The homomorphism lattice of finite structures, unique characterization, and exact learnability

Balder ten Cate

A|C seminar
Oct 6, 2021

About me

Career path:

ILLC → lvl → INRIA / ENS de Cachan → UC Santa Cruz → LogicBlox → Google → ILLC

About me

Career path:

ILLC → IvI → INRIA / ENS de Cachan → UC Santa Cruz → LogicBlox → Google → ILLC

Past research topics:

- Extended modal languages; well-behaved fragments of first-order logic and fixpoint logic
- XML query languages; automata and logics for finite trees
- Data management and data interoperability (schema mappings, access constraints)
- Knowledge representation and reasoning (ontology-based data access)
- Declarative languages for data analysis and machine learning
- And, conversely, techniques for learning declarative representations

European reintegration fellowship LLAMA (Logic and Learning - an Algebraic and Model-theoretic Approach)

Outline

1. Conjunctive queries (CQs)
2. Main technical content: Homomorphism lattice of finite structures
3. Applications of the above to some problems regarding learning CQs (and other declarative specification languages)

Main reference:

- Balder ten Cate, Victor Dalmau (2020). **Conjunctive Queries: Unique Characterizations and Exact Learnability**. arxiv.org/abs/2008.06824

Conjunctive Queries

A **conjunctive query (CQ)** is a positive-existential-conjunctive FO formula (that is, CQs form the fragment of FO that uses only $\exists$ and $\wedge$)

A **union of conjunctive queries (UCQ)** is a disjunction of CQs (that is, UCQs form the fragment of FO that uses only $\exists$, $\wedge$, and $\vee$ --- suitably normalized).

Conjunctive Queries

A **conjunctive query (CQ)** is a positive-existential-conjunctive FO formula (that is, CQs form the fragment of FO that uses only $\exists$ and $\wedge$)

A **union of conjunctive queries (UCQ)** is a disjunction of CQs (that is, UCQs form the fragment of FO that uses only $\exists$, $\wedge$, and $\vee$ --- suitably normalized).

Why are CQs (and UCQs) interesting?

- CQs are of fundamental importance in databases as they capture the core SELECT-FROM-WHERE construct in SQL.
- They arise naturally in numerous other areas of computer science as well.

Brief digression

Recall that

- **Modal logic** can be viewed as a fragment of FO, and
- In fact, it is the **bisimulation-invariant fragment** of FO
    - (“Van Benthem Characterization Theorem”)

Similarly,

- **UCQ** is a fragment of FO logic, and
- In fact, it is the **homomorphism-preserved fragment** of FO
    - (“Homomorphism Preservation Theorem”)

Brief digression

Recall that

- **Modal logic** can be viewed as a fragment of FO, and
- In fact, it is the **bisimulation-invariant fragment** of FO
    - (“Van Benthem Characterization Theorem”)

Similarly,

- **UCQ** is a fragment of FO logic, and
- In fact, it is the **homomorphism-preserved fragment** of FO
    - (“Homomorphism Preservation Theorem”)

A **homomorphism** is a map $h : dom(A) \rightarrow dom(B)$ that preserves structure (more precise definition will come later in this talk).

Brief digression (ct’d)

Many classic results of model theory “*fail in the finite*” (restricted to finite structures)

- Examples: Compactness, Craig interpolation, Los-Tarski Theorem, ...
- Failure-in-the-finite appears to be the rule rather than the exception.

What is an example of a model-theoretic result that *does* hold in the finite?

Brief digression (ct’d)

Many classic results of model theory “*fail in the finite*” (restricted to finite structures)

- Examples: Compactness, Craig interpolation, Łoś–Tarski Theorem, ...
- Failure-in-the-finite appears to be the rule rather than the exception.

What is an example of a model-theoretic result that *does* hold in the finite?

1. Van Benthem Characterization Theorem (as shown by Rosen, 1997)
2. Homomorphism Preservation Theorem (as shown by Rossman, 2008)

World of modal logic	World of database theory
Modal logic	UCQs
Modal mu-calculus	Monadic Datalog
Bisimulations	Homomorphisms

World of modal logic

World of database theory

UN-FO
("Unary Negation fragment")

Modal logic

UCQs

UN-FP

Modal mu-calculus

Monadic Datalog

UN-bisimulations

Bisimulations

Homomorphisms

World of modal logic

World of database theory

GN-FO
("Guarded Negation fragment")

Guarded fragment

UCQs

GN-FP

Guarded fixpoint logic

Monadic Datalog

GN-bisimulations

Guarded bisimulations

Homomorphisms

End of the Brief digression

(Ask me later if you are interested in this.)

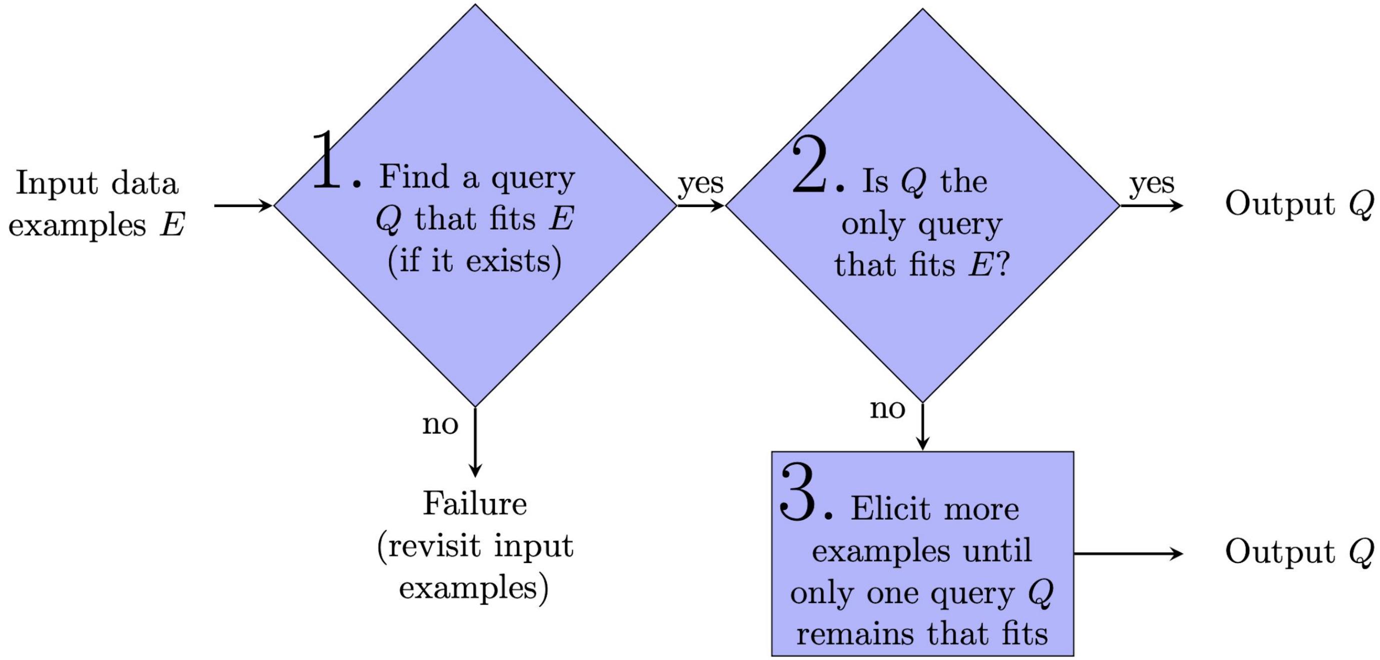
Example-Driven Query Discovery

Scenario: *we are trying to construct a database query from data examples.*

Example-Driven Query Discovery

Scenario: *we are trying to construct a database query from data examples.*

- The database schema **S** and the arity *k* of the query are known.
- A *data example* is a triple: (I, **a**, s)
  - I is a database instance (for schema S)
  - **a** is a k-tuple
  - s in {+, -} indicates if this is a positive or a negative example.
- A query Q …
  - … *fits* data example (I,**a**,+) if **a** $\in$ Q(I)
  - … *fits* data example (I,**a**, -) if **a** $\notin$ Q(I)


Input data examples E
1. Find a query Q that fits E (if it exists)
yes
2. Is Q the only query that fits E?
yes
Output Q
no
Failure (revisit input examples)
no
3. Elicit more examples until only one query Q remains that fits
Output Q

Remainder of this talk

1. Some aspects of the Homomorphism Lattice of Finite Structures
2. Applications to Example-Driven Query Discovery for CQs.
3. (Time permitting:) Schema Mappings and Description Logics

Structures and Homomorphisms

Fix a finite schema **S** consisting of relation symbols and constants symbols.

By a *structure* we will mean a finite relational structure over **S**.

A *homomorphism* from A to B is a function h: dom(A) → dom(B) such that

1. For every fact of A, its image (under h) is a fact of B.
2. $h(c^A) = c^B$ for each constant symbol c

The Homomorphism Lattice

The *Homomorphism Lattice* is the partial order (Str, $\leq_{hom}$)

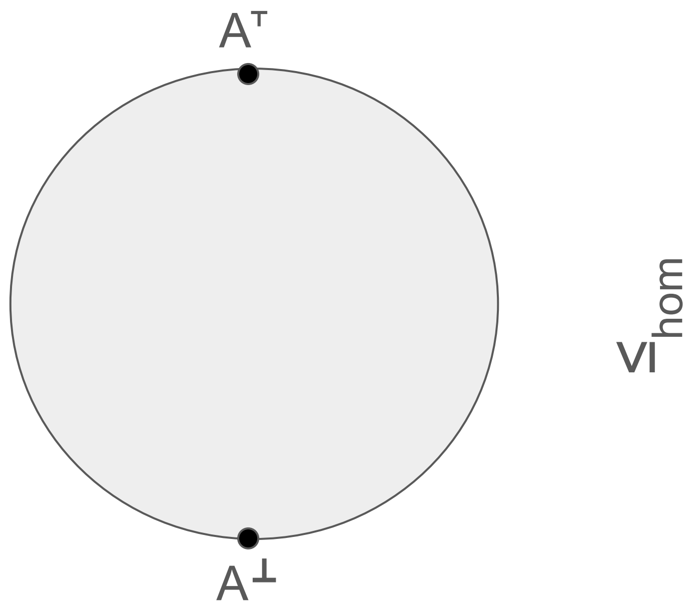
- Str is the set of all structures
- $A \leq_{hom} B$ if there is a homomorphism from A to B.

The Homomorphism Lattice

The *Homomorphism Lattice* is the partial order (Str, $\leq_{hom}$)

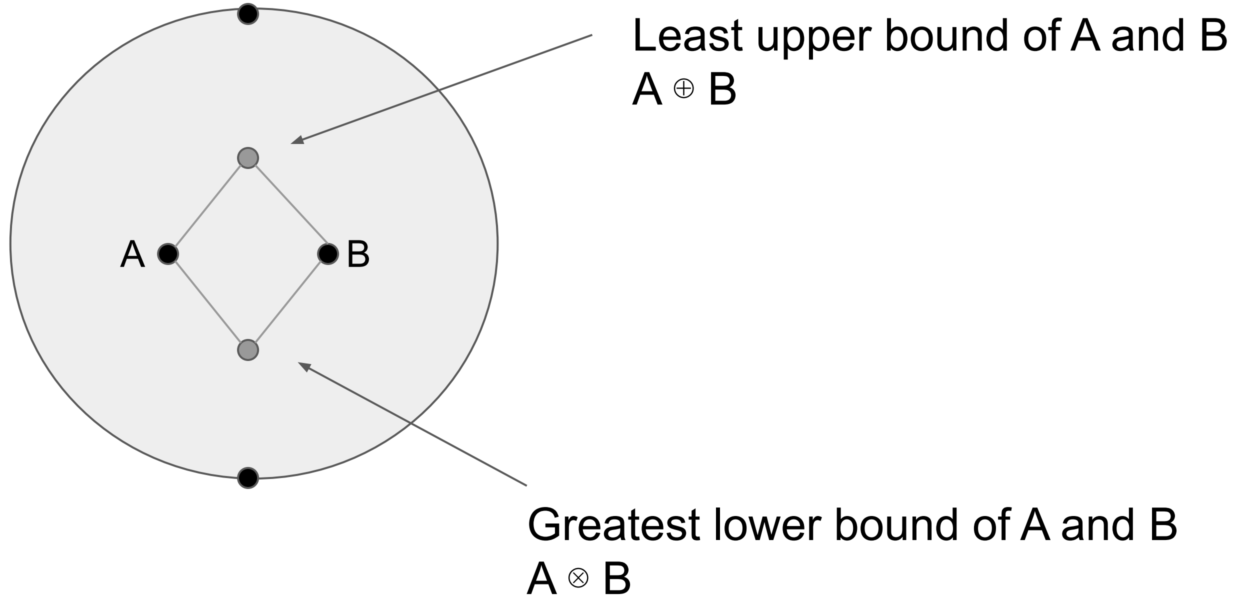
- Str is the set of all structures
- A $\leq_{hom}$ B if there is a homomorphism from A to B.

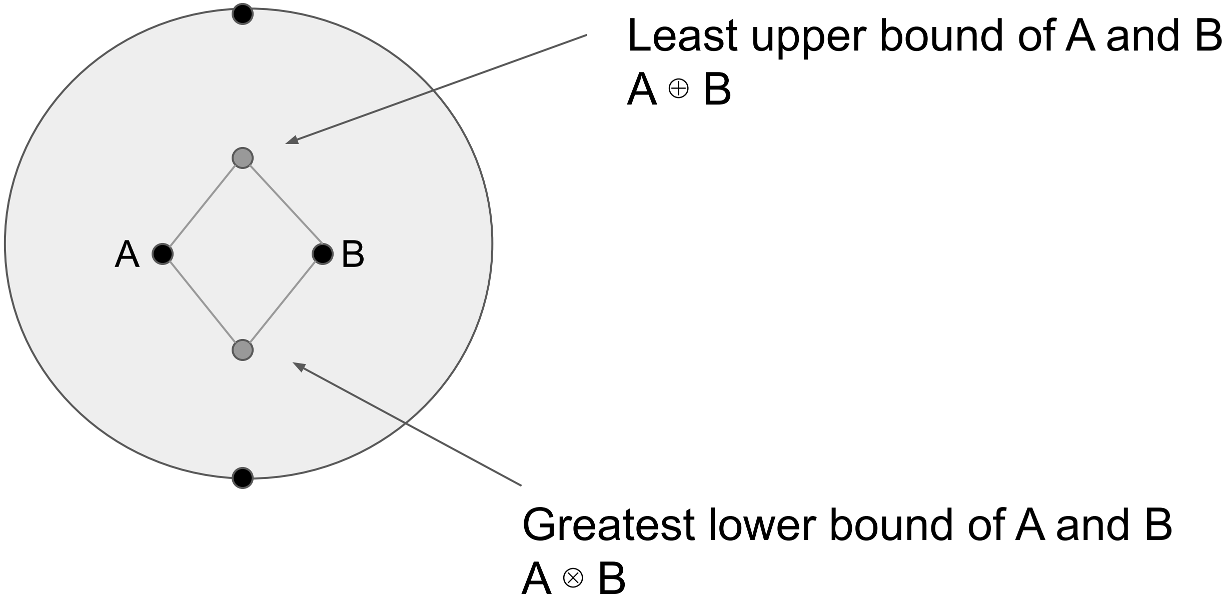
Two structures are *homomorphically equivalent* if A $\leq_{hom}$ B and B $\leq_{hom}$ A.
We will not distinguish between homomorphically equivalent structures.



$A^{\perp}$: structure with no facts; each constant symbol denotes a different element.

$A^{\top}$: single-element structure (all constants denote the same element); all facts.


Least upper bound of A and B
A ⊕ B
A
B
Greatest lower bound of A and B
A ⊗ B



$A \otimes B$ = direct product of A and B

$A \oplus B$ ~= disjoint union of A and B

(in the case with constants the precise construction is a bit more tedious)

The Homomorphism Lattice

Algebra (Str, $\oplus$, $\otimes$, $\top$, $\bot$, …)

The Homomorphism Lattice

Algebra (Str, $\oplus$, $\otimes$, $\top$, $\bot$, …)

One more relevant operation:

for every A and B there is a structure A^B satisfying:

$$C \leq_{hom} A^B \quad \text{iff} \quad (C \otimes B) \leq_{hom} A$$

Small digression

Product Homomorphism Problem (PHP):

> *Given A_1, …., A_n and B, decide whether $(A_1 \otimes \ldots \otimes A_n) \leq_{hom} B$*

Theorem (Willard 2010; tC and Dalmau 2015) :

PHP is NExpTime-complete (even for a fixed schema).

Density

Let **S** = { R } with R a binary relation. Let A_1 = { R(a,b) } and let A_2 = { R(a,b) , R(b,a) }.

Fact 1: $A^{\perp} <_{hom} A_1 <_{hom} A_2$

Density

Let **S** = { R } with R a binary relation. Let A_1 = { R(a,b) } and let A_2 = { R(a,b) , R(b,a) }.

Fact 1: $A^{\perp} <_{hom} A_1 <_{hom} A_2$

Fact 2: There is *no* B such that $A^{\perp} < B < A_1$.

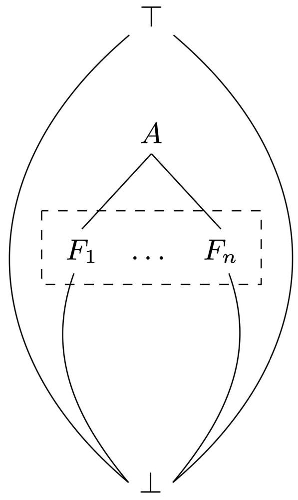
Fact 3: For every $B <_{hom} A_2$ there is a structure B’ such that $B <_{hom} B' <_{hom} A_2$.

Follows from an extension (due to Nesetril and Rodl 1989) of Erdos (1959)’s celebrated theorem on the existence of graphs of high girth and chromatic number

Frontiers

Definition; a *frontier* for A is a finite set of structures $\{ F_1, \ldots, F_n \}$, such that

1. each $F_i <_{hom} A$, and
2. whenever $B <_{hom} A$ then $B \leq_{hom} F_i$ for some F_i



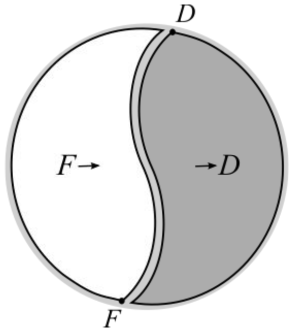
The frontier separates A from the structures strictly below A.

Not every structure has a frontier. A_2 has no frontier.

Dualities

Def 1: A pair of structure (F,D) is a *duality pair* if:

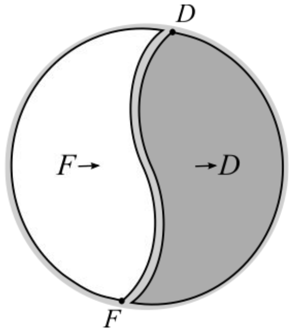
$$\{A \mid F \leq_{hom} A\} = \{A \mid A \nleq_{hom} D)$$



Dualities

Def 1: A pair of structure (F,D) is a *duality pair* if:

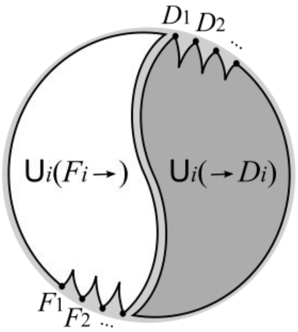
$\{A \mid F \leq_{hom} A\} = \{A \mid A \nleq_{hom} D)$



Def 2: A pair of finite sets $(\{F_1, \ldots, F_n\}, \{D_1, \ldots, D_m\})$

is a *generalized homomorphism duality* if:

$\{A \mid F_i \leq_{hom} A \text{ for some } i \leq n\} = \{A \mid A \nleq_{hom} D_i \text{ for all } i \leq m\}$



Example 1 (duality pair)

Let

- P_{k+1} be the directed path with k+1 elements.
- T_k be the linear order with k elements.

Gallai-Hasse-Roy-Vitaver Theorem (~1965) for directed graphs:

(P_{k+1}, T_k) is a duality pair,

i.e., for every directed graph G, it holds that $P_{k+1} \rightarrow G$ if and only if $G \not\rightarrow T_k$

Example 2 (infinitary homomorphism duality)

A graph is 2-colorable if and only if it does not have a cycle of odd length.

$G \to \mathbf{K}_2$ if and only if $\mathbf{C}_{2n+1} \not\to G$ for all n

where $\mathbf{K}_2$ is the 2-element clique and $\mathbf{C}_n$ is the a cycle of length n.

Frontiers and Dualities

Theorem (from Nesetril and Tardiff 2000, cf. also tC and Dalmau 2020):

- If $\{B_1, \ldots, B_n\}$ is a frontier for A, then

  $(\{A\}, \{ A^{B1} , \ldots, A^{Bn} \})$ is a generalized homomorphism duality

- If $(\{A\},\{B_1, \ldots, B_n\})$ is a generalized homomorphism duality,

  then $\{(A \otimes B_1), \ldots., (A \otimes B_1)\}$ is a frontier for A

(*Statement assumes no constants but naturally extends to the case with constants)

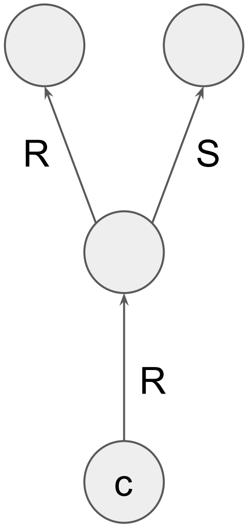
C-acyclicity

A structure is *c-acyclic* if it does not have cycles except for cycles that involve an element named by a constant symbol.

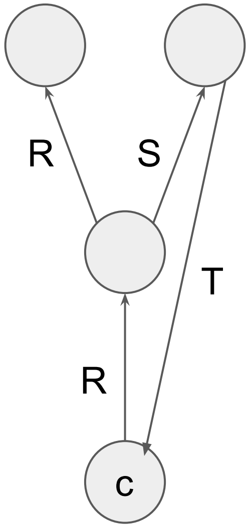
More precise definition:

- The *incidence graph* of a structure A is the bi-partite multi-graph where
  - The nodes of the graph are the elements and facts of A
  - There is an edge between an element and a fact if the element occurs in the fact.
  - If an element occurs multiple times in a fact, each occurrence generates an edge.
- A structure is *c-acyclic* if every cycle in the incidence graph goes through at least one element named by a constant symbol.

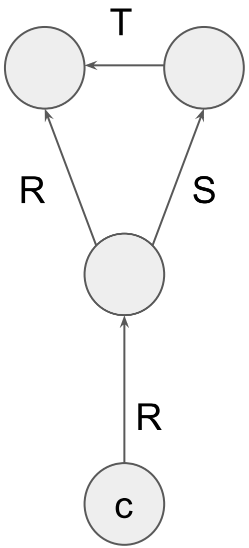
Examples



acyclic



c-acyclic



cyclic

Frontiers: existence and how to construct them

Thm 1: For all structures A, the following are equivalent:

1. A has a frontier
2. A is homomorphically equivalent to a c-acyclic structure.

Thm 2: For c-acyclic structures, a frontier can be computed in polynomial time.

Thm 3: Testing whether a given set of structures { F1, …., Fn } is a frontier for a structure A is NP-complete.

Foniok, Nesetril and Tardiff (2008), tC and Dalmau (2020)

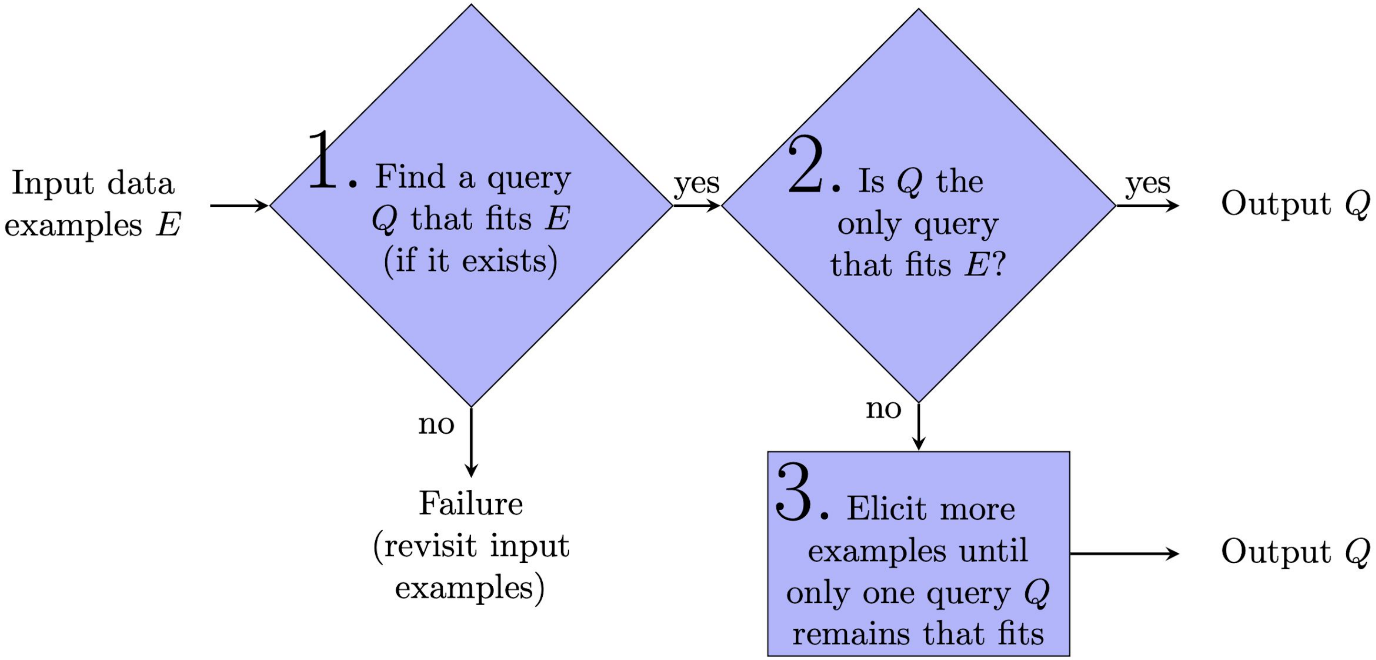
Some further results on frontiers

The class of c-acyclic structures is not "*frontier-closed*" (although every c-acyclic structure has a frontier, it does not necessarily consist of c-acyclic structures).

Thm (tC and Dalmau 2020). The class of acyclic connected structures (with a single constant) is polynomial-time frontier-closed.

Thm (from Nešetřil and Ossona de Mendez 2008; cf. tC and Dalmau 2020): Every class of structures that has bounded expansion admits relatized frontiers.

Example-Driven Query Discovery



Conjunctive Queries

Let's restrict attention to conjunctive queries (CQs).

- Every k-ary CQs over a relational schema **S** can be equivalently viewed as a structure over schema $\mathbf{S} \cup \{c_1, .., c_k\}$.
- Every example corresponds to a structure over $\mathbf{S} \cup \{c_1 \ldots c_k\}$ as well.
- Q fits a (I,**a**, **+**) iff there is a homomorphism from q(**x**) to (I,**a**)
- Q fits a (I,**a**, **-**) iff there is no homomorphism from q(**x**) to (I,**a**)

This sets the stage for us to apply results about the homomorphism lattice.

1. The Fitting Problem

Fitting problem:

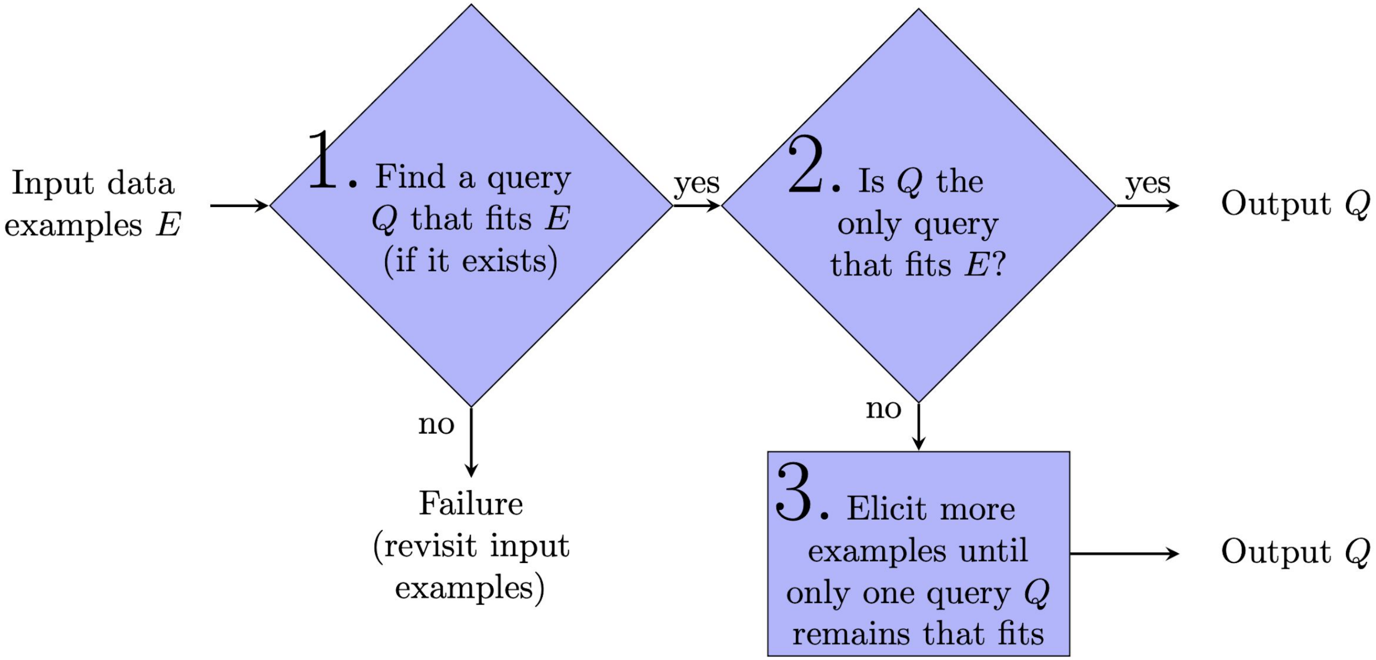
> Input: a finite set E of data examples.
> Decide whether there is a CQ that fits all data examples in E.

Lemma: let Q* be the direct product of the positive examples in E. The following are equivalent:

1. There exists a CQ that fits the data examples in E.
2. Q* fits the data examples in E.
3. Q* does not have a homomorphism to any negative example in E.
4. Q* is the *most-specific fitting CQ* for the data examples in E.

Theorem: the fitting problem is coNExpTime-complete. (by reduction from PHP)

Example-Driven Query Discovery



2. The Uniqueness Problem

Definition: E *uniquely characterizes* Q if Q is the only CQ (up to logical equivalence) that fits the examples in E.

Unique characterization problem:

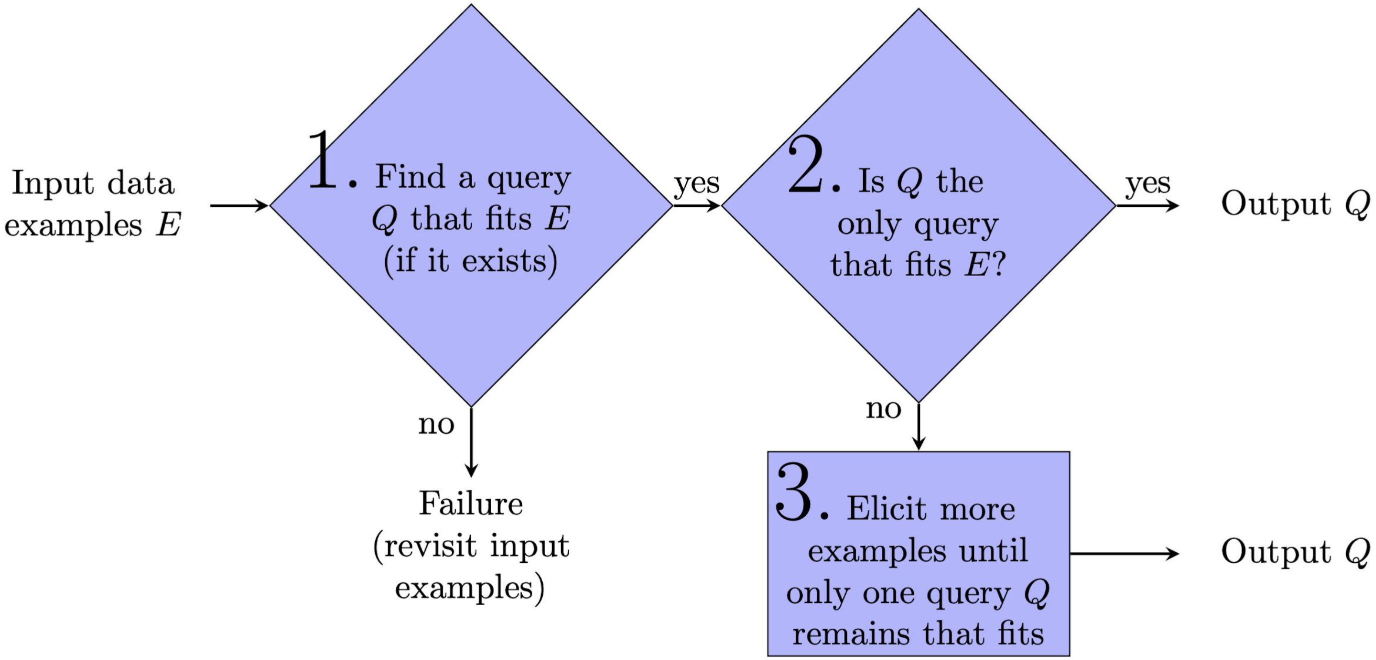
> Input: a finite set of data examples E and a most-specific-fitting CQ Q for E. Decide if E uniquely characterizes Q.

Lemma: The following are equivalent:

1. E uniquely characterizes Q
2. The negative examples in E form a frontier for Q.

Theorem: the unique characterization problem is NP-complete.

Example-Driven Query Discovery



3. Eliciting Further Examples

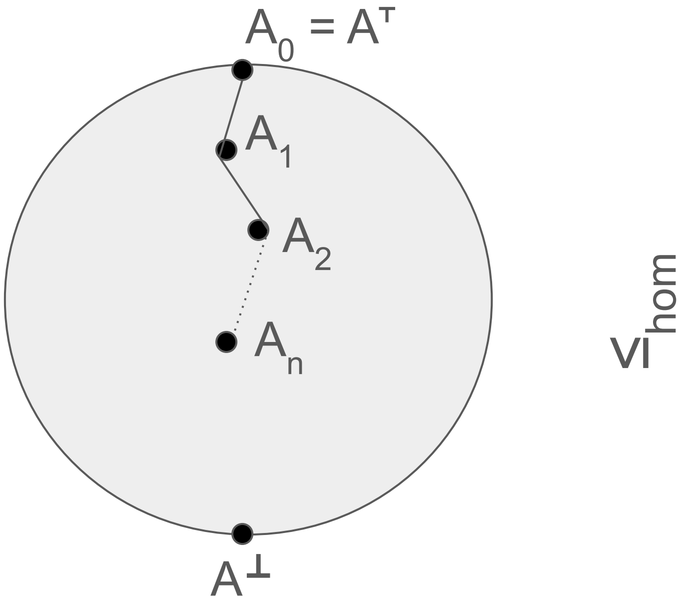
Will finitely many examples even ever be enough to uniquely characterize the target CQ?

Theorem 1: for all CQs Q, the following are equivalent

1. Q is uniquely characterized by a finite set of data examples
2. Q has a frontier.
3. Q is logically equivalent to a c-acyclic CQ.

Theorem 2: the class of c-acyclic CQs is efficiently exactly learnable with membership queries.

Theorem 2 means that, if the “target CQ” is c-acyclic, then there is a PTIME algorithm that, after asking for the label of one or more examples, terminates and identifies it correctly (modulo logical equivalence).



$A_0 = A^\top$
A_1
A_2
A_n
$A^\perp$
$\leq_{hom}$

Unions of Conjunctive Queries

The situation for UCQs is a little different:

- Unique characterizations for UCQs ~ generalized homomorphism dualities
- A UCQ is uniquely characterized by a finite set of examples iff if is logically equivalent to a c-acyclic UCQ.
- For c-acyclic UCQs, a uniquely characterizing set of examples can be effectively constructed but (provably) not in polynomial time.
- The class of c-acyclic UCQs is *not* efficiently exactly learnable with membership queries (but is efficiently exactly learnable with membership and equivalence queries).

Schema Mappings

A schema mapping M=(**S**,**T**,Σ) is a high-level declarative specifications of the relationships between two database schemas.

Two of the most well-studied schema mapping specification languages are LAV ("Local-as-View") and GAV ("Global-as-View") schema mappings.

Schema Mappings

A schema mapping M=(**S**,**T**,Σ) is a high-level declarative specifications of the relationships between two database schemas.

Two of the most well-studied schema mapping specification languages are LAV (“Local-as-View”) and GAV (“Global-as-View”) schema mappings.

Alexe, tC, Kolaitis, and Tan (2011) studied the question when a schema mapping be uniquely characterized by a finite set of data examples.

tC, Dalmau, and Kolaitis (2013) studied efficient exact learnability for GAV schema mappings.

Schema Mappings

Corr. Fix a source schema S and a target schema T .

1. A LAV schema mapping M=(**S**,**T**,Σ) is uniquely characterizable by a finite set of positive and negative schema-mapping examples if and only if M is logically equivalent to a c-acyclic LAV schema mapping.
2. If M is c-acyclic, then a uniquely characterizing set of positive and negative schema-mapping examples can be constructed in PTIME.
3. The class of c-acyclic LAV schema mappings over S, T is efficiently exactly learnable with membership queries.

Question: what happens in the presence of integrity constraints?

Description logics

Description logics (DLs) are formal specification languages used to represent domain knowledge.

The DL concept language *ELI* can be characterized as a notational variant of acyclic connected unary CQs (over a schema with unary and binary relations only).

For example, the *ELI* concept

$$\text{logician} \sqcap \exists \text{HasParent}. \exists \text{HasParent.logician}$$

corresponds to $q(x) = \exists y,z\ .\ (\text{logician}(x) \land \text{HasParent}(x,y) \land \text{HasParent}(y,z) \land \text{logician}(z))$

Description logics

Corr.

1. Every ELI concept expression is uniquely characterizable by a finite collection of positive and negative QA examples, which is PTIME-computable.
2. Furthermore, the class of ELI concept expressions is efficiently exactly learnable with membership queries.

Question: does this hold also in the presence of an ontology (background theory)?

One recent result

Assume a schema **S** consisting of unary and binary relations only.

A *DL-Lite ontology* is a FO theory consisting of

- Inclusion constraints $\alpha(x) \rightarrow \beta(x)$ and/or
- Disjointness constraints of the form $\alpha(x) \rightarrow \neg\beta(x)$, $R(x,y) \rightarrow \neg S(x,y)$, and/or $R(x,y) \rightarrow \neg S(y,x)$

where $\alpha(x)$, $\beta(x)$ are unary projections of relations, i.e., of the form $P(x)$, $\exists y.R(x,y)$ or $\exists y.R(y,x)$

Thm (Funk, Jung and Lutz, DL 2021): Acyclic, connected, unary CQs over **S** (in other words, ELI description logic concepts) are efficiently exactly learnable in the presence of DL-Lite ontologies.

Outlook (LLAMA)

- Unique characterizations and exact learnability for modal formulas
- Unique characterizations and exact learnability in the presence of a background theory

Collaboration with Raoul Koudijs, … (room for more collaborations!)

Thank you!