Conjunctive Queries

So far

- The main languages for querying relational databases are:
  - Relational Algebra (**RA**)
  - Domain Relational Calcuclus (**DRC**)
  - Tuple Relational Calculus (**TRC**)

RA = DRC = TRC

(under the active domain semantics)

- Evaluation is decidable, and highly tractable in data complexity
  - **Foundations of the database industry**
  - The core of SQL is equally expressive to **RA**/**DRC**/**TRC**

- Satisfiability, equivalence and containment are undecidable
  - **Perfect query optimization is impossible**

A Crucial Question

Are there interesting sublanguages of **RA/DRC/TRC** for which perfect

query optimization is possible?

Conjunctive Queries

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

= relational calculus without $\neg$, $\forall$, $\vee$

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

Syntax of Conjunctive Queries (**CQ**)

$$Q(\mathbf{x}) \;:=\; \exists \mathbf{y}\, (R_1(\mathbf{v_1}) \;\wedge\; \cdots \;\wedge\; R_m(\mathbf{v_m}))$$

- $R_1,\ldots,R_m$ are relations
- $\mathbf{x}$, $\mathbf{y}$, $\mathbf{v_1},\ldots,\mathbf{v_m}$ 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

$$Q(\mathbf{x}) \;\text{:-}\; \underbrace{R_1(\mathbf{v_1}),\ldots,R_m(\mathbf{v_m})}$$

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

Conjunctive Queries: Example 1

List all the airlines

Flight	origin	destination	airline
	VIE	LHR	BA
	LHR	EDI	BA
	LGW	GLA	U2
	LCA	VIE	OS

Airport	code	city
	VIE	Vienna
	LHR	London
	LGW	London
	LCA	Larnaca
	GLA	Glasgow
	EDI	Edinburgh

{BA, U2, OS}

π_{airline} Flight

{z | ∃x∃y Flight(x,y,z)}

Q(z) :- Flight(x,y,z)

Conjunctive Queries: Example 2

List the codes of the airports in London

Flight	origin	destination	airline
	VIE	LHR	BA
	LHR	EDI	BA
	LGW	GLA	U2
	LCA	VIE	OS

Airport	code	city
	VIE	Vienna
	LHR	London
	LGW	London
	LCA	Larnaca
	GLA	Glasgow
	EDI	Edinburgh

⇩

{LHR, LGW}

π_{code} $(\sigma_{city=\text{'London'}}$ Airport)

$\{x \mid \exists y \text{ Airport}(x,y) \wedge y = \text{London}\}$

Q(x) :- Airport(x,y), y = London

Conjunctive Queries: Example 2

List the codes of the airports in London

Flight	origin	destination	airline
	VIE	LHR	BA
	LHR	EDI	BA
	LGW	GLA	U2
	LCA	VIE	OS

Airport	code	city
	VIE	Vienna
	LHR	London
	LGW	London
	LCA	Larnaca
	GLA	Glasgow
	EDI	Edinburgh

{LHR, LGW}

$\pi_{code}\ (\sigma_{city=\text{'London'}}\ \text{Airport})$

$\{x \mid \exists y\ \text{Airport}(x,y)\ \land\ y = \text{London}\}$

Q(x) :- Airport(x,London)

Conjunctive Queries: Example 3

List the airlines that fly directly from London to Glasgow

Flight	origin	destination	airline
	VIE	LHR	BA
	LHR	EDI	BA
	LGW	GLA	$\boxed{\text{U2}}$
	LCA	VIE	OS

Airport	code	city
	VIE	Vienna
	LHR	London
	LGW	London
	LCA	Larnaca
	GLA	Glasgow
	EDI	Edinburgh

$\Downarrow$

{U2}

$$\pi_{\text{airline}} \; ((\text{Flight} \bowtie_{\text{origin=code}} (\sigma_{\text{city='London'}} \text{ Airport})) \bowtie_{\text{destination=code}} (\sigma_{\text{city='Glasgow'}} \text{ Airport}))$$

$$\{z \mid \exists x \exists y \exists u \exists v \text{ Airport}(x,u) \wedge u = \text{London} \wedge \text{Airport}(x,u) \wedge u = \text{London}\} \wedge \text{Flight}(x,y,z)\}$$

Conjunctive Queries: Example 3

List the airlines that fly directly from London to Glasgow

Flight	origin	destination	airline
	VIE	LHR	BA
	LHR	EDI	BA
	LGW	GLA	U2
	LCA	VIE	OS

$\Downarrow$

{U2}

Airport	code	city
	VIE	Vienna
	LHR	London
	LGW	London
	LCA	Larnaca
	GLA	Glasgow
	EDI	Edinburgh

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

Homomorphism

- Semantics of conjunctive queries via the key notion of homomorphism

- A substitution from a set of symbols **S** to a set of symbols **T** is a function $h : \mathbf{S} \rightarrow \mathbf{T}$,

  i.e., h is a set of mappings of the form $s \mapsto t$, where $s \in \mathbf{S}$ and $t \in \mathbf{T}$

- A homomorphism from a set of atoms **A** to a set of atoms **B** is a substitution

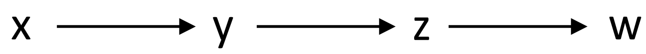
  $h : \text{terms}(\mathbf{A}) \rightarrow \text{terms}(\mathbf{B})$ such that:

  1. t is a constant $\Rightarrow$ $h(t) = t$
  2. $R(t_1,\ldots,t_k) \in \mathbf{A} \Rightarrow h(R(t_1,\ldots,t_k)) = R(h(t_1),\ldots,h(t_k)) \in \mathbf{B}$

(terms(**A**) = {t | t is a variable or constant that occurs in **A**})

Exercise: Find the Homomorphisms

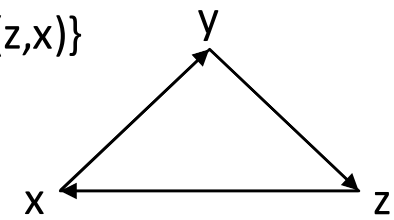
$\mathbf{S_1} = \{P(x,y), P(y,z), P(z,w)\}$



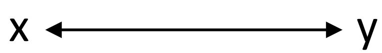
$\{x \mapsto x,\ y \mapsto y,\ z \mapsto z,\ w \mapsto x\}$

$\{x \mapsto x,\ y \mapsto y,\ z \mapsto x,\ w \mapsto y\}$

$\mathbf{S_2} = \{P(x,y), P(y,z), P(z,x)\}$



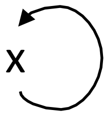
$\mathbf{S_3} = \{P(x,y), P(y,x)\}$



$\{x \mapsto y,\ y \mapsto x,\ z \mapsto y\}$

$\{x \mapsto x,\ y \mapsto y\}$

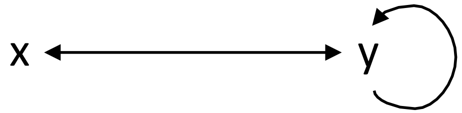
$\mathbf{S_4} = \{P(x,x)\}$



$\{x \mapsto y\}$

$\{x \mapsto y,\ y \mapsto x\}$

$\mathbf{S_5} = \{P(x,y), P(y,x), P(y,y)\}$



Semantics of Conjunctive Queries

- A match of a conjunctive query $Q(x_1,\ldots,x_k)$:- body in a database D is a homomorphism h such that h(body) $\subseteq$ D

- The answer to $Q(x_1,\ldots,x_k)$:- body over D is the set of k-tuples

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

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

Conjunctive Queries: Example

List the airlines that fly directly from London to Glasgow

Flight	origin	destination	airline
	VIE	LHR	BA
	LHR	EDI	BA
	LGW	GLA	U2
	LCA	VIE	OS

Airport	code	city
	VIE	Vienna
	LHR	London
	LGW	London
	LCA	Larnaca
	GLA	Glasgow
	EDI	Edinburgh

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

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

Complexity of **CQ**

Theorem: It holds that:

- BQE(**CQ**) is NP-complete (combined complexity)
- BQE[D](**CQ**) is NP-complete, for a fixed database D (query complexity)
- BQE[Q](**CQ**) is in LOGSPACE, for a fixed query Q ∈ **CQ** (data complexity)

Proof:

(NP-membership) Consider a database D, and a Boolean CQ Q :- body

Guess a substitution h : terms(body) → terms(D)

Verify that h is a match of Q in D, i.e., h(body) ⊆ D

(NP-hardness) Reduction from 3-colorability

(LOGSPACE-membership) Inherited from BQE[Q](**DRC**)

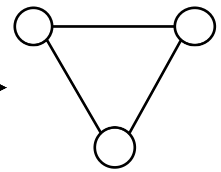
NP-hardness

(NP-hardness) Reduction from 3-colorability

3COL

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

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

Lemma: **G** is 3-colorable iff **G** can be mapped to $\mathbf{K_3}$, i.e., **G** $\xrightarrow{\text{hom}}$ 

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

the Boolean CQ that represents **G**

Complexity of **CQ**

Theorem: It holds that:

- BQE(**CQ**) is NP-complete (combined complexity)
- BQE[D](**CQ**) is NP-complete, for a fixed database D (query complexity)
- BQE[Q](**CQ**) is in LOGSPACE, for a fixed query Q ∈ **CQ** (data complexity)

Proof:

(NP-membership) Consider a database D, and a Boolean CQ Q :- body

Guess a substitution h : terms(body) → terms(D)

Verify that h is a match of Q in D, i.e., h(body) ⊆ D

(NP-hardness) Reduction from 3-colorability

(LOGSPACE-membership) Inherited from BQE[Q](**DRC**)

What About Optimization of CQs?

SAT(**CQ**)

Input: a query $Q \in$ **CQ**

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

EQUIV(**CQ**)

Input: two queries $Q_1 \in$ **CQ** and $Q_2 \in$ **CQ**

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

CONT(**CQ**)

Input: two queries $Q_1 \in$ **CQ** and $Q_2 \in$ **CQ**

Question: $Q_1 \subseteq Q_2$? or $Q_1(D) \subseteq Q_2(D)$ for every (finite) 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(**x**) :- body, D[Q] is obtained from body by replacing each variable x with a new constant $c(x) = \underline{x}$

- 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 : {variables in body} → {new constants} is a bijection, where c(body) = D[Q] and $c^{-1}(D[Q])$ = body

Satisfiability of CQs

SAT(**CQ**)

Input: a query Q ∈ **CQ**

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

Theorem: A query Q ∈ **CQ** is always satisfiable - SAT(**CQ**) ∈ O(1)-time

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

Equivalence and Containment of CQs

EQUIV(**CQ**)

Input: two queries $Q_1 \in$ **CQ** and $Q_2 \in$ **CQ**

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

CONT(**CQ**)

Input: two queries $Q_1 \in$ **CQ** and $Q_2 \in$ **CQ**

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

$Q_1 \equiv Q_2$ iff $Q_1 \subseteq Q_2$ and $Q_2 \subseteq Q_1$

$Q_1 \subseteq Q_2$ iff $Q_1 \equiv (Q_1 \wedge Q_2)$

…thus, we can safely focus on CONT(**CQ**)

Homomorphism Theorem

A query homomorphism from $Q_1(x_1,\ldots,x_k)$:- $body_1$ to $Q_2(y_1,\ldots,y_k)$:- $body_2$

is a substitution $h : terms(body_1) \rightarrow terms(body_2)$ such that:

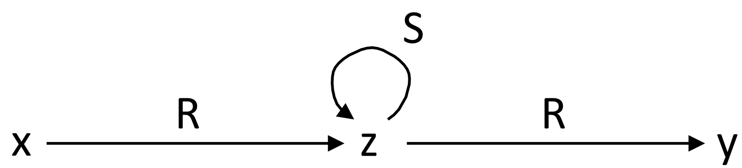
1. h is a homomorphism from $body_1$ to $body_2$
2. $(h(x_1),\ldots,h(x_k)) = (y_1,\ldots,y_k)$

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

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



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

We expect that $Q_1 \subseteq Q_2$. Why?

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 $\Rightarrow$ $Q_1 \subset Q_2$

Homomorphism Theorem: Proof

Assume that $Q_1(x_1,\ldots,x_k)$:- $body_1$ and $Q_2(y_1,\ldots,y_k)$:- $body_2$

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

- Clearly, $(c(x_1),\ldots,c(x_k)) \in Q_1(D[Q_1])$ - recall that $D[Q_1] = c(body_1)$
- Since $Q_1 \subseteq Q_2$, we conclude that $(c(x_1),\ldots,c(x_k)) \in Q_2(D[Q_1])$
- Therefore, there exists a homomorphism h such that $h(body_2) \subseteq D[Q_1] = c(body_1)$ and $h((y_1,\ldots,y_k)) = (c(x_1),\ldots,c(x_k))$
- By construction, $c^{-1}(c(body_1)) = body_1$ and $c^{-1}((c(x_1),\ldots,c(x_k))) = (x_1,\ldots,x_k)$
- Therefore, $c^{-1} \circ h$ is a query homomorphism from Q_2 to Q_1

$$
\begin{tikzcd}
Q_2(y_1,\ldots,y_k) \arrow[d] & :- & body_2 \arrow[d, "h"] \arrow[dd, bend left=60, "c^{-1} \circ h"] \\
Q_1(c(x_1),\ldots,c(x_k)) \arrow[d] & :- & c(body_1) \arrow[d, "c^{-1}"] \\
Q_1(x_1,\ldots,x_k) & :- & body_1
\end{tikzcd}
$$

Homomorphism Theorem: Proof

Assume that $Q_1(x_1,\ldots,x_k)$:- $body_1$ and $Q_2(y_1,\ldots,y_k)$:- $body_2$

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

- Consider a database D, and a tuple **t** such that **t** $\in Q_1(D)$
- We need to show that **t** $\in Q_2(D)$
- Clearly, there exists a homomorphism g such that $g(body_1) \subseteq D$ and $g((x_1,\ldots,x_k)) = \mathbf{t}$
- By hypothesis, there exists a query homomorphism h from Q_2 to Q_1
- Therefore, $g(h(body_2)) \subseteq D$ and

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

$$\begin{tikzcd}
Q_2(y_1,\ldots,y_k) \arrow[d] & \text{:-} & body_2 \arrow[d, "h"] \arrow[dd, bend left=60, "g \circ h"] \\
Q_1(x_1,\ldots,x_k) \arrow[d] & \text{:-} & body_1 \arrow[d, "g"] \\
\mathbf{t} & & D
\end{tikzcd}$$

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:

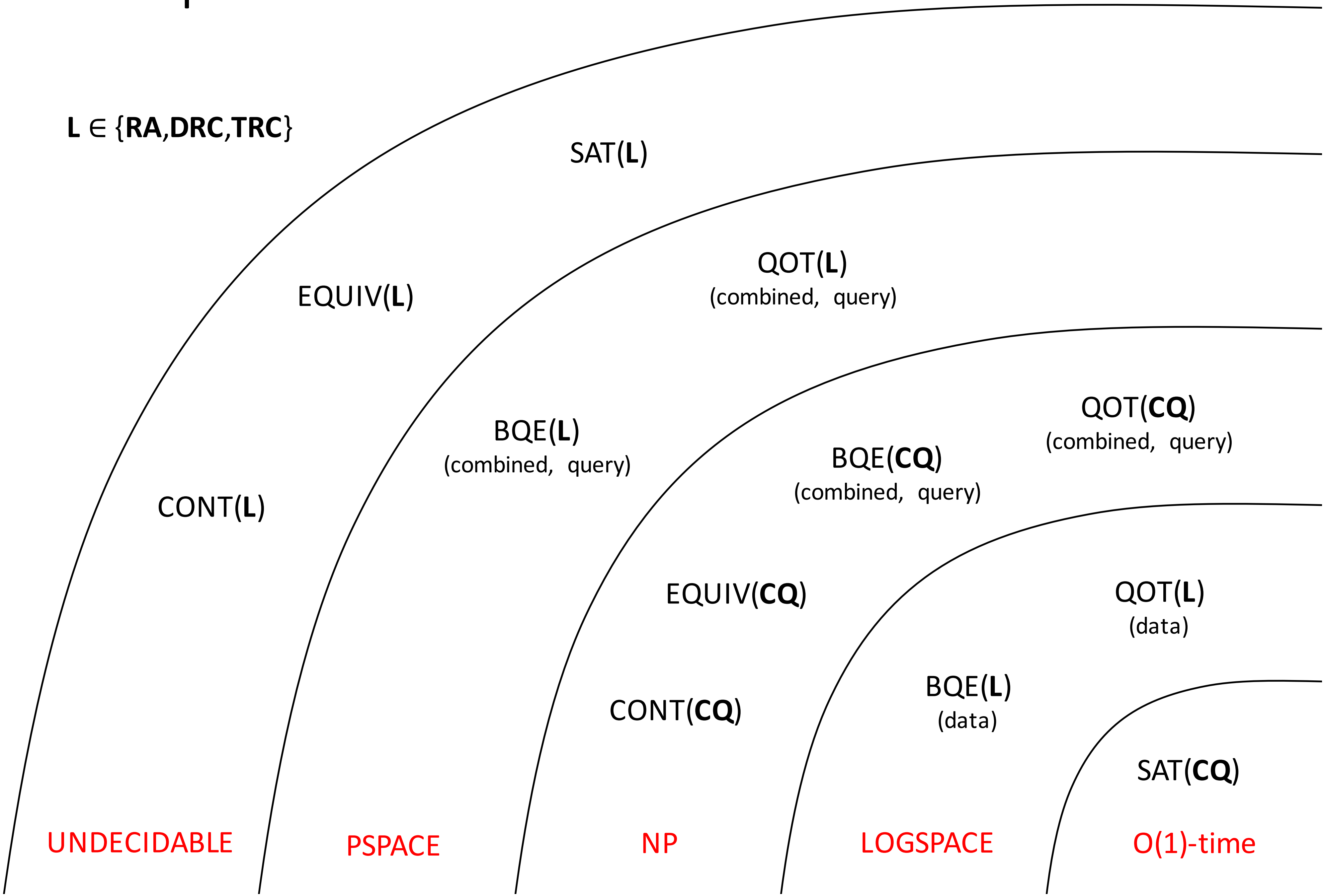
(NP-membership) Guess a substitution, and verify that is a query homomorphism

(NP-hardness) Straightforward reduction from BQE(**CQ**)

By applying the homomorphism theorem we get that:

Corollary: EQUIV(**CQ**) and CONT(**CQ**) are NP-complete

Recap



L ∈ {**RA**,**DRC**,**TRC**}
SAT(**L**)
EQUIV(**L**)
QOT(**L**)
(combined, query)
CONT(**L**)
BQE(**L**)
(combined, query)
QOT(**CQ**)
(combined, query)
BQE(**CQ**)
(combined, query)
EQUIV(**CQ**)
QOT(**L**)
(data)
CONT(**CQ**)
BQE(**L**)
(data)
SAT(**CQ**)
UNDECIDABLE
PSPACE
NP
LOGSPACE
O(1)-time

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 $Q_1(x_1,\ldots,x_k)$:- $body_1$.

If Q_1 is equivalent to a conjunctive query $Q_2(y_1,\ldots,y_k)$:- $body_2$ where $|body_2| < |body_1|$,

then Q_1 is equivalent to a query $Q_3(x_1,\ldots,x_k)$:- $body_3$ such that $body_3 \subseteq body_1$

The above theorem says that to minimize a conjunctive query $Q_1(\mathbf{x})$:- body we simply need to remove some atoms from body

Minimization Procedure

Minimization($Q(\mathbf{x})$:- body)

Repeat until no change

    choose an atom $\alpha \in$ body

    if there is a query homomorphism from $Q(\mathbf{x})$:- body to $Q(\mathbf{x})$:- body $\setminus \{\alpha\}$

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

Return $Q(\mathbf{x})$:- body

Note: if there is a query homomorphism from $Q(\mathbf{x})$:- body to $Q(\mathbf{x})$:- 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)

Q(x) :- R(x,y), R(x,b), R(a,b), R(u,c), R(u,v), S(a,c,d)

$\{y \mapsto b\}$

Q(x) :- R(x,b), R(a,b), R(u,c), R(u,v), S(a,c,d)

$\{v \mapsto c\}$

Q(x) :- R(x,b), R(a,b), R(u,c), S(a,c,d)

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?

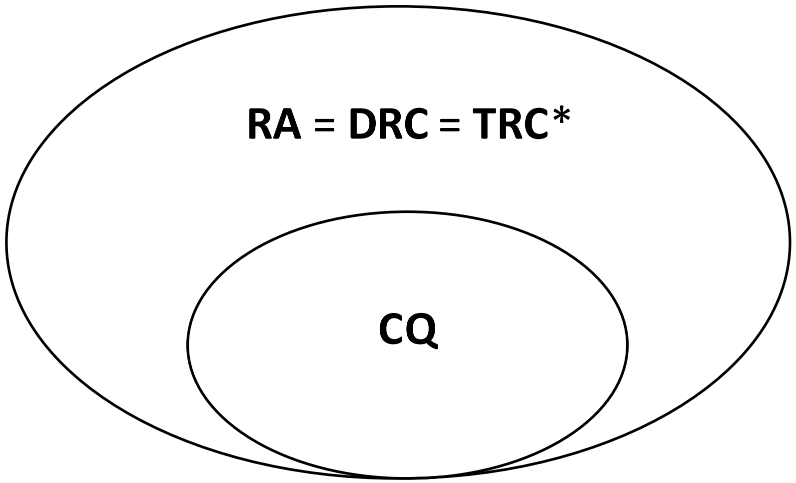
Theorem: Consider a conjunctive query Q. Let Q_1 and Q_2 be minimal conjunctive queries such that $Q_1 \equiv Q$ and $Q_2 \equiv Q$. Then, Q_1 and Q_2 are isomorphic (i.e., they are the same up to variable renaming)

Therefore, given a conjunctive query Q, the result of Minimization(Q) is unique (up to variable renaming) and is called the core of Q

Recap

- The main relational query languages - **RA/DRC/TRC**
  - Evaluation is decidable - foundations of the database industry
  - Perfect query optimization is impossible

- Conjunctive queries - an important query language
  - All the relevant algorithmic problems are decidable
  - Query minimization



*under the active domain semantics