Open-World Finite Query Answering with Number Restrictions

Antoine Amarilli[1,2], Michael Benedikt[2]

[1]Télécom ParisTech, Paris, France

[2]University of Oxford, Oxford, United Kingdom

July 7, 2015

Open-world query answering (QA)

- Open-world query answering:
  - Relational instance I (ground facts), correct but incomplete
  - Boolean conjunctive query q
  - Consider all possible completions $J \supseteq I$
  - Is q certain, i.e., holds on all completions? If yes, $I \models_{\mathrm{unr}} q$

Constraints

- Impose constraints on the possible completions
  - Inclusion dependencies: some facts imply more facts:
    $\forall xd \text{ Employee}(x, d) \Rightarrow \exists y \text{ Advises}(x, y, d)$
    $\text{Employee}[1, 2] \subseteq \text{Advises}[1, 3]$

Constraints

- Impose constraints on the possible completions
  - Inclusion dependencies: some facts imply more facts:
    $\forall xd\ \text{Employee}(x, d) \Rightarrow \exists y\ \text{Advises}(x, y, d)$
    $\text{Employee}[1, 2] \subseteq \text{Advises}[1, 3]$

→ Unary inclusion dependencies (UIDs): one shared position

Constraints

- Impose constraints on the possible completions
  - Inclusion dependencies: some facts imply more facts:
    $\forall xd\ \mathsf{Employee}(x, d) \Rightarrow \exists y\ \mathsf{Advises}(x, y, d)$
    $\mathsf{Employee}[1, 2] \subseteq \mathsf{Advises}[1, 3]$

  $\rightarrow$ Unary inclusion dependencies (UIDs): one shared position

  - Functional dependencies (FDs): uniqueness constraints:
    $\forall xx'yd\ \mathsf{Advises}(x, y, d) \wedge \mathsf{Advises}(x', y, d) \rightarrow x = x'$
    $\mathsf{Advises}[2, 3] \rightarrow \mathsf{Advises}[1]$

Constraints

- Impose constraints on the possible completions
  - Inclusion dependencies: some facts imply more facts:
    $\forall xd\ \mathrm{Employee}(x, d) \Rightarrow \exists y\ \mathrm{Advises}(x, y, d)$
    $\mathrm{Employee}[1, 2] \subseteq \mathrm{Advises}[1, 3]$
  - → Unary inclusion dependencies (UIDs): one shared position
  - Functional dependencies (FDs): uniqueness constraints:
    $\forall xx'yd\ \mathrm{Advises}(x, y, d) \wedge \mathrm{Advises}(x', y, d) \rightarrow x = x'$
    $\mathrm{Advises}[2, 3] \rightarrow \mathrm{Advises}[1]$

→ Consider all $J \supseteq I$ that satisfy the constraints

→ Is q certain on all such completions? If yes, $I, \Sigma \models_{\mathrm{unr}} q$

Finite vs infinite

Instance: List of adviser–advisee pairs: Advises$(a, b), \ldots$

UID: Each advisee advises someone
Advises$[2] \subseteq$ Advises$[1]$

FD: Each advisee has only one adviser
Advises$[2] \rightarrow$ Advises$[1]$

Query: Are all advisers themselves advised by someone?

Finite vs infinite

Instance: List of adviser–advisee pairs: $\mathsf{Advises}(a, b), \ldots$

UID: Each advisee advises someone
$\mathsf{Advises}[2] \subseteq \mathsf{Advises}[1]$

FD: Each advisee has only one adviser
$\mathsf{Advises}[2] \rightarrow \mathsf{Advises}[1]$

Query: Are all advisers themselves advised by someone?

- q is not certain:
  $\mathsf{Advises}(a, b), \mathsf{Advises}(b, b_2), \mathsf{Advises}(b_2, b_3), \ldots$
- q is certain on all finite completions

Finite vs infinite

Instance: List of adviser–advisee pairs: $\text{Advises}(a, b), \ldots$

UID: Each advisee advises someone
$\text{Advises}[2] \subseteq \text{Advises}[1]$

FD: Each advisee has only one adviser
$\text{Advises}[2] \rightarrow \text{Advises}[1]$

Query: Are all advisers themselves advised by someone?

- q is not certain:
  $\text{Advises}(a, b), \text{Advises}(b, b_2), \text{Advises}(b_2, b_3), \ldots$
- q is certain on all finite completions

$\rightarrow$ Finite QA: only finite completions. If yes, $I, \Sigma \models_{\text{fin}} q$

Finite vs infinite

Instance: List of adviser–advisee pairs: $\mathsf{Advises}(a, b), \ldots$

UID: Each advisee advises someone
$\mathsf{Advises}[2] \subseteq \mathsf{Advises}[1]$

FD: Each advisee has only one adviser
$\mathsf{Advises}[2] \rightarrow \mathsf{Advises}[1]$

Query: Are all advisers themselves advised by someone?

- q is not certain:
  $\mathsf{Advises}(a, b), \mathsf{Advises}(b, b_2), \mathsf{Advises}(b_2, b_3), \ldots$
- q is certain on all finite completions

→ Finite QA: only finite completions. If yes, $I, \Sigma \models_{\mathrm{fin}} q$

→ Σ finitely controllable: $\forall I \, \forall q, \; I, \Sigma \models_{\mathrm{fin}} q$ iff $I, \Sigma \models_{\mathrm{unr}} q$

Table of contents

Existing results

- (Finite) open-world query answering is undecidable for IDs and FDs [Calì et al., 2003]

Existing results

- (Finite) open-world query answering is undecidable for IDs and FDs [Calì et al., 2003]
- IDs alone are finitely controllable [Rosati, 2006]

Existing results

- (Finite) open-world query answering is undecidable for IDs and FDs [Calì et al., 2003]
- IDs alone are finitely controllable [Rosati, 2006]
- Infinite open-world query answering is decidable for UIDs and FDs [Calì et al., 2012] but not finitely controllable (see above; [Rosati, 2006])

Existing results

- (Finite) open-world query answering is undecidable for IDs and FDs [Calì et al., 2003]
- IDs alone are finitely controllable [Rosati, 2006]
- Infinite open-world query answering is decidable for UIDs and FDs [Calì et al., 2012] but not finitely controllable (see above; [Rosati, 2006])
- (Finite) open-world query answering is decidable for IDs and FDs with relations of arity ≤ 2 [Pratt-Hartmann, 2009, Ibáñez-García et al., 2014]

Problem and main result

- Study finite QA for UIDs and FDs on arbitrary signatures
- Is it decidable? What is the complexity?

Problem and main result

- Study finite QA for UIDs and FDs on arbitrary signatures
- Is it decidable? What is the complexity?

Theorem

Finite open-world query answering for UIDs and FDs has PTIME data complexity and NP-complete combined complexity.

Main idea

- Find which UIDs and FDs are implied over finite instances
  - → This is PTIME [Cosmadakis et al., 1990]
  - → It differs from unrestricted implication
- Finite closure: add all finitely implied dependencies

Main idea

- Find which UIDs and FDs are implied over finite instances
  - → This is PTIME [Cosmadakis et al., 1990]
  - → It differs from unrestricted implication
- Finite closure: add all finitely implied dependencies

→ Taking the finite closure ensure finite controllability?

Main idea

- Find which UIDs and FDs are implied over finite instances
  - → This is PTIME [Cosmadakis et al., 1990]
  - → It differs from unrestricted implication
- Finite closure: add all finitely implied dependencies

→ Taking the finite closure ensure finite controllability?

Theorem

For any instance I, conjunctive query q, UIDs and FDs Σ, letting Σ^* be the finite closure of Σ, we have $I, \Sigma \models_{\mathrm{fin}} q$ iff $I, \Sigma^* \models_{\mathrm{unr}} q$.

Table of contents

Rephrasing the result

- Assume that the UIDs and FDs Σ are finitely closed

Rephrasing the result

- Assume that the UIDs and FDs Σ are finitely closed
- Assume that an infinite completion J satisfies Σ but not q

Rephrasing the result

- Assume that the UIDs and FDs Σ are finitely closed
- Assume that an infinite completion J satisfies Σ but not q

$\rightarrow$ Construct a finite completion J' satisfying Σ but not q:

- Start with I
- Follow J and add facts to satisfy the UIDs of Σ...
- ... using finitely many elements (reuse them)
- ... but respecting the FDs of Σ
- ... and without making q inadvertantly true

Constructing the finite completion

Make assumptions, we lift them from bottom to top:

Constructing the finite completion

Make assumptions, we lift them from bottom to top:

5. Assume that the query q is acyclic
   e.g., $\exists xy\ R(x)S(x,y)$ but not $\exists xyz\ R(x,y)S(y,z)T(z,x)$

Constructing the finite completion

Make assumptions, we lift them from bottom to top:

5. Assume that the query q is acyclic
   e.g., $\exists xy\ R(x)S(x,y)$ but not $\exists xyz\ R(x,y)S(y,z)T(z,x)$

4. Assume that there are only unary FDs
   e.g., $R[1] \rightarrow R[2]$ but not $R[1,3] \rightarrow R[2]$

Constructing the finite completion

Make assumptions, we lift them from bottom to top:

5. Assume that the query q is acyclic
   e.g., $\exists xy\ R(x) S(x, y)$ but not $\exists xyz\ R(x, y) S(y, z)\, T(z, x)$

4. Assume that there are only unary FDs
   e.g., $R[1] \rightarrow R[2]$ but not $R[1, 3] \rightarrow R[2]$

3. Assume a certain reversibility property on the UIDs and FDs

Constructing the finite completion

Make assumptions, we lift them from bottom to top:

5. Assume that the query q is acyclic
   e.g., $\exists xy\, R(x)S(x,y)$ but not $\exists xyz\, R(x,y)S(y,z)\,T(z,x)$

4. Assume that there are only unary FDs
   e.g., $R[1] \rightarrow R[2]$ but not $R[1,3] \rightarrow R[2]$

3. Assume a certain reversibility property on the UIDs and FDs

2. Assume we have the trivial query $\perp$

Constructing the finite completion

Make assumptions, we lift them from bottom to top:

5. Assume that the query q is acyclic
   e.g., $\exists xy\, R(x) S(x,y)$ but not $\exists xyz\, R(x,y) S(y,z)\, T(z,x)$
4. Assume that there are only unary FDs
   e.g., $R[1] \rightarrow R[2]$ but not $R[1,3] \rightarrow R[2]$
3. Assume a certain reversibility property on the UIDs and FDs
2. Assume we have the trivial query $\perp$
1. Assume that the relations all have arity two

Constructing the finite completion

Make assumptions, we lift them from bottom to top:

5. Assume that the query q is acyclic
   e.g., $\exists xy\, R(x) S(x, y)$ but not $\exists xyz\, R(x, y) S(y, z)\, T(z, x)$
4. Assume that there are only unary FDs
   e.g., $R[1] \rightarrow R[2]$ but not $R[1, 3] \rightarrow R[2]$
3. Assume a certain reversibility property on the UIDs and FDs
2. Assume we have the trivial query $\perp$
1. Assume that the relations all have arity two
0. Solve the problem!

Constructing the finite completion

Make assumptions, we lift them from bottom to top:

5. Assume that the query q is acyclic
   e.g., $\exists xy\ R(x)S(x,y)$ but not $\exists xyz\ R(x,y)S(y,z)T(z,x)$
4. Assume that there are only unary FDs
   e.g., $R[1] \rightarrow R[2]$ but not $R[1,3] \rightarrow R[2]$
3. Assume a certain reversibility property on the UIDs and FDs
2. Assume we have the trivial query $\bot$
1. Assume that the relations all have arity two
0. Solve the problem!
   $\rightarrow$ Exemplified soon

Constructing the finite completion

Make assumptions, we lift them from bottom to top:

5. Assume that the query q is acyclic
   e.g., $\exists xy\ R(x)S(x,y)$ but not $\exists xyz\ R(x,y)S(y,z)T(z,x)$

4. Assume that there are only unary FDs
   e.g., $R[1] \rightarrow R[2]$ but not $R[1,3] \rightarrow R[2]$

3. Assume a certain reversibility property on the UIDs and FDs

2. Assume we have the trivial query $\perp$

1. Assume that the relations all have arity two
   - $\rightarrow$ Reuse elements or extend previous case – skipped

0. Solve the problem!
   - $\rightarrow$ Exemplified soon

Constructing the finite completion

Make assumptions, we lift them from bottom to top:

5. Assume that the query q is acyclic
   e.g., $\exists xy\ R(x)S(x,y)$ but not $\exists xyz\ R(x,y)S(y,z)T(z,x)$
4. Assume that there are only unary FDs
   e.g., $R[1] \rightarrow R[2]$ but not $R[1,3] \rightarrow R[2]$
3. Assume a certain reversibility property on the UIDs and FDs
2. Assume we have the trivial query $\perp$
   - $\rightarrow$ Exemplified soon
1. Assume that the relations all have arity two
   - $\rightarrow$ Reuse elements or extend previous case – skipped
0. Solve the problem!
   - $\rightarrow$ Exemplified soon

Constructing the finite completion

Make assumptions, we lift them from bottom to top:

5. Assume that the query q is acyclic
   e.g., $\exists xy\ R(x) S(x, y)$ but not $\exists xyz\ R(x, y) S(y, z) T(z, x)$
4. Assume that there are only unary FDs
   e.g., $R[1] \to R[2]$ but not $R[1, 3] \to R[2]$
3. Assume a certain reversibility property on the UIDs and FDs
   $\rightarrow$ Decomposition of UIDs based on [Cosmadakis et al., 1990] – skipped
2. Assume we have the trivial query $\perp$
   $\rightarrow$ Exemplified soon
1. Assume that the relations all have arity two
   $\rightarrow$ Reuse elements or extend previous case – skipped
0. Solve the problem!
   $\rightarrow$ Exemplified soon

Constructing the finite completion

Make assumptions, we lift them from bottom to top:

5. Assume that the query q is acyclic
   e.g., $\exists xy\, R(x) S(x,y)$ but not $\exists xyz\, R(x,y) S(y,z) T(z,x)$
4. Assume that there are only unary FDs
   e.g., $R[1] \rightarrow R[2]$ but not $R[1,3] \rightarrow R[2]$
   - $\rightarrow$ Recombination of elements in patterns when reusing – skipped
3. Assume a certain reversibility property on the UIDs and FDs
   - $\rightarrow$ Decomposition of UIDs based on [Cosmadakis et al., 1990] – skipped
2. Assume we have the trivial query $\perp$
   - $\rightarrow$ Exemplified soon
1. Assume that the relations all have arity two
   - $\rightarrow$ Reuse elements or extend previous case – skipped
0. Solve the problem!
   - $\rightarrow$ Exemplified soon

Constructing the finite completion

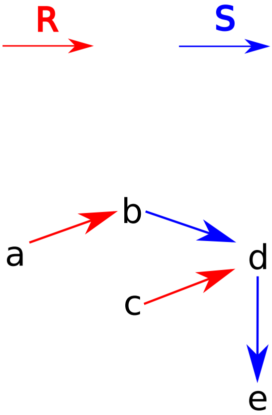
Make assumptions, we lift them from bottom to top:

5. Assume that the query q is acyclic
   e.g., $\exists xy\, R(x)S(x,y)$ but not $\exists xyz\, R(x,y)S(y,z)T(z,x)$
   - $\rightarrow$ Product with group of high girth inspired by [Otto, 2002] – skipped
4. Assume that there are only unary FDs
   e.g., $R[1] \rightarrow R[2]$ but not $R[1,3] \rightarrow R[2]$
   - $\rightarrow$ Recombination of elements in patterns when reusing – skipped
3. Assume a certain reversibility property on the UIDs and FDs
   - $\rightarrow$ Decomposition of UIDs based on [Cosmadakis et al., 1990] – skipped
2. Assume we have the trivial query $\perp$
   - $\rightarrow$ Exemplified soon
1. Assume that the relations all have arity two
   - $\rightarrow$ Reuse elements or extend previous case – skipped
0. Solve the problem!
   - $\rightarrow$ Exemplified soon

0. Satisfying UIDs and UFDs in arity-two

$$\xrightarrow{\mathbf{R}} \qquad \xrightarrow{\mathbf{S}}$$

0. Satisfying UIDs and UFDs in arity-two

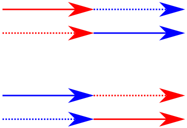


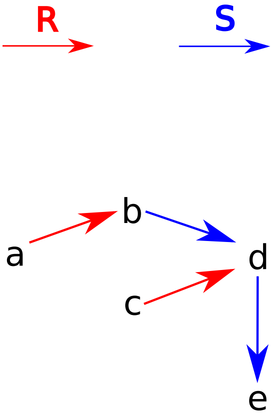
0. Satisfying UIDs and UFDs in arity-two

UIDs:

$R[2] \overset{\subseteq}{\supseteq} S[1]$

$S[2] \overset{\subseteq}{\supseteq} R[1]$





0. Satisfying UIDs and UFDs in arity-two

UIDs:

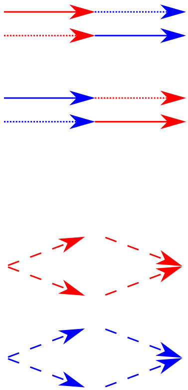
$R[2] \overset{\subseteq}{\supseteq} S[1]$

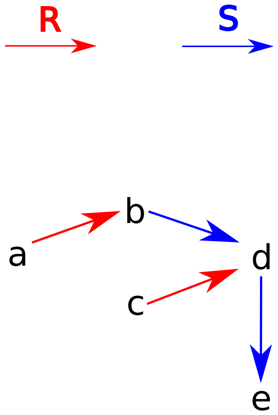
$S[2] \overset{\subseteq}{\supseteq} R[1]$

UFDs:

$R[1] \rightleftarrows R[2]$

$S[1] \rightleftarrows S[2]$



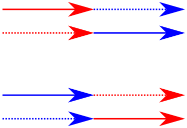


0. Satisfying UIDs and UFDs in arity-two

UIDs:

$R[2] \underset{\supseteq}{\subseteq} S[1]$

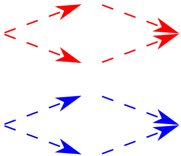
$S[2] \underset{\supseteq}{\subseteq} R[1]$

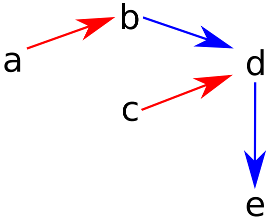


UFDs:

$R[1] \rightleftarrows R[2]$

$S[1] \rightleftarrows S[2]$





0. Satisfying UIDs and UFDs in arity-two

UIDs:

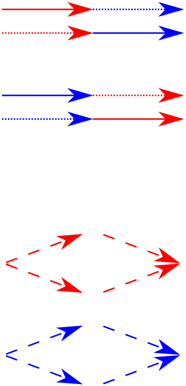
$R[2] \overset{\subseteq}{\supseteq} S[1]$

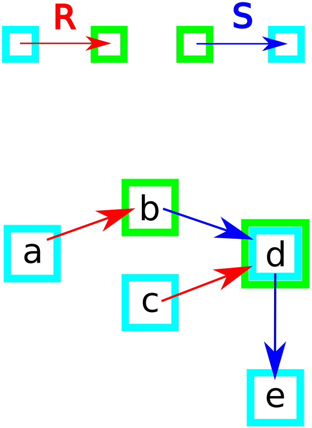
$S[2] \overset{\subseteq}{\supseteq} R[1]$

UFDs:

$R[1] \rightleftarrows R[2]$

$S[1] \rightleftarrows S[2]$

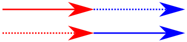




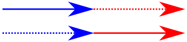
0. Satisfying UIDs and UFDs in arity-two

UIDs:

$R[2] \mathrel{\substack{\subseteq\\ \supseteq}} S[1]$



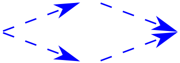
$S[2] \mathrel{\substack{\subseteq\\ \supseteq}} R[1]$

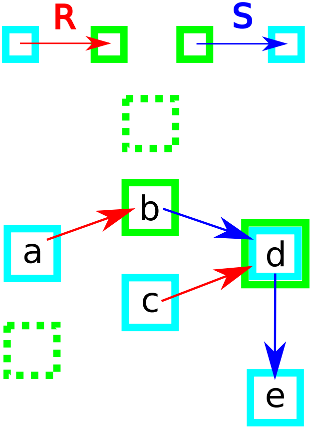


UFDs:

$R[1] \rightleftarrows R[2]$

$S[1] \rightleftarrows S[2]$



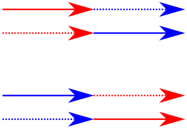


0. Satisfying UIDs and UFDs in arity-two

UIDs:

$R[2] \mathrel{\substack{\subseteq \\ \supseteq}} S[1]$

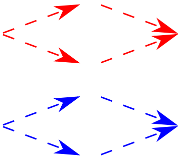
$S[2] \mathrel{\substack{\subseteq \\ \supseteq}} R[1]$

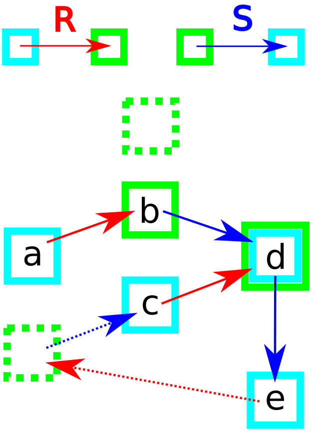


UFDs:

$R[1] \rightleftarrows R[2]$

$S[1] \rightleftarrows S[2]$



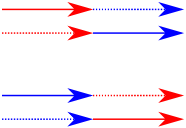


0. Satisfying UIDs and UFDs in arity-two

UIDs:

$R[2] \subseteq\supseteq S[1]$

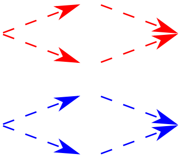
$S[2] \subseteq\supseteq R[1]$

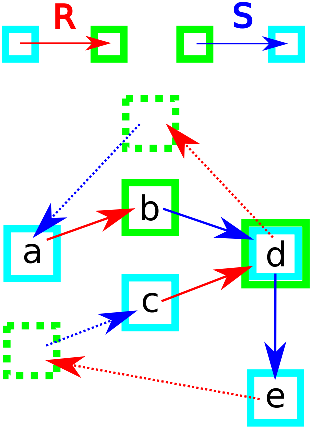


UFDs:

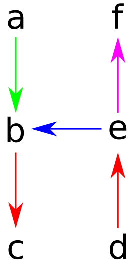
$R[1] \rightleftarrows R[2]$

$S[1] \rightleftarrows S[2]$





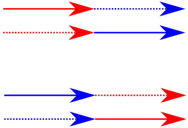
2. Adding acyclic queries



UIDs:

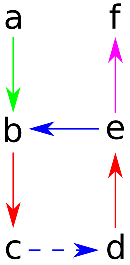
$R[2] \overset{\subseteq}{\supseteq} S[1]$

$S[2] \overset{\subseteq}{\supseteq} R[1]$



and UFDs...

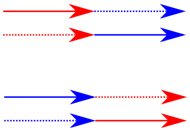
2. Adding acyclic queries



UIDs:

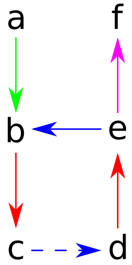
$R[2] \substack{\subseteq \\ \supseteq} S[1]$

$S[2] \substack{\subseteq \\ \supseteq} R[1]$



and UFDs...

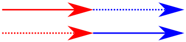
2. Adding acyclic queries



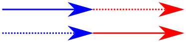
query:

UIDs:

R[2] $\subseteq\supseteq$ S[1]

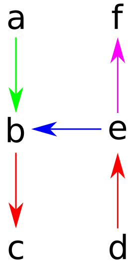


S[2] $\subseteq\supseteq$ R[1]



and UFDs...

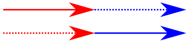
2. Adding acyclic queries



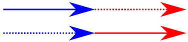
query:

UIDs:

$R[2] \mathrel{\substack{\subseteq \\ \supseteq}} S[1]$

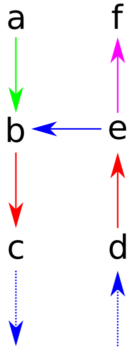


$S[2] \mathrel{\substack{\subseteq \\ \supseteq}} R[1]$



and UFDs...

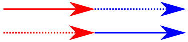
2. Adding acyclic queries



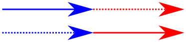
query:

UIDs:

$R[2] \overset{\subseteq}{\supseteq} S[1]$

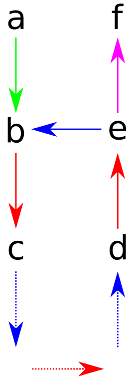


$S[2] \overset{\subseteq}{\supseteq} R[1]$



and UFDs...

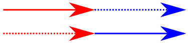
2. Adding acyclic queries



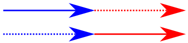
query:

UIDs:

R[2] $\subseteq \supseteq$ S[1]



S[2] $\subseteq \supseteq$ R[1]



and UFDs...

Table of contents

Summary

- Finite open-world query answering with:
  - relational instance
  - arbitrary arity signature
  - UIDs and FDs
  - conjunctive query

Summary

- Finite open-world query answering with:
  - relational instance
  - arbitrary arity signature
  - UIDs and FDs
  - conjunctive query

$\rightarrow$ We have finite controllability up to finite closure

Summary

- Finite open-world query answering with:
  - relational instance
  - arbitrary arity signature
  - UIDs and FDs
  - conjunctive query

$\rightarrow$ We have finite controllability up to finite closure

- Future work:
  - Simplify the proof
  - More expressive frontier-one logics [Ibáñez-García et al., 2014]
  - Finite controllability and closure for path FDs in arity-two?

Summary

- Finite open-world query answering with:
  - relational instance
  - arbitrary arity signature
  - UIDs and FDs
  - conjunctive query

→ We have finite controllability up to finite closure

- Future work:
  - Simplify the proof
  - More expressive frontier-one logics [Ibáñez-García et al., 2014]
  - Finite controllability and closure for path FDs in arity-two?

Thanks for your attention!

References I

Calì, A., Gottlob, G., and Pieris, A. (2012).
Towards more expressive ontology languages: The query answering problem.
Artif. Intel., 193.

Calì, A., Lembo, D., and Rosati, R. (2003).
On the decidability and complexity of query answering over inconsistent and incomplete databases.
In *PODS*.

Cosmadakis, S. S., Kanellakis, P. C., and Vardi, M. Y. (1990).
Polynomial-time implication problems for unary inclusion dependencies.
JACM, 37(1).

References II

Ibáñez-García, Y., Lutz, C., and Schneider, T. (2014).
Finite model reasoning in Horn description logics.
In *KR*.

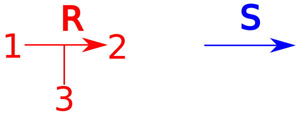
Otto, M. (2002).
Modal and guarded characterisation theorems over finite transition systems.
In *LICS*.

Pratt-Hartmann, I. (2009).
Data-complexity of the two-variable fragment with counting quantifiers.
Inf. Comput., 207(8).

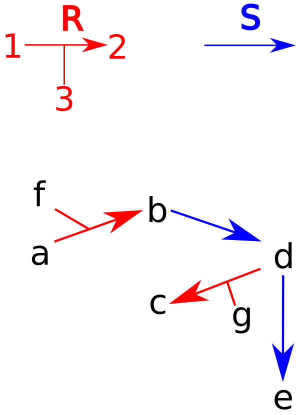
References III

Rosati, R. (2006).
On the decidability and finite controllability of query processing in databases with incomplete information.
In *PODS*.

1. Supporting arbitrary arity



1. Supporting arbitrary arity

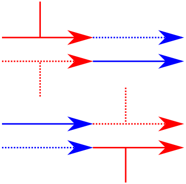


1. Supporting arbitrary arity

UIDs:

$R[2] \subseteq\supseteq S[1]$

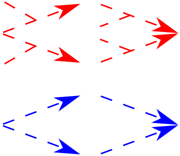
$S[2] \subseteq\supseteq R[1]$

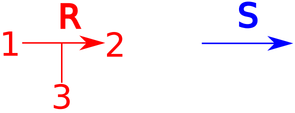


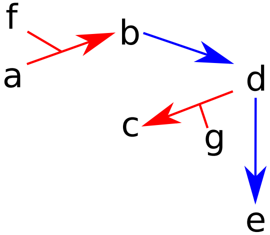
UFDs:

$R[1] \rightleftarrows R[2]$

$S[1] \rightleftarrows S[2]$





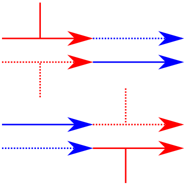


1. Supporting arbitrary arity

UIDs:

$R[2] \overset{\subseteq}{\supseteq} S[1]$

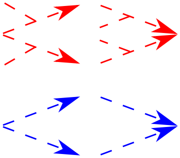
$S[2] \overset{\subseteq}{\supseteq} R[1]$

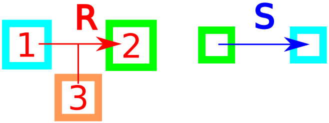


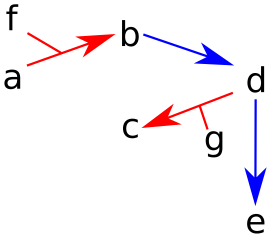
UFDs:

$R[1] \rightleftarrows R[2]$

$S[1] \rightleftarrows S[2]$





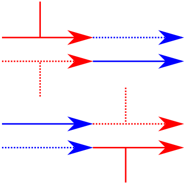


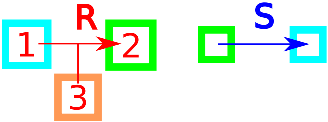
1. Supporting arbitrary arity

UIDs:

$R[2] \subseteq\supseteq S[1]$

$S[2] \subseteq\supseteq R[1]$

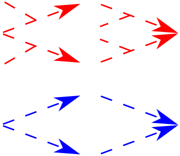


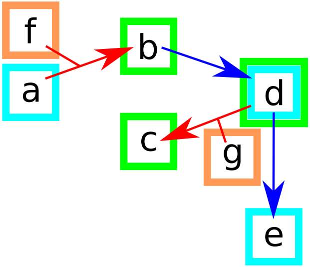


UFDs:

$R[1] \rightleftarrows R[2]$

$S[1] \rightleftarrows S[2]$



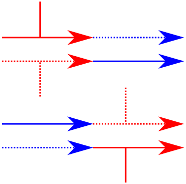


1. Supporting arbitrary arity

UIDs:

$R[2] \overset{\subseteq}{\supseteq} S[1]$

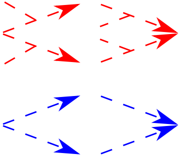
$S[2] \overset{\subseteq}{\supseteq} R[1]$

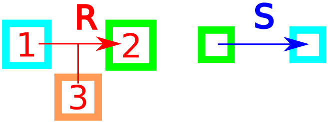


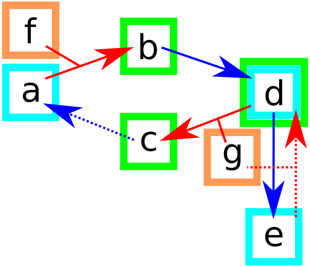
UFDs:

$R[1] \rightleftarrows R[2]$

$S[1] \rightleftarrows S[2]$







1. Supporting arbitrary arity

UIDs:

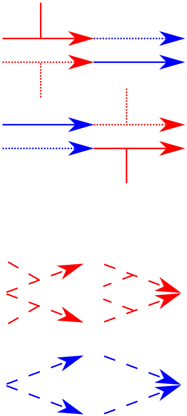
$R[2] \overset{\subseteq}{\supseteq} S[1]$

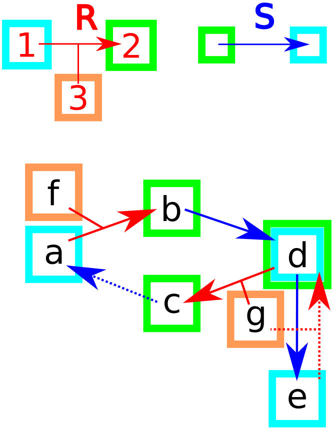
$S[2] \overset{\subseteq}{\supseteq} R[1]$

UFDs:

$R[1] \rightleftarrows R[2]$

$S[1] \rightleftarrows S[2]$





- When no UFDs: reuse elements from suitable positions

1. Supporting arbitrary arity

UIDs:

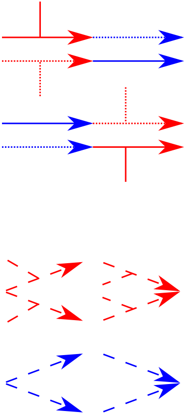
$R[2] \overset{\subseteq}{\supseteq} S[1]$

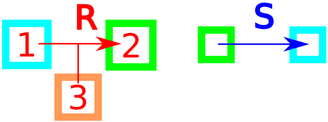
$S[2] \overset{\subseteq}{\supseteq} R[1]$

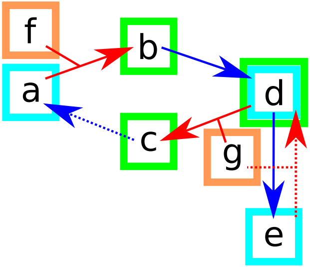
UFDs:

$R[1] \rightleftarrows R[2]$

$S[1] \rightleftarrows S[2]$







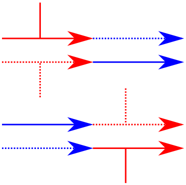
- When no UFDs: reuse elements from suitable positions
  $\rightarrow$ Must saturate initially to make sure such elements exist

1. Supporting arbitrary arity

UIDs:

$R[2] \subseteq\supseteq S[1]$

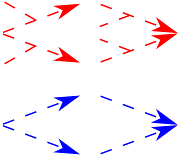
$S[2] \subseteq\supseteq R[1]$

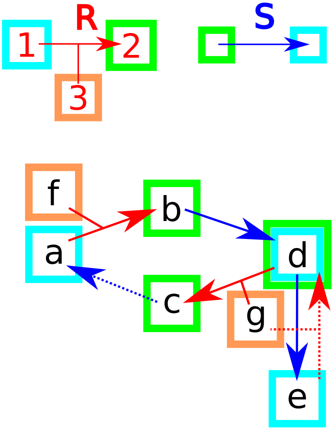


UFDs:

$R[1] \rightleftarrows R[2]$

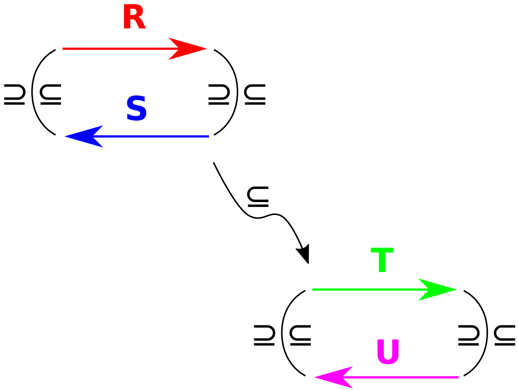
$S[1] \rightleftarrows S[2]$



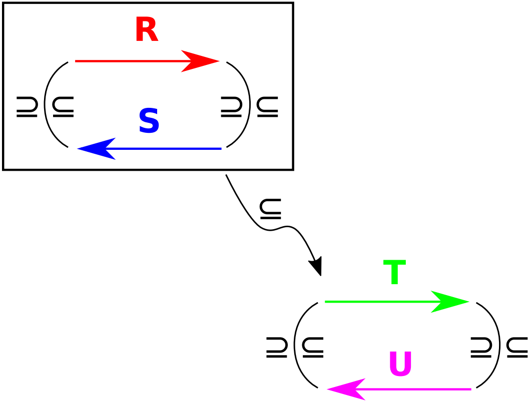


- When no UFDs: reuse elements from suitable positions
  - $\rightarrow$ Must saturate initially to make sure such elements exist
- With UFDs: bijective maps within $\rightleftarrows$-classes

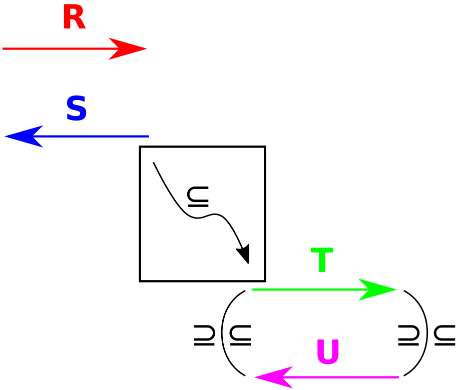
3. From reversible to arbitrary UIDs and UFDs



3. From reversible to arbitrary UIDs and UFDs



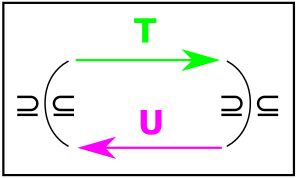
3. From reversible to arbitrary UIDs and UFDs



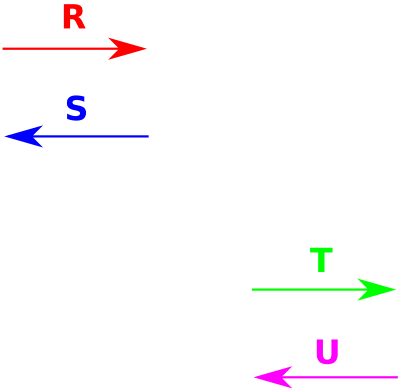
3. From reversible to arbitrary UIDs and UFDs

$$\xrightarrow{\mathbf{R}}$$

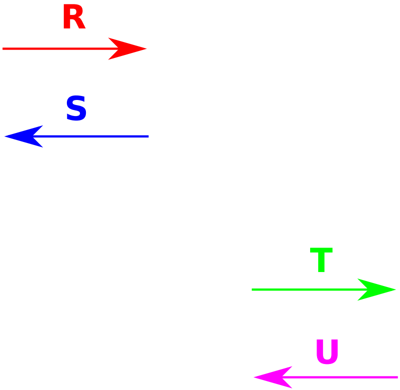
$$\xleftarrow{\mathbf{S}}$$



3. From reversible to arbitrary UIDs and UFDs

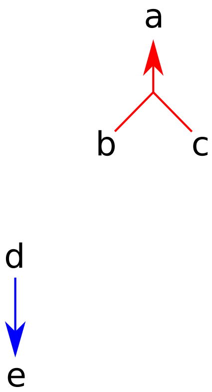


3. From reversible to arbitrary UIDs and UFDs



- UIDs can be decomposed in reversible subsets
- The subsets can be solved independently
- Relies on the finite implication construction [Cosmadakis et al., 1990]

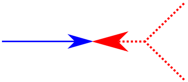
4. From UFDs to FDs

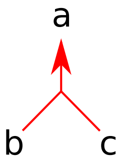


4. From UFDs to FDs

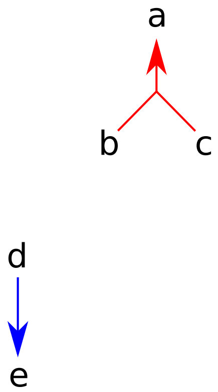
UID:

$S[2] \subseteq R[2]$



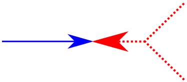


4. From UFDs to FDs



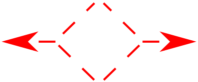
UID:

S[2] ⊆ R[2]

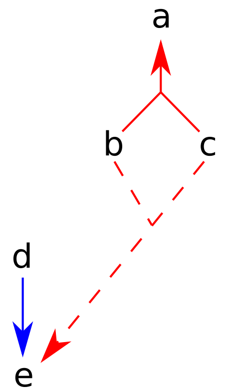


FD:

R[1,3] → R[2]

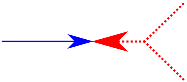


4. From UFDs to FDs



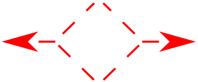
UID:

S[2] ⊆ R[2]

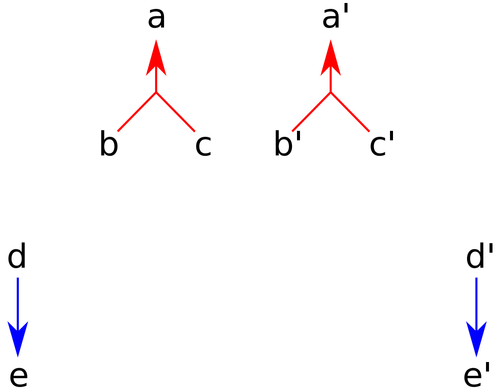


FD:

R[1,3] → R[2]

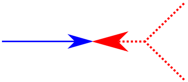


4. From UFDs to FDs



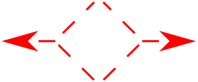
UID:

S[2] ⊆ R[2]

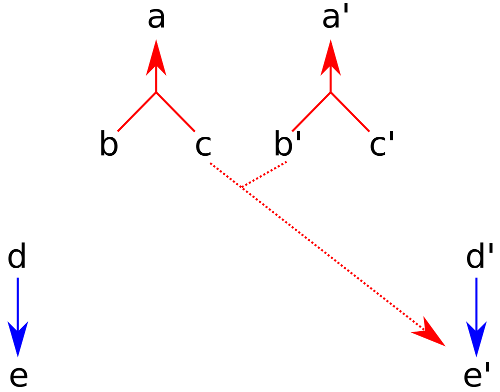


FD:

R[1,3] → R[2]

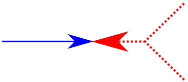


4. From UFDs to FDs



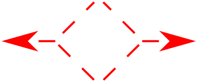
UID:

S[2] ⊆ R[2]

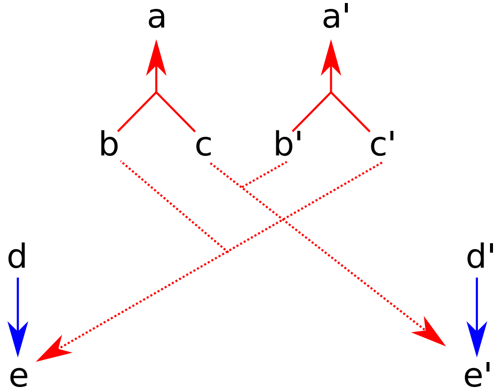


FD:

R[1,3] → R[2]

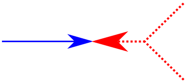


4. From UFDs to FDs



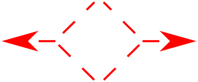
UID:

S[2] ⊆ R[2]

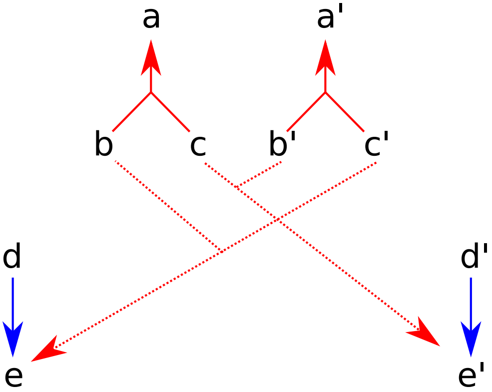


FD:

R[1,3] → R[2]



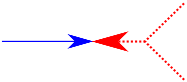
4. From UFDs to FDs



- Create initially many reusable elements
- Reuse them in new combinations

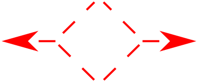
UID:

S[2] ⊆ R[2]

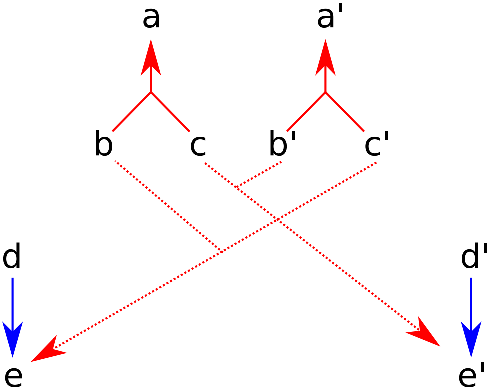


FD:

R[1,3] → R[2]

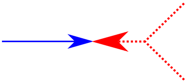


4. From UFDs to FDs



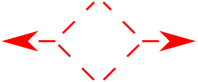
UID:

S[2] ⊆ R[2]



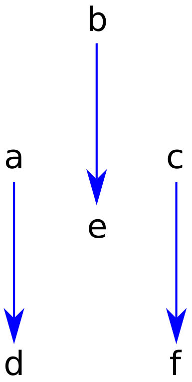
FD:

R[1,3] → R[2]

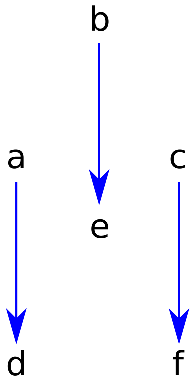


- Create initially many reusable elements
- Reuse them in new combinations
- Dense interpretations lemma:
  linearly many elements give super-linearly many combinations

5. From acyclic queries to arbitrary queries



5. From acyclic queries to arbitrary queries

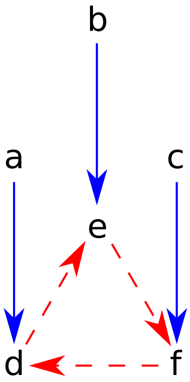


UID:

$S[2] \subseteq R[1]$

$S[2] \subseteq R[2]$

5. From acyclic queries to arbitrary queries

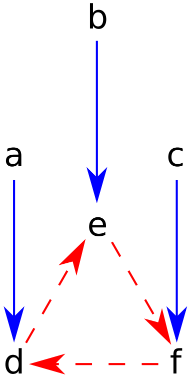


UID:

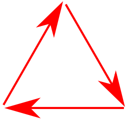
$S[2] \subseteq R[1]$

$S[2] \subseteq R[2]$

5. From acyclic queries to arbitrary queries



query:

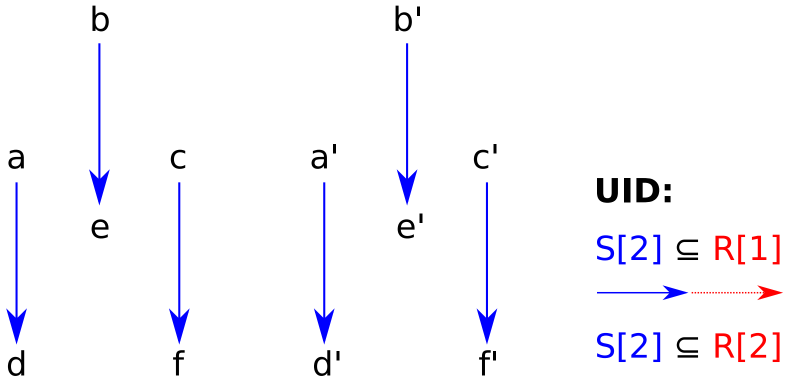


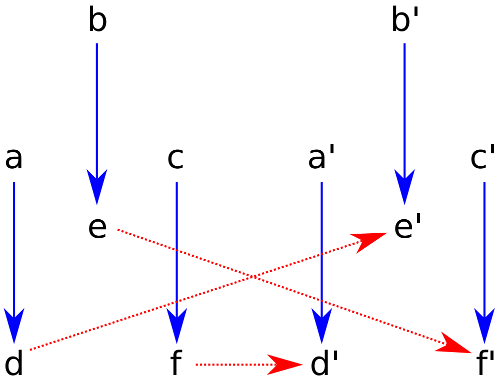
UID:

$S[2] \subseteq R[1]$

$S[2] \subseteq R[2]$

5. From acyclic queries to arbitrary queries



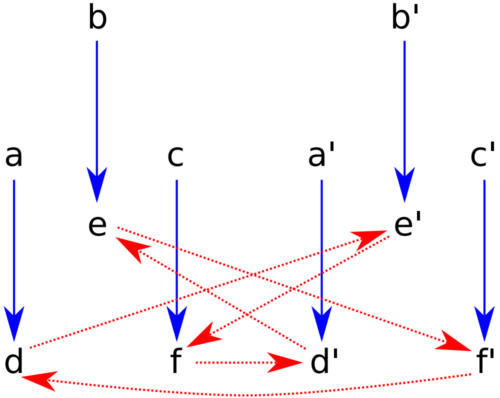


UID:

S[2] ⊆ R[1]

S[2] ⊆ R[2]

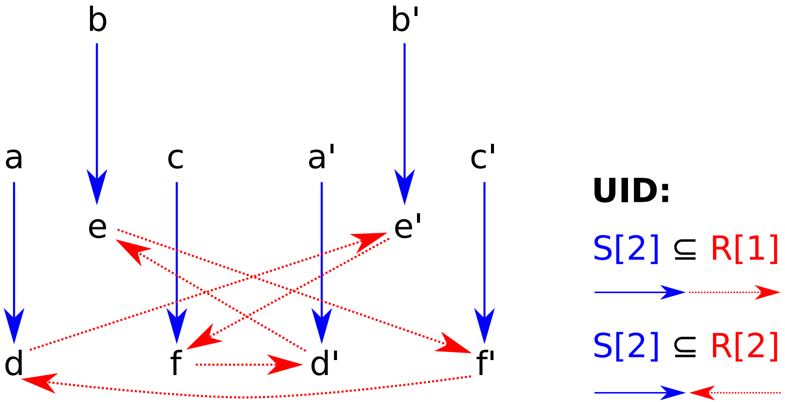
5. From acyclic queries to arbitrary queries



UID:

S[2] ⊆ R[1]

S[2] ⊆ R[2]



- Product with a group of high girth [Otto, 2002]
- Must be tweaked to avoid violating FDs