Conjunctive queries

Giuseppe De Giacomo

Università di Roma "La Sapienza"

Corso di Seminari di Ingegneria del Software:
Data and Service Integration
Laurea Specialistica in Ingegneria Informatica
Università degli Studi di Roma "La Sapienza"
A.A. 2005-06

FOL queries

A FOL query is an (open) FOL formula.

Let ϕ be a FOL query with free variables $(x_1, \ldots, x_k)$, then we sometimes write it as $\phi(x_1, \ldots, x_k)$.

Given an interpretation $\mathcal{I}$, the assignments we are interested in are those that map the variables $x_1, \ldots, x_k$ (and only those). We will write such assignment explicitly sometimes: i.e., $\alpha(x_i) = a_i$ $(i = 1, \ldots, k)$, is written simply as $\langle a_1, \ldots, a_k \rangle$.

Now we define the answer to a query $\phi(x_1, \ldots, x_k)$ as follows

$$\phi(x_1, \ldots, x_k)^{\mathcal{I}} = \{(a_1, \ldots, a_k) \mid \mathcal{I}, \langle a_1, \ldots, a_k \rangle \models \phi(x_1, \ldots, x_k)\}$$

Note: We will also use the notation: $\phi^{\mathcal{I}}$, keeping the free variables implicit, and $\phi(\mathcal{I})$ making apparent that ϕ becomes a functions from interpretations to set of tuples.

Conjunctive queries (CQs)

A conjunctive query (CQ) q is a query of the form

$$\exists \vec{y}.conj(\vec{x}, \vec{y})$$

where $conj(\vec{x}, \vec{y})$ is a conjunction (an "and") of atoms and equalities, with free variables $\vec{x}$ and $\vec{y}$.

- CQs are the most frequently asked queries
- CQs correspond to relational algebra Select-Project-Join (SPJ) queries

CQs: datalog notation

A conjunctive query $q = \exists \vec{y}.conj(\vec{x}, \vec{y})$ is denoted in datalog notation as

$$q(\vec{x'}) \leftarrow conj'(\vec{x'}, \vec{y'})$$

where $conj'(\vec{x'}, \vec{y'})$ is the list of atoms in $conj(\vec{x}, \vec{y})$ obtained after having equated the variables $\vec{x}, \vec{y}$ according to the equalities in $conj(\vec{x}, \vec{y})$. As a result of such equality elimination, we have that $\vec{x'}$ and $\vec{y'}$ can actually contain constants and multiple occurrences of the same variable.

We call $q(\vec{x'})$ the head of q, and $conj'(\vec{x'}, \vec{y'})$ the body. Moreover, we call the variables in $\vec{x'}$ the distinguished variables of q and those in $\vec{y'}$ the non-distinguished variables.

Example

- Consider an interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, E^{\mathcal{I}})$, where $E^{\mathcal{I}}$ is a binary relation – *note that such interpretation is a (directed) graph*;
- the following CQ q returns all nodes that participate to a triangle in the graph:

$$\exists y, z. E(x, y) \land E(y, z) \land E(z, x)$$

- the query q in datalog notation becomes:

$$q(x) \leftarrow E(x, y), E(y, z), E(z, x)$$

- the query q in SQL is ($E(x, y) \rightsquigarrow$ `Edge(F,S)`):

```
select e1.F
from Edge e1, Edge e2, Edge e3
where e1.S=e2.F, e2.S=e3.F, e3.S=e1.F
```

Nondeterministic CQ evaluation algorithm

```
boolean ConjTruth(𝓘,α,∃y⃗. conj(x⃗,y⃗)) {
  GUESS assignment α[y⃗ ↦ a⃗] {
     return Truth(𝓘,α[x⃗ ↦ a⃗],conj(x⃗,y⃗));
}

boolean Truth(𝓘,α,φ)) {
  if(φ is t_1 = t_2)
     return TermEval(t_1) = TermEval(t_2);
  if(φ is P(t_1,...,t_k))
     return P^𝓘(TermEval(t_1),...,TermEval(t_k));
  if(φ is ψ ∧ ψ')
     return Truth(𝓘,α,ψ) ∧ Truth(𝓘,α,ψ');
}
```



```
o ∈ Δ^𝓘 TermEval(𝓘,α,t) {
   if(t is a variable x) return α(x);
   if(t is a constant c) return c^𝓘;
}
```

CQ evaluation: combined, data, query complexity

Combined complexity: complexity of $\{\langle \mathcal{I}, \alpha, q \rangle \mid \mathcal{I}, \alpha \models q\}$, i.e., interpretation, tuple, and query part of the input:

- time: exponential
- space: NP *(NP-complete –see below for hardness)*

Data complexity: complexity of $\{\langle \mathcal{I}, \alpha \rangle \mid \mathcal{I}, \alpha \models q\}$, i.e., interpretation fixed (not part of the input):

- time: polynomial
- space: LogSpace *(LogSpace-complete –see [Vardi82] for hardness)*

Query complexity: complexity of $\{\langle \alpha, q \rangle \mid \mathcal{I}, \alpha \models q\}$, i.e., query fixed (not part of the input):

- time: exponential
- space: NP *(NP-complete –see below for hardness)*

3-colorability

3-colorability: Given a graph $G = (V, E)$, is it 3-colorable?

Thm: 3-colorability is NP-complete.

can we deduce 3-colorability to conjunctive query evaluation?
YES

Reduction from 3-colorability to CQ evaluation

Let $G = (V, E)$ be a graph, we define:

- Interpretation: $\mathcal{I} = (\Delta^{\mathcal{I}}, E^{\mathcal{I}})$ where:
  - $\Delta^{\mathcal{I}} = \{r, g, b\}$
  - $E^{\mathcal{I}} = \{(r, g), (g, r), (r, b), (b, r), (b, g), (g, b)\}$
- Conjunctive query: Let $V = \{x_1, \ldots, x_n\}$, then consider the boolean conjunctive query q defined as:

$$\exists x_1, \ldots, x_n. \bigwedge_{(x_i, x_j) \in E} E(x_i, x_j) \wedge E(x_j, x_i)$$

Thm: G is 3-colorable iff $\mathcal{I} \models q$.

Thm: CQ evaluation is NP-hard in query and combined complexity.

Homomorphism

Let $\mathcal{I} = (\Delta^{\mathcal{I}}, P^{\mathcal{I}}, \ldots, c^{\mathcal{I}}, \ldots)$ and $\mathcal{J} = (\Delta^{\mathcal{J}}, P^{\mathcal{J}}, \ldots, c^{\mathcal{J}}, \ldots)$ be two interpretation over the same alphabet (for simplicity, we consider only constants as functions). Then an homomorphism form $\mathcal{I}$ to $\mathcal{J}$ is a mapping $h : \Delta^{\mathcal{I}} \rightarrow \Delta^{\mathcal{J}}$ such that:

- $h(c^{\mathcal{I}}) = c^{\mathcal{J}}$
- $h(P^{\mathcal{I}}(a_1, \ldots, a_k)) = P^{\mathcal{J}}(h(a_1), \ldots, h(a_k))$

Note: An isomorphism is a homomorphism, which is one-to-one and onto.

Thm: FOL is unable to distinguish between interpretations that are isomorphic – any standard book on logic.

Recognition problem and boolean query evaluation

Consider the recognition problem associated to the evaluation of a query q, then

$$\mathcal{I}, \alpha \models q(\vec{x}) \text{ iff } \mathcal{I}' \models q(\vec{c})$$

where $\mathcal{I}'$ is identical to $\mathcal{I}$ but includes a new constant c which is interpreted as $c^{\mathcal{I}'} = \alpha(x)$.

That is, we can reduce the recognition problem to the evaluation of a boolean query.

Canonical interpretation of a (boolean) CQ

Let q be a conjunctive query

$$\exists x_1, \ldots, x_n.conj$$

then the canonical interpretation $\mathcal{I}_q$ associated with q is the interpretation $\mathcal{I}_q = (\Delta^{\mathcal{I}_q}, P^{\mathcal{I}_q}, \ldots, c^{\mathcal{I}_q}, \ldots)$, where

- $\Delta^{\mathcal{I}_q} = \{x_1, \ldots, x_n\} \cup \{c \mid c \text{ constant occurring in } q\}$, i.e., all the variables and constants
- $c^{\mathcal{I}_q} = c$ for all constants in q
- $(t_1, t_2) \in P^{\mathcal{I}_q}$ iff the atom $P(t_1, t_2)$ occurs in q

Sometime the procedure for obtaining the canonical interpretation is call freezing of q.

Example Given the boolean query q:

$$q(c) \leftarrow E(c, y), E(y, z), E(z, c)$$

the canonical structure $\mathcal{I}_q$ is defined as

$$\mathcal{I}_q = (\Delta^{\mathcal{I}_q}, E^{\mathcal{I}_q}, c^{\mathcal{I}_q})$$

where

- $\Delta^{\mathcal{I}_q} = \{y, z, c\}$
- $c^{\mathcal{I}_q} = c$
- $E^{\mathcal{I}_q} = \{(c, y), (y, z), (z, c)\}$

Canonical interpretation and query evaluation

Thm [Chandra&Merlin77]: For (boolean) CQs, $\mathcal{I} \models q$ iff there exists an homomorphism from $\mathcal{I}_q$ to $\mathcal{I}$.

Proof.

$\Rightarrow$ Let $\mathcal{I} \models q$, let α be the assignment to an existential variables that makes the query true in $\mathcal{I}$, and let $\bar{\alpha}$ be its extension to constants. Then $\bar{\alpha}$ is an homomorphism from $\mathcal{I}_q$ to $\mathcal{I}$.

$\Leftarrow$ Let h be an homomorphism from $\mathcal{I}_q$ to $\mathcal{I}$, then restricting h to the variables only we obtain an assignment of the existential variables that makes q true in $\mathcal{I}$. $\square$

In other words (the recognition problem associated to) query evaluation can be reduced to finding an homomorphism.

Finding an homomorphism between two interpretations (aka relational structure) is also known as solving a CSP (Constraint Satisfaction Problem), well-studied in AI –see also [Kolaitis&Vardi98].

Query containment

Query containment: given two FOL queries ϕ and ψ check whether $\phi \subseteq \psi$ for all interpretations $\mathcal{I}$ and all assignments α we have that

$$\mathcal{I}, \alpha \models \phi \text{ implies } \mathcal{I}, \alpha \models \psi$$

(In logical terms check whether $\phi \models \psi$.)

Note: of special interest in query optimization.

Thm: For FOL queries, query containment is undecidible.

Proof: Reduction from FOL logical implication.$\square$

Query containment for CQs

For CQs, query containment can be reduced to query evaluation!

Step 1 – freeze the free variables: $q(\vec{x}) \subseteq q'(\vec{x})$ iff

- $\mathcal{I}, \alpha \models q(\vec{x})$ implies $\mathcal{I}, \alpha \models q'(\vec{x})$, for all $\mathcal{I}$ and α; or equivalently
- $\mathcal{I}' \models q(\vec{c})$ implies $\mathcal{I}' \models q'(\vec{c})$, for all $\mathcal{I}'$, where $\vec{c}$ are new constants, and $\mathcal{I}'$ extends $\mathcal{I}$ to the new constants as follows $c^{\mathcal{I}'} = \alpha(x)$.

Step 2 – construct the canonical intepretation of the CQ on the left $q(\vec{c})$
consider the canonical interpretation $\mathcal{I}_{q(\vec{c})}$ ...

Step 3 – evaluate the CQ on the right $q'(\vec{c})$ on $\mathcal{I}_{q(\vec{c})}$

.... check whether $\mathcal{I}_{q(\vec{c})} \models q'(\vec{c})$.

Query containment for CQs (cont.)

Thm [Chandra&Merlin77]: For CQs, $q(\vec{x}) \subseteq q'(\vec{x})$ iff $\mathcal{I}_{q(\vec{c})} \models q'(\vec{c})$, where $\vec{c}$ are new constants.

Proof.

$\Rightarrow$ Assume that $q(\vec{c}) \subseteq q'(\vec{c})$:

- since $\mathcal{I}_{q(\vec{c})} \models q(\vec{c})$ it follows that $\mathcal{I}_{q(\vec{c})} \models q'(\vec{c})$.

$\Leftarrow$ Assume that $\mathcal{I}_{q(\vec{c})} \models q'(\vec{c})$.

- by Thm[Chandra&Merlin77] on homomorphism, for every $\mathcal{I}$ such that $\mathcal{I} \models q(\vec{c})$ there exists an homomorphism h from $\mathcal{I}_{q(\vec{c})}$ to $\mathcal{I}$;
- on the other hand, since $\mathcal{I}_{q(\vec{c})} \models q'(\vec{c})$, again by Thm[Chandra&Merlin77] on homomorphism, there exists an homomorphism h' from $\mathcal{I}_{q'(\vec{c})}$ to $\mathcal{I}_{q(\vec{c})}$;
- the mapping $h \circ h'$ obtained composing h and h' is an homomorphism

from $\mathcal{I}_{q'(\vec{c})}$ to $\mathcal{I}$. Hence, once again for Thm[Chandra&Merlin77] on homomorphism, $\mathcal{I} \models q'(\vec{c})$.

So we can conclude $q(\vec{c}) \subseteq q'(\vec{c})$. $\Box$

Thm: Containment of CQs is NP-complete.

Union of conjunctive queries (UCQs)

A union of conjunctive queries (UCQ) q is a query of the form

$$\bigvee_{i=1,\ldots,n} \exists \vec{y_i}.conj_i(\vec{x}, \vec{y_i})$$

where each $conj_i(\vec{x}, \vec{y_i})$ is, as before, a conjunction of atoms and equalities with free variables $\vec{x}$ and $\vec{y_i}$.

Note: Obviously, conjunctive queries are a subset of union of conjunctive queries.

UCQs: datalog notation

The datalog notation is then extended to union of conjunctive queries as follows. A union of conjunctive queries

$$q = \bigvee_{i=1,\ldots,n} \exists \vec{y_i}.conj_i(\vec{x}, \vec{y_i})$$

is denoted in datalog notation as

$$q = \{ q_1, \ldots, q_n \}$$

where each q_i is the datalog expression corresponding to the conjunctive query $q_i = \{ \vec{x} \mid \exists \vec{y_i}.conj_i(\vec{x}, \vec{y_i}) \}$.

UCQs: query evaluation

Form the definition of FOL query we have that:

$$\mathcal{I}, \alpha \models \bigvee_{i=1,\ldots,n} \exists \vec{y_i}.conj_i(\vec{x}, \vec{y_i})$$

iff

$$\mathcal{I}, \alpha \models \exists \vec{y_i}.conj_i(\vec{x}, \vec{y_i}) \qquad \text{for some } i = 1, \ldots, n.$$

Hence to evaluate a UCQ q, we simply evaluate a number (linear in the size of q of conjunctive queries in isolation.

Hence, evaluating UCQs has the same complexity of evaluating CQs.

UCQs: combined, data, query complexity

Combined complexity: complexity of $\{\langle \mathcal{I}, \alpha, q\rangle \mid \mathcal{I}, \alpha \models q\}$, i.e., interpretation, tuple, and query part of the input:

- time: exponential
- space: NP-complete

Data complexity: complexity of $\{\langle \mathcal{I}, \alpha\rangle \mid \mathcal{I}, \alpha \models q\}$, i.e., interpretation fixed (not part of the input):

- time: polynomial
- space: LogSpace-complete

Query complexity: complexity of $\{\langle \alpha, q\rangle \mid \mathcal{I}, \alpha \models q\}$, i.e., query fixed (not part of the input):

- time: exponential
- space: NP-complete

Query containment for UCQs

Thm: For UCQs, $\{q_1, \ldots, q_k\} \subseteq \{q'_1, \ldots, q'_n\}$ iff for all q_i there is a q'_j such that $q_i \subseteq q'_j$.

Proof.

$\Leftarrow$ Obvious.

$\Rightarrow$ If the containment holds, then we have $\{q_1(\vec{c}), \ldots, q_k(\vec{c})\} \subseteq \{q'_1(\vec{c}), \ldots, q'_n(\vec{c})\}$, where $\vec{c}$ are new variables:

- now consider $\mathcal{I}_{q_i(\vec{c})}$, we have $\mathcal{I}_{q_i(\vec{c})} \models q_i(\vec{c})$, and hence $\mathcal{I}_{q_i(\vec{c})} \models \{q_1(\vec{c}), \ldots, q_k(\vec{c})\}$;
- by the containment we have that $\mathcal{I}_{q_i(\vec{c})} \models \{q'_1(\vec{c}), \ldots, q'_n(\vec{c})\}$, that is there exists a $q'_j(\vec{c})$ such that $\mathcal{I}_{q_i(\vec{c})} \models q'_j(\vec{c})$;
- hence, by the Thm[Chandra&Merlin77] on containment of CQs, we have

that $q_i \subseteq q'_j$.$\square$