

# Finite Model Theory

## Unit 4

Dan Suciu

Spring 2018

# 599c: Finite Model Theory

## Unit 4: Query Containment and Equivalence

# Resources

- Abitebou, Hull, Vianu, *Database Theory* (Alice book)
- Simon's Institute: *Logical Structures in Computation Boot Camp*, 2016  
<https://simons.berkeley.edu/workshops/logic2016-boot-camp>  
See Kolaiti's tutorial on Logic and Databases
- Cerignou, Vollmer, *Boolean Constraint Satisfaction Problem*.

# Query

Fix a vocabulary  $\sigma$ .

An FO **query** is defined by formula  $Q(\mathbf{x})$  with  $k$  free variables  
 $Q$  maps  $\mathbf{A} \in \text{STRUCT}[\sigma]$  to the relation  $Q(\mathbf{A}) \subseteq A^k$ :

$$Q(\mathbf{A}) \stackrel{\text{def}}{=} \{\mathbf{a} \subseteq A^k \mid \mathbf{A} \models Q[\mathbf{a}]\}$$

**discuss** connection to FO reduction  $\text{STRUCT}[\sigma] \rightarrow \text{STRUCT}[\tau]$ .

When  $k = 0$  then we call it a **Boolean** query:  $Q(\mathbf{D})$  is true or false.

**Warning:** we use conflicting notations  $Q(\mathbf{A})$  and  $Q(\mathbf{x})$ .

# Problem Definition

## Definition (Query Containment)

We say that  $Q_1$  is contained in  $Q_2$ ,  $Q_1 \subseteq Q_2$  if for all  $\mathbf{A}$ ,  $Q_1(\mathbf{A}) \subseteq Q_2(\mathbf{A})$ .

The **containment problem** for a language  $L$  is:

given  $Q_1, Q_2 \in L$  check if  $Q_1 \subseteq Q_2$ .

When  $Q_1, Q_2$  are Boolean queries, then containment is logical implication:

$Q_1 \rightarrow Q_2$ .

## Definition (Query Equivalence)

We say that  $Q_1$  is equivalent to  $Q_2$ ,  $Q_1 \equiv Q_2$  if for all  $\mathbf{A}$ ,  $Q_1(\mathbf{A}) = Q_2(\mathbf{A})$ .

The **equivalence problem** for a language  $L$  is:

given  $Q_1, Q_2 \in L$  check if  $Q_1 \equiv Q_2$ .

## Discussion

- If  $L$  is closed under  $\wedge$  or closed under  $\vee$  then containment and equivalence have the same complexity. **proof in class**

Thus, containment and equivalence are essentially the same problem.

- However, it is undecidable for FO:

### Theorem

*The problem “given  $Q_1, Q_2 \in FO$ , is  $Q_1 \subseteq Q_2$ ?” is undecidable.*

**proof in class**

- Thus, we study containment for fragments  $L \subseteq FO$ .

# The Homomorphism Problem

Fix two structures  $\mathbf{A} = (A, R_1^A, \dots, R_m^A)$ ,  $\mathbf{B} = (B, R_1^B, \dots, R_m^B)$ .

A **homomorphism**  $f : \mathbf{A} \rightarrow \mathbf{B}$  is a function  $f : A \rightarrow B$  s.t.  $f(R_j^A) \subseteq R_j^B$  for  $j = 1, m$ .

## Definition (The Homomorphism Problem)

The homomorphism problem is: given two structures  $\mathbf{A}, \mathbf{B}$ , check if there exists a homomorphism  $h : \mathbf{A} \rightarrow \mathbf{B}$

# The Homomorphism Problem: Complexity

Find  $f : \mathbf{A} \rightarrow \mathbf{B}$

## Theorem

- (1) The homomorphism problem is NP-hard in general.*
- (2) There exists a fixed  $\mathbf{B}$  s.t. the homomorphism problem is NP-hard.*

Prove (2) in class, twice: 3-colorability (ternary domain of  $\mathbf{B}$ ), 3SAT (binary domain of  $\mathbf{B}$ ).



# Conjunctive Query

A **Conjunctive Query** (CQ) is a query of the form:

$$Q(\mathbf{x}) = \exists \mathbf{y} (R_{j_1}(\mathbf{u}_1) \wedge R_{j_2}(\mathbf{u}_2) \wedge \dots)$$

We often write it in datalog notation, dropping  $\exists$ :

$$Q(\mathbf{x}) \leftarrow R_{j_1}(\mathbf{u}_1) \wedge R_{j_2}(\mathbf{u}_2) \wedge \dots$$

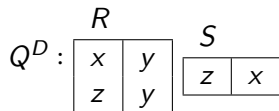
Each  $R_{j_i}(\mathbf{u}_i)$  is called an *atom*, or a *subgoal*.

## Homomorphism and CQ Evaluation

The **canonical database** of a Boolean CQ  $Q$ , denoted  $Q^D$ , is the following:

- Domain =  $\{x_1, \dots, x_n\}$  (all variables of  $Q$ )
- Relation  $R_j^{Q^D} =$  all atoms  $R_j(\mathbf{u})$  in  $Q$ .

E.g.:  $Q = R(x, y) \wedge R(z, y) \wedge S(z, x)$



CQ evaluation is the same as the homomorphism problem:

### Fact

For any structure (database)  $\mathbf{D}$ ,  $\mathbf{D} \models Q$  iff there exists a homomorphism  $Q^D \rightarrow \mathbf{D}$ .

# The Constraint Satisfaction Problem (CSP)

Fix a domain  $D$  and a set of *logical relations*,  $\mathbf{D} = (R_1^D, \dots, R_m^D)$ .

Fix  $n$  variables  $x_1, \dots, x_n$ .

A *constraint* is an expression  $R_j(x_{i_1}, \dots, x_{i_k})$ .

## Definition

A *Constraint Satisfaction Problem* is a set  $Q$  of constraints.

A solution is  $f : \{x_1, \dots, x_n\} \rightarrow D$  s.t. for every constraint  $R_j(x_{i_1}, \dots, x_{i_k})$ ,  $(f(x_{j_1}), \dots, f(x_{j_k})) \in R_j^D$ .

If  $D = \{0, 1\}$  then we call it a Boolean CSP.

# Examples

3-colorability.  $Q$  = the graph; logical relation =  $E^D$ :

red	green
red	blue
green	blue

3SAT is a CSP **in class**

# Homomorphism and the CSP

## Fact

*The CSP problem has a solution iff there exists a homomorphism  $Q \rightarrow D$ .*

The homomorphism goes from the problem  $Q$  to the logical relations  $D$ .

## Discussion

- CQ Evaluation and CSP are the same thing! And they are the same as the homomorphism problem:

$$f : \mathbf{A} \rightarrow \mathbf{B}$$

- But they look at different “sides”:
  - CSP: fix logical relations  $\mathbf{B}$ , the input is the problem  $\mathbf{A}$ .  
NP-hard in general.  
Schaefer’s dichotomy for Boolean CSP into PTIME v.s. NP-hard.
  - CQ: fix the query  $\mathbf{A}$ , the input is the database  $\mathbf{B}$ .  
Always in PTIME (data complexity).

# The Homomorphism Theorem for Containment of CQ

Consider Boolean queries only; extension to non-Boolean is straightforward.

## Theorem

Let  $Q_1, Q_2$  be CQ. The following are equivalent:

- $Q_1 \subseteq Q_2$
- There exists a homomorphism  $f : Q_2 \rightarrow Q_1$ .
- $Q_2$  is true on the canonical database given by  $Q_1$ .

Consequence:  $Q_1 \equiv Q_2$  iff there exists two homomorphisms  $Q_2 \rightarrow Q_1$  and  $Q_1 \rightarrow Q_2$ .

## Example

In class prove that  $Q_3 \subseteq Q_2 \equiv Q_1$ :

$$Q_1 \leftarrow E(x, y), E(z, y), E(z, u), E(u, v)$$

$$Q_2 \leftarrow E(r, s), E(s, t)$$

$$Q_3 \leftarrow E(a, b), E(b, c), E(c, d)$$



## CQ Query Minimization

A CQ  $Q$  is called *minimal* if:

forall  $Q'$ , if  $Q' \equiv Q$ , then  $Q'$  has at least as many atoms as  $Q$ .

### Theorem

*If  $Q \equiv Q'$  and both are minimal, then  $Q, Q'$  are isomorphic.*

Proof. Let  $f : Q \rightarrow Q'$ ,  $g : Q' \rightarrow Q$  be two homomorphisms.

Then  $g \circ f : Q \rightarrow Q$  is also a homomorphism.

Since  $Q$  is minimal,  $g \circ f$  must be surjective. *why?*

Since the body of  $Q$  is finite (has finitely many atoms),  $g \circ f$  is a bijection.

Hence both  $f, g$  are bijections, i.e. isomorphisms.

# The Minimization Procedure

Given  $Q$ , we want to find the (unique) minimal query  $Q_m$  s.t.  $Q \equiv Q_m$ .

(1) Start with  $Q' = Q$ .

(2) For each atom  $R_j$  of  $Q'$ , check if there exists a homomorphism  $f : Q' \rightarrow Q' - \{R_j\}$ ; if yes, then set  $Q' = Q' - \{R_j\}$  and continue.

(3) If no such  $R_j$  exists, then stop and return  $Q_m = Q'$ .

**Prove in class:** this procedure returns the unique minimal query equivalent to  $Q$ .

Note: the minimal query is always a subset of the atoms of  $Q$ !

## Discussion

- CQ query evaluation is CSP *from the other side*, and in PTIME.
- CQ query containment/equivalence is CSP *from both ends*, and NP-complete.
- To minimize  $Q$ , simply remove atoms one by one, in any order, until no other removal is possible.
- If  $G$  is a graph, then a **core** is a subgraph  $G_0 \subseteq G$  s.t. (a) there exists a homomorphism  $G \rightarrow G_0$ , and (b)  $G_0$  is smallest with this property.  
**is the core unique? how does one find it?**

# Clauses

A Knowledge Base (in AI) is often described by a collection of *clauses*:

$$C = \forall \mathbf{x} (L_1 \vee L_2 \vee \dots)$$

where each literal is some  $R(\mathbf{u})$  or  $\neg R(\mathbf{u})$ .

## Fact

*If  $C, C'$  are two positive clauses (w/o negation) then the implication problem  $C \rightarrow C'$  is decidable and co-NP complete.*

proof in class (reduction to CQ)

Note: this fact seems little known!

# Unions of Conjunctive Queries

A **Conjunctive Query** (CQ) is a query of the form:

$$Q(\mathbf{x}) = \exists \mathbf{y} (R_{j_1}(\mathbf{u}_1) \wedge R_{j_2}(\mathbf{u}_2) \wedge \dots)$$

A **Union of Conjunctive Queries** (UCQ) is a query of the form:

$$Q(\mathbf{x}) = Q_1(\mathbf{x}) \vee Q_2(\mathbf{x}) \vee \dots$$

where  $Q_1, Q_2, \dots$  are CQ's with the same free variables.

## Example

Equivalently, a UCQ is a non-recursive datalog program. Example:

$$P_1(x, y) \leftarrow E(x, y)$$

$$P_2(x, y) \leftarrow P_1(x, y)$$

$$P_3(x, y) \leftarrow P_2(x, y)$$

$$P_4(x, y) \leftarrow P_3(x, y)$$

$$Q(x, y) \leftarrow P_4(x, y)$$

$$P_2(x, y) \leftarrow P_1(x, z) \wedge P_1(z, y)$$

$$P_3(x, y) \leftarrow P_2(x, z) \wedge P_2(z, y)$$

$$P_4(x, y) \leftarrow P_3(x, z) \wedge P_3(z, y)$$

$$Q \leftarrow P_4(x, z) \wedge P_4(z, y)$$

How much larger is the UCQ compared to the datalog program?

## Containment for UCQ

We discuss Boolean queries only; non-Boolean queries are handled similarly, straightforwardly:

$$Q = Q_1 \vee Q_2 \vee \dots \vee Q_m$$
$$Q' = Q'_1 \vee Q'_2 \vee \dots \vee Q'_n$$

### Theorem

*$Q \subseteq Q'$  iff  $\forall i \exists j$  such that  $Q_i \subseteq Q'_j$ . Hence, containment of UCQ is NP-complete.*

Proof in class

# Minimizing UCQ

$$Q = Q_1 \vee Q_2 \vee \dots \vee Q_m$$

(1) Minimize each CQ  $Q_j$ .

(2) For all  $i$ , if there exists  $j$  s.t.  $Q_i \subseteq Q_j$ , then remove  $Q_i$ .

(3) The remaining query is minimal, and unique up to isomorphism. **proof in class**



## Domain-Independent Queries

$Q$  is called *domain-independent* if for any two structures  $\mathbf{D}, \mathbf{D}'$  with the same relations but different domains, we have  $Q(\mathbf{D}) = Q(\mathbf{D}')$ :

$$\mathbf{D} = (D, R_1^D, \dots, R_m^D)$$

$$\mathbf{D}' = (D', R_1^D, \dots, R_m^D)$$

Which queries are domain independent?

$$\exists x \exists y R(x, y)$$

$$\exists x \exists y \neg R(x, y)$$

$$\exists x \exists y (R(x) \wedge \neg S(x, y))$$

$$\exists x \exists y (R(x) \wedge \neg S(x, y) \wedge T(y))$$

$$\forall y S(y)$$

$$\forall x \forall y (R(x, y) \rightarrow S(y))$$

In databases we consider **only** domain-independent queries.

Checking if  $Q$  is domain independent is undecidable in general **why?**

# Monotone Queries

Two structures are contained,  $\mathbf{A} \subseteq \mathbf{B}$ , if the domains and all their relations are contained:  $A \subseteq B, R_j^A \subseteq R_j^B, j = 1, m$ .

A query  $Q$  is **monotone** if  $\mathbf{A} \subseteq \mathbf{B}$  implies  $Q(\mathbf{A}) \subseteq Q(\mathbf{B})$

Checking if  $Q$  is monotone is undecidable in general **why?**

## More Query Languages

The languages  $CQ^{<}$ ,  $CQ^{\neg}$ ,  $CQ^{<,\neg}$  extend CQ with  $<$  or  $\neg$  respectively; similarly UCQ.

Examples **to which language do they belong?**

$$\exists y \exists z \text{Friend}(x, y) \wedge \text{Friend}(y, z) \wedge \text{Boss}(z)$$

$$\exists y \exists z \text{Friend}(x, y) \wedge \text{Friend}(y, z) \wedge \neg \text{Boss}(z)$$

$$\exists y \exists z \text{Friend}(x, y) \wedge \text{Friend}(y, z) \wedge \text{Boss}(z) \wedge x < z$$

**In class** do we need  $=$  in CQ, i.e.  $CQ^{=}$ ?

# Summary of Query Languages

Syntax	FO fragment	Domain independent?	Monotone?
CQ	$FO(\exists, \wedge)$	yes	yes
$CQ^<$	$FO(\exists, \wedge, <)$	yes	yes
$CQ^\neg$	$FO(\exists, \wedge, \neg)$ (Negation Normal Form)	no	no
UCQ	$FO(\exists, \vee, \wedge)$	yes	yes
$UCQ^<$	$FO(\exists, \vee, \wedge, <)$	yes	yes
$UCQ^\neg$	$FO(\exists, \vee, \wedge, \neg)$ (Negation Normal Form)	no	no

# Decidability

## Theorem

*The containment problem for UCQ<sup><,¬</sup> is decidable.*

Proof: consider Boolean queries only.

Any UCQ<sup><,¬</sup> query can be written as  $\exists \mathbf{x} \varphi(\mathbf{x})$ . Then:

$$\begin{aligned}
 Q_1 \subseteq Q_2 & \quad \text{iff } \models \exists \mathbf{x} \varphi_1(\mathbf{x}) \rightarrow \exists \mathbf{y} \varphi_2(\mathbf{y}) \\
 & \quad \text{iff } \models (\neg \exists \mathbf{x} \varphi_1(\mathbf{x})) \vee (\exists \mathbf{y} \varphi_2(\mathbf{y})) \\
 & \quad \text{iff } \models (\forall \mathbf{x} \neg \varphi_1(\mathbf{x})) \vee (\exists \mathbf{y} \varphi_2(\mathbf{y})) \\
 & \quad \text{iff } \models \forall \mathbf{x} \exists \mathbf{y} (\neg \varphi_1(\mathbf{x}) \vee \varphi_2(\mathbf{y}))
 \end{aligned}$$

The latter is the negation of a Bernays-Schönfinkel formula  $\exists^* \forall^*$ , hence validity is decidable.

## Containment Procedure for $CQ^<$

Main idea: it is insufficient to treat  $<$  as any other predicate.

$$Q_1 = R(x, y) \wedge R(y, z) \wedge x < z \qquad Q_2 = R(u, v), u < v$$

Then  $Q_1 \subseteq Q_2$  **why?** yet there is no homomorphism  $Q_2 \rightarrow Q_1$  that maps  $u < v$  to some  $<$ -atom.

Solution: expand  $Q_1$  by considering all linear orders of variables

$$Q_{11} = R(x, y) \wedge R(y, z) \wedge y < x < z \quad Q_{12} = R(x, y) \wedge R(y, z) \wedge x = y < z \quad Q_{13}$$

$$Q_{14} = R(x, y) \wedge R(y, z) \wedge x < y = z \quad Q_{15} = R(x, y) \wedge R(y, z) \wedge x < z < y$$

Prove in class:  $Q_1 \equiv Q_{11} \vee \dots \vee Q_{15} \subseteq Q_2$ .

### Theorem

*The Containment problem for  $CQ^<$  (and for  $UCQ^<$ ) is  $\Pi_2^P$ -complete.*

## Negation

Once we add negation, a query may not be domain independent.  
 Problem: the abbreviated syntax suggests two interpretations. E.g.

$$Q \leftarrow R(x, y) \wedge \neg S(y, z)$$

Interpretation 1:  $\exists x \exists y \exists z (R(x, y) \wedge \neg S(y, z))$

Interpretation 2: the result of this datalog program:

$$\begin{aligned} \text{Not}S(y) &\leftarrow S(y, z) \\ Q &\leftarrow R(x, y) \wedge \text{Not}S(y) \end{aligned}$$

Note: this means  $\exists x \exists y \forall z (R(x, y) \wedge \neg S(y, z))$

# Negation: Interpretation 1

## Theorem

*Containment of  $CQ^\neg$  queries under interpretation 1 is  $\Pi_2^P$  complete.*

Curiously, I could never find a published proof!



## Negation: Interpretation 2

### Theorem

*Containment of  $CQ^-$  queries under interpretation 2 is undecidable.*

## Discussion

- Containment of  $FO(\exists, \vee, \wedge)$  is decidable (and in  $\Pi_2^P$ ) because of Bernays-Schönfinkel.
- Better complexities (meaning NP) for various fragments.
- Checking containment  $Q_1 \subseteq Q_2$  is related to query evaluation of  $Q_2$  on some database(s) derived from  $Q_1$ .
- All results discussed here carry over to implication of universally quantified clauses. **seems little known in the AI community**