Foundations of Query Languages

Dr. Fang Wei

Lehrstuhl für Datenbanken und Informationssysteme
Universität Freiburg

SS 2011

Problemes

Let Q, Q_1, Q_2 be conjunctive queries.

Containment: $Q_1 \sqsubseteq Q_2$, i.e., $Q_1(\mathcal{I}) \subseteq Q_2(\mathcal{I})$ for any instance $\mathcal{I}$?

Equivalence: $Q_1 \equiv Q_2$, i.e., $Q_1 \sqsubseteq Q_2$ and $Q_2 \sqsubseteq Q_1$?

Minimization: Given Q_1, construct an equivalent query Q_2, which has as most as much subgoals in its body as Q_1 and is minimal in the sense, that any query Q_3 being equivalent to Q_2 has at least as much subgoals in the body as Q_2.

Q_2 is called *minimal*.

Example

$$\begin{gathered}
Sales(Part,\ Supplier,\ Customer), \\
Part(PName,\ Type), \\
Cust(CName,\ CAddr), \\
Supp(SName,\ SAddr).
\end{gathered}$$

Equivalent queries:

$$Q: \quad ans(T) \leftarrow Sales(P, S, C), Part(P, T), Cust(C, A), Supp(S, A)$$

$$\begin{aligned}
Q': \quad ans(T) \leftarrow\ & Sales(P, S, C), Part(P, T), Cust(C, A), Supp(S, A), \\
& Sales(P', S', C'), Part(P', T)
\end{aligned}$$

Lemma

Let

$$\begin{array}{ll} Q_1: & ans(\vec{U}) \leftarrow R_1(\vec{U_1}), \ldots, R_n(\vec{U_n}) \\ Q_2: & ans(\vec{U}) \leftarrow S_1(\vec{V_1}), \ldots, S_m(\vec{V_m}) \end{array}$$

be conjunctive queries, where

$$\{R_1(\vec{U_1}), \ldots, R_n(\vec{U_n})\} \supseteq \{S_1(\vec{V_1}), \ldots, S_m(\vec{V_m})\}$$

Then $Q_1 \sqsubseteq Q_2$.

Substitution

- A *substitution* θ over a set of variables $\mathcal{V}$ is a mapping from $\mathcal{V}$ to $\mathcal{V} \cup$ **dom**, where **dom** a corresponding domain.
- We extend θ to constants $a \in$ **dom** and relation names $R \in \mathcal{R}$, where $\theta(a) = a$, resp. $\theta(R) = R$.

Example

Consider

$$Q: \quad ans(T) \leftarrow Sales(P,S,C), Part(P,T), Cust(C,A), Supp(S,A)$$

$$Q': \quad ans(T) \leftarrow Sales(P,S,C), Part(P,T), Cust(C,A), Supp(S,A), Sales(P',S',C'), Part(P',T)$$

and θ:

X	P	P'	S	S'	C	C'	T	A
$\theta(X)$	P	P	S	S	C	C	T	A

Containment Mapping

Given conjunctive queries

$$
\begin{aligned}
Q_1 : \quad & ans(\vec{U}) \leftarrow R_1(\vec{U_1}), \ldots, R_n(\vec{U_n}) \\
Q_2 : \quad & ans(\vec{V}) \leftarrow S_1(\vec{V_1}), \ldots, S_m(\vec{V_m})
\end{aligned}
$$

Substitution θ is called *containment mapping* (a.k.a. Homomorphism) from Q_2 to Q_1, if Q_2 can be transformed by means of θ to become Q_1:

- $\theta(ans(\vec{V})) = ans(\vec{U})$,
- for $i = 1, \ldots, m$ there exists a $j \in \{1, \ldots, n\}$, such that $\theta(S_i(\vec{V_i})) = R_j(\vec{U_j})$.

Example

$$Q: \quad ans(T) \leftarrow Sales(P,S,C), Part(P,T), Cust(C,A), Supp(S,A)$$

$$Q': \quad ans(T) \leftarrow Sales(P,S,C), Part(P,T), Cust(C,A), Supp(S,A), Sales(P',S',C'), Part(P',T)$$

θ:

X	P	P'	S	S'	C	C'	T	A
$\theta(X)$	P	P	S	S	C	C	T	A

θ is a containment mapping.

Theorem

Let

$$
\begin{aligned}
Q_1 : \quad & ans(\vec{U}) \leftarrow R_1(\vec{U_1}), \ldots, R_n(\vec{U_n}) \\
Q_2 : \quad & ans(\vec{V}) \leftarrow S_1(\vec{V_1}), \ldots, S_m(\vec{V_m})
\end{aligned}
$$

be conjunctive queries.
$Q_1 \sqsubseteq Q_2$ iff there exists a containment mapping θ from Q_2 to Q_1.

Proof "$\Leftarrow$":

There exists containment mapping θ.
Let $\mathcal{I}$ be an instance of Q_1 and let $\mu \in Q_1(\mathcal{I})$.
There exists a substitution τ, such that $\tau(\vec{U}_j) \in \mathcal{I}(R_j)$, $j \in \{1, \ldots, n\}$ and $\mu = \tau(\vec{U})$.
Consider a substitution $\tau' = \tau \circ \theta$ and further $\tau'(S_i(\vec{V}_i))$.
There holds $\tau'(\vec{V}_i) \in \mathcal{I}(S_i)$, $i \in \{1, \ldots, m\}$ and therefore also $\mu = \tau'(\vec{V})$.
That is, $\mu \in Q_2(\mathcal{I})$.

Canonical Instance

Let Q be a conjunctive $ans(\vec{U}) \leftarrow R_1(\vec{U_1}), \ldots, R_n(\vec{U_n})$ over a database schema $\mathcal{R}$. The *canonical instance* $\mathcal{I}_Q$ to Q is constructed as follows.

- $\mathcal{I}_Q$ is an instance of $\mathcal{R} = \{R_1, \ldots, R_n\}$.
- Let τ be a substitution, which assigns to any X in Q an unique constant a_X.
- For any literal $R(t_1, \ldots, t_n)$ in the body, insert a tupel of the form $(\tau(t_1), \ldots, \tau(t_n))$ into $\mathcal{I}_Q(R)$; we also write $\tau(R(t_1, \ldots, t_n)) \in \mathcal{I}_Q(R)$.
- No other tuples are inserted into $\mathcal{I}_Q(R)$.

τ is called *canonical substitution*.

Example

$Q: \quad ans(T) \leftarrow Sales(P,S,C), Part(P,T), Cust(C,A), Supp(S,A)$

$Q': \quad ans(T) \leftarrow Sales(P,S,C), Part(P,T), Cust(C,A), Supp(S,A), Sales(P',S',C'), Part(P',T)$

$\mathcal{I}_Q$:

Sales

a_P	a_S	a_C

Part

a_P	a_T

Cust

a_C	a_A

Supp

a_S	a_A

$\mathcal{I}_{Q'}$:

Sales

a_P	a_S	a_C
$a_{P'}$	$a_{S'}$	$a_{C'}$

Part

a_P	a_T
$a_{P'}$	a_T

Cust

a_C	a_A

Supp

a_S	a_A

Proof "$\Rightarrow$":

$Q_1 \sqsubseteq Q_2$.
Consider $\mathcal{I}_{Q_1}$ and the corresponding canonical substitution τ.
Then $\tau(ans(\vec{U})) \in Q_1(\mathcal{I}_{Q_1})$.
Because of $Q_1 \sqsubseteq Q_2$ further $\tau(ans(\vec{U})) \in Q_2(\mathcal{I}_{Q_1})$.
Thus, there exists a substitution ρ, such that $\rho(S_i(\vec{V}_i)) = \tau(R_j(\vec{U}_j))$, $1 \leq i \leq m$, $j \in \{1, \ldots, n\}$ und $\rho(ans(\vec{V})) = \tau(ans(\vec{U}))$.
$\tau^{-1} \circ \rho$ is a containment mapping.

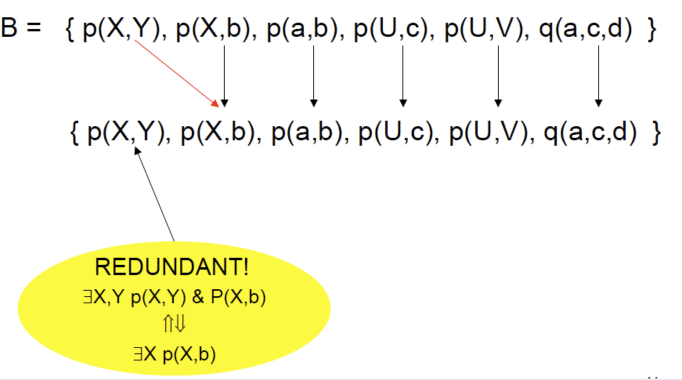
CQ minimization

- A conjunctive query Q is minimal, if it contains minimal number of subgoals (subgoal = body atom)
- This means: whenewer we drop an atomfrom the body of Q, we get a query Q', such that $Q' \neq Q$.
- The task of the CQ minimization: minimize the number of subgoalsof a given CQ.

Example

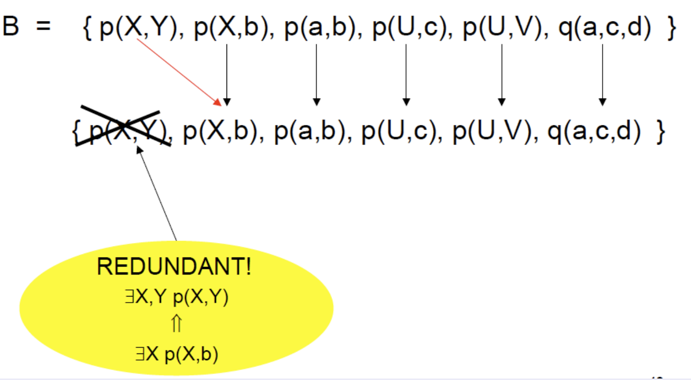
$$true \leftarrow p(X, Y), p(X, b), p(a, b), p(U, c), p(U, V), q(a, c, d).$$

Example



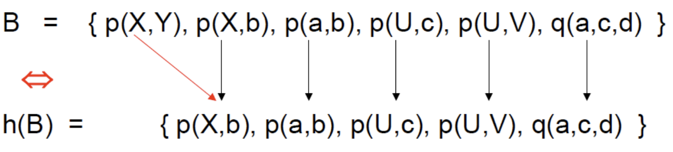
Containment mapping: $h : Y \to b$.

Example



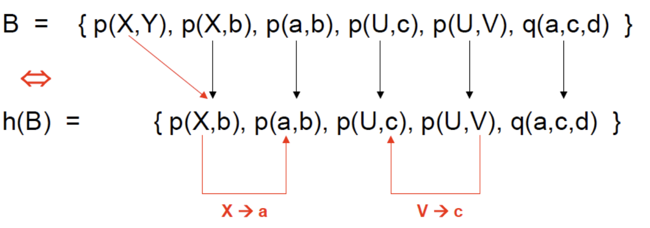
Containment mapping: $h : Y \to b$.

Example



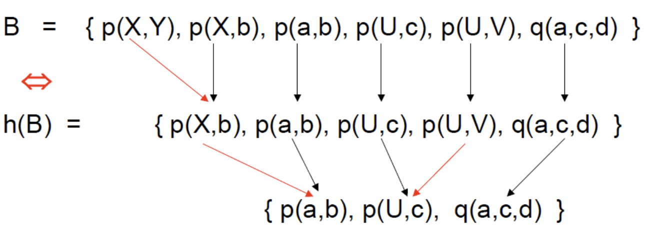
Containment mapping: $h : Y \rightarrow b$.

Example



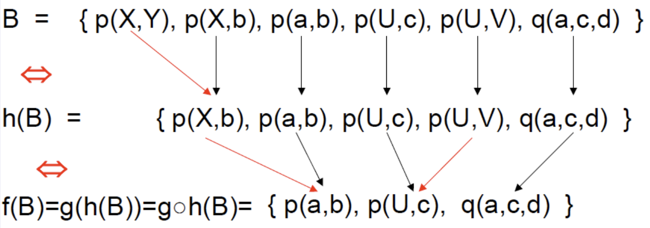
Containment mapping: $g : X \rightarrow a, V \rightarrow c$.

Example



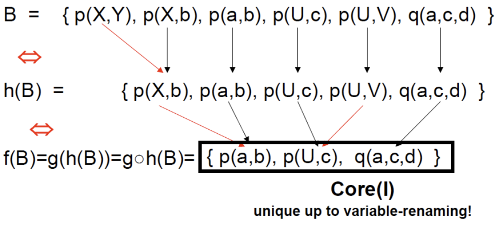
Containment mapping: $g : X \rightarrow a, V \rightarrow c$.

Example



Containment mapping: $f : Y \to b, X \to a, V \to c$.

Example



Containment mapping: $f : Y \to b, X \to a, V \to c.$

Minimization algorithm

- Can we just check the subgoalsone by one? (if one subgoalcan not be removed at contain moment, could it be removed at a later moment?)
- Does the sequence of the elimination count? (or, is the minimized query unique?)

Minimization algorithm

- Can we just check the subgoals one by one?
  Yes. Take the subgoal u, in a CQ Q_1, check whether there is a containment mapping h, such that $h(Q_1) \subseteq (Q_1 - u)$. If there exists such a containment mapping, then u can be removed. Otherwise, u can not be removed forever. Because Q_1 is getting smaller.
- Does the sequence of the elimination count? (or, is the minimized query unique?)
  –homework