CSE599d: Advanced Query Processing

Lecture 1: Introduction

Dan Suciu

University of Washington

Motivation

- Query Processing (QP), Query Optimization (QP): mature science.
- 544 is insufficient; Generals Reading Exam is insufficient
- No good comprehensive, up-to-date textbook or survey.
- This course fills the gap. This is material that everyone with a PhD in data management should know.

Organization

- Lectures Monday/Wednesday. Class discussions: please participate.
- Some lectures based on research papers, which you are welcome to read, but are not required. It's my job to distill those papers in the lectures.
- There will be a few light-weight homework assignments: submit on Canvas. Credit/parial-credit/no-credit.
  If you take this course for a grade, then HWs are mandatory.
- See `www.cs.washington.edu/cse599d`, subscribe to the calendar.

Course Topics (may still change)

- Provenance, Semirings, K-Relations.
- QO via Dynamic Programming: DPSize, DPSub, DPccp, IKKBZ.
- Extensible QO
  - Cascades; (guest lecture on SCOPE by Marc Friedman on Feb. 9)
  - Chase and GMap
- Yannakakis' algorithm, tree decomposition, modern approaches.
- The AGM Bound, Generic Join, Umbra's solution, FreeJoin.
- Entropic inequalities and the LpBound.

Prerequisites

544 or (at least) 344. You should know:

- SQL: select-from-where-groupby-having; subquereis.
- Relational algebra: $\sigma, \Pi, \bowtie, \cup, -$, outer joins.
- Basic identities:
$$\sigma(R \bowtie S) = \sigma(R) \bowtie S, (R \bowtie S) \bowtie T = R \bowtie (S \bowtie T), \text{ etc.}$$
- Basic cost/cardinality estimation: $\texttt{Est}(\sigma_{A=5}(R)) = \frac{|R|}{V(R,A)}$.
- System-R dynamic programming for join reordering.

Outline

- Today: Conjunctive Queries and Containment
- Wednesday: Provenance Semirings
- Next week we start Query Optimization

Query Equivalence

- Two queries Q, Q' are equivalent if they return the same answer on any input database:

$$\boxed{\forall D, Q(D) = Q'(D)}$$

- Query optimization is all about query equivalence.
- Checking equivalence of two SQL queries is undecidable.
- Checking equivalence of two conjunctive queries is decidable.

[Chandra and Merlin, 1977]

What are Conjunctive Queries?

Conjunctive Queries

A Conjunctive Query (CQ) is:

$$\boxed{Q(\boldsymbol{x}_0) = \exists \boldsymbol{y}\ (\bigwedge_i R_i(\boldsymbol{x}_i))}$$

Conjunctive Queries

A Conjunctive Query (CQ) is:

$$\boxed{Q(\boldsymbol{x}_0) = \exists \boldsymbol{y}\ (\bigwedge_i R_i(\boldsymbol{x}_i))}$$

- $Q(x,y) = \exists z(E(x,z) \wedge E(z,y))$ or just $Q(x,y) = E(x,z) \wedge E(z,y)$.
- Set semantics (for now!).
- FO formula restricted to $=, \wedge, \exists$; not allowed $\vee, \forall, \neg$.

Conjunctive Queries

A Conjunctive Query (CQ) is:

$$\boxed{Q(\boldsymbol{x}_0) = \exists \boldsymbol{y}\ (\bigwedge_i R_i(\boldsymbol{x}_i))}$$

- $Q(x,y) = \exists z(E(x,z) \wedge E(z,y))$ or just $Q(x,y) = E(x,z) \wedge E(z,y)$.
- Set semantics (for now!).
- FO formula restricted to $=, \wedge, \exists$; not allowed $\vee, \forall, \neg$.
- What fragment of RA?

Conjunctive Queries

A Conjunctive Query (CQ) is:

$$\boxed{Q(\boldsymbol{x}_0) = \exists \boldsymbol{y}\ (\bigwedge_i R_i(\boldsymbol{x}_i))}$$

- $Q(x,y) = \exists z(E(x,z) \wedge E(z,y))$ or just $Q(x,y) = E(x,z) \wedge E(z,y)$.
- Set semantics (for now!).
- FO formula restricted to $=, \wedge, \exists$; not allowed $\vee, \forall, \neg$.
- What fragment of RA? RA restricted to $\sigma, \Pi, \bowtie$.

Conjunctive Queries

A Conjunctive Query (CQ) is:

$$\boxed{Q(\boldsymbol{x}_0) = \exists \boldsymbol{y}\ (\bigwedge_i R_i(\boldsymbol{x}_i))}$$

- $Q(x,y) = \exists z(E(x,z) \wedge E(z,y))$ or just $Q(x,y) = E(x,z) \wedge E(z,y)$.
- Set semantics (for now!).
- FO formula restricted to $=, \wedge, \exists$; not allowed $\vee, \forall, \neg$.
- What fragment of RA? RA restricted to $\sigma, \Pi, \bowtie$.
- What fragment of SQL?

Conjunctive Queries

A Conjunctive Query (CQ) is:

$$\boxed{Q(\boldsymbol{x}_0) = \exists \boldsymbol{y}\ (\bigwedge_i R_i(\boldsymbol{x}_i))}$$

- $Q(x, y) = \exists z(E(x, z) \wedge E(z, y))$ or just $Q(x, y) = E(x, z) \wedge E(z, y)$.
- Set semantics (for now!).
- FO formula restricted to $=, \wedge, \exists$; not allowed $\vee, \forall, \neg$.
- What fragment of RA? RA restricted to $\sigma, \Pi, \bowtie$.
- What fragment of SQL? SELECT DISTINCT-FROM-WHERE...

Unions of Conjunctive Queries

A Union of Conjunctive Queries (UCQ) is:

$$\boxed{Q(\boldsymbol{x}) = \bigvee_i Q_i(\boldsymbol{x})}$$

where all Q_i's are CQs, and have the same sets of free variables.

Unions of Conjunctive Queries

A Union of Conjunctive Queries (UCQ) is:

$$\boxed{Q(\boldsymbol{x}) = \bigvee_i Q_i(\boldsymbol{x})}$$

where all Q_i's are CQs, and have the same sets of free variables.

- $Q(x, y) = E(x, y) \bigvee \exists z(E(x, z) \wedge E(z, y))$.
- FO formulas restricted to $=, \wedge, \exists, \vee$; not allowed $\forall, \neg$
- RA restricted to $\sigma, \Pi, \bowtie, \cup$.

Terminology

- Boolean query: no head vars: $Q() = \exists x \exists y \exists z (E(x,y) \wedge E(y,z))$.
- Full query: no existential vars: $Q(x,y,z) = E(x,y) \wedge E(y,z)$.
- We often omit the existential quantifiers, and write for example:

$$Q(x) = R(x,y) \wedge S(y,z) \wedge T(z,x).$$

Equivalent Concepts

Equivalent Concepts

- Boolean CQ: $Q() = R(x, y, z) \wedge S(x, u) \wedge S(y, v) \wedge S(z, w) \wedge R(u, v, w)$

Equivalent Concepts

- Boolean CQ: $Q() = R(x, y, z) \wedge S(x, u) \wedge S(y, v) \wedge S(z, w) \wedge R(u, v, w)$
- Database Instance:

$R =$

x	y	z
u	v	w

$S =$

x	u
y	v
z	w

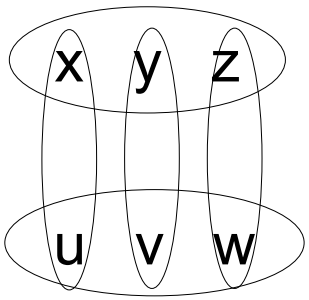
Equivalent Concepts

- Boolean CQ: $Q() = R(x, y, z) \wedge S(x, u) \wedge S(y, v) \wedge S(z, w) \wedge R(u, v, w)$
- Database Instance:

$$R = \begin{array}{|c|c|c|} \hline x & y & z \\ u & v & w \\ \hline \end{array} \qquad S = \begin{array}{|c|c|} \hline x & u \\ y & v \\ z & w \\ \hline \end{array}$$

- (Labeled) Hypergraph, $G = (V, E)$:

$$V = \{x, y, z, u, v, w\},$$
$$E = \{\{x, y, z\}, \{u, v, w\}, \{x, u\}, \{y, v\}, \{z, w\}\}$$



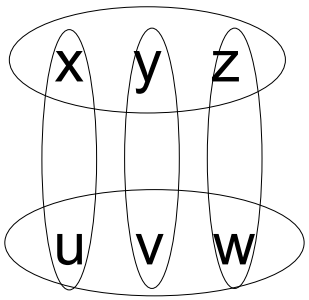
Equivalent Concepts

- Boolean CQ: $Q() = R(x,y,z) \wedge S(x,u) \wedge S(y,v) \wedge S(z,w) \wedge R(u,v,w)$
- Database Instance:

$$R = \begin{array}{|c|c|c|} \hline x & y & z \\ \hline u & v & w \\ \hline \end{array} \qquad S = \begin{array}{|c|c|} \hline x & u \\ \hline y & v \\ \hline z & w \\ \hline \end{array}$$

- (Labeled) Hypergraph, $G = (V, E)$:

$$V = \{x, y, z, u, v, w\},$$
$$E = \{\{x, y, z\}, \{u, v, w\}, \{x, u\}, \{y, v\}, \{z, w\}\}$$



We will often switch back-and-forth between these equivalent notions

Discussion

- We discuss mostly Boolean queries. For non-Boolean queries, remember to treat head variables as distinguished.
- Set semantics!
- May use constants too, e.t. $Q(x) = E(x, y) \wedge E(y, 'a')$.
- A CQ is a hypergraph. Thus, we can use concepts from graph theory:

Homomorphisms

Homomorphisms

Review: Graph Homomoprhisms

A graph homomorphism $h : (V, E) \rightarrow (V', E')$ is ??????

Review: Graph Homomoprhisms

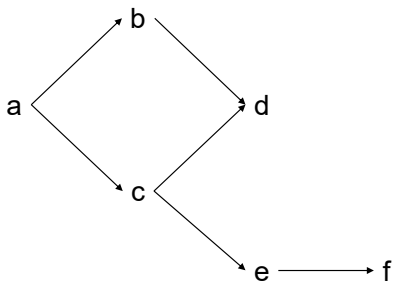
A graph homomorphism $h : (V, E) \rightarrow (V', E')$ is a function $h : V \rightarrow V'$ that maps every edge in the first graph to an edge in the second graph:

$$\boxed{\forall x, y \in V \text{ if } (x, y) \in E \text{ then } (h(x), h(y)) \in E'}$$

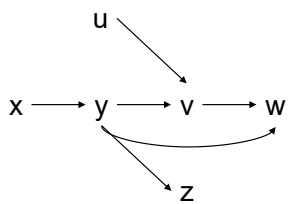
Review: Graph Homomoprhisms

A graph homomorphism $h : (V, E) \rightarrow (V', E')$ is a function $h : V \rightarrow V'$ that maps every edge in the first graph to an edge in the second graph:

$$\forall x, y \in V \text{ if } (x, y) \in E \text{ then } (h(x), h(y)) \in E'$$



G

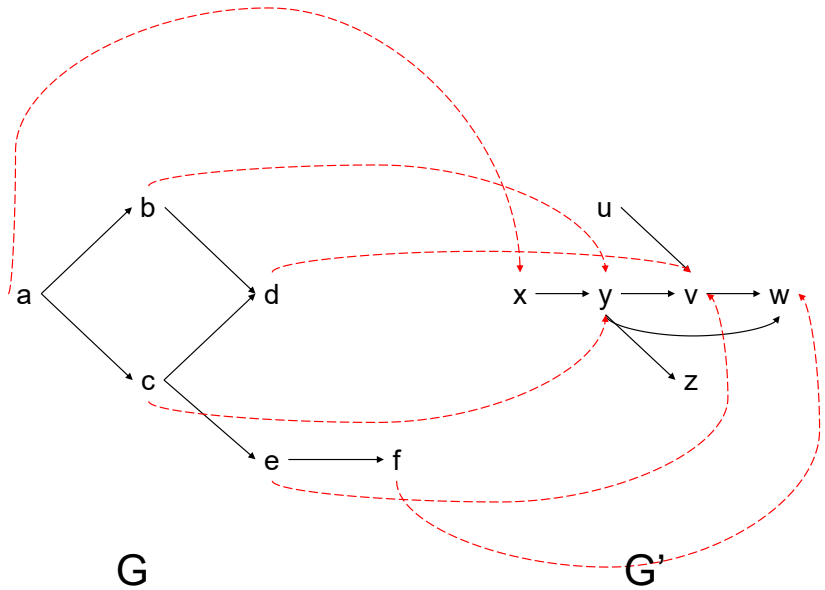


G'

Review: Graph Homomoprhisms

A graph homomorphism $h : (V, E) \rightarrow (V', E')$ is a function $h : V \rightarrow V'$ that maps every edge in the first graph to an edge in the second graph:

$$\forall x, y \in V \text{ if } (x, y) \in E \text{ then } (h(x), h(y)) \in E'$$



Homomorphisms of Boolean Queries

$$Q() = R_1(\boldsymbol{x}_1) \wedge \cdots \wedge R_m(\boldsymbol{x}_m), \qquad Q'() = S_1(\boldsymbol{y}_1) \wedge \cdots \wedge S_n(\boldsymbol{y}_n).$$

A homomorphism $h : Q' \to Q$ is a function $h : \texttt{Vars}(Q') \to \texttt{Vars}(Q)$ that maps each atom in Q' to an atom in Q

$$\boxed{\forall j = 1, n, \exists i = 1, m\text{: symbol } R_i \text{ is the same as } S_j, \text{ and } h(\boldsymbol{y}_j) = \boldsymbol{x}_i}$$

Homomorphisms of Boolean Queries

$$Q() = R_1(\boldsymbol{x}_1) \wedge \cdots \wedge R_m(\boldsymbol{x}_m), \qquad Q'() = S_1(\boldsymbol{y}_1) \wedge \cdots \wedge S_n(\boldsymbol{y}_n).$$

A homomorphism $h : Q' \to Q$ is a function $h : \texttt{Vars}(Q') \to \texttt{Vars}(Q)$ that maps each atom in Q' to an atom in Q

$$\boxed{\forall j = 1, n, \exists i = 1, m\text{: symbol } R_i \text{ is the same as } S_j, \text{ and } h(\boldsymbol{y}_j) = \boldsymbol{x}_i}$$

How would you modify the definition to account for:

- Constants in the query, e.g. $Q() = E(x, y) \wedge E(y, 5) \wedge E(y, x)$
- Non-Boolean queries, e.g. $Q(x) = E(x, y) \wedge E(y, z)$.

(Discussion in class only)

CQ Evaluation and Homomorphisms

Computing $Q(\boldsymbol{D})$:
(1) find all $h : Q \to \boldsymbol{D}$, (2) return the set $\{h(\mathtt{Head}(Q)) \mid h : Q \to \boldsymbol{D}\}$.

CQ Evaluation and Homomorphisms

Computing $Q(\boldsymbol{D})$:
(1) find all $h : Q \to \boldsymbol{D}$, (2) return the set $\{h(\mathtt{Head}(Q)) \mid h : Q \to \boldsymbol{D}\}$.

$$Q(x) = R(x) \wedge S(x, y) \wedge T(y, 'a')$$

$R =$

x
1
2

$S =$

x	y
1	10
1	20
2	20

$T =$

y	z
10	$'a'$
10	b
20	$'a'$

CQ Evaluation and Homomorphisms

Computing $Q(\boldsymbol{D})$:
(1) find all $h : Q \to \boldsymbol{D}$, (2) return the set $\{h(\mathtt{Head}(Q)) \mid h : Q \to \boldsymbol{D}\}$.

$$Q(x) = R(x) \wedge S(x,y) \wedge T(y,'a')$$

$R =$

x
1
2

$S =$

x	y
1	10
1	20
2	20

$T =$

y	z
10	$'a'$
10	b
20	$'a'$

We list all homomorphisms $\{x, y, 'a'\} \to \{1, 2, 10, 20, 'a', b\}$:

$h =$

x	y	$'a'$
1	10	$'a'$
1	20	$'a'$
2	20	$'a'$

CQ Evaluation and Homomorphisms

Computing $Q(\boldsymbol{D})$:
(1) find all $h : Q \to \boldsymbol{D}$, (2) return the set $\{h(\mathtt{Head}(Q)) \mid h : Q \to \boldsymbol{D}\}$.

$$Q(x) = R(x) \wedge S(x,y) \wedge T(y,'a')$$

$R =$

x
1
2

$S =$

x	y
1	10
1	20
2	20

$T =$

y	z
10	$'a'$
10	b
20	$'a'$

We list all homomorphisms $\{x, y, 'a'\} \to \{1, 2, 10, 20, 'a', b\}$:

$h =$

x	y	$'a'$
1	10	$'a'$
1	20	$'a'$
2	20	$'a'$

Final answer after duplicate elimination: $Q(\boldsymbol{D}) = \{1, 2\}$.

Discussion

- If Q is Boolean, then $Q(\boldsymbol{D}) \equiv \exists h : Q \rightarrow \boldsymbol{D}$.
- If Q is a Full CQ, then $Q(\boldsymbol{D}) = \{h \mid h : Q \rightarrow \boldsymbol{D}\}$.
- Same as the nested-loop semantics that we use in 344/544 (why?)
- Homomorphism is used to check query containment and equivalence. Next.

Query Containment and Equivalence

[Chandra and Merlin, 1977]

Query Equivalence and Query Containment

Definition (Equivalence)

Q_1, Q_2 are equivalent if $\forall \boldsymbol{D}, Q_1(\boldsymbol{D}) = Q_2(\boldsymbol{D})$. Notation: $Q_1 \equiv Q_2$.

Query Equivalence and Query Containment

Definition (Equivalence)

Q_1, Q_2 are equivalent if $\forall \boldsymbol{D}, Q_1(\boldsymbol{D}) = Q_2(\boldsymbol{D})$. Notation: $Q_1 \equiv Q_2$.

Definition (Containment)

Q_1 is contained in Q_2 if $\forall \boldsymbol{D}, Q_1(\boldsymbol{D}) \subseteq Q_2(\boldsymbol{D})$. Notation: $Q_1 \subseteq Q_2$.

Query Equivalence and Query Containment

Definition (Equivalence)

Q_1, Q_2 are equivalent if $\forall \boldsymbol{D}, Q_1(\boldsymbol{D}) = Q_2(\boldsymbol{D})$. Notation: $Q_1 \equiv Q_2$.

Definition (Containment)

Q_1 is contained in Q_2 if $\forall \boldsymbol{D}, Q_1(\boldsymbol{D}) \subseteq Q_2(\boldsymbol{D})$. Notation: $Q_1 \subseteq Q_2$.

Fact

Equivalence and containment are (almost) the same problem.

Query Equivalence and Query Containment

Definition (Equivalence)

Q_1, Q_2 are equivalent if $\forall \boldsymbol{D}, Q_1(\boldsymbol{D}) = Q_2(\boldsymbol{D})$. Notation: $Q_1 \equiv Q_2$.

Definition (Containment)

Q_1 is contained in Q_2 if $\forall \boldsymbol{D}, Q_1(\boldsymbol{D}) \subseteq Q_2(\boldsymbol{D})$. Notation: $Q_1 \subseteq Q_2$.

Fact

Equivalence and containment are (almost) the same problem.

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

Query Equivalence and Query Containment

Definition (Equivalence)

Q_1, Q_2 are equivalent if $\forall \boldsymbol{D}, Q_1(\boldsymbol{D}) = Q_2(\boldsymbol{D})$. Notation: $Q_1 \equiv Q_2$.

Definition (Containment)

Q_1 is contained in Q_2 if $\forall \boldsymbol{D}, Q_1(\boldsymbol{D}) \subseteq Q_2(\boldsymbol{D})$. Notation: $Q_1 \subseteq Q_2$.

Fact

Equivalence and containment are (almost) the same problem.

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

$$\boxed{Q_1 \subseteq Q_2} \text{ iff } \boxed{Q_1 \equiv Q_1 \wedge Q_2}$$

Containment for CQs

The canonical database associated to a CQ Q is the following: its domain is $\texttt{Vars}(Q)$, and its tuples are the atoms of Q. Notation: $\boldsymbol{D}_Q$.

Theorem Let Q_1, Q_2 be Boolean CQs. The following are equivalent:

- Containment holds: $Q_1 \subseteq Q_2$
- There exists a homomorphism $h : Q_2 \to Q_1$
- $Q_2(\boldsymbol{D}_{Q_1}) = \text{true}$.

Containment for CQs

The canonical database associated to a CQ Q is the following: its domain is $\texttt{Vars}(Q)$, and its tuples are the atoms of Q. Notation: $\boldsymbol{D}_Q$.

Theorem Let Q_1, Q_2 be Boolean CQs. The following are equivalent:

- Containment holds: $Q_1 \subseteq Q_2$
- There exists a homomorphism $h : Q_2 \rightarrow Q_1$
- $Q_2(\boldsymbol{D}_{Q_1}) = \text{true}$.

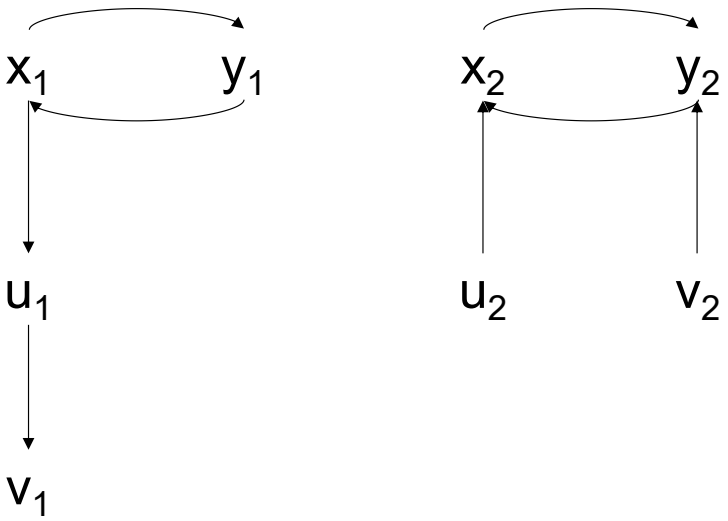
Extend the statement to the case when Q_1, Q_2 are not Boolean

An Example

$$Q_1() = E(x_1, y_1) \wedge E(y_1, x_1) \wedge E(x_1, u_1) \wedge E(u_1, v_1)$$
$$Q_2() = E(x_2, y_2) \wedge E(y_2, x_2) \wedge E(u_2, x_2) \wedge E(v_2, y_2)$$

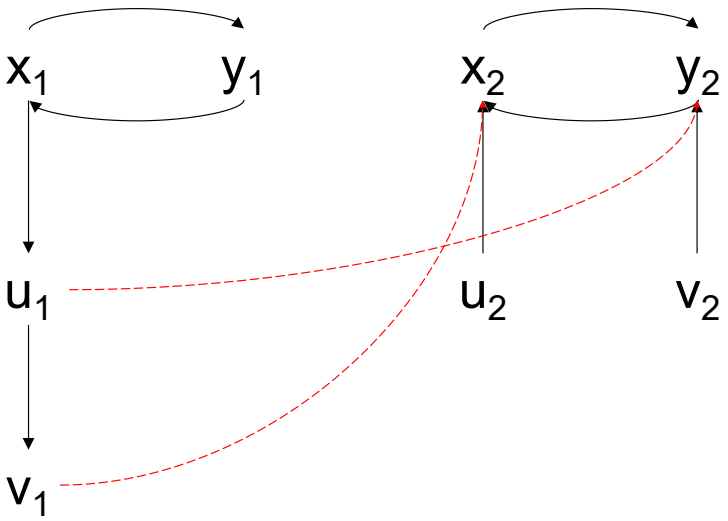
An Example

$$Q_1() = E(x_1, y_1) \wedge E(y_1, x_1) \wedge E(x_1, u_1) \wedge E(u_1, v_1)$$
$$Q_2() = E(x_2, y_2) \wedge E(y_2, x_2) \wedge E(u_2, x_2) \wedge E(v_2, y_2)$$



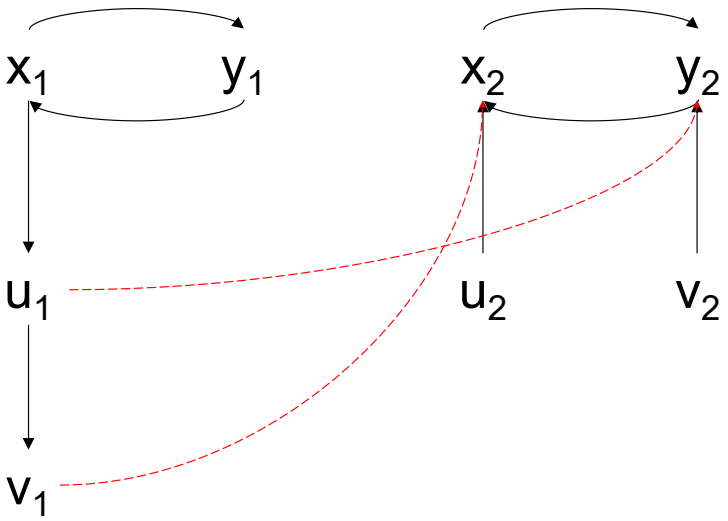
An Example

$$Q_1() = E(x_1, y_1) \wedge E(y_1, x_1) \wedge E(x_1, u_1) \wedge E(u_1, v_1)$$
$$Q_2() = E(x_2, y_2) \wedge E(y_2, x_2) \wedge E(u_2, x_2) \wedge E(v_2, y_2)$$



An Example

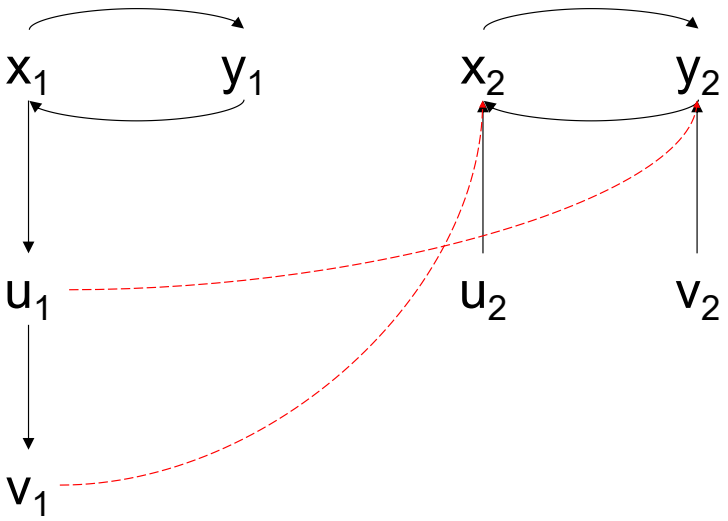
$$Q_1() = E(x_1, y_1) \wedge E(y_1, x_1) \wedge E(x_1, u_1) \wedge E(u_1, v_1)$$
$$Q_2() = E(x_2, y_2) \wedge E(y_2, x_2) \wedge E(u_2, x_2) \wedge E(v_2, y_2)$$



$$Q_2 \subseteq Q_1$$

An Example

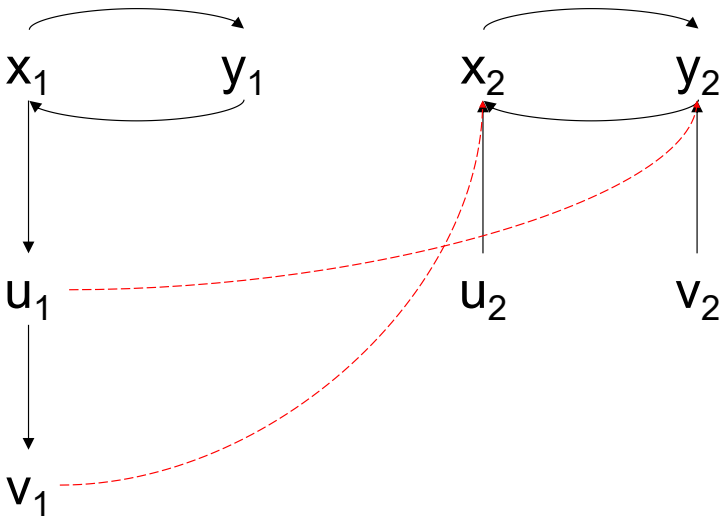
$$Q_1() = E(x_1, y_1) \wedge E(y_1, x_1) \wedge E(x_1, u_1) \wedge E(u_1, v_1)$$
$$Q_2() = E(x_2, y_2) \wedge E(y_2, x_2) \wedge E(u_2, x_2) \wedge E(v_2, y_2)$$



$$Q_2 \subseteq Q_1$$
$$Q_1 \subseteq Q_2$$

An Example

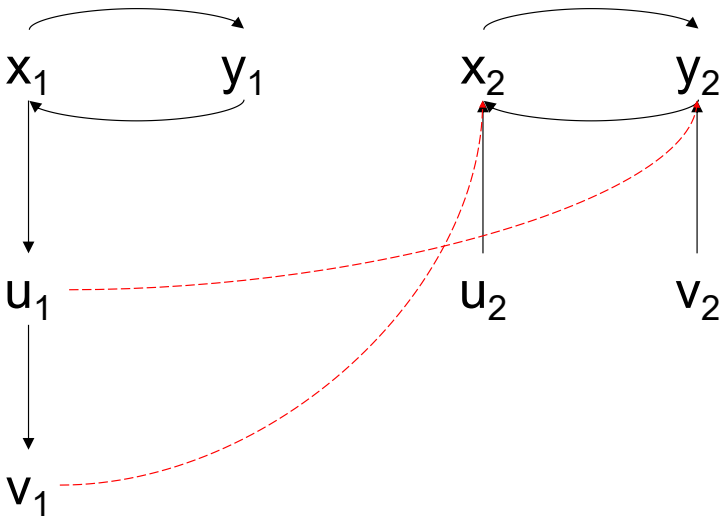
$$Q_1() = E(x_1, y_1) \wedge E(y_1, x_1) \wedge E(x_1, u_1) \wedge E(u_1, v_1)$$
$$Q_2() = E(x_2, y_2) \wedge E(y_2, x_2) \wedge E(u_2, x_2) \wedge E(v_2, y_2)$$



$Q_2 \subseteq Q_1$
$Q_1 \subseteq Q_2$
$Q_1 \equiv Q_2$

An Example

$$Q_1() = E(x_1, y_1) \wedge E(y_1, x_1) \wedge E(x_1, u_1) \wedge E(u_1, v_1)$$
$$Q_2() = E(x_2, y_2) \wedge E(y_2, x_2) \wedge E(u_2, x_2) \wedge E(v_2, y_2)$$



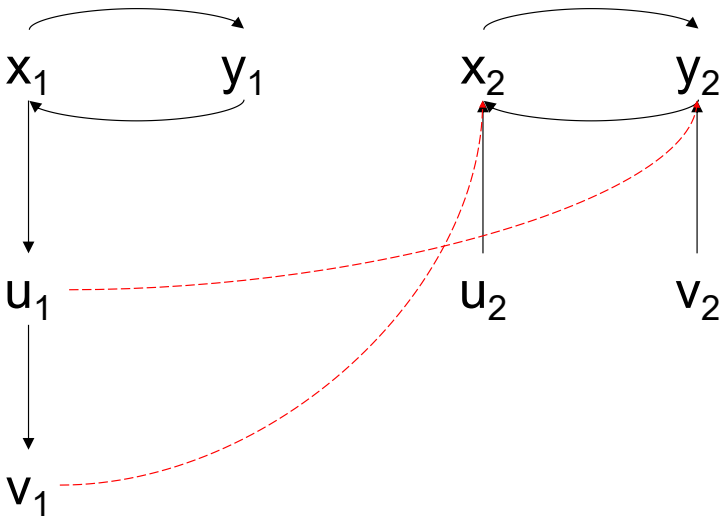
$Q_2 \subseteq Q_1$
$Q_1 \subseteq Q_2$
$Q_1 \equiv Q_2$

What changes if the queries are not Boolean?

$$Q_1(u_1) = E(x_1, y_1) \wedge E(y_1, x_1) \wedge E(x_1, u_1) \wedge E(u_1, v_1)$$
$$Q_2(u_2) = E(x_2, y_2) \wedge E(y_2, x_2) \wedge E(u_2, x_2) \wedge E(v_2, y_2)$$

An Example

$$Q_1() = E(x_1, y_1) \wedge E(y_1, x_1) \wedge E(x_1, u_1) \wedge E(u_1, v_1)$$
$$Q_2() = E(x_2, y_2) \wedge E(y_2, x_2) \wedge E(u_2, x_2) \wedge E(v_2, y_2)$$



$Q_2 \subseteq Q_1$
$Q_1 \subseteq Q_2$
$Q_1 \equiv Q_2$

What changes if the queries are not Boolean?

$$Q_1(u_1) = E(x_1, y_1) \wedge E(y_1, x_1) \wedge E(x_1, u_1) \wedge E(u_1, v_1)$$
$$Q_2(u_2) = E(x_2, y_2) \wedge E(y_2, x_2) \wedge E(u_2, x_2) \wedge E(v_2, y_2)$$

$Q_1 \not\subseteq Q_2$ and $Q_2 \not\subseteq Q_1$

Containment of UCQs

Let $Q = Q_1 \vee Q_2 \vee \cdots$, $Q' = Q'_1 \vee Q'_2 \vee \cdots$ The following are equivalent:

- Containment holds: $Q \subseteq Q'$
- Every Q_i is contained in some Q_j: $\forall i \exists j, Q_i \subseteq Q'_j$.

Proof in class.

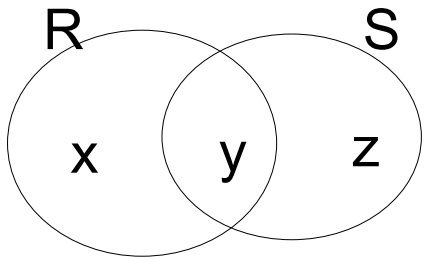
Join/Semi-join Identities: Idempotence

$$R \bowtie S = (R \ltimes S) \bowtie S$$

Join/Semi-join Identities: Idempotence

$$R \bowtie S = (R \ltimes S) \bowtie S$$

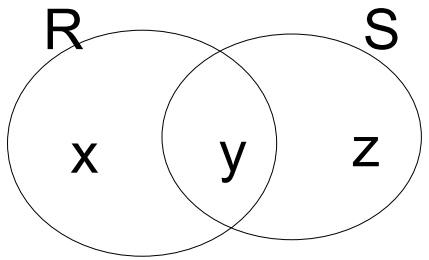
Denote $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}$ the set of variables:



Join/Semi-join Identities: Idempotence

$$R \bowtie S = (R \ltimes S) \bowtie S$$

Denote $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}$ the set of variables:

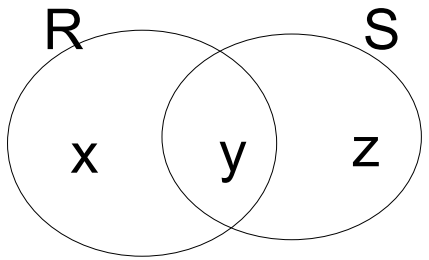


$$Q_1(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}) = R(\boldsymbol{x}, \boldsymbol{y}) \wedge S(\boldsymbol{y}, \boldsymbol{z})$$

Join/Semi-join Identities: Idempotence

$$R \bowtie S = (R \ltimes S) \bowtie S$$

Denote $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}$ the set of variables:



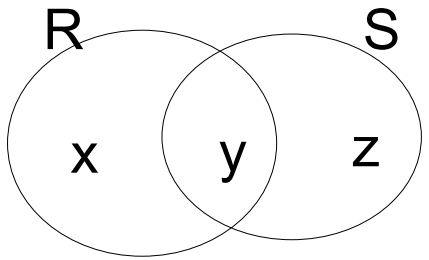
$$Q_1(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}) = R(\boldsymbol{x}, \boldsymbol{y}) \wedge S(\boldsymbol{y}, \boldsymbol{z})$$

$$Q_2(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}) = (\exists \boldsymbol{z}\, (R(\boldsymbol{x}, \boldsymbol{y}) \wedge S(\boldsymbol{y}, \boldsymbol{z}))) \wedge S(\boldsymbol{y}, \boldsymbol{z})$$

Join/Semi-join Identities: Idempotence

$$R \bowtie S = (R \ltimes S) \bowtie S$$

Denote $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}$ the set of variables:



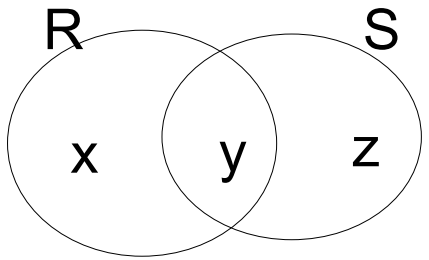
$$\begin{aligned} Q_1(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}) &= R(\boldsymbol{x}, \boldsymbol{y}) \wedge S(\boldsymbol{y}, \boldsymbol{z}) \\ Q_2(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}) &= (\exists \boldsymbol{z}\, (R(\boldsymbol{x}, \boldsymbol{y}) \wedge S(\boldsymbol{y}, \boldsymbol{z}))) \wedge S(\boldsymbol{y}, \boldsymbol{z}) \\ &= \exists \boldsymbol{u} R(\boldsymbol{x}, \boldsymbol{y}) \wedge S(\boldsymbol{y}, \boldsymbol{u}) \wedge S(\boldsymbol{y}, \boldsymbol{z}) \end{aligned}$$

We renamed $\exists z$ to $\exists u$ so it doesn't clash with the head variable $\boldsymbol{z}$.

Join/Semi-join Identities: Idempotence

$$R \bowtie S = (R \ltimes S) \bowtie S$$

Denote x, y, z the set of variables:



$$\begin{aligned} Q_1(x,y,z) &= R(x,y) \wedge S(y,z) \\ Q_2(x,y,z) &= (\exists z\,(R(x,y) \wedge S(y,z))) \wedge S(y,z) \\ &= \exists u R(x,y) \wedge S(y,u) \wedge S(y,z) \end{aligned}$$

We renamed $\exists z$ to $\exists u$ so it doesn't clash with the head variable z.

$h_1 : Q_1 \to Q_2$ maps $(x,y,z) \mapsto (x,y,z)$.

$h_2 : Q_2 \to Q_1$ maps $(x,u,y,z) \mapsto (x,z,y,z)$.

Therefore, $Q_1 \equiv Q_2$.

Discussion

Checking query containment and equivalence is not only decidable, but the same as query evaluation!

In principle, any query optimization technique also applies to checking equivalence too.

Limitations:

- Restricted to CQs. Some extensions exists (add $x < y$, or add one level of negation), but require more complex techniques.
- Set semantics only. Good for reasoning about semijoins, but not for general RA expressions.

Query Minimization

Motivation

Query Minimization:

Given Q, find Q_{min} such that $Q \equiv Q_{\text{min}}$ and Q_{min} has the smallest number of atoms.

We will show:

- Q_{min} is unique up to isomorphism.
- Q_{min} can be obtained from Q by repeatedly dropping atoms, as long as $Q \rightarrow Q_{\text{min}}$.
- In graph theory, Q_{min} is called the core of Q.

An Example

$$Q(x) = E(x, y) \wedge E(y, z) \wedge E(x, w)$$

An Example

$$Q(x) = E(x, y) \wedge E(y, z) \wedge E(x, w)$$

$$Q_{\mathsf{min}}(x) = E(x, y) \wedge E(y, z)$$

A Simple Fact

Let Q' be obtained from Q by dropping some atoms.

Does any containment hold between Q and Q'?

A Simple Fact

Let Q' be obtained from Q by dropping some atoms.

Does any containment hold between Q and Q'?

$Q \subseteq Q'$ because of the inclusion homomorphism $Q' \rightarrow Q$

Uniqueness

Fact Q_{min} is unique up to isomorphism.

Proof: Let Q', Q'' be minimal and equivalent to Q.

Since $Q' \equiv Q''$, there exists $h : Q' \rightarrow Q''$, and $h' : Q'' \rightarrow Q'$.

$h' \circ h : Q' \rightarrow Q'$ is surjective, otherwise $Q \equiv \text{Im}(h' \circ h)$ violating minimality.

Thus, $h' \circ h$ is an isomorphism (since its domain is finite).

Finding Q_{min}

Start with the query $Q() = R_1(\boldsymbol{x}_1) \wedge R_2(\boldsymbol{x}_2) \wedge \cdots$

- Find an atom $R_i(\boldsymbol{x}_i)$ such that, if Q' is the query obtained by removing the atom, then there exists homomorphism $Q \rightarrow Q'$.
- Remove the atom, replace Q with Q' and repeat
- Stop when no more atom can be removed.

$Q \rightarrow Q' \rightarrow Q'' \rightarrow \cdots \rightarrow Q_{\text{min}}$

Minimizing UCQ

A UCQ query $\boxed{Q = Q_1 \vee Q_2 \vee \cdots}$ is minimal if:

- each CQ Q_i is minimal
- for all i, j, $Q_i \subseteq Q_j$ implies $i = j$.

Query minimization:

- Minimize each Q_i for $i = 1, 2, \ldots$
- Remove Q_i whenever $\exists j \neq i$ s.t. $Q_i \subseteq Q_j$.

Summary

- Query containment/minimization: classical result in database theory.
- In practice? Not so much. Real queries have bag semantics query minimization does not apply: $Q_1(x) = R(x) \wedge R(x)$ is not equivalent to $Q_2(x) = R(x)$.
- However the theory becomes quite relevant for reasoning about semi-joins and query rewriting using views, which is a major topic for database systems.

Chandra, A. K. and Merlin, P. M. (1977).
Optimal implementation of conjunctive queries in relational data bases.
In Hopcroft, J. E., Friedman, E. P., and Harrison, M. A., editors, *Proceedings of the 9th Annual ACM Symposium on Theory of Computing, May 4-6, 1977, Boulder, Colorado, USA*, pages 77–90. ACM.