Logic and Databases

Phokion G. Kolaitis

UC Santa Cruz & IBM Research - Almaden

Lecture 2

Three Fundamental Problems in Databases

Let L be a database query language.

- The Query Evaluation Problem:
  Given a query q in L and a database D, evaluate q(D)

- The Query Containment Problem:
  Given queries q and q' in L, is q $\subseteq$ q'?

- The Query Equivalence Problem:
  Given queries q and q' in L, is q $\equiv$ q'?

Vardi's Taxonomy of the Query Evaluation Problem

M.Y Vardi, "The Complexity of Relational Query Languages", 1982

- Definition: Let L be a database query language.
  - The combined complexity of L is the decision problem: given an L-sentence and a database instance D, is φ true on D? (does D satisfy φ?) (in symbols, does $D \models \varphi$?)
  - The data complexity of L is the family of the following decision problems P_φ, where φ is an L-sentence: given a database instance D, does $D \models \varphi$?
  - The query complexity of L is the family of the following decision problems P_D, where D is a database instance: given an L-sentence φ, does $D \models \varphi$?

Summary

- Relational Algebra and Relational Calculus have “essentially” the same expressive power.
- The Query Equivalence Problem for Relational Calculus is undecidable.
- The Query Containment Problem for Relational Calculus is undecidable.
- The Query Evaluation Problem for Relational Calculus:
  - Data Complexity is in LOGSPACE
  - Combined Complexity is PSPACE-complete
  - Query Complexity is PSPACE-complete.

Sublanguages of Relational Calculus

- Question: Are there interesting sublanguages of relational calculus for which the Query Containment Problem and the Query Evaluation Problem are “easier” than the full relational calculus?

- Answer:
  - Yes, the language of conjunctive queries is such a sublanguage.
  - Moreover, conjunctive queries are the most frequently asked queries against relational databases.

Conjunctive Queries

- Definition: A conjunctive query is a query expressible by a relational calculus formula in prenex normal form built from atomic formulas $R(y_1,\ldots,y_n)$, and $\wedge$ and $\exists$ only.

$$\{ (x_1,\ldots,x_k): \exists z_1 \ldots \exists z_m \chi(x_1, \ldots,x_k, z_1,\ldots,z_k) \},$$

  where $\chi(x_1, \ldots,x_k, z_1,\ldots,z_k)$ is a conjunction of atomic formulas of the form $R(y_1,\ldots,y_m)$.

  - Equivalently, a conjunctive query is a query expressible by a relational algebra expression of the form

    $\pi_X(\sigma_\Theta(R_1\times \ldots\times R_n))$, where

    Θ is a conjunction of equality atomic formulas (equijoin).
  - Equivalently, a conjunctive query is a query expressible by an SQL expression of the form

    SELECT <list of attributes>
    FROM   <list of relation names>
    WHERE <conjunction of equalities>

Conjunctive Queries

- Every natural join is a conjunctive query with no existentially quantified variables
  P(A,B,C), R(B,C,D) two relation symbols
  - $P \bowtie R = \{(x,y,z,w): P(x,y,z) \wedge R(y,z,w)\}$
  - q(x,y,z,w) :-- P(x,y,z), R(y,z,w)
    (no variables are existentially quantified)
  -
    ```
    SELECT P.A, P.B, P.C, R.D
    FROM   P, R
    WHERE P.B = R.B  AND  P.C = R.C
    ```

Conjunctive Query Evaluation and Containment

- Definition: Two fundamental problems about CQs
  - Conjunctive Query Evaluation (CQE):
    Given a conjunctive query q and a database D, find q(D).

  - Conjunctive Query Containment (CQC):
    - Given two k-ary conjunctive queries q_1 and q_2,
      is it true that $q_1 \subseteq q_2$?
      (i.e., for every instance D, we have that $q_1(D) \subseteq q_2(D)$)
    - Given two Boolean conjunctive queries q_1 and q_2,
      is it true that $q_1 \models q_2$?
      (i.e., for all D, if $D \models q_1$, then $D \models q_2$)
      CQC is logical implication.

The Homomorphism Problem

- Definition: The Homomorphism Problem
  Given two database instances D and F, is there a homomorphism $h: D \rightarrow F$?
- Notation: $D \rightarrow F$ denotes that a homomorphism from D to F exists.
- Theorem: The Homomorphism Problem is NP-complete.

  Proof: Easy reduction from 3-Colorabilty

  G is 3-colorable if and only if $G \rightarrow K_3$.

Homomorphism Problem & Conjunctive Queries

- Theorem: Chandra & Merlin, 1977
  - CQE and CQC are the “same” problem.
  - Moreover, each is an NP-complete problem.

- Question: What is the common link?
- Answer:
  - Both CQE and CQC are “equivalent” to the Homomorphism Problem.
  - The link is established by bringing into the picture
    - Canonical conjunctive queries and
    - Canonical database instances.

Canonical CQs and Canonical Instances

- Definition: Canonical Conjunctive Query
  Given an instance $D = (R_1, \ldots, R_m)$, the canonical CQ of D is the Boolean conjunctive query Q^D with (a renaming of) the elements of D as variables and the facts of D as conjuncts, where a fact of D is an expression $R_i(a_1,\ldots,a_m)$ such that $(a_1,\ldots,a_m) \in R_i$.

- Example:
  D consists of E(a,b), E(b,c), E(c,a)
  - Q^D is given by the rule:
    Q^D :-- E(x,z), E(z,y), E(y,x)
  - Alternatively, Q^D is
    $$\exists x \exists y \exists z \, (E(x,z) \wedge E(z,y) \wedge E(y,x))$$

Canonical Conjunctive Query

- Example: K_3, the complete graph with 3 nodes

  K_3 is a database instance with one binary relation E, where

  $E = \{(b,r), (r,b), (b,g), (g,b), (r,g), (g,r)\}$

- The canonical conjunctive query Q^{K_3} of K_3 is given by the rule:

  Q^{K_3} :- E(x,y),E(y,x),E(x,z),E(z,x),E(y,z),E(z,y)

- The canonical conjunctive query Q^{K_3} of K_3 is also given by the relational calculus expression:

  $\exists x,y,z(E(x,y) \wedge E(y,x) \wedge E(x,z) \wedge E(z,x) \wedge E(y,z) \wedge E(z,y))$

Canonical Database Instance

- Definition: Canonical Instance

  Given a CQ Q, the canonical instance of Q is the instance D^Q with the variables of Q as elements and the conjuncts of Q as facts.

- Example:

  Conjunctive query Q :-- E(x,y),E(y,z),E(z,w)

  - Canonical instance D^Q consists of the facts E(x,y), E(y,z),E(z,w).
  - In other words, $E^{D^Q} = \{(x,y), (y,z), (z,w)\}$.

Canonical CQs and Canonical Instances

Magic Lemma: Assume that Q is a Boolean conjunctive query and F is a database instance. Then the following statements are equivalent.

- $F \models Q$.
- There is a homomorphism $h: D^Q \rightarrow F$.

Proof: Let Q be $\exists x_1 \ldots \exists x_m \varphi(x_1,\ldots,x_m)$.

1. $\Rightarrow$ 2. Assume that $F \models Q$. Hence, there are elements $a_1, \ldots, a_m$ in adom(F) such that $F \models \varphi(a_1,\ldots,a_m)$. The function h with $h(x_i) = a_i$, for $i=1,\ldots,m$, is a homomorphism from D^Q to F.

2. $\Rightarrow$ 1. Assume that there is a homomorphism $h: D^Q \rightarrow F$. Then the values $h(x_i) = a_i$, for $i = 1,\ldots, m$, give values for the interpretation of the existential quantifiers $\exists x_i$ of Q in adom(F) so that $F \models \varphi(a_1,\ldots,a_m)$.

Homomorphisms, CQE, and CQC

The Homomorphism Theorem: Chandra & Merlin – 1977
For Boolean CQs Q and Q', the following are equivalent:

- $Q \subseteq Q'$
- There is a homomorphism h: $D^{Q'} \to D^{Q}$
- $D^{Q} \models Q'$.

In dual form:

The Homomorphism Theorem: Chandra & Merlin – 1977
For instances D and D', the following are equivalent:

- There is a homomorphism h: $D \to D'$
- $D' \models Q^{D}$
- $Q^{D'} \subseteq Q^{D}$

Homomorphisms, CQE, and CQC

The Homomorphism Theorem: Chandra & Merlin – 1977
For Boolean CQs Q and Q', the following are equivalent:

1. $Q \subseteq Q'$
2. There is a homomorphism h: $D^{Q'} \rightarrow D^{Q}$
3. $D^{Q} \models Q'$.

Proof:
1. $\Rightarrow$ 2. Assume $Q \subseteq Q'$. Since $D^{Q} \models Q$, we have that $D^{Q} \models Q'$.
Hence, by the Magic Lemma, there is a homomorphism from $D^{Q'}$ to D^{Q}.
2. $\Rightarrow$ 3. It follows from the other direction of the Magic Lemma.
3. $\Rightarrow$ 1. Assume that $D^{Q} \models Q'$. So, by the Magic Lemma, there is a homomorphism h: $D^{Q'} \rightarrow D^{Q}$. We have to show that if $F \models Q$, then $F \models Q'$.
Well, if $F \models Q$, then (by the Magic Lemma), there is a homomorphism h': $D^{Q} \rightarrow F$. The composition $h' \circ h$: $D^{Q'} \rightarrow F$ is a homomorphism, hence (once again by the Magic Lemma!), we have that $F \models Q'$.

Illustrating the Homomorphism Theorem

- **Example:**
  - Q : $\exists x_1 \exists x_2 (E(x_1,x_2) \wedge E(x_2,x_1))$
  - Q': $\exists x_1 \exists x_2 \exists x_3 \exists x_4 (E(x_1,x_2) \wedge E(x_2,x_1) \wedge E(x_2,x_3) \wedge E(x_3,x_2) \wedge E(x_3,x_4) \wedge E(x_4,x_3) \wedge E(x_4,x_1) \wedge E(x_1,x_4))$

  Then:
- $Q \subseteq Q'$

  Homomorphism h: $D^{Q'} \rightarrow D^{Q}$ with

  $h(x_1) = x_1$, $h(x_2) = x_2$, $h(x_3) = x_1$, $h(x_4) = x_2$.
- $Q' \subseteq Q$

  Homomorphism h': $D^{Q} \rightarrow D^{Q'}$ with $h'(x_1) = x_1$, $h(x_2) = x_2$.
- Hence, $Q \equiv Q'$.

Illustrating the Homomorphism Theorem

Example: 3-Colorability

For a graph G=(V,E), the following are equivalent:

- G is 3-colorable
- There is a homomorphism h: $G \rightarrow K_3$
- $K_3 \models Q^G$
- $Q^{K3} \subseteq Q^G$.

Combined complexity of CQC and CQE

Corollary: The following problems are NP-complete:

- Given two (Boolean) conjunctive queries Q and Q', is $Q \subseteq Q'$?
- Given a Boolean conjunctive query Q and an instance D, does $D \models Q$?

Proof:

(a) Membership in NP follows from the Homom. Theorem:

$Q \subseteq Q'$ if and only if there is a homomorphism $h: D^{Q'} \rightarrow D^{Q}$

(b) NP-hardness follows from 3-Colorability:

G is 3-colorable if and only if $Q^{K_3} \subseteq Q^{G}$.

Conjunctive Query Equivalence

- The Conjunctive Query Equivalence Problem: Given two conjunctive queries Q and Q', is $Q \equiv Q'$?

- Corollary: For conjunctive queries Q and Q', we have that $Q \equiv Q'$ if and only if $D^Q \equiv_h D^{Q'}$.

- Corollary: The Conjunctive Query Equivalence Problem is NP-complete.

  Proof:

  - The following problem is NP-complete:
    Given a graph H containing a K_3, is H 3-colorable?

  - Let H be a graph containing a K_3. Then
    H is 3-colorable if and only if $Q^H \equiv Q^{K_3}$.

The Complexity of Database Query Languages

	Relational Calculus	Conjunctive Queries
Query Evaluation Problem: Combined Complexity / Query Complexity	PSPACE-complete	NP-complete
Query Evaluation Problem: Data Complexity	In LOGSPACE (hence, in P)	In LOGSPACE (hence, in P)
Query Equivalence Problem	Undecidable	NP-complete
Query Containment Problem	Undecidable	NP-complete

Beyond Conjunctive Queries

- What can we say about query languages of intermediate expressive power between conjunctive queries and the full relational calculus?
- Conjunctive queries form the sublanguage of relational algebra obtained by using only cartesian product, projection, and selection with equality conditions.
- The next step would be to consider relational algebra expressions that also involve union.

Beyond Conjunctive Queries

- Definition:
  - A union of conjunctive queries is a query expressible by an expression of the form $q_1 \cup q_2 \cup \ldots \cup q_m$, where each q_i is a conjunctive query.
  - A monotone query is a query expressible by a relational algebra expression which uses only union, cartesian product, projection, and selection with equality condition.
- Fact:
  - Every union of conjunctive queries is a monotone query.
  - Every monotone query is equivalent to a union of conjunctive queries, but the union may have exponentially many disjuncts.

    (normal form for monotone queries).
  - Monotone queries are precisely the queries expressible by relational calculus expressions using $\wedge$, $\vee$, and $\exists$ only.

Unions of CQs and Monotone Queries

- Union of Conjunctive Queries

  $E \cup \pi_{1,4} (\sigma_{\$2=\$3} (E \times E))$ or, as a relational calculus expression,

  $E(x_1,x_2) \vee \exists z(E(x_1,z) \wedge E(z,x_2))$

- Monotone Query

  Consider the relation schemas $R_1(A,B)$, $R_2(A,B)$, $R_3(B,C)$, $R_4(B,C)$.

  The monotone query

  $(R_1 \cup R_2) \bowtie (R_3 \cup R_4)$

  is equivalent to the following union of conjunctive queries:

  $(R_1 \bowtie R_3) \cup (R_1 \bowtie R_4) \cup (R_2 \bowtie R_3) \cup (R_2 \bowtie R_4)$.

The Containment Problem for Unions of CQs

Theorem: Sagiv and Yannakakis – 1981

Let $q_1 \cup q_2 \cup \ldots \cup q_m$ and $q'_1 \cup q'_2 \cup \ldots \cup q'_n$ be two unions of conjunctive queries. Then the following are equivalent:

1. $q_1 \cup q_2 \cup \ldots \cup q_m \subseteq q'_1 \cup q'_2 \cup \ldots \cup q'_n$.
2. For every $i \leq m$, there is $j \leq n$ such that $q_i \subseteq q'_j$.

Proof: Use the Homomorphism Theorem

1. $\Rightarrow$ 2. Since $D^{q_i} \models q_i$, we have that $D^{q_i} \models q_1 \cup q_2 \cup \ldots \cup q_m$ hence $D^{q_i} \models q'_1 \cup q'_2 \cup \ldots \cup q'_n$, hence there is some $j \leq n$ such that $D^{q_i} \models q'_j$, hence (by the Homomorphism Theorem) $q_i \subseteq q'_j$.

2. $\Rightarrow$ 1. This direction is obvious.

The Containment Problem for Unions of CQ

- Corollary: The Query Containment Problem for Unions of Conjunctive Queries is NP-complete.

  Proof:
  - Membership in NP follows from the Sagiv-Yannakakis Theorem.
    - Guess m pairs (q'_{k_i}, h_{k_i}) and verify that for every $i \leq m$, the function h_{k_i} is a homomorphism from $D^{q'_{ki}}$ to D^{q_i}.
  - NP-hardness follows from the fact that Conjunctive Query Containment is a special case of this problem.
- Fact: The Query Evaluation Problem for Unions of Conjunctive Queries is NP-complete (combined complexity).

  Proof: Exercise.

The Complexity of Database Query Languages

	Relational Calculus	Conjunctive Queries	Unions of Conjunctive Queries
Query Evaluation: Combined Complexity	PSPACE-complete	NP-complete	NP-complete
Query Evaluation: Data Complexity	In LOGSPACE (hence, in P)	In LOGSPACE (hence, in P)	In LOGSPACE (hence, in P)
Query Equivalence	Undecidable	NP-complete	NP-complete
Query Containment	Undecidable	NP-complete	NP-complete

Monotone Queries

- Even though monotone queries have the same expressive power as unions of conjunctive queries, the containment problem for monotone queries has higher complexity than the containment problem for unions of conjunctive queries (syntax/complexity tradeoff)

- Theorem: Sagiv and Yannakakis – 1982
  The containment problem for monotone queries is Π_2^p-complete.

- Note: The prototypical Π_2^p-complete problem is $\forall\exists$-SAT, i.e., the restriction of QBF to formulas of the form

$$\forall x_1 \ldots \forall x_m \exists y_1 \ldots \exists y_n \varphi.$$

The Complexity of Database Query Languages

	Relational Calculus	Conjunctive Queries	Unions of Conjunctive Queries	Monotone Queries
Query Eval.: Combined Complexity	PSPACE-complete	NP-complete	NP-complete	NP-complete
Query Eval.: Data Complexity	In LOGSPACE (hence, in P)	In LOGSPACE (hence, in P)	In LOGSPACE (hence, in P)	In LOGSPACE (hence, in P)
Query Equivalence	Undecidable	NP-complete	NP-complete	Π_2^p-complete
Query Containment	Undecidable	NP-complete	NP-complete	Π_2^p-complete

Conjunctive Queries with Inequalities

- Definition: Conjunctive queries with inequalities form the sublanguage of relational algebra obtained by using only cartesian product, projection, and selection with equality and inequality ($\neq$, $<$, $\leq$) conditions.

- Example: Q(x,y):-- E(x,z), E(z,w),E(w,y), $z \neq w$, $z < y$.

- Theorem: (Klug – 1988, van der Meyden – 1992)
  - The query containment problem for conjunctive queries with inequalities is Π_2^p-complete.
  - The query evaluation problem for conjunctive queries with inequalities in NP-complete.

The Complexity of Database Query Languages

	Relational Calculus	Conjunctive Queries	Unions of Conjunctive Queries	Monotone Queries/ Conj.Queries with Inequal.
Query Eval.: Combined Complexity	PSPACE-complete	NP-complete	NP-complete	NP-complete
Query Eval.: Data Complexity	In LOGSPACE (hence, in P)	In LOGSPACE (hence, in P)	In LOGSPACE (hence, in P)	In LOGSPACE (hence, in P)
Query Equivalence	Undecidable	NP-complete	NP-complete	Π_2^p-complete
Query Containment	Undecidable	NP-complete	NP-complete	Π_2^p-complete

Limitations of Relational Algebra & Calculus

Outline:

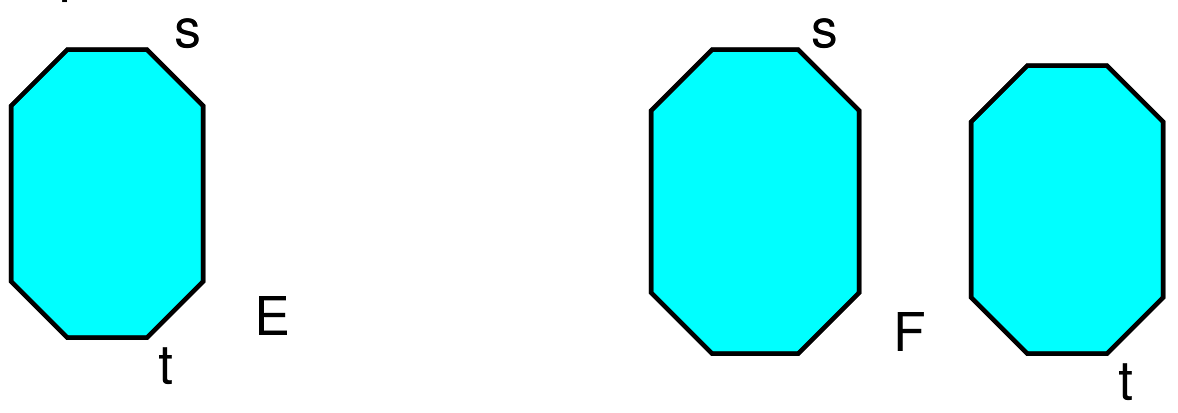
- Relational Algebra and Relational Calculus have substantial expressive power. In particular, they **can** express
  - Natural Join
  - Unions of conjunctive queries
  - …
- However, they **cannot** express recursive queries.
- Datalog is a declarative database query language that augments the language of conjunctive queries with a recursion mechanism.

Transitive Closure and Relational Calculus

Theorem: A. Aho and J. Ullman – 1979 (really, Fraïssé – 1954)
There is **no** relational algebra (or relational calculus) expression that defines the Transitive Closure of a given binary relation E.

Intuition behind this result:

- Relational Calculus queries can only express "local" properties



Overcoming the Limitations of Relational Calculus

- Question: What is to be done to overcome the limitations of the expressive power of relational calculus?
- Answer 1: Embedded Relational Calculus (Embedded SQL):
  - Allow SQL commands inside a conventional programming language, such as C, Java, etc.
  - This is an inferior solution, as it destroys the high-level character of SQL.
- Answer 2:
  - Augment relational calculus with a high-level declarative mechanism for recursion.
  - Conceptually, this a superior solution as it maintains the high-level declarative character of relational calculus.

Datalog

- Datalog = “Conjunctive Queries + Recursion”
- Datalog was introduced by Chandra and Harel in 1982 and has been studied by the research community in depth:
  - Hundreds of research papers in major database conferences; numerous doctoral dissertations.
  - Recent applications outside databases:
    - Specification of network properties
    - Access control languages
    - Business analytics (LogicBlox)
- SQL:1999 and subsequent versions of the SQL standard provide support for a sublanguage of Datalog, called linear Datalog.

Datalog Syntax

Definition: A Datalog program π is a finite set of rules each expressing a conjunctive query

$$T(x_1,\ldots,x_k) \text{ :-- } R_1(\mathbf{u}_1), \ldots, R_n(\mathbf{u}_n),$$

where each variable x_i occurs in the body of the rule.

- Some relational symbols occurring in the heads of the rules may also occur in the bodies of the rules

  (unlike the rules for conjunctive queries).
  - These relational symbols are the recursive relational symbols; they are also known as intensional database predicates (IDBs).

- The remaining relational symbols in the rules are known as the extensional database predicates (EDBs).

Datalog

- Example: Datalog program for Transitive Closure

  T(x,y) :- E(x,y)

  T(x,y) :- E(x,z), T(z,y)

  – E is the EDB predicate
  – T is the IDB predicate
  – The intuition is that the Datalog program gives a recursive specification of the IDB predicate T in terms of the EDB E.

- Example: Another Datalog program for Transitive Closure

  T(x,y) :- E(x,y)

  T(x,y) :- T(x,z), T(z,y)

  ("divide and conquer" algorithm for Transitive Closure)

Datalog

- Example: Paths of Even and Odd Length

  Consider the Datalog program:

  ```
  ODD(x,y)  :-  E(x,y)
  ODD(x,y)  :-  E(x,z), EVEN(z,y)
  EVEN(x,y) :-  E(x,z), ODD(z,y).
  ```

  - E is the EDB predicate
  - EVEN and ODD are the IDB predicates.
  - This program gives a recursive specification of the IDB predicates EVEN and ODD in terms of the EDB predicate E.
  - This is a Datalog program expressing mutual recursion.

Datalog Semantics

- Question: What is the precise semantics of a Datalog program?
- Short Answer: Datalog is a fragment of Least Fixed-Point Logic LFP, hence it inherits the semantics of LFP.
- Fact: Datalog coincides with Existential Positive LFP
- Example:
  - Datalog program π

    T(x,y) :- E(x,y)

    T(x,y) :- E(x,z), T(z,y)
  - Existential Positive First-Order Formula

    $\varphi(x,y,T) \equiv E(x,y) \vee \exists\, z\,(E(x,z) \wedge T(z,y))$
  - The semantics of the Datalog program π is the least fixed-point of the formula $\varphi(x,y,T)$.

Datalog Semantics

- Question: What is the precise semantics of a Datalog program?
- Long Answer (self-contained): Datalog programs can be given two different types of semantics.
  - Declarative Semantics (denotational semantics)
    - Smallest solutions of recursive specifications.
    - Least fixed-points of monotone operators.
  - Procedural Semantics (operational semantics)
    - An iterative process for computing the "meaning" of Datalog programs.
- Main Result: The declarative semantics coincides with the procedural semantics.

Declarative Semantics of Datalog Programs

- Each Datalog program can be viewed as a recursive specification of its IDB predicates.
- This specification is expressed using relational algebra operators
  - The body of each rule uses π, σ, and cartesian product $\times$
  - All rules having the same predicate in the head are combined using union.
  - The recursive specification is given by equations involving unions of conjunctive queries.
- Example: T(x,y) :- E(x,y)

  T(x,y) :- T(x,z), T(z,y)
  - Recursive equation:

$$T = E \cup \pi_{1,4}(\sigma_{\$2=\$3} (T \times T))$$

Declarative Semantics of Datalog Programs

Example: Consider the Datalog program:

```
ODD(x,y)  :-  E(x,y)
ODD(x,y)  :-  E(x,z), EVEN(z,y)
EVEN(x,y) :-  E(x,z), ODD(z,y).
```

- System of recursive equations:

$$\mathrm{ODD} = \mathrm{E} \cup \pi_{1,4}(\sigma_{\$2=\$3}\ (\mathrm{E}\times\mathrm{EVEN}))$$

$$\mathrm{EVEN} = \pi_{1,4}(\sigma_{\$2=\$3}\ (\mathrm{E}\times\mathrm{ODD})).$$

Declarative Semantics of Datalog Programs

- Recursive equations arising from Datalog programs **need not** have a unique solution.

- Example: Consider the recursive equation:

$$T = E \cup \pi_{1,4}(\sigma_{\$2=\$3}(T \times T))$$

Let E = { (1,2), (2,3) }.

Then both T_1 and T_2 satisfy this recursive equation, where

- T_1 = { (1,2), (2,3), (1,3) }
- T_2 = { (1,2),(2,1),(2,3),(3,2),(1,3),(3,2),(1,1),(2,2),(3,3) }.

Furthermore, this recursive equation has many other solutions.

Declarative Semantics of Datalog Programs

- Theorem: Every recursive equation arising from a Datalog program has a smallest solution (smallest w.r.t. the $\subseteq$ partial order).

- Example: Datalog program

  T(x,y) :- E(x,y)

  T(x,y):- T(x,z), T(z,y)

  – Recursive equation:

  $$T = E \cup \pi_{1,4}(\sigma_{\$2=\$3}(T \times T))$$

  – The smallest solution of this recursive equation is the Transitive Closure of E.

- Note: A special case of the Knaster-Tarski Theorem for smallest solutions of recursive equations arising from monotone operators (Datalog uses monotone relational algebra operators only).

Procedural Semantics of Datalog Programs

Definition: Let π be a Datalog program. The procedural semantics of π are obtained by the following bottom-up evaluation of the recursive predicates (IDBs) of π:

1. Set all IDBs of π to $\emptyset$.
2. Apply all rules of π in parallel; update the IDBs by evaluating the bodies of the rules on the given database D.
3. Repeat until no IDB predicate changes.
4. Return the values of the IDB predicates obtained at the end of Step 3.

Procedural Semantics of Datalog Programs

Example: Datalog program for Transitive Closure

T(x,y) :- E(x,y)

T(x,y):- E(x,z),T(z,y)

– Bottom-up evaluation:

$T^0 = \emptyset$

$T^{n+1} = \{(a,b): E(a,b) \vee \exists z(E(a,z) \wedge T^n(z,b))\}$

Fact: The following statements are true:

– T^n ={ (a,b): there is a path of length at most n from a to b }

– Transitive Closure of E = $\cup_{n \geq 1} T^n$.

Proof: By induction on n.

Procedural Semantics of Datalog Programs

Example: Another Datalog program for Transitive Closure

T(x,y) :- E(x,y)

T(x,y):- T(x,z),T(z,y)

– Bottom-up evaluation:

$T^0 = \varnothing$

$T^{n+1} = \{(a,b): E(a,b) \lor \exists z(T^n(a,z) \land T^n(z,b))\}$

Fact: The following statements are true:

– T^n ={ (a,b): there is a path of length at most 2^n from a to b }
– Transitive Closure of E = $\cup_{n \geq 1} T^n$.

Proof: By induction on n.

Procedural Semantics of Datalog Programs

Example: Consider the Datalog program

```
ODD(x,y)   :-  E(x,y)
ODD(x,y)   :-  E(x,z), EVEN(z,y)
EVEN(x,y)  :-  E(x,z), ODD(z,y)
```

- Bottom-up evaluation:

$$\mathrm{ODD}^0 = \emptyset$$

$$\mathrm{EVEN}^0 = \emptyset$$

$$\mathrm{ODD}^{n+1} = \{(a,b): E(a,b) \vee \exists z(E(a,z) \wedge \mathrm{EVEN}^n(z,b))\}$$

$$\mathrm{EVEN}^{n+1} = \{(a,b): \exists z(E(a,z) \wedge \mathrm{ODD}^n(z,b))\}$$

Fact: The following statements are true:

- $\cup_{n \geq 1} \mathrm{ODD}^n$ = { (a,b): there is a path of odd length from a to b }
- $\cup_{n \geq 1} \mathrm{EVEN}^n$ = { (a,b): there is a path of even length from a to b }.

Proof: By induction on n.

Declarative vs. Procedural Datalog Semantics

Theorem: Let π be a Datalog program. Then the following statements are true:

- The bottom-up evaluation of the procedural semantics of π terminates within a number of steps bounded by a polynomial in the size of the database instance D (= size of the EDB predicates).
- The declarative semantics of π coincides with the procedural semantics of π.

Declarative vs. Procedural Datalog Semantics

Proof: For simplicity, assume that π has a single IDB T of arity k.

- By induction on n, show that $T^n \subseteq T^{n+1}$, for every n.

  (this uses the monotonicity of unions of conjunctive queries).
- Hence, $T^0 \subseteq T^1 \subseteq \ldots \subseteq T^n \subseteq T^{n+1} \subseteq \ldots$
- Since each $T^n \subseteq \mathrm{adom}(D)^k$, there is an $m \leq |\mathrm{adom}(D)|^k$ such that $T^m = T^{m+1}$.
- Since $T^m = T^{m+1}$, we have that the procedural semantics produces a solution to the recursive equation arising from π.
- By induction on n, show that if T^* is another solution of this recursive equation, then $T^n \subseteq T^*$, for all n

  (use the monotonicity of unions of conjunctive queries again).
- In particular, $T^m \subseteq T^*$, hence T^m is the smallest solution of this recursive equation.

The Query Evaluation Problem for Datalog

Theorem: Let π be a Datalog program.

There is a polynomial-time algorithm such that, given a database instance D, it evaluates π on D.

Proof: The bottom-up evaluation of the procedural semantics of π on a database D runs in polynomial time because:

- The number of iterations is bounded by a polynomial in the size of D.
- Each step of the iteration can be carried out in polynomial time (why?).

Corollary: The data complexity of Datalog is in P.

The Query Evaluation Problem for Datalog

Corollary: The data complexity of Datalog is in P.

Theorem: The combined complexity of Datalog is EXPTIME-complete.

Note: Recall that

$$\mathrm{P} \subseteq \mathrm{NP} \subseteq \mathrm{PSPACE} \subseteq \mathrm{EXPTIME}.$$

Thus, Datalog has higher combined complexity than relational calculus (since the combined complexity of relational calculus is PSPACE-complete).

Some Interesting Datalog Programs

Example: Non 2-Colorability can be expressed by a Datalog program.

Fact: A graph E is 2-colorable if and only if it does not contain a cycle of odd length.

Datalog program for Non 2-Colorability:

```
ODD(x,y)    :--    E(x,y)
ODD(x,y)    :--    E(x,z), EVEN(z,y)
EVEN(x,y)   :--   E(x,z), ODD(z,y).
 Q              :--   ODD(x,x)
```

Sanity check: Can you find a Datalog program for Non 3-Colorability?

Some Interesting Datalog Programs

Example: Path Systems Problem

```
T(x) :--   A(x)
T(x) :--   R(x,y,z), T(y), T(z)
```

Theorem: S. Cook – 1974
Evaluating this Datalog program is a P-complete problem (via logspace-reductions).

Note:

- Path Systems was the first problem shown to be P-complete.
- In particular, it is highly unlikely that Path Systems is in LOGSPACE.
- Thus, Datalog has higher data complexity than relational calculus.

Linear Datalog

Example: Give a linear Datalog program that computes the binary query COUSIN from the binary relation schema PARENT

```
SIBLING(x,y)  :-   PARENT(z,x), PARENT(z,y)
COUSIN(x,y)   :-  PARENT(z,x), PARENT(w,y), SIBLING(z,w)
COUSIN(x,y)   :-  PARENT(z,x), PARENT(w,y), COUSIN(z,w).
```

Fact: COUSIN(Barack Obama, Dick Cheney)
Actually, COUSIN^8(Barack Obama, Dick Cheney)

http://www.msnbc.msn.com/id/21340764/

Fact: COUSIN(Sarah Palin, Princess Diana).
Actually, COUSIN^{10}(Sarah Palin, Princess Diana)

http://www.dailymail.co.uk/news/worldnews/article-1073249/Sarah-Palin-Princess-Diana-cousins-genealogists-reveal.html

Linear Datalog

- Definition: A Datalog program π is linearizable if there is a linear Datalog program π^* that is equivalent to π.

- Example: The following Datalog program is linearizable:

```
T(x,y) :- E(x,y)
T(x,y) :- T(x,z), T(z,y)
```

- Example: The following Datalog program is **not** linearizable:

```
T(x) :-  A(x)
T(x) :-  R(x,y,z), T(y), T(z)
```

(the proof of this fact is non-trivial).

Datalog and SQL

- SQL:1999 and subsequent versions of the SQL standard provide support for linear Datalog programs (but **not** for non-linear ones)

- Syntax:

WITH RECURSIVE T AS

<Datalog program for T>

<query involving T>

- Semantics:
  - Compute T as the semantics of <Datalog program for T>
  - The result of the previous step is a temporary relation that is then used, together with other EDBS, as if it were a stored relation (an EDB) in <query involving T>.

Datalog vs. First-Order Logic

Facts:

- Unions of Conjunctive Queries are contained in Datalog
- Relational Calculus is not contained in Datalog
  - Datalog cannot express universal first-order sentences.
- Datalog is not contained in Relational Calculus
  - Transitive closure is not expressible in Relational Calculus

Corollary to Rossman's Theorem:

Datalog ∩ Relational Calculus = Unions of Conjunctive Queries

Proof:

Datalog queries are preserved under homomorphisms.

Datalog vs. First-Order Logic

Theorem (Ajtai-Guvevich – 1987)

The following statements are equivalent for a Datalog program π:

1. π is bounded.
2. The query q defined by π is expressible in first-order logic.

Proof: Use Rossman's Theorem (2005) for: (2) implies (1).

- If q is first-order expressible, then, by Rossman, q is equivalent to a finite union of conjunctive queries $q'_1 \cup \ldots \cup q'_m$.
- Therefore,

$$q_1 \cup \ldots \cup q_n \cup \ldots \equiv q'_1 \cup \ldots \cup q'_m$$

- By Sagiv and Yannakakis, it follows that there is some N such that

$$q_1 \cup \ldots \cup q_n \cup \ldots \equiv q_1 \cup \ldots \cup q_{N.}$$

- Hence, π is bounded.

Datalog with Negation

- Question: What if we allow negation in the bodies of Datalog rules?

- Examples:
  - T(x) :- ¬ T(x)
    (the recursive specification has **no** solutions!)
  - S(x) :- E(x,y), ¬ S(y)

- Note: Several different semantics for Datalog programs with negation have been proposed over the years (see Ch. 15 of AHV):
    - Stratified datalog programs
    - Well-founded semantics
    - Inflationary semantics
    - Stable model semantics
    - …

Query Equivalence and Containment for Datalog

- Note: Recall that the following are known about the Query Evaluation Problem for Datalog queries:
  - The data complexity of Datalog is in P.
  - The combined complexity of Datalog is EXPTIME-complete.

- Questions:
  - What about the Query Equivalence Problem for Datalog:

    Given two Datalog programs π and π', is π equivalent to π'?

    (do they return the same answer on every database instance?)
  - What about the Query Containment Problem for Datalog:

    Given two Datalog programs π and π', is $\pi \subseteq \pi$'?

    (is $\pi(I) \subseteq \pi'(I)$, on every database instance I?)

Query Equivalence and Containment for Datalog

Theorem: O. Shmueli – 1987

- The query equivalence problem for Datalog queries is undecidable. In fact, it is undecidable even for Datalog queries with a single binary IDB.
- Consequently, the query containment problem for Datalog queries is undecidable, even for Datalog programs with a single binary IDB.

Hint of Proof:

- Reduction from Context-Free Grammar Equivalence:

  Given two context-free grammars G and G’, is L(G) = L(G’)?

Decidable Containment for Datalog Fragments

Theorem (Cosmadakis, Gaifman, Kanellakis, Vardi – 1988)

The query containment problem for monadic Datalog queries (all IDBs in the Datalog program are unary) is decidable.

In fact, it is in 2EXPTIME-time and is EXPTIME-hard.

Theorem (Reutter, Romero, Vardi – 2015)

The query containment problem for regular queries (a large fragment of binary Datalog) is 2EXPSPACE-complete.

Note: Both these results make systematic use of automata-theoretic techniques.

The Complexity of Database Query Languages

	Relational Calculus	Conjunctive Queries	Unions of Conjunctive Queries	Datalog Queries
Query Eval.: Combined Complexity	PSPACE-complete	NP-complete	NP-complete	EXPTIME-complete
Query Eval.: Data Complexity	In LOGSPACE (hence, in P)	In LOGSPACE (hence, in P)	In LOGSPACE (hence, in P)	P-complete
Query Equivalence	Undecidable	NP-complete	NP-complete	Undecidable
Query Containment	Undecidable	NP-complete	NP-complete	Undecidable