Containment of Conjunctive Queries Over Conceptual Schemata

Andrea Calì

Faculty of Computer Science
Free University of Bolzano

KRDB Seminar
Bolzano, 10th May 2006

What is query containment?

Definition

Given two queries Q_1 and Q_2 we say Q_1 is contained in Q_2, denoted $Q_1 \subseteq Q_2$, if for every database D we have

$$Q_1(D) \subseteq Q_2(D)$$

$Q(D)$: result of evaluation of Q over D

Query containment

- Fundamental issue in query optimisation
- We consider conjunctive queries over schemata schemata with constraints
- The presence of constraints makes query containment checking difficult
- Need for reasoning on constraints imposed by the database schema
- Q_1 contained in Q_2 under Σ, denoted $Q_1 \subseteq_\Sigma Q_2$, if for every database D that satisfies Σ we have $Q_1(D) \subseteq Q_2(D)$

Conjunctive query containment: algorithm

1. **freeze** $body(Q_1)$ and $head(Q_1)$ by turning each variable into a distinct (fresh) constant
2. evaluate Q_2 over the frozen body of Q_1
3. $Q_1 \subseteq Q_2$ iff the evaluation returns the frozen head of Q_1

Testing containment amounts to checking the existence of a **query homomorphism** from Q_2 to Q_1 [Chandra & Merlin 1977].

Example

From [Ullman 1997]

$$
\begin{array}{llll}
Q_1: & p(X,Z) & \leftarrow & a(X,Y), a(Y,Z) \\
Q_2: & p(X,Z) & \leftarrow & a(X,U), a(V,Z)
\end{array}
$$

Example

From [Ullman 1997]

$$
\begin{array}{llll}
Q_1: & p(X,Z) & \leftarrow & a(X,Y), a(Y,Z) \\
Q_2: & p(X,Z) & \leftarrow & a(X,U), a(V,Z)
\end{array}
$$

Frozen $body(Q_1)$:

$$
\begin{array}{l}
a(0,1) \leftarrow \\
a(1,2) \leftarrow
\end{array}
$$

Example

From [Ullman 1997]

$$
\begin{array}{llcl}
Q_1 : & p(X,Z) & \leftarrow & a(X,Y), a(Y,Z) \\
Q_2 : & p(X,Z) & \leftarrow & a(X,U), a(V,Z)
\end{array}
$$

Frozen $body(Q_1)$:

$$
\begin{array}{l}
a(0,1) \leftarrow \\
a(1,2) \leftarrow
\end{array}
$$

Frozen $head(Q_1)$: $p(0,2) \leftarrow$

Example (contd.)

Applying Q_2 to the frozen $body(Q_1)$, we find a substitution:

$$X \rightarrow 0,\ U \rightarrow 1,\ V \rightarrow 1,\ Z \rightarrow 2$$

that yelds $p(0,2)$ which is the frozen head of Q_1.

Example (contd.)

Applying Q_2 to the frozen $body(Q_1)$, we find a substitution:

$$X \rightarrow 0, \ U \rightarrow 1, \ V \rightarrow 1, \ Z \rightarrow 2$$

that yelds $p(0,2)$ which is the frozen head of Q_1. Therefore $Q_1 \subseteq Q_2$.

Example (contd.)

Applying Q_2 to the frozen $body(Q_1)$, we find a substitution:

$$X \to 0,\ U \to 1,\ V \to 1,\ Z \to 2$$

that yelds $p(0,2)$ which is the frozen head of Q_1. Therefore $Q_1 \subseteq Q_2$.

Note

The frozen body of Q_1 is a representative of (a piece of) all databases that provide an answer to Q_1

Outline

Query containment under constraints

Definition

Given a set Σ of database dependencies, we say Q_1 **is contained in** Q_2 under Σ, denoted $Q_1 \subseteq_\Sigma Q_2$, if **for every database** D such that $D \models \Sigma$ we have

$$Q_1(D) \subseteq Q_2(D)$$

Our setting

Queries

- conjunctive queries (CQs)

Dependencies

1. key dependencies (KDs)
   $key(R) = \{A_1, \ldots, A_k\}$
2. inclusion dependencies (IDs) (generalisation of foreign key dependencies)
   $R_1[A_1, \ldots, A_m] \subseteq R_2[B_1, \ldots, B_m]$

QC under constraints: example

Schema

$$\text{employee}(\mathit{Emp\_id}, \mathit{Salary}, \mathit{Dept})$$
$$\text{department}(\mathit{Dept}, \mathit{Location})$$

with constraint employee[3] $\subseteq$ department[1].

QC under constraints: example

Schema

$$\text{employee}(Emp\_id, Salary, Dept)$$
$$\text{department}(Dept, Location)$$

with constraint employee[3] $\subseteq$ department[1].

Queries

$$Q_1 : \quad p(X) \quad \leftarrow \quad \text{employee}(X, Y, Z), \text{department}(Z, W)$$
$$Q_2 : \quad p(X) \quad \leftarrow \quad \text{employee}(X, Y, Z)$$

QC under constraints: example

Schema

$$\begin{array}{l} \text{employee}(Emp\_id, Salary, Dept) \\ \text{department}(Dept, Location) \end{array}$$

with constraint employee[3] $\subseteq$ department[1].

Queries

$$\begin{array}{llll} Q_1 : & p(X) & \leftarrow & \text{employee}(X, Y, Z), \text{department}(Z, W) \\ Q_2 : & p(X) & \leftarrow & \text{employee}(X, Y, Z) \end{array}$$

$Q_1 \subseteq Q_2$ and $Q_2 \not\subseteq Q_1$, but notice that $Q_2 \subseteq_\Sigma Q_1$ (queries are equivalent under Σ).

Checking QC under database dependencies

Intuition

- Once we freeze Q_1 we are constructing a generic database that provides an answer to Q_1
- When we freeze, we must construct a database that satisfies Σ
- We do that by constructing the chase of the frozen query

The chase

The chase "repairs" the frozen body of Q_1 in two ways:

1. "collapsing" pairs of facts that violate a KD (**KD chase rule**)
2. adding facts when an ID is violated (**ID chase rule**)

Containment test under dependencies

Theorem [Johnson & Klug 1982]

$Q_2 \subseteq_\Sigma Q_1$ iff there is a homomorphism that maps $body(Q_2)$ on the chase of the frozen $body(Q_1)$, and the head of Q_2 to the frozen head of Q_1

Note

The chase is a representative for all databases that satisfy Σ and provide an answer for Q_1.

Infinite chase: example

Relations $R/2,\ S/2$

Dependencies

$$\begin{array}{ll} \sigma_1 : & R[1] \subseteq S[1] \\ \sigma_2 : & S[2] \subseteq R[1] \\ \sigma_3 : & S[2] \subseteq S[1] \\ \gamma 1 : & key(R)=\{1\} \end{array}$$

Infinite chase: example

Relations $R/2,\ S/2$

Dependencies

$$\begin{array}{llll} \sigma_1 : & R[1] & \subseteq & S[1] \\ \sigma_2 : & S[2] & \subseteq & R[1] \\ \sigma_3 : & S[2] & \subseteq & S[1] \\ \gamma 1 : & \multicolumn{3}{l}{key(R) = \{1\}} \end{array}$$

Initial database (frozen body)

$$R(a, \alpha_0) \leftarrow$$
$$R(a, b) \leftarrow$$

greek letters denote (frozen) nondistinguished variables (NDVs) and fresh constants

Infinite chase: example (contd.)

KD chase rule

We collapse the first two facts (due to γ_1) by forcing $\alpha_0 = b$.

Infinite chase: example (contd.)

KD chase rule

We collapse the first two facts (due to γ_1) by forcing $\alpha_0 = b$.

ID chase rule

Added facts due to IDs:

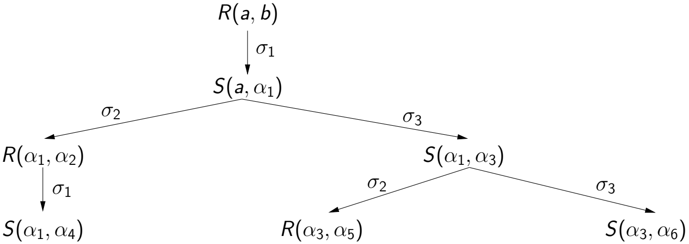
$$
\begin{array}{l}
S(a, \alpha_1) \leftarrow \\
R(\alpha_1, \alpha_2) \leftarrow \\
S(\alpha_1, \alpha_3) \leftarrow \\
S(\alpha_1, \alpha_4) \leftarrow \\
R(\alpha_3, \alpha_5) \leftarrow \\
S(\alpha_3, \alpha_6) \leftarrow \\
\ldots
\end{array}
$$

(...ad infinitum!)

Chase graph

- Facts in the frozen body of Q_1 have level 0. A fact derived from the ID chase rule from another fact that is at level k has level $k + 1$
- If fact f_2 is derived from fact f_1 by an ID σ, there is an arc (f_1, f_2) labelled with σ

Example: chase graph



Undecidability of QC under KDs and IDs

Known result

QC under general functional dep. and IDs is undecidable [Chandra & Vardi 1982]

Undecidability of QC under KDs and IDs

Known result

QC under general functional dep. and IDs is undecidable [Chandra & Vardi 1982]

Theorem

QC under general KDs and IDs is undecidable

Proof sketch: Reduction from implication of KDs and IDs. Consider R/n, S/m, a set of dep. Σ and a constraint $\sigma: \ R[1,\ldots,k] \subseteq S[1,\ldots,k]$.

$$
\begin{array}{llll}
Q_1 : & Q() & \leftarrow & R(X_1,\ldots,X_k,\ldots,X_n) \\
Q_2 : & Q() & \leftarrow & R(X_1,\ldots,X_k,\ldots,X_n), \\
 & & & S(X_1,\ldots,X_k,Y_1,\ldots,Y_{m-k})
\end{array}
$$

it is easy to see that $Q_1 \subseteq_\Sigma Q_2$ iff $\Sigma \models \sigma$.

QC under IDs alone

Theorem [Johnson & Klug JCSS 1984]

Containment is decidable in PSPACE.

Proof sketch:

- only a finite portion of the chase is necessary
- notion of **equivalent conjuncts** (agree on non-fresh constants)
- given a fact, an equivalent conjunct is found within $\delta = |\Sigma| \cdot (W+1)^W$, W maximum "width" of IDs in Σ
- Taking into account joins in Q_2: the necessary depth is $|Q_2| \cdot \delta$
- A nontrivial guess shows memberships in PSPACE
- PSPACE-hardness is also proved (like undecidability)

QC under KDs and IDs: decidable cases

- unary IDs
- key-based IDs [Johnson & Klug 1984]; limited class, but is more general than forign keys
- **non-key-conflicting IDs** [Calì & al. PODS 2003]; more general class than key-based IDs

Non-key-conflicting IDs

Definition

Non-key-conflicting IDs (NKCIDs) are of the form

$$R_1[\mathbf{A}_1] \subseteq R_2[\mathbf{A}_2]$$

where either:

1. no KD is defined over r_2
2. $\mathbf{A}_2$ is **not** a strict superset of $key(R_2)$

Separation Theorem

Theorem

Given Q_1, Q_2, Σ (NKCIDs), if the chase w.r.t. Σ does not fail in the first applications of the FD chase rule:

$$Q_1 \subseteq Q_2 \text{ iff } Q_1 \subseteq_\Sigma Q_2$$

Proof

Based on the fact that if KDs are not violated in the first step of the chase, they are never violated

Summary of complexity results

KDs	**IDs**	**complexity**
no	GEN	PSPACE
yes	no	NP
yes	FK	PSPACE
yes	NKC	PSPACE
yes	1KC	undecidable
yes	GEN	undecidable

Containment of queries over conceptual schemata

- We consider a conceptual model (Extended ER, EER) derived by enriching Chen's ER model
- Conjunctive queries formulated on predicates referring the the constructs of the conceptual schema
- Need for checking containment under constraints derived from the conceptual schema
- Constraints are represented with inclusion dependencies (IDs) and key dependencies (KDs)

⋆ Decidability of query containment approaches

Extended ER schemata

We consider ER schemata enriched with:

- IS-A among entities **and relationships**
- mandatory (at least once) participation constraints
- functional (at most once) participation constraints

Extended ER schemata

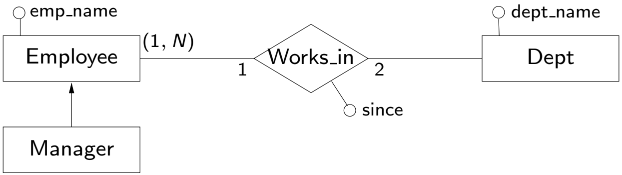
We consider ER schemata enriched with:

- IS-A among entities **and relationships**
- mandatory (at least once) participation constraints
- functional (at most once) participation constraints

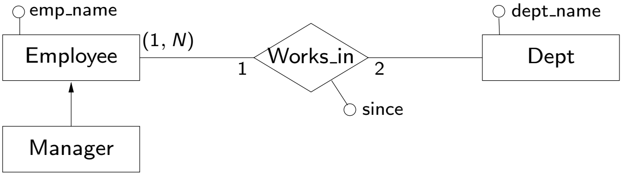
Relational representation of EER schemata:

- entities $\longrightarrow$ unary relations
- relationships $\longrightarrow$ n-ary relations
- attributes $\longrightarrow$ n-ary relations (binary for entities)
- mandatory participation constraints $\longrightarrow$ IDs
- functional participation constraints $\longrightarrow$ KDs

Representing and querying ER schemata: example



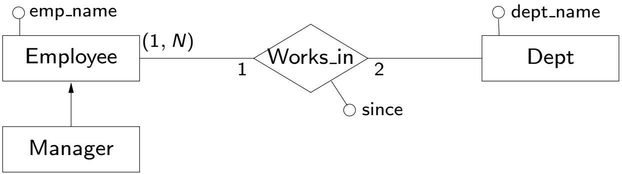
Representing and querying ER schemata: example



Constraints: they are always key and inclusion dependencies

$$\text{employee}[1] \subseteq \text{works\_in}[1]$$
$$key(\text{works\_in}) = \{1\}$$
$$\text{manager}[1] \subseteq \text{employee}[1]$$
$$\ldots$$

Representing and querying ER schemata: example



Constraints: they are always key and inclusion dependencies

$$\text{employee}[1] \subseteq \text{works\_in}[1]$$
$$\mathit{key}(\text{works\_in}) = \{1\}$$
$$\text{manager}[1] \subseteq \text{employee}[1]$$
$$\ldots$$

$$Q(X) \leftarrow \text{manager}(X), \text{works\_in}(X, Y), \text{since}(X, Y, 1999)$$
$$\text{dept}(Y), \text{dept\_name}(Y, \mathit{toy\_dept})$$

The chase as a tool for containment checking: recall

The chase of a query database is obtained by:

1. "freezing" the query:
   - turn each atom into a fact
   - leave constants unaltered
   - turn variables into "fresh" constants
2. adding facts to satisfy IDs;
3. collapsing (if possible) facts to satisfy KDs

The chase as a tool for containment checking: recall

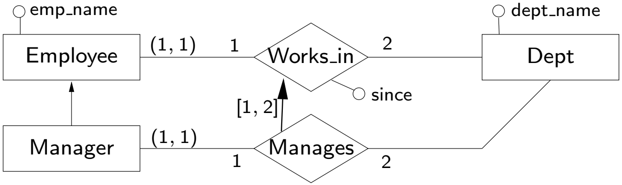
The **chase** of a query database is obtained by:

1. "freezing" the query:
   - turn each atom into a fact
   - leave constants unaltered
   - turn variables into "fresh" constants
2. adding facts to satisfy IDs;
3. collapsing (if possible) facts to satisfy KDs

To check $Q_1 \subseteq_\Sigma Q_2$:

1. we consider the chase of Q_1
2. we evaluate Q_2 over such chase
3. if the evaluation returns the frozen head of Q_1, then $Q_1 \subseteq_\Sigma Q_2$

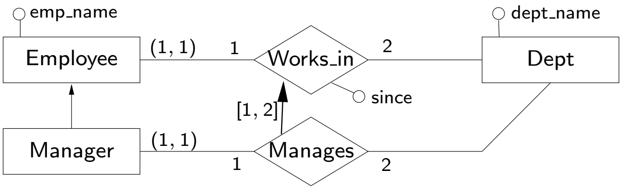
Chase for EER dependencies: example



Initial facts (frozen query):

$$\text{manager}(m) \leftarrow$$
$$\text{manages}(m, d) \leftarrow$$

Chase for EER dependencies: example



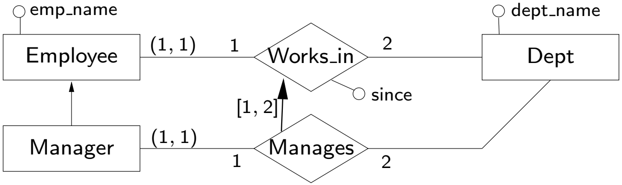
Initial facts (frozen query):

$$\text{manager}(m) \leftarrow$$
$$\text{manages}(m, d) \leftarrow$$

Facts added in the chase:

$$\text{employee}(m) \leftarrow$$
$$\text{works\_in}(m, \alpha) \leftarrow$$
$$\text{works\_in}(m, d) \leftarrow$$
$$\text{dept}(\alpha) \leftarrow$$
$$\text{dept}(d) \leftarrow$$

Chase for EER dependencies: example



Initial facts (frozen query):

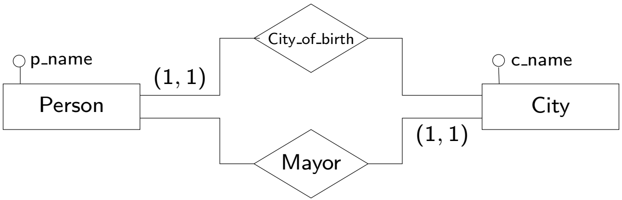
$\text{manager}(m) \leftarrow$
$\text{manages}(m, d) \leftarrow$

Facts added in the chase:

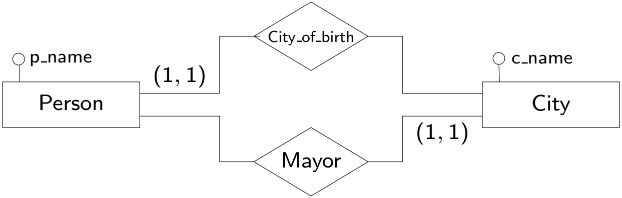
$\text{employee}(m) \leftarrow$
$\text{works\_in}(m, \alpha) \leftarrow$
$\text{works\_in}(m, d) \leftarrow$
$\text{dept}(\alpha) \leftarrow$
$\text{dept}(d) \leftarrow$

We must deduce $\alpha = d$ (α is a fresh constant) and replace this value in all the segment of the chase constructed so far.

The need for unbounded models: example



The need for unbounded models: example



Initial fact (frozen query):

$$\text{person}(p) \leftarrow$$

Graph representation of the chase

Chase for EER dependencies: properties

- Violations of KDs are possible only in a very initial segment
- We prove that a finite segment of the chase, until depth $|\Sigma| \cdot W! \cdot |Q_2|$, is sufficient to test containment
  - W is the maximum number of attributes involved by an ID in Σ;
- In principle we do not know how far we should go with the chase until we stop, since collapses may propagate back from some "deep" (late) part

Query answering under EER dependencies: decidability and complexity

Containment check $Q_1 \subseteq_\Sigma Q_2$:

Main result

Propagation of collapses between constants does not go back more than a fixed “distance” in the chase; such distance is $|\Sigma| \cdot W!$

Query answering under EER dependencies: decidability and complexity

Containment check $Q_1 \subseteq_\Sigma Q_2$:

Main result

Propagation of collapses between constants does not go back more than a fixed "distance" in the chase; such distance is $|\Sigma| \cdot W!$

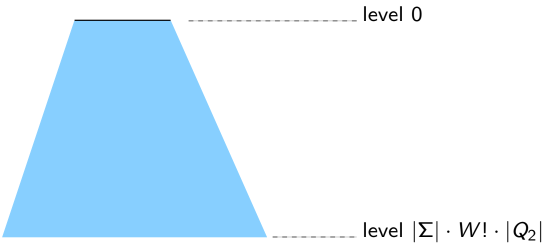
Technique

We proceed until level $|\Sigma| \cdot W! \cdot |Q_2|$ (initial segment) plus another $|\Sigma| \cdot W!$ levels

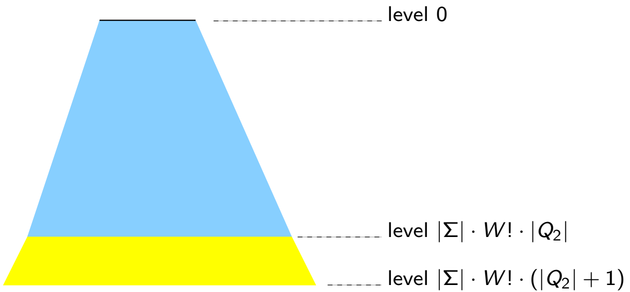
1. after that no collapse will affect the inital segment;
2. the segment we have is enough for checking query containment

Construction of the relevant segment of the chase

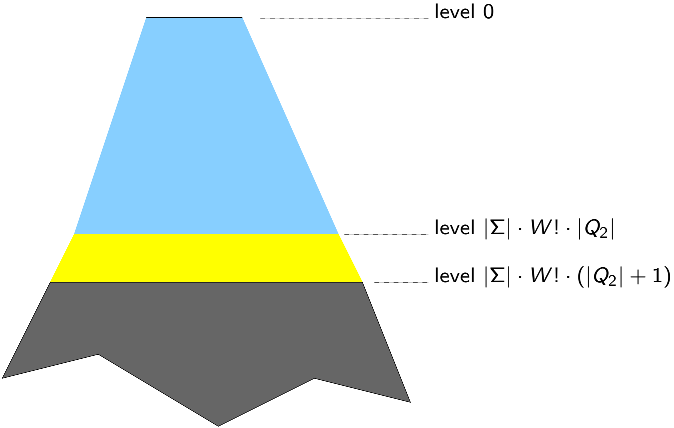
Construction of the relevant segment of the chase (cont.)



Construction of the relevant segment of the chase (cont.)



Construction of the relevant segment of the chase (cont.)



Complexity of containment

Complexity of checking $Q_1 \subseteq_\Sigma Q_2$ (upper bound):

- exponential in $|Q_2|$
- polynomial in $|Q_1|$
- exponential in $|\Sigma|$
- double exponential in W

Conclusions

- Techniques for checking conjunctive query containment under constraints
- Use of the chase for containment checking
- Decidable cases and tractability
- Decidability of query containment under EER constraints
  - ⋆ proof technique based on chase
  - ⋆ results extend to querying incomplete data
- Characterisation of complexity of query containment in different cases

Related work

Containment under KDs and IDs

- Chase technique: [Beeri & Vardi JACM 1984]
- Case of IDs alone: treated in [Johnson & Klug JCSS 1984]: containment in PSPACE
- Extension to in [Calì et al. PODS 2003] to KDs and **non-key-conflicting IDs** (NKCIDs), again in PSPACE

Related work (contd.)

Containment on EER schemata

- Decidability of the QC problem on EER schemata derived from [Calvanese et al. PODS 1998] by encoding into CPDL;
- We achieve a lower complexity by providing a direct insight into the structure of the chase
- Other relevant approaches:
  - DL-Lite [Calvanese et al. KR 2005] (captures conceptual schemata without IS-A among n-ary roles)
  - [Ortiz et al. 2005] (higher expressiveness and complexity)

The End

Part of the results are from a joint work with:

- Domenico Lembo
- Riccardo Rosati

Thanks to:

- Maurizio Lenzerini
- Michael Kifer
- Leo Bertossi
- Diego Calvanese
- TONES (Thinking ONtologiES) EU project

Slides typeset with LaTeX2ε