Semantics of Query Answering in Data Exchange

André Hernich

Department of Computer Science
Humboldt University Berlin

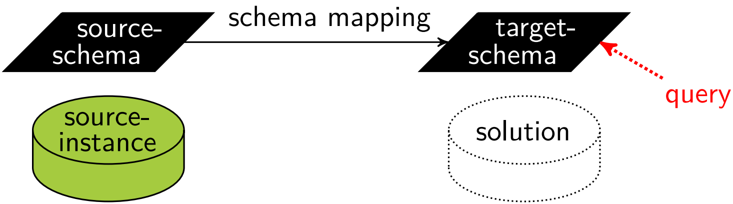
DEIS 2010, Dagstuhl

Outline

1. Goals of Query Answering in Data Exchange
2. The Basic Query Answering Semantics
3. Alternative Semantics

Query Answering in Data Exchange

Goal: Answer queries posed against target data
(Fagin, Kolaitis, Miller, Popa '03)



Example

Source instance:

Book

title	author
Algebra	Lang
Logic	Hodges

Solution:

Author

id	name
1	Lang
2	Hodges

Publ

title	a_id
Algebra	1
Logic	2

Schema mapping:

- $\forall t\, \forall a\, (\ \mathrm{Book}(t, a)\ \rightarrow\ \exists id\ \mathrm{Author}(id, a) \wedge \mathrm{Publ}(t, id)\ )$

Example

Source instance:

Book

title	author
Algebra	Lang
Logic	Hodges

Solution:

Author

id	name
1	Lang
2	Hodges

Publ

title	a_id
Algebra	1
Logic	2

Schema mapping:

- $\forall t\, \forall a\, (\ \mathrm{Book}(t, a)\ \rightarrow\ \exists id\ \mathrm{Author}(id, a) \wedge \mathrm{Publ}(t, id)\ )$

Example query over target schema

Who are the authors of “Algebra”?

$$Q(a)\ :=\ \exists id\, (\ \mathrm{Publ}(\text{“Algebra”}, id) \wedge \mathrm{Author}(id, a)\ )$$

Fundamental Issues

1. What is the “right” answer to/semantics of a query?

Example

Source instance:

Book

title	author
Algebra	Lang
Logic	Hodges

Solution:

Author

id	name
1	Lang
2	Hodges

Publ

title	a_id
Algebra	1
Logic	2

Schema mapping:

- $\forall t \, \forall a \, (\, \mathrm{Book}(t, a) \;\rightarrow\; \exists id \; \mathrm{Author}(id, a) \wedge \mathrm{Publ}(t, id) \,)$

Example query over target schema

Who are the authors of “Algebra”?

$$Q(a) \;:=\; \exists id \, (\, \mathrm{Publ}(\text{“Algebra”}, id) \wedge \mathrm{Author}(id, a) \,)$$

Fundamental Issues

1. What is the "right" answer to/semantics of a query?

   Problem: many solutions with *different* sets of answers

Fundamental Issues

1. What is the “right” answer to/semantics of a query?
   Problem: many solutions with *different* sets of answers
2. Which solutions are appropriate for query answering?
   Problem: queries have to be answered *without* source instance

Fundamental Issues

1. What is the "right" answer to/semantics of a query?
   Problem: many solutions with *different* sets of answers
2. Which solutions are appropriate for query answering?
   Problem: queries have to be answered *without* source instance
3. What is the complexity of query answering?
   (computing the solution & evaluating the query)

Outline

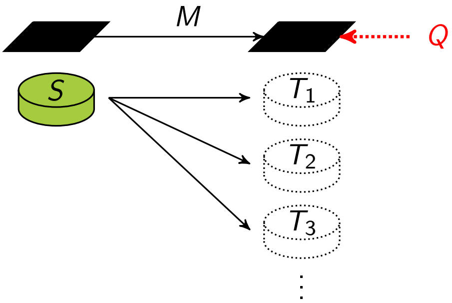
1. Goals of Query Answering in Data Exchange
2. The Basic Query Answering Semantics
3. Alternative Semantics

The Certain Answers Semantics

Idea: return "safe" answers

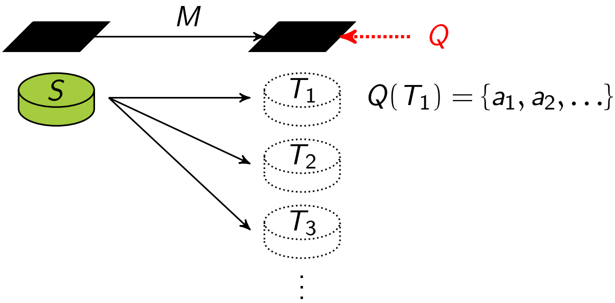
The Certain Answers Semantics

Idea: return "safe" answers



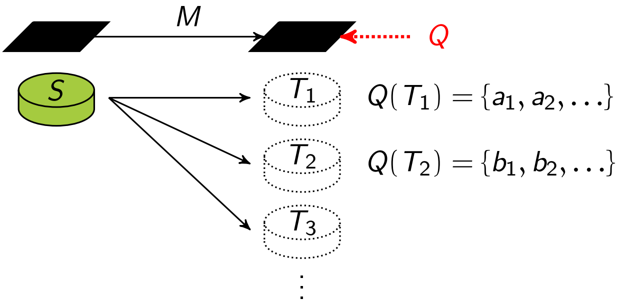
The Certain Answers Semantics

Idea: return “safe” answers

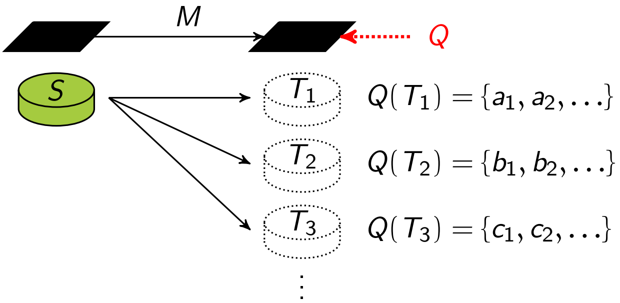


The Certain Answers Semantics

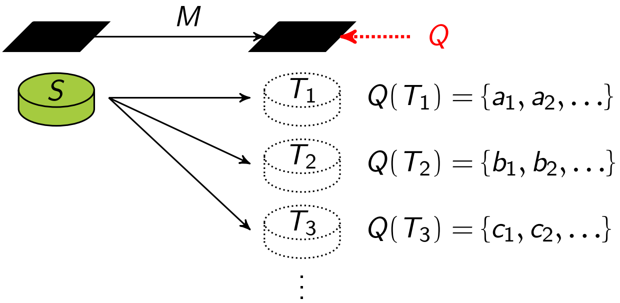
Idea: return "safe" answers



Idea: return "safe" answers



Idea: return “safe” answers



Definition (Fagin, Kolaitis, Miller, Popa '03)

a is a certain answer to Q on M and S

$\iff$ $a \in Q(T)$ for all solutions T for S under M

Example

Source instance:

Book

title	author
Algebra	Lang
Logic	Hodges

Solution:

Author

id	name
1	Lang
2	Hodges

Publ

title	a_id
Algebra	1
Logic	2

Schema mapping:

- $\forall t\, \forall a\, \big(\, \mathrm{Book}(t, a) \;\rightarrow\; \exists id\; \mathrm{Author}(id, a) \wedge \mathrm{Publ}(t, id) \,\big)$

Example

Source instance:

Book

title	author
Algebra	Lang
Logic	Hodges

Solution:

Author

id	name
1	Lang
2	Hodges

Publ

title	a_id
Algebra	1
Logic	2

Schema mapping:

- $\forall t\, \forall a \;(\; \mathrm{Book}(t, a) \;\rightarrow\; \exists id\; \mathrm{Author}(id, a) \wedge \mathrm{Publ}(t, id) \;)$

Query: Who are the authors of "Algebra"?

$$Q(a) \;:=\; \exists id \;(\; \mathrm{Publ}(\text{“Algebra”}, id) \wedge \mathrm{Author}(id, a) \;)$$

Example

Source instance:

Book

title	author
Algebra	Lang
Logic	Hodges

Solution:

Author

id	name
1	Lang
2	Hodges

Publ

title	a_id
Algebra	1
Logic	2

Schema mapping:

- $\forall t\, \forall a\, \big(\, \mathrm{Book}(t, a) \;\rightarrow\; \exists id\; \mathrm{Author}(id, a) \wedge \mathrm{Publ}(t, id) \,\big)$

Query: Who are the authors of "Algebra"?

$$Q(a) \;:=\; \exists id\, \big(\, \mathrm{Publ}(\text{“Algebra”}, id) \wedge \mathrm{Author}(id, a) \,\big)$$

Certain answers: {"Lang"}

Example

Source instance:

Book

title	author
Algebra	Lang
Logic	Hodges

Solution:

Author

id	name
1	Lang
2	Hodges

Publ

title	a_id
Algebra	1
Logic	2

Schema mapping:

- $\forall t \, \forall a \, \big(\, \mathrm{Book}(t, a) \; \rightarrow \; \exists id \; \mathrm{Author}(id, a) \wedge \mathrm{Publ}(t, id) \, \big)$

Query: Who are the authors of "Algebra"?

$$Q(a) \; := \; \exists id \, \big(\, \mathrm{Publ}(\text{"Algebra"}, id) \wedge \mathrm{Author}(id, a) \, \big)$$

Certain answers: {"Lang"}

The Certain Answers and UCQs

Consensus: suitable for unions of conjunctive queries (UCQs)

The Certain Answers and UCQs

Consensus: suitable for unions of conjunctive queries (UCQs)

Theorem (Fagin, Kolaitis, Miller, Popa '03)

For every schema mapping M, source instance S for M, universal solution T for S, and UCQ Q

$$\textit{certain answers to } Q = \{a \in Q(T) \mid a \textit{ is null-free}\}$$

The Certain Answers and UCQs

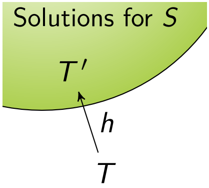
Consensus: suitable for unions of conjunctive queries (UCQs)

Theorem (Fagin, Kolaitis, Miller, Popa '03)

For every schema mapping M, source instance S for M, universal solution T for S, and UCQ Q

$$\textit{certain answers to } Q = \{a \in Q(T) \mid a \textit{ is null-free}\}$$

"Ingredients" for the proof:



$$+ \quad \bar{a} \in Q(T) \Longrightarrow \underbrace{h(\bar{a})}_{=\bar{a}} \in Q(T')$$

The Certain Answers and UCQs

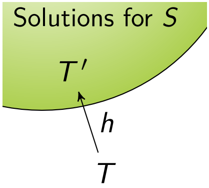
Consensus: suitable for unions of conjunctive queries (UCQs)

Theorem (Fagin, Kolaitis, Miller, Popa '03)

For every schema mapping M, source instance S for M, universal solution T for S, and UCQ Q

$$\textit{certain answers to } Q = \{a \in Q(T) \mid a \textit{ is null-free}\}$$

"Ingredients" for the proof:



$$+ \quad \bar{a} \in Q(T) \implies \underbrace{h(\bar{a})}_{=\bar{a}} \in Q(T')$$

More general: for queries preserved under homomorphisms

. . . and Monotonic Queries in General

- \+ Widely agreed: the certain answers semantics is suitable
- – issue of appropriate solutions and query answering less well understood

. . . and Monotonic Queries in General

+ Widely agreed: the certain answers semantics is suitable
− issue of appropriate solutions and query answering less well understood

(Data) complexity results:

- evaluation of UCQs with ≤ 1 inequality per disjunct in PTIME on universal solutions (Fagin, Kolaitis, Miller, and Popa '03)
- co-NP-complete for CQs with ≥ 2 inequalities (Mądry '05)
- fragments of UCQs with ≤ 2 inequalities per disjunct in PTIME on universal solutions (Arenas, Barceló, Reutter '09)

...and Monotonic Queries in General

+ Widely agreed: the certain answers semantics is suitable
− issue of appropriate solutions and query answering less well understood

(Data) complexity results:

- evaluation of UCQs with ≤ 1 inequality per disjunct in PTIME on universal solutions (Fagin, Kolaitis, Miller, and Popa '03)
- co-NP-complete for CQs with ≥ 2 inequalities (Mądry '05)
- fragments of UCQs with ≤ 2 inequalities per disjunct in PTIME on universal solutions (Arenas, Barceló, Reutter '09)

"Generic" approach: based on extension of universal solutions (Deutsch, Nash, Remmel '08)

. . . and Beyond?

Counter-intuitive answers possible on non-monotonic queries
(Fagin, Arenas, Barceló, Libkin '04; Libkin '06)

. . . and Beyond?

Counter-intuitive answers possible on non-monotonic queries
(Fagin, Arenas, Barceló, Libkin '04; Libkin '06)

Example (copy relation E to E')

Schema mapping: $\forall x \forall y (\ E(x,y) \rightarrow E'(x,y)\ )$

Source instance:

$a \xrightarrow{E} b$

Solution:

$a \xrightarrow{E'} b$

Counter-intuitive answers possible on non-monotonic queries
(Fagin, Arenas, Barceló, Libkin '04; Libkin '06)

Example (copy relation E to E')

Schema mapping: $\forall x \forall y (\ E(x,y) \rightarrow E'(x,y)\ )$

Source instance:

$$a \xrightarrow{E} b$$

Solution:

$$a \xrightarrow{E'} b$$

Query: $Q(x) :=$ Is there exactly one y with $E'(x,y)$?

- Expected answers: $\{a\}$

. . . and Beyond?

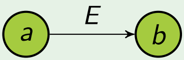
Counter-intuitive answers possible on non-monotonic queries

(Fagin, Arenas, Barceló, Libkin ’04; Libkin ’06)

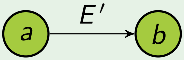
Example (copy relation E to E')

Schema mapping: $\forall x \forall y (\, E(x,y) \rightarrow E'(x,y) \,)$

Source instance:



Solution:



Query: $Q(x) :=$ Is there exactly one y with $E'(x,y)$?

- Expected answers: $\{a\}$
- The certain answers: $\emptyset$

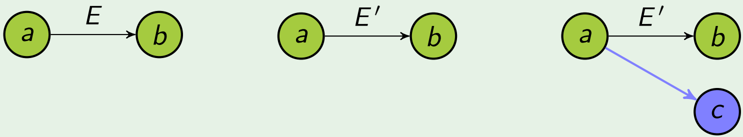
. . . and Beyond?

Counter-intuitive answers possible on non-monotonic queries
(Fagin, Arenas, Barceló, Libkin '04; Libkin '06)

Example (copy relation E to E')

Schema mapping: $\forall x \forall y (\ E(x,y) \rightarrow E'(x,y)\ )$

Source instance: Solution: Another solution:



Query: $Q(x) :=$ Is there exactly one y with $E'(x,y)$?

- Expected answers: $\{a\}$
- The certain answers: $\emptyset$

Outline

Dealing with Non-Monotonic Queries

1. Use the certain answers semantics

2. Use alternative semantics

Dealing with Non-Monotonic Queries

1. Use the certain answers semantics
   - *manually* rule out undesired solutions via *suitable constraints*

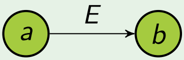
2. Use alternative semantics

Motivating Example Revisited

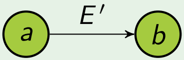
Example (copy relation E to E')

Schema mapping: $\forall x \forall y (\ E(x,y) \rightarrow E'(x,y)\ )$

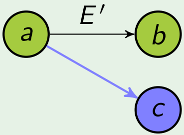
Source instance:



Solution:



Another solution:



Query: $Q(x) :=$ Is there exactly one y with $E'(x,y)$?

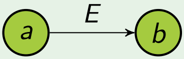
- Expected answers: $\{a\}$
- The certain answers: $\emptyset$

Motivating Example Revisited

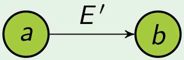
Example (copy relation E to E')

Schema mapping: $\forall x \forall y (\ E(x,y) \rightarrow E'(x,y)\ )$
$\forall x \forall y (\ \neg E(x,y) \rightarrow \neg E'(x,y)\ )$

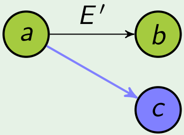
Source instance:



Solution:



Another solution:



Query: $Q(x) :=$ Is there exactly one y with $E'(x,y)$?

- Expected answers: $\{a\}$
- The certain answers: $\emptyset$

Dealing with Non-Monotonic Queries

1. Use the certain answers semantics
   - *manually* rule out undesired solutions via *suitable constraints*

2. Use alternative semantics

Dealing with Non-Monotonic Queries

1. Use the certain answers semantics
   - *manually* rule out undesired solutions via *suitable constraints*
   - requires richer constraint language
   - almost no research in this direction

2. Use alternative semantics

Dealing with Non-Monotonic Queries

1. Use the certain answers semantics
    - *manually* rule out undesired solutions via *suitable constraints*
    - requires richer constraint language
    - almost no research in this direction
2. Use alternative semantics (this talk)

Dealing with Non-Monotonic Queries

1. Use the certain answers semantics
   - *manually* rule out undesired solutions via *suitable constraints*
   - requires richer constraint language
   - almost no research in this direction
2. Use alternative semantics (this talk)
   - *automatically* rule out undesired solutions *via heuristics*
   - no richer constraint language
   - can build on research from non-monotonic reasoning

Dealing with Non-Monotonic Queries

1. Use the certain answers semantics
   - *manually* rule out undesired solutions via *suitable constraints*
   - requires richer constraint language
   - almost no research in this direction

2. Use alternative semantics (this talk)
   - *automatically* rule out undesired solutions *via heuristics*
   - no richer constraint language
   - can build on research from non-monotonic reasoning

Basis: variants of Closed World Assumption (CWA) (Reiter '78)

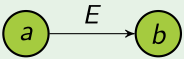
"If something is not mentioned, take it to be false."

Motivating Example Revisited

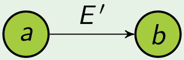
Example (copy relation E to E')

Schema mapping: $\forall x \forall y (\ E(x,y) \rightarrow E'(x,y)\ )$
$\forall x \forall y (\ \neg E(x,y) \rightarrow \neg E'(x,y)\ )$

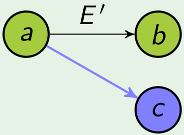
Source instance:



Solution:



Another solution:



Query: $Q(x) :=$ Is there exactly one y with $E'(x,y)$?

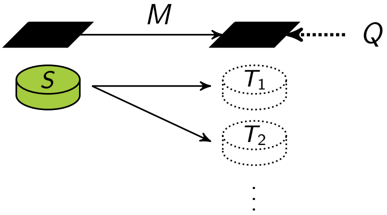
- Expected answers: $\{a\}$
- The certain answers: $\emptyset$

CWA-Semantics

- for schema mappings defined by *s-t tgds, t-tgds, and egds*
  (Libkin '06; H., Schweikardt '07)
- family of semantics, based on CWA-solutions
  (= solutions valid under the CWA-semantics)

CWA-Semantics

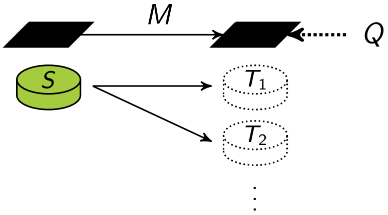
- for schema mappings defined by *s-t tgds, t-tgds, and egds* (Libkin '06; H., Schweikardt '07)
- family of semantics, based on CWA-solutions (= solutions valid under the CWA-semantics)
- CWA-certain answers semantics:



... like the certain answers semantics, *except:*

CWA-Semantics

- for schema mappings defined by *s-t tgds, t-tgds, and egds*
  (Libkin '06; H., Schweikardt '07)
- family of semantics, based on CWA-solutions
  (= solutions valid under the CWA-semantics)
- CWA-certain answers semantics:

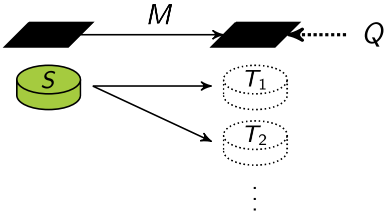


…like the certain answers semantics, *except:*

- the T_i are CWA-solutions

CWA-Semantics

- for schema mappings defined by *s-t tgds, t-tgds, and egds* (Libkin '06; H., Schweikardt '07)
- family of semantics, based on CWA-solutions (= solutions valid under the CWA-semantics)
- CWA-certain answers semantics:



...like the certain answers semantics, *except:*

- the T_i are CWA-solutions
- Q is evaluated under a special semantics for instances with nulls

CWA-Solutions

Rule: all atoms and facts in CWA-solutions must be justified by the source instance and the schema mapping

Criteria

Example

$S = \{ P(a) \} \quad \forall x \big(\; P(x) \; \rightarrow \; \exists y \; E(x, y) \; \big)$

CWA-Solutions

Rule: all atoms and facts in CWA-solutions must be justified by the source instance and the schema mapping

Criteria

1. Derivability

Example

$S = \{ P(a) \} \quad \forall x \big(\; P(x) \rightarrow \exists y \; E(x,y) \; \big)$

CWA-Solutions

Rule: all atoms and facts in CWA-solutions must be justified by the source instance and the schema mapping

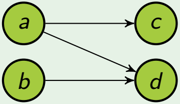
Criteria

1. Derivability

Example

$S = \{ P(a) \} \quad \forall x \big(P(x) \rightarrow \exists y\, E(x, y) \big)$

Solution:



Rule: all atoms and facts in CWA-solutions must be justified by the source instance and the schema mapping

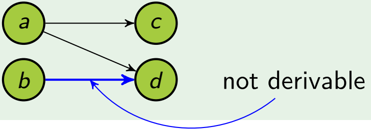
Criteria

1. Derivability

Example

$S = \{ P(a) \} \quad \forall x \big(\; P(x) \to \exists y \; E(x, y) \; \big)$

Solution:



CWA-Solutions

Rule: all atoms and facts in CWA-solutions must be justified by the source instance and the schema mapping

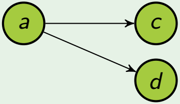
Criteria

1. Derivability

Example

$S = \{ P(a) \} \quad \forall x \big(\ P(x) \ \rightarrow \ \exists y \ E(x,y) \ \big)$

Solution:



CWA-Solutions

Rule: all atoms and facts in CWA-solutions must be justified by the source instance and the schema mapping

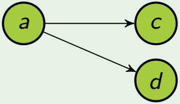
Criteria

1. Derivability
2. Parsimony

Example

$S = \{ P(a) \} \quad \forall x \big(P(x) \rightarrow \exists y\, E(x, y) \big)$

Solution:



CWA-Solutions

Rule: all atoms and facts in CWA-solutions must be justified by the source instance and the schema mapping

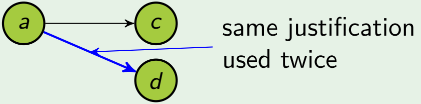
Criteria

1. Derivability
2. Parsimony

Example

$S = \{ P(a) \} \quad \forall x \big(\; P(x) \; \rightarrow \; \exists y \; E(x, y) \; \big)$

Solution:



CWA-Solutions

Rule: all atoms and facts in CWA-solutions must be justified by the source instance and the schema mapping

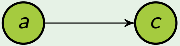
Criteria

1. Derivability
2. Parsimony

Example

$S = \{ P(a) \} \quad \forall x \big(\ P(x) \ \rightarrow \ \exists y \ E(x,y) \ \big)$

Solution:



CWA-Solutions

Rule: all atoms and facts in CWA-solutions must be justified by the source instance and the schema mapping

Criteria

1. Derivability
2. Parsimony
3. No invented facts

Example

$S = \{ P(a) \} \quad \forall x \big(\; P(x) \; \rightarrow \; \exists y \; E(x, y) \; \big)$

Solution:

$$a \longrightarrow c$$

CWA-Solutions

Rule: all atoms and facts in CWA-solutions must be justified by the source instance and the schema mapping

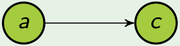
Criteria

1. Derivability
2. Parsimony
3. No invented facts

Example

$S = \{ P(a) \} \quad \forall x \big(\; P(x) \rightarrow \exists y \; E(x, y) \; \big)$

Solution:



contant c
is invented

CWA-Solutions

Rule: all atoms and facts in CWA-solutions must be justified by the source instance and the schema mapping

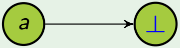
Criteria

1. Derivability
2. Parsimony
3. No invented facts

Example

$S = \{ P(a) \} \quad \forall x \big(P(x) \to \exists y\, E(x, y) \big)$

Solution:



CWA-Solutions

Rule: all atoms and facts in CWA-solutions must be justified by the source instance and the schema mapping

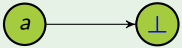
Criteria

1. Derivability
2. Parsimony
3. No invented facts

Example

$S = \{\, P(a) \,\} \quad \forall x \big(\; P(x) \;\rightarrow\; \exists y \; E(x, y) \;\big)$

unique CWA-solution:



CWA-Solutions

Rule: all atoms and facts in CWA-solutions must be justified by the source instance and the schema mapping

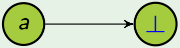
Criteria

1. Derivability
2. Parsimony
3. No invented facts

Example

$S = \{ P(a) \} \quad \forall x \big(P(x) \rightarrow \exists y \, E(x, y) \big)$

unique CWA-solution:



Characterization (Libkin '06; H., Schweikardt '07)

CWA-solutions = universal solutions derivable from the source instance using a certain variant of the chase

E.g., core solution = minimal CWA-solution

Query Evaluation under the CWA-Semantics

Theorem (Libkin '06)

For every schema mapping M defined by s-t tgds, every source instance S, and every query Q,

$$\text{CWA-certain answers to } Q \text{ on } M \text{ and } S \;=\; \Box Q(T),$$

where T = canonical solution for S under M.

Query Evaluation under the CWA-Semantics

Theorem (Libkin '06)

For every schema mapping M defined by s-t tgds, every source instance S, and every query Q,

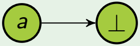
$$\text{CWA-certain answers to } Q \text{ on } M \text{ and } S = \Box Q(T),$$

where T = canonical solution for S under M.

What is $\Box Q(T)$?

- T may contain incomplete information in the form of nulls

Example



Query Evaluation under the CWA-Semantics

Theorem (Libkin '06)

For every schema mapping M defined by s-t tgds, every source instance S, and every query Q,

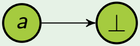
$$\text{CWA-certain answers to } Q \text{ on } M \text{ and } S \; = \; \Box Q(T),$$

where T = canonical solution for S under M.

What is $\Box Q(T)$?

- T may contain incomplete information in the form of nulls
- Possible worlds of T: instances arising from T by assigning constants to nulls

Example



Query Evaluation under the CWA-Semantics

Theorem (Libkin '06)

For every schema mapping M defined by s-t tgds, every source instance S, and every query Q,

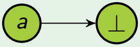
$$\text{CWA-certain answers to } Q \text{ on } M \text{ and } S \; = \; \Box Q(T),$$

where T = canonical solution for S under M.

What is $\Box Q(T)$?

- T may contain incomplete information in the form of nulls
- Possible worlds of T: instances arising from T by assigning constants to nulls

Example



Possible worlds:

Query Evaluation under the CWA-Semantics

Theorem (Libkin '06)

For every schema mapping M defined by s-t tgds, every source instance S, and every query Q,

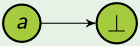
$$\textit{CWA-certain answers to } Q \textit{ on } M \textit{ and } S \;=\; \Box Q(T),$$

where T = canonical solution for S under M.

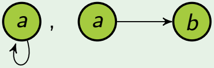
What is $\Box Q(T)$?

- T may contain incomplete information in the form of nulls
- Possible worlds of T: instances arising from T by assigning constants to nulls

Example



Possible worlds:



Query Evaluation under the CWA-Semantics

Theorem (Libkin '06)

For every schema mapping M defined by s-t tgds, every source instance S, and every query Q,

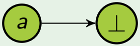
$$\text{CWA-certain answers to } Q \text{ on } M \text{ and } S = \Box Q(T),$$

where T = canonical solution for S under M.

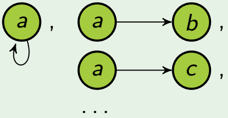
What is $\Box Q(T)$?

- T may contain incomplete information in the form of nulls
- Possible worlds of T: instances arising from T by assigning constants to nulls

Example



Possible worlds:



Query Evaluation under the CWA-Semantics

Theorem (Libkin '06)

For every schema mapping M defined by s-t tgds, every source instance S, and every query Q,

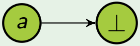
$$\text{CWA-certain answers to } Q \text{ on } M \text{ and } S \;=\; \Box Q(T),$$

where T = canonical solution for S under M.

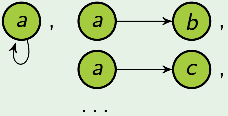
What is $\Box Q(T)$?

- T may contain incomplete information in the form of nulls
- Possible worlds of T: instances arising from T by assigning constants to nulls
- $\Box Q(T)$: the certain answers to Q over the possible worlds of T

Example



Possible worlds:



Generalization and Restriction of the CWA-Semantics

Modifications of the CWA-semantics
(both for schema mappings defined by s-t tgds only):

- "Mixed world" semantics (Libkin, Sirangelo '08)

- Endomorphic images semantics (Afrati, Kolaitis '08)

Generalization and Restriction of the CWA-Semantics

Modifications of the CWA-semantics
(both for schema mappings defined by s-t tgds only):

- "Mixed world" semantics (Libkin, Sirangelo '08)
  - based on generalized notion of possible worlds of an instance
  - generalized constraint language (annotated s-t tgds)
- Endomorphic images semantics (Afrati, Kolaitis '08)

Generalization and Restriction of the CWA-Semantics

Modifications of the CWA-semantics
(both for schema mappings defined by s-t tgds only):

- "Mixed world" semantics (Libkin, Sirangelo '08)
  - based on generalized notion of possible worlds of an instance
  - generalized constraint language (annotated s-t tgds)
- Endomorphic images semantics (Afrati, Kolaitis '08)
  - based on restricted notion of possible worlds of an instance
  - shown to be suitable for special aggregate queries

Two Natural Properties

Two natural properties are "missing":

1. Invariance under logically equivalent schema mappings
2. Reflection of "standard semantics" of constraints

Two Natural Properties

Two natural properties are "missing":

1. Invariance under logically equivalent schema mappings
2. Reflection of "standard semantics" of constraints

Reflection of "Standard Semantics" of Constraints

Example

Schema mapping:

$$\forall x (\ P(x) \rightarrow \exists y\, E(x, y)\ )$$

Source instance: $S = \{P(a)\}$

Unique CWA-solution: $a \longrightarrow \perp$

Reflection of "Standard Semantics" of Constraints

Example

Schema mapping:

$$\forall x(\ P(x) \to \exists y\, E(x,y)\ )$$

Source instance: $S = \{P(a)\}$

Unique CWA-solution: $a \longrightarrow \perp$

Example query: $Q :=$ Is there exactly one y with $E(a,y)$?

CWA-answers: yes

Reflection of “Standard Semantics” of Constraints

Example

Schema mapping:

$$\forall x\big(\ P(x) \rightarrow \exists y\, E(x,y)\ \big)\ \equiv\ \forall x\big(\ P(x) \rightarrow \bigvee_{y \in Const} E(x,y)\ \big)$$

Source instance: $S = \{P(a)\}$

Unique CWA-solution: $a \longrightarrow \perp$

Example query: $Q :=$ Is there exactly one y with $E(a,y)$?

CWA-answers: yes

Reflection of "Standard Semantics" of Constraints

Example

Schema mapping:

$$\forall x\big(\ P(x) \rightarrow \exists y\, E(x,y)\ \big) \ \equiv\ \forall x\big(\ P(x) \rightarrow \bigvee_{y \in Const} E(x,y)\ \big)$$

Source instance: $S = \{P(a)\}$

Unique CWA-solution: $a \longrightarrow \perp$

Example query: $Q :=$ Is there exactly one y with $E(a,y)$?

CWA-answers: yes

Desired answer: no

The GCWA*-Semantics

Definition (H. '10, restricted version)

1. GCWA*-solutions:
   ground solutions that are unions of minimal solutions
2. GCWA*-answers:
   the certain answers over GCWA*-solutions

The GCWA*-Semantics

Definition (H. '10, restricted version)

1. GCWA*-solutions:
   ground solutions that are unions of minimal solutions
2. GCWA*-answers:
   the certain answers over GCWA*-solutions

- inspired by semantics for deductive databases:
  GCWA (Minker '82) and EGCWA (Yahya, Henschen '85)
- invariant under logically equivalent schema mappings
- *intuitively:* reflects "standard semantics" of constraints

Motivating Example Revisited

Example

Schema mapping: $\forall x (\ P(x) \rightarrow \exists y\, E(x, y)\ )$

Source instance: $S = \{P(a)\}$

GCWA*solutions:

Motivating Example Revisited

Example

Schema mapping: $\forall x \big(\; P(x) \rightarrow \exists y \, E(x, y) \; \big)$

Source instance: $S = \{P(a)\}$

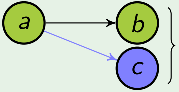
GCWA*solutions: $\left. a \longrightarrow b \;\right\}$ union of one minimal solution

Example

Schema mapping: $\forall x (\ P(x) \rightarrow \exists y\, E(x, y)\ )$

Source instance: $S = \{P(a)\}$

GCWA*solutions:



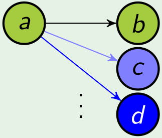
union of two minimal solutions

Motivating Example Revisited

Example

Schema mapping: $\forall x (\ P(x) \rightarrow \exists y\, E(x, y)\ )$

Source instance: $S = \{P(a)\}$

GCWA*solutions: 

union of three minimal solutions

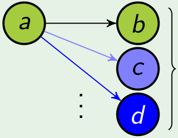
Motivating Example Revisited

Example

Schema mapping: $\forall x(\ P(x) \rightarrow \exists y\, E(x,y)\ )$

Source instance: $S = \{P(a)\}$

GCWA*solutions:



union of three minimal solutions

Query: $Q :=$ Is there exactly one y with $E(a,y)$?

GCWA*-answers: no (as desired)

Basic Results

- for monotonic queries: GCWA*-answers $=$ certain answers
  (actually true for almost all of the preceding semantics)

Basic Results

- for monotonic queries: GCWA*-answers = certain answers
  (actually true for almost all of the preceding semantics)
- There is a simple schema mapping M defined by s-t tgds, and a Boolean CQ Q with one negated atom for which

EVAL(M, Q)	
Input:	source instance S
Question:	Are the GCWA*-answers to Q on M and S non-empty?

  is co-NP-hard

  (simple reduction from clique problem)

Basic Results

- for monotonic queries: GCWA*-answers = certain answers (actually true for almost all of the preceding semantics)
- There is a simple schema mapping M defined by s-t tgds, and a Boolean CQ Q with one negated atom for which

EVAL(M, Q)	
Input:	source instance S
Question:	Are the GCWA*-answers to Q on M and S non-empty?

  is co-NP-hard

  (simple reduction from clique problem)

- There is a simple schema mapping M defined by s-t tgds, and a Boolean FO query Q for which EVAL(M, Q) is undecidable.

Evaluation of Universal Queries

universal query: FO query of the form $\forall \bar{x}\, \varphi$, φ quantifier-free

Theorem (H. '10)

For every properly restricted schema mapping M and for each universal query Q there is a polynomial time algorithm for:

Input: *the core solution for some source instance S for M*
Output: *the GCWA*-answers to Q on M and S*

Evaluation of Universal Queries

universal query: FO query of the form $\forall \bar{x}\, \varphi$, φ quantifier-free

Theorem (H. '10)

For every properly restricted schema mapping M and for each universal query Q there is a polynomial time algorithm for:

Input: the core solution for some source instance S for M

Output: the GCWA*-answers to Q on M and S

Restriction: M specified by packed s-t tgds

$$\forall \bar{x} \forall \bar{y} \Big(\varphi(\bar{x}, \bar{y}) \rightarrow \exists \bar{z} \; \cdots R(\cdots z \cdots) \wedge \cdots \wedge R'(\cdots z \cdots) \cdots \Big)$$

Evaluation of Universal Queries

universal query: FO query of the form $\forall \bar{x}\, \varphi$, φ quantifier-free

Theorem (H. '10)

For every properly restricted schema mapping M and for each universal query Q there is a polynomial time algorithm for:

Input: *the core solution for some source instance S for M*
Output: *the GCWA*-answers to Q on M and S*

Restriction: M specified by packed s-t tgds

$$\forall \bar{x} \forall \bar{y} \Big(\varphi(\bar{x}, \bar{y}) \rightarrow \exists \bar{z}\ \cdots R(\cdots z \cdots) \wedge \cdots \wedge R'(\cdots z \cdots) \cdots \Big)$$

Recall: Here the core solution can be computed in polynomial time

Step 1/4: Reduction to Satisfiability Problem

M: schema mapping, defined by packed s-t tgds
Q: universal query (Boolean)

Input: source instance S (for the moment)
Question: Are the GCWA*-answers to Q non-empty?

Step 1/4: Reduction to Satisfiability Problem

M: schema mapping, defined by packed s-t tgds
Q: universal query (Boolean)

Input: source instance S (for the moment)
Question: Are the GCWA*-answers to Q non-empty?

- Idea: test whether there is a GCWA*-solution T with $T \models \neg Q$

Step 1/4: Reduction to Satisfiability Problem

M: schema mapping, defined by packed s-t tgds
Q: universal query (Boolean)

Input: source instance S (for the moment)

Question: Are the GCWA*-answers to Q non-empty?

- Idea: test whether there is a GCWA*-solution T with $T \models \neg Q$
- Observation:

$$\neg Q \equiv \neg \forall \bar{x}\, \varphi(\bar{x}) \qquad \varphi\text{: quantifier-free}$$

Step 1/4: Reduction to Satisfiability Problem

M: schema mapping, defined by packed s-t tgds
Q: universal query (Boolean)

Input: source instance S (for the moment)

Question: Are the GCWA*-answers to Q non-empty?

- Idea: test whether there is a GCWA*-solution T with $T \models \neg Q$
- Observation:

$$\neg Q \;\equiv\; \exists \bar{x} \neg \varphi(\bar{x}) \qquad\qquad \varphi\text{: quantifier-free}$$

Step 1/4: Reduction to Satisfiability Problem

M: schema mapping, defined by packed s-t tgds
Q: universal query (Boolean)

Input: source instance S (for the moment)

Question: Are the GCWA*-answers to Q non-empty?

- Idea: test whether there is a GCWA*-solution T with $T \models \neg Q$
- Observation:

$$\neg Q \;\equiv\; \exists \bar{x} \bigvee_{i=1}^{n} \varphi_i(\bar{x}_i)$$

φ_i: conjunction of atoms or negated atoms

Step 1/4: Reduction to Satisfiability Problem

M: schema mapping, defined by packed s-t tgds
Q: universal query (Boolean)

Input: source instance S (for the moment)
Question: Are the GCWA*-answers to Q non-empty?

- Idea: test whether there is a GCWA*-solution T with $T \models \neg Q$
- Observation:

$$\neg Q \;\equiv\; \bigvee_{i=1}^{n} \exists \bar{x}_i \, \varphi_i(\bar{x}_i)$$

φ_i: conjunction of atoms or negated atoms

Step 1/4: Reduction to Satisfiability Problem

M: schema mapping, defined by packed s-t tgds
Q: universal query (Boolean)

Input: source instance S (for the moment)
Question: Are the GCWA*-answers to Q non-empty?

- Idea: test whether there is a GCWA*-solution T with $T \models \neg Q$
- Observation:

$$\neg Q \equiv \bigvee_{i=1}^{n} \exists \bar{x}_i \, \varphi_i(\bar{x}_i)$$

φ_i: conjunction of atoms or negated atoms

- Remains: test whether for some i there is a GCWA*-solution T for S with

$$T \models \exists \bar{x}_i \, \varphi_i(\bar{x}_i)$$

Step 1/4: Reduction to Satisfiability Problem

M: schema mapping, defined by packed s-t tgds
Q: universal query (Boolean)

Input: source instance S (for the moment)
Question: Are the GCWA*-answers to Q non-empty?

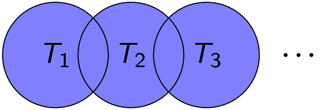
- Idea: test whether there is a GCWA*-solution T with $T \models \neg Q$
- Observation:

$$\neg Q \equiv \bigvee_{i=1}^{n} \exists \bar{x}_i \, \varphi_i(\bar{x}_i)$$

φ_i: conjunction of atoms or negated atoms

- Remains: test whether for some i there is a GCWA*-solution T for S with

$$T \models \exists \bar{x}_i \, \varphi_i(\bar{x}_i)$$



Step 1/4: Reduction to Satisfiability Problem

M: schema mapping, defined by packed s-t tgds
Q: universal query (Boolean)

Input: source instance S (for the moment)
Question: Are the GCWA*-answers to Q non-empty?

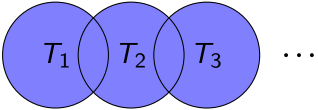
- Idea: test whether there is a GCWA*-solution T with $T \models \neg Q$
- Observation:

$$\neg Q \;\equiv\; \bigvee_{i=1}^{n} \exists \bar{x}_i \, \varphi_i(\bar{x}_i)$$

φ_i: conjunction of atoms or negated atoms

- Remains: test whether for some i there is a set $\mathcal{T}$ of ground minimal solutions for S with $1 \leq |\mathcal{T}|$ and

$$\bigcup \mathcal{T} \;\models\; \exists \bar{x}_i \, \varphi_i(\bar{x}_i)$$



Step 1/4: Reduction to Satisfiability Problem

M: schema mapping, defined by packed s-t tgds
Q: universal query (Boolean)

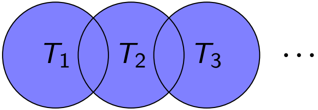
Input: source instance S (for the moment)
Question: Are the GCWA*-answers to Q non-empty?

- Idea: test whether there is a GCWA*-solution T with $T \models \neg Q$
- Observation:

$$\neg Q \equiv \bigvee_{i=1}^{n} \exists \bar{x}_i \, \varphi_i(\bar{x}_i) \qquad \varphi_i\text{: conjunction of atoms or negated atoms}$$

- Remains: test whether for some i there is a set $\mathcal{T}$ of ground minimal solutions for S with $1 \leq |\mathcal{T}| \leq |\varphi_i|$ and

$$\bigcup \mathcal{T} \models \exists \bar{x}_i \, \varphi_i(\bar{x}_i)$$



Step 2/4: Reformulation in Terms of the Core

Query: $\exists \bar{x}\, \varphi(\bar{x})$, φ conjunction of atoms and neg. atoms

Question: Are there ground minimal solutions $T_1, \ldots, T_{|\varphi|}$ for S with

$$\bigcup_i T_i \models \exists \bar{x}\, \varphi(\bar{x}) \; ?$$

Step 2/4: Reformulation in Terms of the Core

Query: $\exists \bar{x}\, \varphi(\bar{x})$, φ conjunction of atoms and neg. atoms

Question: Are there ground minimal solutions $T_1, \dots, T_{|\varphi|}$ for S with

$$\bigcup_i T_i \models \exists \bar{x}\, \varphi(\bar{x}) \ ?$$

Lemma

ground minimal solutions for S

$=$ *minimal possible worlds of the core solution for S*

Step 2/4: Reformulation in Terms of the Core

Query: $\exists\bar{x}\,\varphi(\bar{x})$, φ conjunction of atoms and neg. atoms

Question: Are there ground minimal solutions $T_1, \ldots, T_{|\varphi|}$ for S with

$$\bigcup_i T_i \models \exists\bar{x}\,\varphi(\bar{x}) \; ?$$

Lemma

ground minimal solutions for S

$= $ *minimal possible worlds of the core solution for S*

New question: Are there minimal possible worlds $T_1, \ldots, T_{|\varphi|}$ of the core solution for S with $\bigcup_i T_i \models \exists\bar{x}\,\varphi(\bar{x})$?

Step 3/4: Find Appropriate Minimal Instances

Lemma

M: schema mapping defined by packed s-t tgds
Q: query $\exists \bar{x}\, \varphi(\bar{x})$, φ conjunction of atoms and negated atoms

There is a polynomial time algorithm for

Input:	*core solution C for some source instance S for M*
Question:	*Are there minimal possible worlds $T_1, \ldots, T_{\lvert\varphi\rvert}$ of C with $\bigcup_i T_i \models Q$*

Step 3/4: Find Appropriate Minimal Instances

Lemma

M: schema mapping defined by packed s-t tgds
Q: query $\exists \bar{x}\, \varphi(\bar{x})$, φ conjunction of atoms and negated atoms

There is a polynomial time algorithm for

Input:	*core solution C for some source instance S for M*
Question:	*Are there minimal possible worlds $T_1, \ldots, T_{\vert\varphi\vert}$ of C with $\bigcup_i T_i \models Q$*

Problems to overcome:

- In general, infinitely many minimal possible worlds of C
  Solution: canonical representation

Step 3/4: Find Appropriate Minimal Instances

Lemma

M: schema mapping defined by packed s-t tgds
Q: query $\exists \bar{x}\, \varphi(\bar{x})$, φ conjunction of atoms and negated atoms

There is a polynomial time algorithm for

Input: *core solution C for some source instance S for M*
Question: *Are there minimal possible worlds $T_1, \ldots, T_{|\varphi|}$ of C with $\bigcup_i T_i \models Q$*

Problems to overcome:

- In general, infinitely many minimal possible worlds of C
  Solution: canonical representation
- Still exponentially many instances
  Solution: reduce set of instances that need to be considered to polynomial size

Step 4/4: A Special Case

Reduction for special case: given atom $R(\bar{a})$, test whether $R(\bar{a})$ belongs to some minimal instance in $poss(C)$

❶ Key property: number of nulls in atom blocks of C bounded by a constant (Fagin, Kolaitis, Popa '03)

Step 4/4: A Special Case

Reduction for special case: given atom $R(\bar{a})$, test whether $R(\bar{a})$ belongs to some minimal instance in $poss(C)$

1. Key property: number of nulls in atom blocks of C bounded by a constant (Fagin, Kolaitis, Popa '03)

$$C = \{E(a,\bot), \\ E(b,a) \\ R(a,\bot,\bot')\}$$

Gaifman graph:

$$\begin{array}{ccc} E(a,\bot) & \quad & E(b,a) \\ | & & \\ R(a,\bot,\bot') & & \end{array}$$

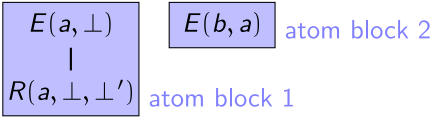
Step 4/4: A Special Case

Reduction for special case: given atom $R(\bar{a})$, test whether $R(\bar{a})$ belongs to some minimal instance in $poss(C)$

1. Key property: number of nulls in atom blocks of C bounded by a constant (Fagin, Kolaitis, Popa '03)

$$C = \{E(a, \perp), \\ E(b, a) \\ R(a, \perp, \perp')\}$$

Gaifman graph:



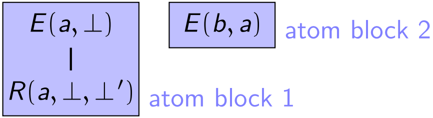
Step 4/4: A Special Case

Reduction for special case: given atom $R(\bar{a})$, test whether $R(\bar{a})$ belongs to some minimal instance in $poss(C)$

1. Key property: number of nulls in atom blocks of C bounded by a constant (Fagin, Kolaitis, Popa '03)

$$C = \{E(a, \perp), \\ E(b, a) \\ R(a, \perp, \perp')\}$$

Gaifman graph:



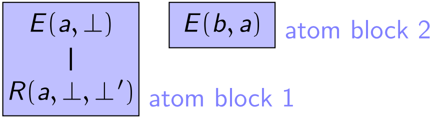
2. First idea: use minimal instances arising from atom blocks of C by replacing nulls with constants ...

Reduction for special case: given atom $R(\bar{a})$, test whether $R(\bar{a})$ belongs to some minimal instance in $poss(C)$

1. Key property: number of nulls in atom blocks of C bounded by a constant (Fagin, Kolaitis, Popa '03)

$$C = \{E(a, \perp), E(b, a), R(a, \perp, \perp')\}$$

Gaifman graph:



2. First idea: use minimal instances arising from atom blocks of C by replacing nulls with constants . . . fails

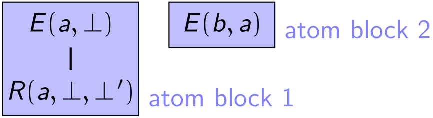
Step 4/4: A Special Case

Reduction for special case: given atom $R(\bar{a})$, test whether $R(\bar{a})$ belongs to some minimal instance in $poss(C)$

1. Key property: number of nulls in atom blocks of C bounded by a constant (Fagin, Kolaitis, Popa '03)

$$C = \{E(a, \perp), \; E(b, a) \; R(a, \perp, \perp')\}$$

Gaifman graph:



2. First idea: use minimal instances arising from atom blocks of C by replacing nulls with constants . . . fails
3. Instead: consider the cores of images of C under special mappings

Step 4/4: A Special Case

Reduction for special case: given atom $R(\bar{a})$, test whether $R(\bar{a})$ belongs to some minimal instance in $poss(C)$

1. Key property: number of nulls in atom blocks of C bounded by a constant (Fagin, Kolaitis, Popa '03)

$$C = \{E(a, \perp), \\ E(b, a) \\ R(a, \perp, \perp')\}$$

Gaifman graph:

$\boxed{E(a, \perp) \text{ — } R(a, \perp, \perp')}$ atom block 1

$\boxed{E(b, a)}$ atom block 2

2. First idea: use minimal instances arising from atom blocks of C by replacing nulls with constants . . . fails

3. Instead: consider the cores of images of C under special mappings . . . here packed s-t tgds come into play

Summary

- Widely agreed: for monotonic queries use the certain answers

Summary

- Widely agreed: for monotonic queries use the certain answers
  - answering queries preserved under homomorphisms well understood

Summary

- Widely agreed: for monotonic queries use the certain answers
  - answering queries preserved under homomorphisms well understood
  - few results for more general monotonic queries

Summary

- Widely agreed: for monotonic queries use the certain answers
  - answering queries preserved under homomorphisms well understood
  - few results for more general monotonic queries
- Several semantics for non-monotonic queries

Summary

- Widely agreed: for monotonic queries use the certain answers
  - answering queries preserved under homomorphisms well understood
  - few results for more general monotonic queries
- Several semantics for non-monotonic queries
  - based on rules for excluding undesired solutions

Summary

- Widely agreed: for monotonic queries use the certain answers
  - answering queries preserved under homomorphisms well understood
  - few results for more general monotonic queries
- Several semantics for non-monotonic queries
  - based on rules for excluding undesired solutions
  - each reflects a certain intuition about what “not mentioned” by a source instance and schema mapping means

Summary

- Widely agreed: for monotonic queries use the certain answers
  - answering queries preserved under homomorphisms well understood
  - few results for more general monotonic queries
- Several semantics for non-monotonic queries
  - based on rules for excluding undesired solutions
  - each reflects a certain intuition about what "not mentioned" by a source instance and schema mapping means
  - query evaluation may be hard, is not really understood

Open Problems

Lots of open problems, e.g.:

- When is (non-monotonic) query answering tractable?

Open Problems

Lots of open problems, e.g.:

- When is (non-monotonic) query answering tractable?
  - For which queries and schema mappings?

Open Problems

Lots of open problems, e.g.:

- When is (non-monotonic) query answering tractable?
  - For which queries and schema mappings?
  - . . . and under which semantics?

Open Problems

Lots of open problems, e.g.:

- When is (non-monotonic) query answering tractable?
  - For which queries and schema mappings?
  - . . . and under which semantics?
  - Data complexity? Combined complexity?

Open Problems

Lots of open problems, e.g.:

- When is (non-monotonic) query answering tractable?
  - For which queries and schema mappings?
  - . . . and under which semantics?
  - Data complexity? Combined complexity?
- Alternative approaches, e.g., stick with the certain answers, but use richer constraint language

Bibliography

- Fagin, Kolaitis, Miller, and Popa. Data exchange: Semantics and query answering. ICDT 2003
- Libkin. Data exchange and incomplete information. PODS 2006
- H. and Schweikardt. CWA-solutions for data exchange settings with target dependencies. PODS 2007
- Libkin and Sirangelo. Data exchange and schema mappings in open and closed worlds. PODS 2008
- Afrati and Kolaitis. Answering aggregate queries in data exchange. PODS 2008
- H. and Schweikardt. Logic and data exchange: Which solutions are good solutions? In *Logic and the Foundations of Game and Decision Theory (LOFT 8)*, 2008
- H. Answering non-monotonic queries in relational data exchange. ICDT 2010