi_{\mathbf{S}}(\bar{x}) \to \psi_{\mathbf{T}}(\bar{x})$ in $\mathcal{T}$ and
  - for every tuple $\bar{a}$ of values in $G_{\mathbf{S}}$, we have that

if $\bar{a}$ is in the evaluation of $\varphi_{\mathbf{S}}$ over $G_{\mathbf{S}}$, then
$\bar{a}$ is in the evaluation of $\psi_{\mathbf{T}}$ over $G_{\mathbf{T}}$.

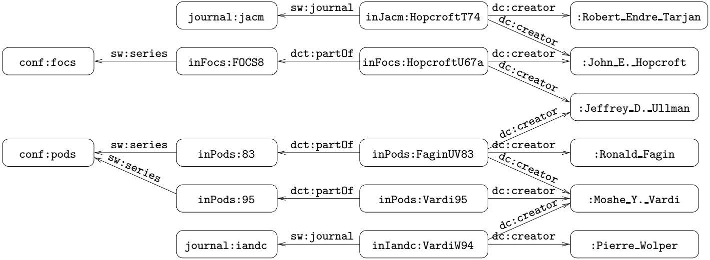
$\mathrm{Sol}_{\mathcal{M}}(G_{\mathbf{S}})$ is the set of solutions for $G_{\mathbf{S}}$ under $\mathcal{M}$.

Example

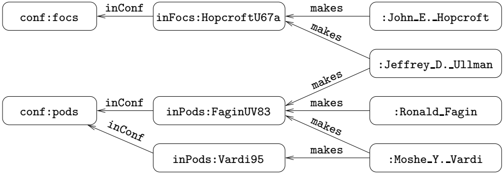


$$(y, \mathtt{creator}^{-}, x) \wedge (x, \mathtt{partOf} \cdot \mathtt{series}, w) \longrightarrow (y, \mathtt{makes}, x) \wedge (x, \mathtt{inConf}, w)$$

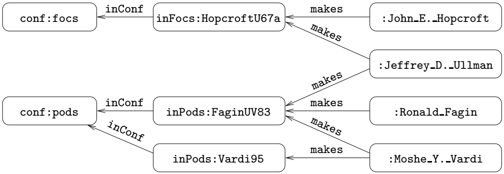
Example



$$(y, \texttt{creator}^{-}, x) \wedge (x, \texttt{partOf} \cdot \texttt{series}, w) \longrightarrow (y, \texttt{makes}, x) \wedge (x, \texttt{inConf}, w)$$

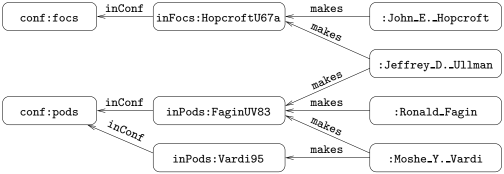


Example

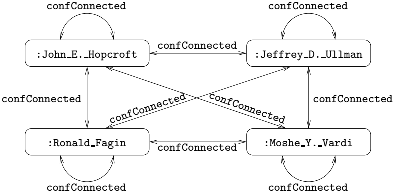


$$(\texttt{makes} \cdot \texttt{makes}^{-})^{+} \longrightarrow \texttt{confConnected}$$

Example



$$(\texttt{makes} \cdot \texttt{makes}^{-})^{+} \longrightarrow \texttt{confConnected}$$



Interesting expressive power

Example

Copy from source to target all paths of the form

$$a(aa)^*b$$

changing the first a by a', remaining aa by a'', and b by b'

We can express this by NRE-mappings

Interesting expressive power

Example

Copy from source to target all paths of the form

$$a(aa)^*b$$

changing the first a by a', remaining aa by a'', and b by b'

We can express this by NRE-mappings

$$a \cdot [(aa)^*b] \quad \rightarrow \quad a'$$

Interesting expressive power

Example

Copy from source to target all paths of the form

$$a(aa)^*b$$

changing the first a by a', remaining aa by a'', and b by b'

We can express this by NRE-mappings

$$\begin{aligned} a \cdot [(aa)^* b] &\rightarrow a' \\ [(a^- a^-)^* a^-] \cdot aa \cdot [(aa)^* b] &\rightarrow a'' \end{aligned}$$

Interesting expressive power

Example

Copy from source to target all paths of the form

$$a(aa)^*b$$

changing the first a by a', remaining aa by a'', and b by b'

We can express this by NRE-mappings

$$\begin{array}{rcl} a \cdot [(aa)^* b] & \rightarrow & a' \\ [(a^- a^-)^* a^-] \cdot aa \cdot [(aa)^* b] & \rightarrow & a'' \\ [(a^- a^-)^* a^-] \cdot b & \rightarrow & b' \end{array}$$

Interesting expressive power

Example

Copy from source to target all paths of the form

$$a(aa)^*b$$

changing the first a by a', remaining aa by a'', and b by b'

We can express this by NRE-mappings

$$\begin{aligned} a \cdot [(aa)^* b] &\rightarrow a' \\ [(a^- a^-)^* a^-] \cdot aa \cdot [(aa)^* b] &\rightarrow a'' \\ [(a^- a^-)^* a^-] \cdot b &\rightarrow b' \end{aligned}$$

Any regular source path can be *synchronized* in the same way

Outline

Graph mapping language

Computing solutions & answering queries

Composing graph schema mappings

Graph patterns as universal representatives

Graph patterns are graphs such that

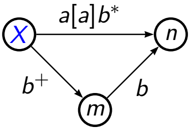
- Nodes can be labeled with *null values*
- Edges can be labeled with (nested) regular expressions

Graph patterns as universal representatives

Graph patterns are graphs such that

- Nodes can be labeled with *null values*
- Edges can be labeled with (nested) regular expressions

π:



Graph patterns: semantics

Semantics of graph patterns in terms of homomorphisms:

Given a pattern π, graph database G is in $\text{rep}(\pi)$ iff there exists homomorphism h from nulls in π to nodes in G s.t.

for every (u, expr, v) in π there is a path in G from $h(u)$ to $h(v)$ that satisfies expr.

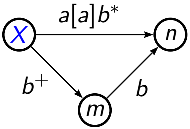
Graph patterns: semantics

Semantics of graph patterns in terms of homomorphisms:

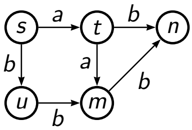
Given a pattern π, graph database G is in $\text{rep}(\pi)$ iff there exists homomorphism h from nulls in π to nodes in G s.t.

for every $(u, expr, v)$ in π there is a path in G from $h(u)$ to $h(v)$ that satisfies $expr$.

π:



G:



$X \longrightarrow s$

$G \in \text{rep}(\pi)$

Computing universal representatives

Definition

$\pi_{\mathbf{T}}$ is a *universal representative* for graph $G_{\mathbf{S}}$ under $\mathcal{M}$ if

$$\mathsf{Sol}_{\mathcal{M}}(G_{\mathbf{S}}) = \mathsf{rep}(\pi_{\mathbf{T}})$$

Computing universal representatives

Definition

$\pi_{\mathbf{T}}$ is a *universal representative* for graph $G_{\mathbf{S}}$ under $\mathcal{M}$ if

$$\mathrm{Sol}_{\mathcal{M}}(G_{\mathbf{S}}) = \mathrm{rep}(\pi_{\mathbf{T}})$$

Proposition

- *Given graph $G_{\mathbf{S}}$ and mapping $\mathcal{M}$, a universal representative always exists and can be computed in polynomial space*
- *For fixed $\mathcal{M}$ it can be computed in polynomial time*

just a simple adaptation of the chase procedure...

Feasible universal representative computation

Universal representatives can be in general of size exponential in the size of the mapping

Proposition

Computing universal representatives is $\mathbf{FP}^{\mathbf{NP}[\log]}$-hard even restricted to inputs ensuring univ representatives of polynomial size

Feasible universal representative computation

Universal representatives can be in general of size exponential in the size of the mapping

Proposition

Computing universal representatives is $\mathbf{FP}^{\mathbf{NP}[\log]}$*-hard even restricted to inputs ensuring univ representatives of polynomial size*

Proposition

Given NRE-to-CNRE mapping $\mathcal{M}$ *a universal representative can be computed in* $O(|G_{\mathbf{S}}|^2 \times |\mathcal{M}|)$ *(tight bound)*

Certain answers

Definition

$$\underline{\text{certain}}_{\mathcal{M}}(Q_{\mathbf{T}}, G_{\mathbf{S}}) = \bigcap_{G_{\mathbf{T}} \in \text{Sol}_{\mathcal{M}}(G_{\mathbf{S}})} Q_{\mathbf{T}}(G_{\mathbf{T}})$$

Certain answers

Definition

$$\underline{\text{certain}}_{\mathcal{M}}(Q_{\mathbf{T}}, G_{\mathbf{S}}) = \bigcap_{G_{\mathbf{T}} \in \text{Sol}_{\mathcal{M}}(G_{\mathbf{S}})} Q_{\mathbf{T}}(G_{\mathbf{T}})$$

Observation: if $\pi_{\mathbf{T}}$ is a unviersal representative, then

$$\underline{\text{certain}}_{\mathcal{M}}(Q_{\mathbf{T}}, G_{\mathbf{S}}) = \bigcap_{G_{\mathbf{T}} \in \text{rep}(\pi_{\mathbf{T}})} Q_{\mathbf{T}}(G_{\mathbf{T}})$$

Certain answers

Definition

$$\underline{\text{certain}}_{\mathcal{M}}(Q_{\mathbf{T}}, G_{\mathbf{S}}) = \bigcap_{G_{\mathbf{T}} \in \text{Sol}_{\mathcal{M}}(G_{\mathbf{S}})} Q_{\mathbf{T}}(G_{\mathbf{T}})$$

Observation: if $\pi_{\mathbf{T}}$ is a unviersal representative, then

$$\underline{\text{certain}}_{\mathcal{M}}(Q_{\mathbf{T}}, G_{\mathbf{S}}) = \bigcap_{G_{\mathbf{T}} \in \text{rep}(\pi_{\mathbf{T}})} Q_{\mathbf{T}}(G_{\mathbf{T}})$$

CertAns

Input: Graph $G_{\mathbf{S}}$, mapping $\mathcal{M}$, target query $Q_{\mathbf{T}}$, and tuple $\bar{a}$

Ouput: Is $\bar{a}$ in $\underline{\text{certain}}_{\mathcal{M}}(Q_{\mathbf{T}}, G_{\mathbf{S}})$?

Complexity of computing certain answers

Theorem

(1) CertAns *is in EXPSPACE for CNRE-to-CNRE mappings and CNRE queries*

(2) CertAns *is EXPSPACE-hard for CRPQ-to-CRPQ mappings and CRPQ queries*

Complexity of computing certain answers

Theorem

(1) CERTANS *is in EXPSPACE for CNRE-to-CNRE mappings and CNRE queries*

(2) CERTANS *is EXPSPACE-hard for CRPQ-to-CRPQ mappings and CRPQ queries*

- (2) follows from known EXPSPACE-hard complexity of query containment for CRPQs (Calvanese et al.)

Complexity of computing certain answers

Theorem

(1) CertAns *is in EXPSPACE for CNRE-to-CNRE mappings and CNRE queries*

(2) CertAns *is EXPSPACE-hard for CRPQ-to-CRPQ mappings and CRPQ queries*

- (2) follows from known EXPSPACE-hard complexity of query containment for CRPQs (Calvanese et al.)
- (1) needed the adaptation of techniques in (Calvanese et al.):

Complexity of computing certain answers

Theorem

(1) CertAns *is in EXPSPACE for CNRE-to-CNRE mappings and CNRE queries*

(2) CertAns *is EXPSPACE-hard for CRPQ-to-CRPQ mappings and CRPQ queries*

- (2) follows from known EXPSPACE-hard complexity of query containment for CRPQs (Calvanese et al.)
- (1) needed the adaptation of techniques in (Calvanese et al.):

  Alternating 2-way automata to represent *canonical solutions*

Complexity of computing certain answers

Theorem

(1) CertAns *is in EXPSPACE for CNRE-to-CNRE mappings and CNRE queries*

(2) CertAns *is EXPSPACE-hard for CRPQ-to-CRPQ mappings and CRPQ queries*

- (2) follows from known EXPSPACE-hard complexity of query containment for CRPQs (Calvanese et al.)
- (1) needed the adaptation of techniques in (Calvanese et al.):
  Alternating 2-way automata to represent *canonical solutions*

$$e_1 \cdot [e_2] \cdot [e_3] \cdot (e_4 \cdot [e_5])^*$$

Complexity of computing certain answers

Theorem

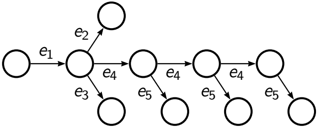
(1) CERTANS *is in EXPSPACE for CNRE-to-CNRE mappings and CNRE queries*

(2) CERTANS *is EXPSPACE-hard for CRPQ-to-CRPQ mappings and CRPQ queries*

- (2) follows from known EXPSPACE-hard complexity of query containment for CRPQs (Calvanese et al.)
- (1) needed the adaptation of techniques in (Calvanese et al.):

  Alternating 2-way automata to represent *canonical solutions*

$$e_1 \cdot [e_2] \cdot [e_3] \cdot (e_4 \cdot [e_5])^*$$



Complexity of computing certain answers

Theorem

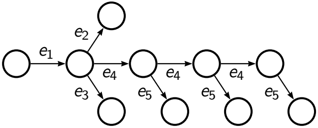
(1) CertAns *is in EXPSPACE for CNRE-to-CNRE mappings and CNRE queries*

(2) CertAns *is EXPSPACE-hard for CRPQ-to-CRPQ mappings and CRPQ queries*

- (2) follows from known EXPSPACE-hard complexity of query containment for CRPQs (Calvanese et al.)
- (1) needed the adaptation of techniques in (Calvanese et al.):

  Alternating 2-way automata to represent *canonical solutions*

  $e_1 \cdot [e_2] \cdot [e_3] \cdot (e_4 \cdot [e_5])^*$



  need to run over (a restricted class of) trees

Even data complexity is hard

$\textsc{CertAns}(\mathcal{M}, Q_{\mathbf{T}})$

Input: Graph $G_{\mathbf{S}}$, and tuple $\bar{a}$

Ouput: Is $\bar{a}$ in $\underline{\text{certain}}_{\mathcal{M}}(Q_{\mathbf{T}}, G_{\mathbf{S}})$?

Even data complexity is hard

$\textsc{CertAns}(\mathcal{M}, Q_{\mathbf{T}})$

Input: Graph $G_{\mathbf{S}}$, and tuple $\bar{a}$

Ouput: Is $\bar{a}$ in $\underline{\text{certain}}_{\mathcal{M}}(Q_{\mathbf{T}}, G_{\mathbf{S}})$?

Theorem

1. $\textsc{CertAns}(\mathcal{M}, Q_{\mathbf{T}})$ *is coNP-complete for every CNRE-to-CNRE mapping and CNRE query.*
2. $\textsc{CertAns}(\mathcal{M}, Q_{\mathbf{T}})$ *is coNP-hard even for RPQ-to-RPQ mappings and RPQ queries.*

Even data complexity is hard

$\textsc{CertAns}(\mathcal{M}, Q_{\mathbf{T}})$

Input: Graph $G_{\mathbf{S}}$, and tuple $\bar{a}$

Ouput: Is $\bar{a}$ in $\underline{\text{certain}}_{\mathcal{M}}(Q_{\mathbf{T}}, G_{\mathbf{S}})$?

Theorem

1. $\textsc{CertAns}(\mathcal{M}, Q_{\mathbf{T}})$ *is coNP-complete for every CNRE-to-CNRE mapping and CNRE query.*
2. $\textsc{CertAns}(\mathcal{M}, Q_{\mathbf{T}})$ *is coNP-hard even for RPQ-to-RPQ mappings and RPQ queries.*

In the paper:

- Structural properties ensuring tractable data complexity

Tractable query answering

High complexity if we allow conjunctions in rules or regular expressions in the right-side

- Need to focus on GAV mappings.

Tractable query answering

High complexity if we allow conjunctions in rules or regular expressions in the right-side

- Need to focus on GAV mappings.

By just computing a universal representative we obtain

Corollary

For NRE-GAV mappings and NRE queries, CertAns *can be solved in time*

$$O(|G_{\mathbf{S}}|^2 \times |\mathcal{M}| \times \ |expr|)$$

Tractable query answering

High complexity if we allow conjunctions in rules or regular expressions in the right-side

- Need to focus on GAV mappings.

By just computing a universal representative we obtain

Corollary

For NRE-GAV mappings and NRE queries, CERTANS *can be solved in time*

$$O(|G_{\mathbf{S}}|^2 \times |\mathcal{M}| \times \ |expr|)$$

But we can do better

Tractable query answering

High complexity if we allow conjunctions in rules or regular expressions in the right-side

- Need to focus on GAV mappings.

By just computing a universal representative we obtain

Corollary

For NRE-GAV mappings and NRE queries, CERTANS *can be solved in time*

$$O(|G_{\mathbf{S}}|^2 \times |\mathcal{M}| \times \ |expr|)$$

But we can do better

Theorem

For NRE-GAV mappings and NRE queries, CERTANS *can be solved in time*

$$O(|G_{\mathbf{S}}| \times |\mathcal{M}| \times \ |expr|)$$

Outline

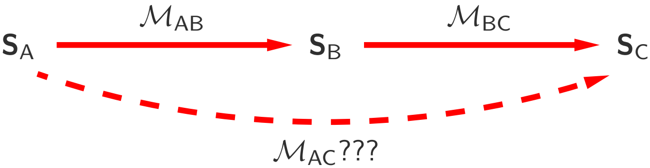
Composing mappings

$\mathbf{S}_A$ $\mathbf{S}_B$ $\mathbf{S}_C$

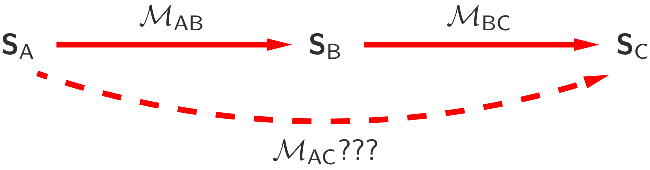
Composing mappings

$$\mathbf{S}_{\mathrm{A}} \xrightarrow{\mathcal{M}_{\mathrm{AB}}} \mathbf{S}_{\mathrm{B}} \xrightarrow{\mathcal{M}_{\mathrm{BC}}} \mathbf{S}_{\mathrm{C}}$$

Composing mappings



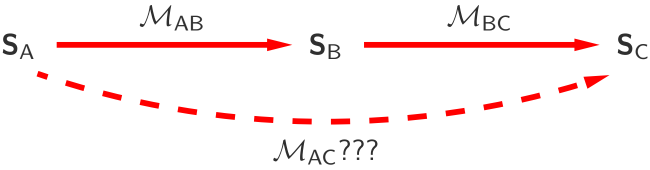
Composing mappings



Intuitively, $\mathcal{M}_{AC}$ must have the same effect as applying $\mathcal{M}_{AB}$ and then $\mathcal{M}_{BC}$

$$\mathcal{M}_{AC} = \mathcal{M}_{AB} \circ \mathcal{M}_{BC}$$

Composing mappings



Intuitively, $\mathcal{M}_{AC}$ must have the same effect as applying $\mathcal{M}_{AB}$ and then $\mathcal{M}_{BC}$

$$\mathcal{M}_{AC} = \mathcal{M}_{AB} \circ \mathcal{M}_{BC}$$

- how to compute the composition?
- what is the language needed to express it?
- is there a language closed under composition?

CRPQs are not suitable for composing graph mappings

CRPQs are not suitable for composing graph mappings

Example

$\mathcal{M}_1$: $\exists u \; (x, \texttt{creator}^-, y) \wedge (y, \texttt{partOf} \cdot \texttt{series}, u) \; \rightarrow \; (x, \texttt{confAuthor}, y)$

CRPQs are not suitable for composing graph mappings

Example

$\mathcal{M}_1$: $\exists u\ (x, \texttt{creator}^-, y) \wedge (y, \texttt{partOf} \cdot \texttt{series}, u) \ \rightarrow\ (x, \texttt{confAuthor}, y)$

$\mathcal{M}_2$: $(x, (\texttt{confAuthor} \cdot \texttt{confAuthor}^-)^+, y) \ \rightarrow\ (x, \texttt{confConnected}, y)$

CRPQs are not suitable for composing graph mappings

Example

$\mathcal{M}_1$: $\exists u \ (x, \texttt{creator}^-, y) \wedge (y, \texttt{partOf} \cdot \texttt{series}, u) \ \rightarrow \ (x, \texttt{confAuthor}, y)$

$\mathcal{M}_2$: $(x, (\texttt{confAuthor} \cdot \texttt{confAuthor}^-)^+, y) \ \rightarrow \ (x, \texttt{confConnected}, y)$

$$\mathcal{M}_1 \circ \mathcal{M}_2???$$

CRPQs are not suitable for composing graph mappings

Example

$\mathcal{M}_1$: $\exists u\ (x, \mathtt{creator}^-, y) \wedge (y, \mathtt{partOf} \cdot \mathtt{series}, u) \ \rightarrow\ (x, \mathtt{confAuthor}, y)$

$\mathcal{M}_2$: $(x, (\mathtt{confAuthor} \cdot \mathtt{confAuthor}^-)^+, y) \ \rightarrow\ (x, \mathtt{confConnected}, y)$

$$\mathcal{M}_1 \circ \mathcal{M}_2???$$

Example

$\mathcal{M}_1$: $(x, \mathtt{creator}^- \cdot [\ \mathtt{partOf} \cdot \mathtt{series}\ ], y) \ \rightarrow\ (x, \mathtt{confAuthor}, y)$

CRPQs are not suitable for composing graph mappings

Example

$\mathcal{M}_1$: $\exists u \; (x, \texttt{creator}^-, y) \wedge (y, \texttt{partOf} \cdot \texttt{series}, u) \; \rightarrow \; (x, \texttt{confAuthor}, y)$

$\mathcal{M}_2$: $(x, (\texttt{confAuthor} \cdot \texttt{confAuthor}^-)^+, y) \; \rightarrow \; (x, \texttt{confConnected}, y)$

$$\mathcal{M}_1 \circ \mathcal{M}_2???$$

Example

$\mathcal{M}_1$: $(x, \texttt{creator}^- \cdot [\, \texttt{partOf} \cdot \texttt{series} \,], y) \; \rightarrow \; (x, \texttt{confAuthor}, y)$

$\mathcal{M}_2$: $(x, (\texttt{confAuthor} \cdot \texttt{confAuthor}^-)^+, y) \; \rightarrow \; (x, \texttt{confConnected}, y)$

CRPQs are not suitable for composing graph mappings

Example

$\mathcal{M}_1$: $\exists u \;\; (x, \mathtt{creator}^-, y) \wedge (y, \mathtt{partOf} \cdot \mathtt{series}, u) \;\rightarrow\; (x, \mathtt{confAuthor}, y)$

$\mathcal{M}_2$: $(x, (\mathtt{confAuthor} \cdot \mathtt{confAuthor}^-)^+, y) \;\rightarrow\; (x, \mathtt{confConnected}, y)$

$$\mathcal{M}_1 \circ \mathcal{M}_2???$$

Example

$\mathcal{M}_1$: $\mathtt{creator}^- \cdot [\, \mathtt{partOf} \cdot \mathtt{series} \,] \quad \rightarrow \quad \mathtt{confAuthor}$

$\mathcal{M}_2$: $(\mathtt{confAuthor} \cdot \mathtt{confAuthor}^-)^+ \quad \rightarrow \quad \mathtt{confConnected}$

CRPQs are not suitable for composing graph mappings

Example

$\mathcal{M}_1$: $\exists u\ (x, \texttt{creator}^-, y) \wedge (y, \texttt{partOf} \cdot \texttt{series}, u) \ \rightarrow\ (x, \texttt{confAuthor}, y)$

$\mathcal{M}_2$: $(x, (\texttt{confAuthor} \cdot \texttt{confAuthor}^-)^+, y) \ \rightarrow\ (x, \texttt{confConnected}, y)$

$$\mathcal{M}_1 \circ \mathcal{M}_2???$$

Example

$\mathcal{M}_1$: $\texttt{creator}^- \cdot [\, \texttt{partOf} \cdot \texttt{series} \,] \quad \rightarrow \quad \texttt{confAuthor}$

$\mathcal{M}_2$: $(\texttt{confAuthor} \cdot \texttt{confAuthor}^-)^+ \quad \rightarrow \quad \texttt{confConnected}$

$$\mathcal{M}_1 \circ \mathcal{M}_2:$$
$$((\texttt{creator}^- \cdot [\, \texttt{partOf} \cdot \texttt{series} \,]) \cdot ([\, \texttt{partOf} \cdot \texttt{series} \,] \cdot \texttt{creator}))^+ \ \rightarrow\ \texttt{confConnected}$$

CRPQs are not suitable for composing graph mappings

Example

$\mathcal{M}_1$: $\exists u\ (x, \mathtt{creator}^-, y) \wedge (y, \mathtt{partOf} \cdot \mathtt{series}, u) \ \rightarrow\ (x, \mathtt{confAuthor}, y)$

$\mathcal{M}_2$: $(x, (\mathtt{confAuthor} \cdot \mathtt{confAuthor}^-)^+, y) \ \rightarrow\ (x, \mathtt{confConnected}, y)$

$$\mathcal{M}_1 \circ \mathcal{M}_2???$$

Example

$\mathcal{M}_1$: $\mathtt{creator}^- \cdot [\ \mathtt{partOf} \cdot \mathtt{series}\ ] \quad \rightarrow \quad \mathtt{confAuthor}$

$\mathcal{M}_2$: $(\mathtt{confAuthor} \cdot \mathtt{confAuthor}^-)^+ \quad \rightarrow \quad \mathtt{confConnected}$

$$\mathcal{M}_1 \circ \mathcal{M}_2:$$
$$(\mathtt{creator}^- \cdot [\ \mathtt{partOf} \cdot \mathtt{series}\ ] \cdot \mathtt{creator})^+ \ \rightarrow\ \mathtt{confConnected}$$

NRE-GAV mappings are closed under composition

Theorem

The composition of NRE-GAV mappings can always be specified by an NRE-GAV mapping

NRE-GAV mappings are closed under composition

Theorem

The composition of NRE-GAV mappings can always be specified by an NRE-GAV mapping

Corollary

The composition of tree-shaped C2RPQ-GAV mappings can always be specified by an NRE-GAV mapping

Composition in the presence of conjunctions

Known result in relational data exchange:

- CQ-GAV mappings are closed under composition

Composition in the presence of conjunctions

Known result in relational data exchange:

- CQ-GAV mappings are closed under composition

Proposition

There exist CRPQ-GAV mappings s.t. their composition cannot be specified by a CNRE-GAV mapping

Composition in the presence of conjunctions

Known result in relational data exchange:

- CQ-GAV mappings are closed under composition

Proposition

There exist CRPQ-GAV mappings s.t. their composition cannot be specified by a CNRE-GAV mapping

Open question:
What is the language needed to compose CRPQ-GAV mappings?

Concluding remarks

We have initiated the study of Graph Data Exchange

- Some techniques can be adapted from the relational case
- Query answering is highly complex
- Schema mapping operators is a challenging topic
- NREs add expressive power compared with 2RPQs maintaining the complexity plus giving good properties for composition

Concluding remarks

We have initiated the study of Graph Data Exchange

- Some techniques can be adapted from the relational case
- Query answering is highly complex
- Schema mapping operators is a challenging topic
- NREs add expressive power compared with 2RPQs maintaining the complexity plus giving good properties for composition

We would like to explore new formalisms to specify mappings

- Can we add expressive power maintaining the complexity?

Concluding remarks

We have initiated the study of Graph Data Exchange

- Some techniques can be adapted from the relational case
- Query answering is highly complex
- Schema mapping operators is a challenging topic
- NREs add expressive power compared with 2RPQs maintaining the complexity plus giving good properties for composition

We would like to explore new formalisms to specify mappings

- Can we add expressive power maintaining the complexity?
  - Good candidate to start: GraphXPath

Concluding remarks

We have initiated the study of Graph Data Exchange

- Some techniques can be adapted from the relational case
- Query answering is highly complex
- Schema mapping operators is a challenging topic
- NREs add expressive power compared with 2RPQs maintaining the complexity plus giving good properties for composition

We would like to explore new formalisms to specify mappings

- Can we add expressive power maintaining the complexity?
  - Good candidate to start: GraphXPath
- More natural (and expressive) synchronization between paths

$$(a/a')(aa/a'')^*(b/b')$$

Schema Mappings and Data Exchange for Graph Databases

Pablo Barceló   Jorge Pérez   Juan Reutter

Universidad de Chile, PUC Chile

Outline