

Conjunctive Queries: Evaluation and Static Analysis

(Chapters 14, 15 and 16 of DBT)

A Core Relational Query Language

Conjunctive Queries (CQ)

= $\{\sigma, \pi, \bowtie\}$ -fragment of relational algebra

= relational calculus using only \exists and \wedge

= simple SELECT-FROM-WHERE SQL queries
(only AND and equality in the WHERE clause)

Syntax of Conjunctive Queries

$$Q(\mathbf{x}) := \exists \mathbf{y} (R_1(\mathbf{v}_1) \wedge \cdots \wedge R_m(\mathbf{v}_m))$$

- R_1, \dots, R_m are relation names
- $\mathbf{x}, \mathbf{y}, \mathbf{v}_1, \dots, \mathbf{v}_m$ are tuples of variables
- each variable mentioned in \mathbf{v}_i appears either in \mathbf{x} or \mathbf{y}
- the variables in \mathbf{x} are free called **distinguished** or **output variables**

It is very convenient to see conjunctive queries as rule-based queries of the form

$$Q(\mathbf{x}) :- \underbrace{R_1(\mathbf{v}_1), \dots, R_m(\mathbf{v}_m)}$$

this is called the **body** of Q that can be seen as a set of atoms

Pattern Matching Problem

List the airlines that fly directly from London to Glasgow

Flight(VIE,LHR,BA),	Airport(VIE,Vienna),
Flight(LHR,EDI,BA),	Airport(LHR,London),
Flight(LGW,GLA,U2),	Airport(LGW,London),
Flight(LCA,VIE,OS),	Airport(LCA,Larnaca),
	Airport(GLA,Glasgow),
	Airport(EDI,Edinburgh)

Q(z) :- Airport(x,London), Airport(y,Glasgow), Flight(x,y,z)

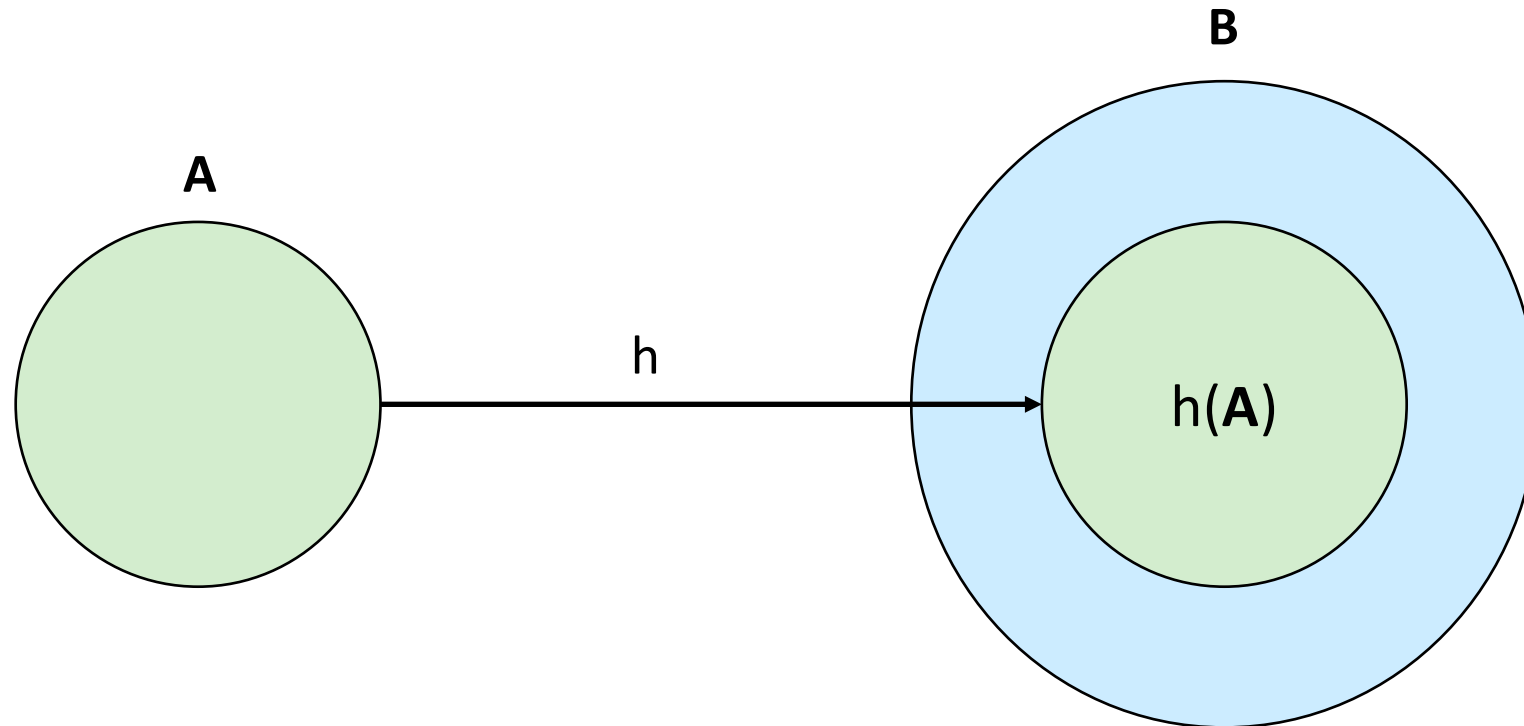
Pattern Matching Problem

List the airlines that fly directly from London to Glasgow

Flight(VIE,LHR,BA),	Airport(VIE,Vienna),
Flight(LHR,EDI,BA),	Airport(LHR,London),
Flight(LGW,GLA,U2),	Airport(LGW,London),
Flight(LCA,VIE,OS),	Airport(LCA,Larnaca),
	Airport(GLA,Glasgow),
	Airport(EDI,Edinburgh)

Q(z) :- Airport(x,London), Airport(y,Glasgow), Flight(x,y,z)

Homomorphism



$h : \text{terms}(\mathbf{A}) \rightarrow \text{terms}(\mathbf{B})$ that is the identity on constants

Semantics of Conjunctive Queries

- A **match** of a conjunctive query $Q(x_1, \dots, x_k) :- \text{body}$ in a database D is a homomorphism h from the set of atoms **body** to the set of atoms D

- The **answer** to $Q(x_1, \dots, x_k) :- \text{body}$ over D is the set of k -tuples

$$Q(D) := \{(h(x_1), \dots, h(x_k)) \mid h \text{ is a match of } Q \text{ in } D\}$$

- The answer consists of the witnesses for the **distinguished variables** of Q

Pattern Matching Problem

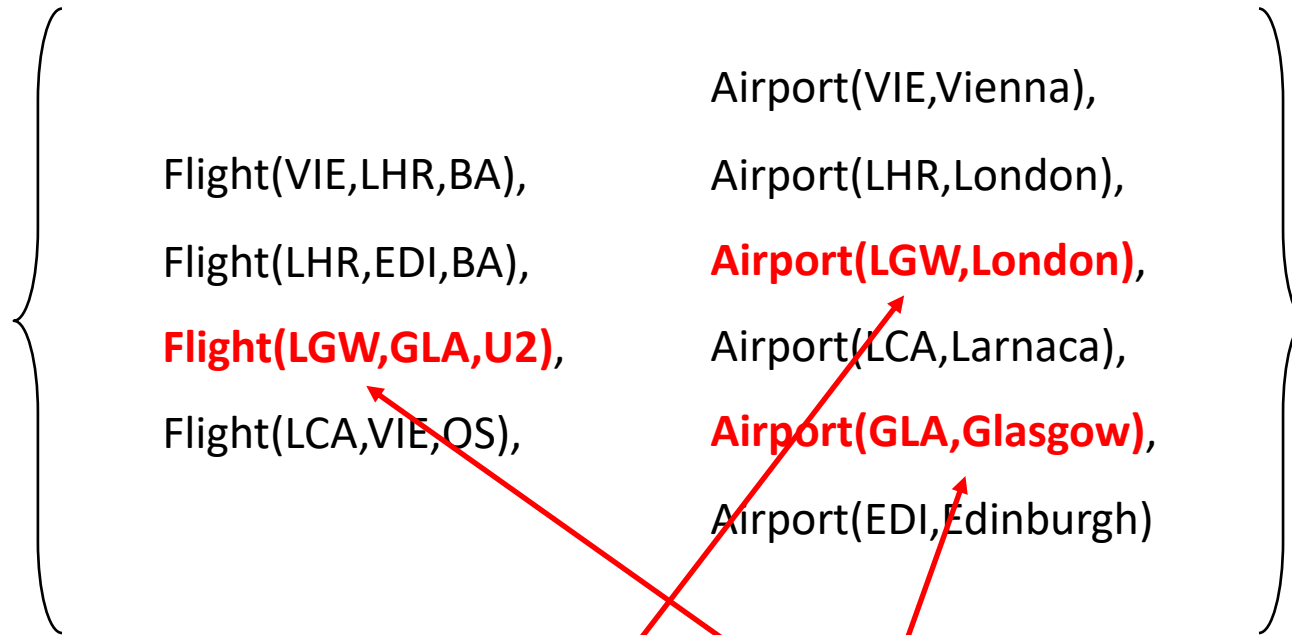
List the airlines that fly directly from London to Glasgow

Flight(VIE,LHR,BA),	Airport(VIE,Vienna),
Flight(LHR,EDI,BA),	Airport(LHR,London),
Flight(LGW,GLA,U2),	Airport(LGW,London),
Flight(LCA,VIE,OS),	Airport(LCA,Larnaca),
	Airport(GLA,Glasgow),
	Airport(EDI,Edinburgh)

Q(z) :- Airport(x,London), Airport(y,Glasgow), Flight(x,y,z)

Pattern Matching Problem

List the airlines that fly directly from London to Glasgow



$\{x \mapsto \text{LGW}, y \mapsto \text{GLA}, z \mapsto \text{U2},$
 $\text{London} \mapsto \text{London}, \text{Glasgow} \mapsto \text{Glasgow}\}$

Q(z) :- Airport(x,London), Airport(y,Glasgow), Flight(x,y,z)

Query Evaluation

- Understand the complexity of evaluating a conjunctive query over a database
- What to measure? Queries may have a large output and it would be misleading to count the output as “complexity”
- We therefore consider the following decision problem for **CQ**

CQ-Evaluation

Input: a database D , a CQ $Q(x_1, \dots, x_k) :- \text{body}$, and a tuple (a_1, \dots, a_k) of values

Question: $(a_1, \dots, a_k) \in Q(D)$?

combined complexity

Data Complexity of Query Evaluation

- Measures the complexity in terms of the size of the database - the query is fixed
- Meaningful in practice since the database is usually much bigger than the query
- We consider the following decision problem for a fixed CQ $Q(x_1, \dots, x_k) :- \text{body}$

Q-Evaluation

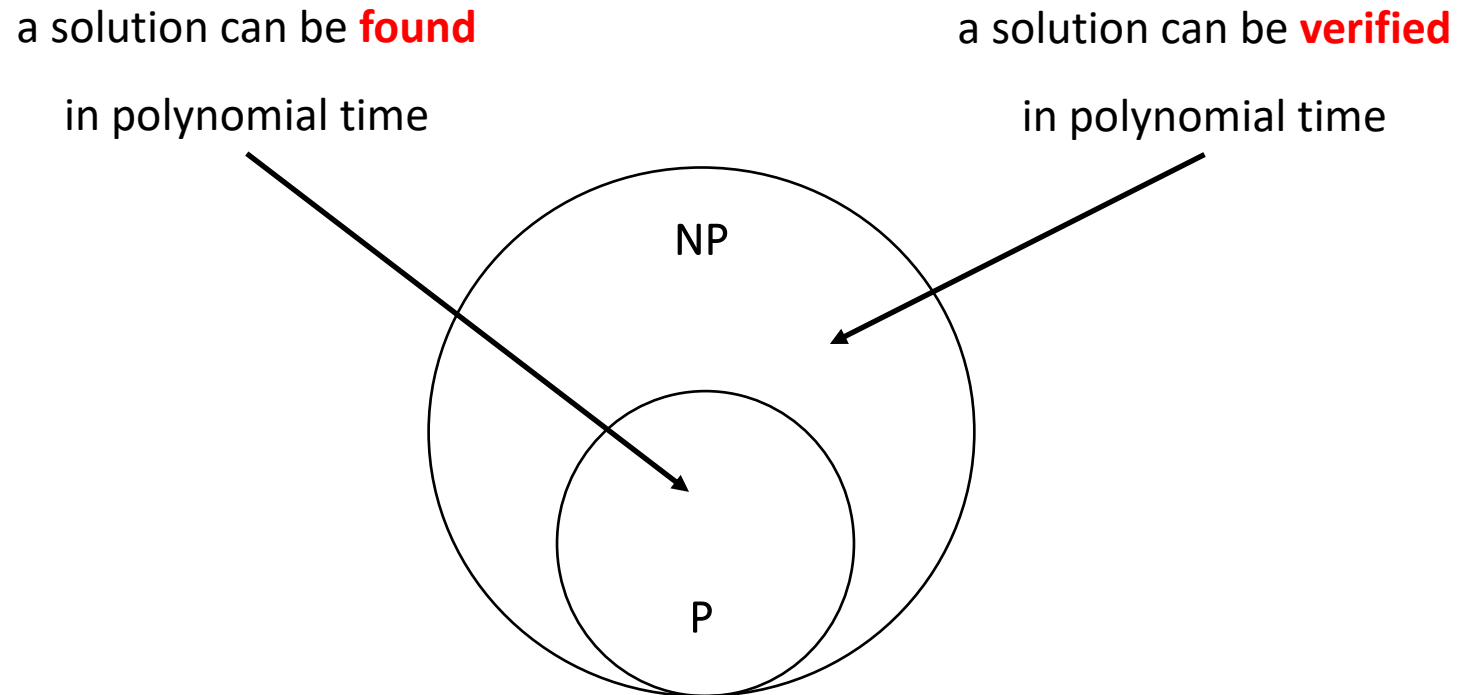
Input: a database D , and a tuple (a_1, \dots, a_k) of values

Question: $(a_1, \dots, a_k) \in Q(D)$?

Complexity of Query Evaluation

Theorem: CQ-Evaluation is NP-complete and in PTIME in data complexity

Few Words about NP



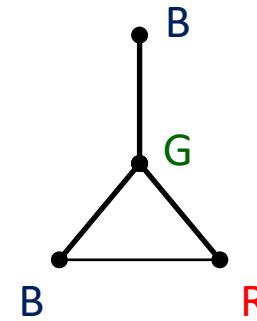
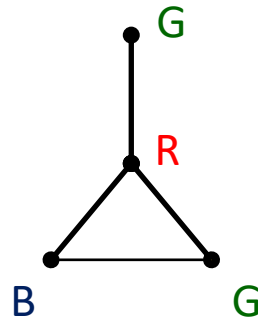
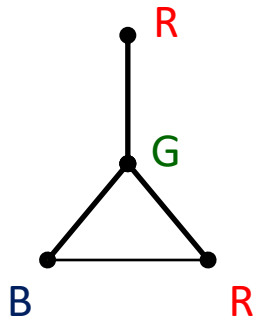
- $P \subseteq NP$, but it is open whether $P \subset NP$ or $P = NP$ - one of the most important questions in mathematics and theoretical computer science
- One of the **Millennium Prize Problems** selected by the Clay Mathematics Institute in 2000 (1 million USD prize for the first correct solution)

3-Colorability Problem

3COL

Input: an undirected graph $G = (V, E)$

Question: is there a function $c : V \rightarrow \{R, G, B\}$ such that $(v, u) \in E \Rightarrow c(v) \neq c(u)$?



3-Colorability Problem

3COL

Input: an undirected graph $G = (V, E)$

Question: is there a function $c : V \rightarrow \{R, G, B\}$ such that $(v, u) \in E \Rightarrow c(v) \neq c(u)$?

Theorem: 3COL is NP-complete

Proof:

- **Guess** a function $c : V \rightarrow \{R, G, B\}$ in polynomial time, and **verify** that $(v, u) \in E \Rightarrow c(v) \neq c(u)$ in polynomial time
- 3COL is one of the most difficult problems in the complexity class NP (it is unlikely to be solvable in polynomial time unless $P = NP$)

Complexity of Query Evaluation

Theorem: CQ-Evaluation is NP-complete and in PTIME in data complexity

Proof:

(NP-membership) Guess-and-verify:

- Consider a database D , a CQ $Q(x_1, \dots, x_k) :- \text{body}$, and a tuple (a_1, \dots, a_k) of values
- Guess a substitution $h : \text{terms}(\text{body}) \rightarrow \text{terms}(D)$ that is the identity on constants
- Verify that h is a match of Q in D , i.e., $h(\text{body}) \subseteq D$ and $(h(x_1), \dots, h(x_k)) = (a_1, \dots, a_k)$

(NP-hardness) Reduction from 3COL

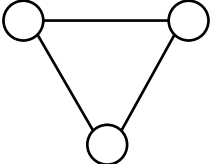
NP-hardness

(NP-hardness) Reduction from 3COL

3COL

Input: an undirected graph $\mathbf{G} = (V, E)$

Question: is there a function $c : V \rightarrow \{R, G, B\}$ such that $(v, u) \in E \Rightarrow c(v) \neq c(u)$?

Lemma: \mathbf{G} is 3-colorable iff \mathbf{G} can be mapped to \mathbf{K}_3 , i.e., $\mathbf{G} \xrightarrow{\text{hom}}$ 

therefore, \mathbf{G} is 3-colorable iff there is a match of $Q_{\mathbf{G}}$ in $D = \{E(x, y), E(y, z), E(z, x)\}$

the Boolean CQ that represents \mathbf{G}

Complexity of Query Evaluation

Theorem: CQ-Evaluation is NP-complete and in PTIME in data complexity

Proof:

(NP-membership) Guess-and-verify:

- Consider a database D , a CQ $Q(x_1, \dots, x_k) :- \text{body}$, and a tuple (a_1, \dots, a_k) of values
- Guess a substitution $h : \text{terms}(\text{body}) \rightarrow \text{terms}(D)$ that is the identity on constants
- Verify that h is a match of Q in D , i.e., $h(\text{body}) \subseteq D$ and $(h(x_1), \dots, h(x_k)) = (a_1, \dots, a_k)$

(NP-hardness) Reduction from 3-colorability

(in PTIME) For every substitution $h : \text{terms}(\text{body}) \rightarrow \text{terms}(D)$ that is the identity on constants, check if $h(\text{body}) \subseteq D$ and $(h(x_1), \dots, h(x_k)) = (a_1, \dots, a_k)$

Static Analysis

CQ-Satisfiability

Input: a conjunctive query Q

Question: is there a database D such that $Q(D)$ is non-empty?

- If the answer is no, then the input query Q makes no sense
- **CQ-Evaluation** becomes trivial - the answer is always NO!

Static Analysis

CQ-Equivalence

Input: two conjunctive queries Q_1 and Q_2

Question: $Q_1 \equiv Q_2$? or $Q_1(D) = Q_2(D)$ for every database D ?

- Replace a query Q_1 with a query Q_2 that is easier to evaluate
- But, we have to be sure that $Q_1(D) = Q_2(D)$ for every database D

Static Analysis

CQ-Containment

Input: two conjunctive queries Q_1 and Q_2

Question: $Q_1 \subseteq Q_2$? or $Q_1(D) \subseteq Q_2(D)$ for every database D ?

- Equivalence boils down to two containment checks
- Clearly, $Q_1(D) = Q_2(D)$ iff $Q_1(D) \subseteq Q_2(D)$ and $Q_2(D) \subseteq Q_1(D)$

Complexity of Static Analysis

CQ-Satisfiability

Input: a conjunctive query Q

Question: is there a database D such that $Q(D)$ is non-empty?

CQ-Equivalence

Input: two conjunctive queries Q_1 and Q_2

Question: $Q_1 \equiv Q_2$? or $Q_1(D) = Q_2(D)$ for every database D ?

CQ-Containment

Input: two conjunctive queries Q_1 and Q_2

Question: $Q_1 \subseteq Q_2$? or $Q_1(D) \subseteq Q_2(D)$ for every database D ?

Canonical Database

- Convert a conjunctive query Q into a database $D[Q]$ - the **canonical database** of Q
- Given a conjunctive query of the form $Q(\mathbf{x}) :- \text{body}$, $D[Q]$ is obtained from body by replacing each variable x with a new value $c(x) = \underline{x}$
- E.g., given $Q(x,y) :- R(x,y), P(y,z,w), R(z,x)$, then $D[Q] = \{R(\underline{x},\underline{y}), P(\underline{y},\underline{z},\underline{w}), R(\underline{z},\underline{x})\}$
- **Note:** The mapping $c : \{\text{variables in body}\} \rightarrow \{\text{new values}\}$ is a **bijection**, where $c(\text{body}) = D[Q]$ and $c^{-1}(D[Q]) = \text{body}$

Satisfiability of CQs

CQ-Satisfiability

Input: a conjunctive query Q

Question: is there a database D such that $Q(D)$ is non-empty?

Theorem: A conjunctive query Q is always satisfiable

Proof: Due to its canonical database - $Q(D[Q])$ is trivially non-empty

Equivalence and Containment of CQs

CQ-Equivalence

Input: two conjunctive queries Q_1 and Q_2

Question: $Q_1 \equiv Q_2$? or $Q_1(D) = Q_2(D)$ for every database D ?

CQ-Containment

Input: two conjunctive queries Q_1 and Q_2

Question: $Q_1 \subseteq Q_2$? or $Q_1(D) \subseteq Q_2(D)$ for every database D ?

$$Q_1 \equiv Q_2 \text{ iff } Q_1 \subseteq Q_2 \text{ and } Q_2 \subseteq Q_1$$

$$Q_1 \subseteq Q_2 \text{ iff } Q_1 \equiv (Q_1 \wedge Q_2)$$

...thus, we can safely focus on **CQ-Containment**

Homomorphism Theorem

A **query homomorphism** from $Q_1(x_1, \dots, x_k) \text{ :- } \text{body}_1$ to $Q_2(y_1, \dots, y_k) \text{ :- } \text{body}_2$

is a substitution $h : \text{terms}(\text{body}_1) \rightarrow \text{terms}(\text{body}_2)$ such that:

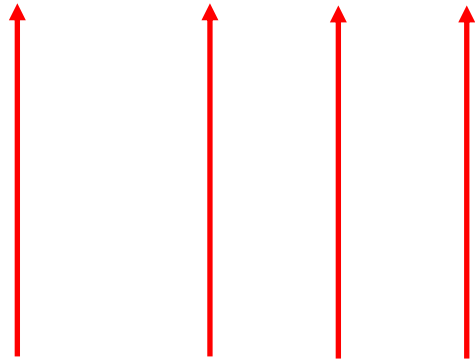
1. h is a homomorphism from body_1 to body_2
2. $(h(x_1), \dots, h(x_k)) = (y_1, \dots, y_k)$

Homomorphism Theorem: Let Q_1 and Q_2 be conjunctive queries. It holds that:

$Q_1 \subseteq Q_2$ iff there exists a query homomorphism from Q_2 to Q_1

Homomorphism Theorem: Example

$Q_1(x,y) :- R(x,z), S(z,z), R(z,y)$



$Q_2(a,b) :- R(a,c), S(c,d), R(d,b)$

$h = \{a \mapsto x, b \mapsto y, c \mapsto z, d \mapsto z\}$

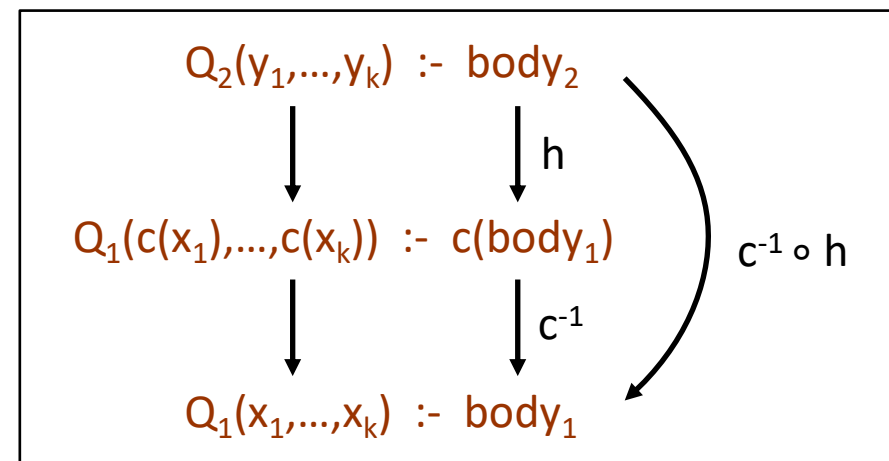
- h is a query homomorphism from Q_2 to $Q_1 \Rightarrow Q_1 \subseteq Q_2$
- But, there is no homomorphism from Q_1 to $Q_2 \Rightarrow Q_1 \not\subseteq Q_2$

Homomorphism Theorem: Proof

Assume that $Q_1(x_1, \dots, x_k) \text{ :- body}_1$ and $Q_2(y_1, \dots, y_k) \text{ :- body}_2$

$(\Rightarrow) Q_1 \subseteq Q_2 \Rightarrow$ there exists a query homomorphism from Q_2 to Q_1

- Clearly, $(c(x_1), \dots, c(x_k)) \in Q_1(D[Q_1])$ - recall that $D[Q_1] = c(\text{body}_1)$
- Since $Q_1 \subseteq Q_2$, we conclude that $(c(x_1), \dots, c(x_k)) \in Q_2(D[Q_1])$
- Therefore, there exists a homomorphism h such that $h(\text{body}_2) \subseteq D[Q_1] = c(\text{body}_1)$
and $h((y_1, \dots, y_k)) = (c(x_1), \dots, c(x_k))$
- By construction, $c^{-1}(c(\text{body}_1)) = \text{body}_1$
and $c^{-1}((c(x_1), \dots, c(x_k))) = (x_1, \dots, x_k)$
- Therefore, $c^{-1} \circ h$ is a
query homomorphism from Q_2 to Q_1



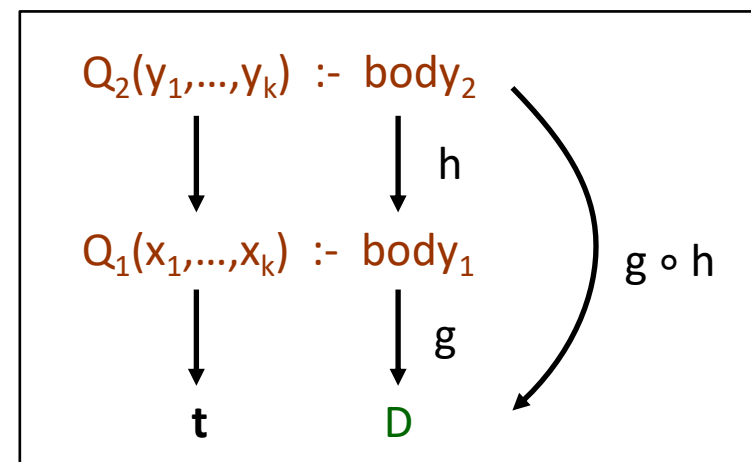
Homomorphism Theorem: Proof

Assume that $Q_1(x_1, \dots, x_k) \text{ :- body}_1$ and $Q_2(y_1, \dots, y_k) \text{ :- body}_2$

$(\Leftarrow) Q_1 \subseteq Q_2 \Leftarrow$ there exists a query homomorphism from Q_2 to Q_1

- Consider a database D , and a tuple \mathbf{t} such that $\mathbf{t} \in Q_1(D)$
- We need to show that $\mathbf{t} \in Q_2(D)$
- Clearly, there exists a homomorphism g such that $g(\text{body}_1) \subseteq D$ and $g((x_1, \dots, x_k)) = \mathbf{t}$
- By hypothesis, there exists a query homomorphism h from Q_2 to Q_1
- Therefore, $g(h(\text{body}_2)) \subseteq D$ and

$g(h((y_1, \dots, y_k))) = \mathbf{t}$, which implies that $\mathbf{t} \in Q_2(D)$



Existence of a Query Homomorphism

Theorem: Let Q_1 and Q_2 be conjunctive queries. The problem of deciding whether there exists a query homomorphism from Q_2 to Q_1 is NP-complete

Proof:

(NP-membership) Guess a substitution and verify that is a query homomorphism

(NP-hardness) Easy reduction from **CQ-Evaluation**

By applying the homomorphism theorem we get that:

Corollary: **CQ-Equivalence** and **CQ-Containment** are NP-complete

Complexity of Static Analysis

CQ-Satisfiability

trivial

Input: a conjunctive query Q

Question: is there a database D such that $Q(D)$ is non-empty?

CQ-Equivalence

NP-complete

Input: two conjunctive queries Q_1 and Q_2

Question: $Q_1 \equiv Q_2$? or $Q_1(D) = Q_2(D)$ for every database D ?

CQ-Containment

NP-complete

Input: two conjunctive queries Q_1 and Q_2

Question: $Q_1 \subseteq Q_2$? or $Q_1(D) \subseteq Q_2(D)$ for every database D ?

Minimizing Conjunctive Queries

- **Goal:** minimize the number of joins in a query
- A conjunctive query Q_1 is **minimal** if there is no conjunctive query Q_2 such that:
 1. $Q_1 \equiv Q_2$
 2. Q_2 has fewer atoms than Q_1
- The task of **CQ minimization** is, given a conjunctive query Q , to compute a minimal one that is equivalent to Q

Minimization by Deletion

By exploiting the homomorphism theorem we can show the following:

Theorem: Consider a conjunctive query $Q_1(x_1, \dots, x_k) :- \text{body}_1$.

If Q_1 is equivalent to a conjunctive query $Q_2(y_1, \dots, y_k) :- \text{body}_2$ where $|\text{body}_2| < |\text{body}_1|$,

then Q_1 is equivalent to a query $Q_3(x_1, \dots, x_k) :- \text{body}_3$ such that $\text{body}_3 \subseteq \text{body}_1$



The above theorem says that to minimize a conjunctive query $Q_1(x_1, \dots, x_k) :- \text{body}$ we simply need to remove some atoms from body

Minimization Procedure

Minimization($Q(x_1, \dots, x_k) :- \text{body}$)

Repeat until no change

choose an atom $\alpha \in \text{body}$ such that the variables x_1, \dots, x_k appear in $\text{body} \setminus \{\alpha\}$

if there is a query homomorphism from $Q(x_1, \dots, x_k) :- \text{body}$ to $Q(x_1, \dots, x_k) :- \text{body} \setminus \{\alpha\}$

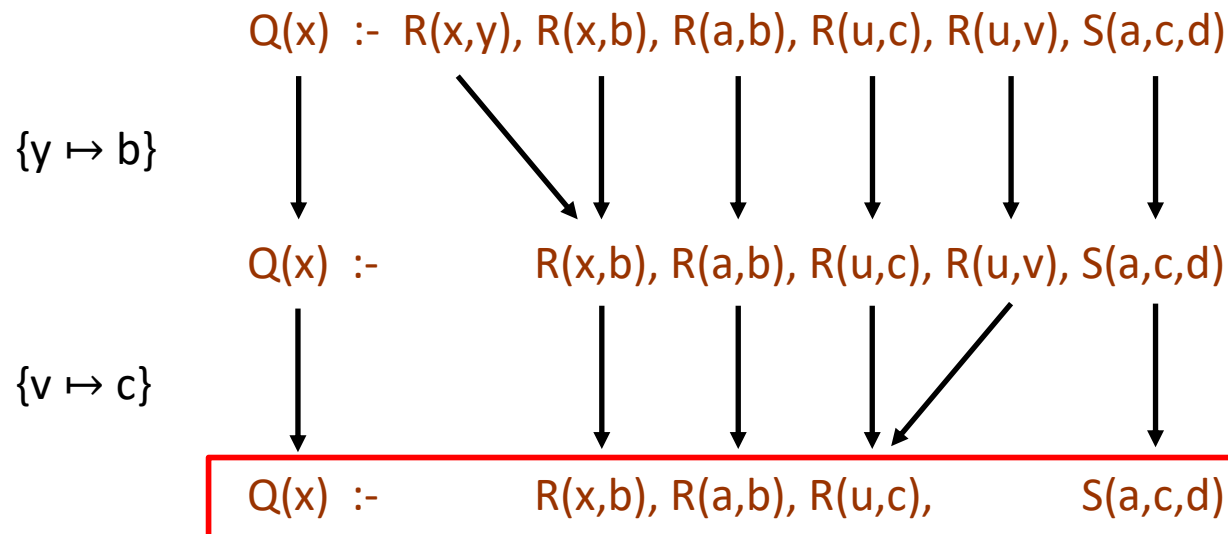
then $\text{body} := \text{body} \setminus \{\alpha\}$

Return $Q(x_1, \dots, x_k) :- \text{body}$

Note: if there is a query homomorphism from $Q(x_1, \dots, x_k) :- \text{body}$ to $Q(x_1, \dots, x_k) :- \text{body} \setminus \{\alpha\}$, then the two queries are equivalent since there is trivially a query homomorphism from the latter to the former query

Minimization Procedure: Example

(a,b,c,d are constants)



minimal query

Note: the mapping $x \mapsto a$ is not valid since x is a distinguished variable

Uniqueness of Minimal Queries

Natural question: does the order in which we remove atoms from the body of the input conjunctive query matter?

Theorem: Consider a conjunctive query Q . Let Q_1 and Q_2 be minimal conjunctive queries such that $Q_1 \equiv Q$ and $Q_2 \equiv Q$. Then, Q_1 and Q_2 are isomorphic (i.e., they are the same up to variable renaming)

Therefore, given a conjunctive query Q , the result of $\text{Minimization}(Q)$ is unique (up to variable renaming) and is called the **core** of Q

Recap

- **CQ-Evaluation** is NP-complete - **combined complexity**
- For a fixed CQ Q , Q -Evaluation is in PTIME - **data complexity**
- A CQ is always satisfiable \Rightarrow **CQ-Satisfiability** is trivial
- **CQ-Equivalence** and **CQ-Containment** are NP-complete
- Improve performance via minimization of CQs