Constraint Satisfaction

and

Logic

Phokion G. Kolaitis

IBM Almaden Research Center

and

UC Santa Cruz

Tutorial Outline

Part I: Queries and Logics

- Queries & Definability of Queries
- First-Order Logic, Existential Second Logic
- Combined, Expression, and Data Complexity

Part II: Logic and CSP Problems

- Conjunctive Queries
- The Chandra-Merlin Theorem
- MMSNP & its extensions.

Part III: Logic and Tractability of CSP

- First-Order Logic and CSP
- Datalog
- Finite-Variable Logics and Pebble Games

Basic Concepts

Definitions:

- *Vocabulary* σ: a set $\sigma = \{R'_1, \ldots, R'_m\}$ of relation symbols of specified arities.
- σ*-structure* $\mathbf{A} = (A, R_1, \ldots, R_m)$:

  a non-empty set A and relations on A such that $\text{arity}(R_i) = \text{arity}(R'_i)$, $1 \leq i \leq m$.
- *Finite* σ*-structure* $\mathbf{A}$: universe A is finite

Examples:

- *Graph:* $\mathbf{G} = (V, E)$, where E is binary.
- *String:* $\mathbf{S} = (\{1, 2, \ldots, n\}, P)$, where P is unary

  $$m \in P \iff \text{the } m\text{-th bit of the string is 1.}$$

  – string 10001 encoded as $(\{1, 2, 3, 4, 5\}, \{1, 5\})$

Basic Concepts

Example: 3-CNF formulas as finite structures

Every 3-CNF formula can be viewed as a finite structure of the form $\mathbf{A} = (A, R_0, R_1, R_2, R_3)$, where each R_i is a ternary relation.

- 3-CNF formula φ with variables $x_1, \ldots, x_n$
- Structure $\mathbf{A}^\varphi = (\{x_1, \ldots, x_n\}, R_0^\varphi, R_1^\varphi, R_2^\varphi, R_3^\varphi)$, where

$$
\begin{aligned}
R_0^\varphi &= \{(x, y, z) : (x \vee y \vee z) \text{ is a clause of } \varphi\} \\
R_1^\varphi &= \{(x, y, z) : (\neg x \vee y \vee z) \text{ is a clause of } \varphi\} \\
R_2^\varphi &= \{(x, y, z) : (\neg x \vee \neg y \vee z) \text{ is a clause of } \varphi\} \\
R_3^\varphi &= \{(x, y, z) : (\neg x \vee \neg y \vee \neg z) \text{ is a clause of } \varphi\}
\end{aligned}
$$

Queries

Definitions:

- *Class $\mathcal{C}$ of structures:* a collection of relational σ-structures closed under isomorphisms.

- *k-ary Query Q on $\mathcal{C}$:*

  a mapping Q with domain $\mathcal{C}$ and such that

  - $Q(\mathbf{A})$ is a k-ary relation on $\mathbf{A}$, for $\mathbf{A} \in \mathcal{C}$;
  - Q is *preserved under isomorphisms*, i.e., if $h : \mathbf{A} \rightarrow \mathbf{B}$ is an isomorphism, then

  $$Q(\mathbf{B}) \;=\; h(Q(\mathbf{A})).$$

- *Boolean Query Q on $\mathcal{C}$:*

  a mapping $Q : \mathcal{C} \rightarrow \{0, 1\}$ preserved under isomorphisms. Thus, Q can be identified with the subclass $\mathcal{C}'$ of $\mathcal{C}$, where

  $$\mathcal{C}' \;=\; \{\mathbf{A} \in \mathcal{C} : \; Q(\mathbf{A}) = 1\}.$$

Examples of Queries

- Path of Length 2: $P2$

  Binary query on graphs $\mathbf{H} = (V, E)$ such that

  $P2(\mathbf{H}) = \{(a, b) \in V^2$: there is a path of length 2 fr

- s-t Connectivity: TC

  Binary query on graphs $\mathbf{H} = (V, E)$ such that

  $TC(\mathbf{H}) = \{(a, b) \in V^2$: there is a path from s to $t\}$.

- Connectivity CN:

  Boolean query on graphs $\mathbf{H} = (V, E)$ such that

$$CN(\mathbf{H}) = \begin{cases} 1 & \text{if } \mathbf{H} \text{ is connected} \\ 0 & \text{otherwise.} \end{cases}$$

- k-Colorability $k \geq 2$

- 3-Sat (with formulas viewed as structures)

Definability of Queries

Let L be a logic and $\mathcal{C}$ a class of structures

- A k-ary query Q on $\mathcal{C}$ is *L-definable* if there is an L-formula $\varphi(x_1, \ldots, x_k)$ with $x_1, \ldots, x_k$ as free variables and such that for every $\mathbf{A} \in \mathcal{C}$

  $$Q(\mathbf{A}) = \{(a_1, \ldots, a_k) \in A^k : \mathbf{A} \models \varphi(a_1, \ldots, a_k)\}.$$

- A Boolean query Q on $\mathcal{C}$ is *L-definable* if there is an L-sentence ψ such that for every $\mathbf{A} \in \mathcal{C}$

  $$Q(\mathbf{A}) = 1 \iff \mathbf{A} \models \psi.$$

First-Order & Second-Order Logic

- **First-Order Logic** FO (on graphs):
  - *first-order variables:* x, y, z, ...
  - *atomic formulas:* $E(x, y)$, $x = y$
  - *formulas:* atomic formulas + connectives + first-order quantifiers $\exists x, \forall x, \exists y, \forall y, \ldots$ that range over the nodes of the graph.
- **Second-Order Logic** SO:

  First-order logic + second-order quantifiers $\exists S$, $\forall S$, $\exists T$, $\forall T$, ... ranging over relations of specified arities on the universe of structures.
- **Existential Second-Order Logic** ESO:

  $(\exists S_1) \cdots (\exists S_m) \varphi(\overline{x}, S_1, \ldots, S_m)$, where φ is FO.
- **Universal Second-Order Logic** USO:

  $(\forall S_1) \cdots (\forall S_m) \varphi(\overline{x}, S_1, \ldots, S_m)$, where φ is FO.

First-Order Definability

Example: On the class $\mathcal{G}$ of finite graphs

- The query PATH OF LENGTH 2 is FO-definable

  $$P2(\mathbf{H}) \;=\; \{(a,b) \in V^2 : \mathbf{H} \models \exists z(E(a,z) \wedge E(z,b))\}.$$

- The queries TRANSITIVE CLOSURE, CONNECTIVITY, k-COLORABILITY, $k \geq 2$, are **not** FO-definable.

Example: On the class of all finite structures with 4 ternary relations:

The query 3-SAT is **not** first-order definable.

Note: Results about non-definability in FO-logic can be proved using Ehrenfeucht-Fraïssé Games.

Second-Order Definability

Fact: The queries DISCONNECTIVITY, k-COLORABILITY, 3-SAT are ESO-definable.

- DISCONNECTIVITY:

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

$$(\forall z \forall w (S(z) \wedge \neg S(w) \rightarrow \neg E(z,w))).$$

- 2-COLORABILITY:

$$\exists R \forall x \forall y (E(x,y) \rightarrow (R(x) \leftrightarrow \neg R(y))).$$

- 3-SAT:

$$\exists S \forall x \forall y \forall z ((R_0(x,y,z) \rightarrow S(x) \vee S(y) \vee S(z)) \wedge$$

$$(R_1(x,y,z) \rightarrow \neg S(x) \vee S(y) \vee S(z)) \wedge$$

$$(R_2(x,y,z) \rightarrow \neg S(x) \vee \neg S(y) \vee S(z)) \wedge$$

$$(R_3(x,y,z) \rightarrow \neg S(x) \vee \neg S(y) \vee \neg S(z))).$$

The Complexity of Logic

Definition: (Vardi – 1982) Let L be a logic.

- The *combined complexity* of L is the following decision problem:

  Given a finite structure $\mathbf{A}$ and an L-sentence ψ, does $\mathbf{A} \models \psi$?

  (i.e., it is the *model checking* problem for L)

- The *data complexity* of L is the family of the following decision problems P_ψ, one for each fixed L-sentence ψ:

  Given a finite structure $\mathbf{A}$, does $\mathbf{A} \models \psi$?

- The *expression complexity* of L is the family of the following decision problems $P_{\mathbf{A}}$, one for each fixed finite structure $\mathbf{A}$:

  Given an L-sentence ψ, does $\mathbf{A} \models \psi$?

The Complexity of Logic

Definition: L a logic and C a complexity class.

- *The data complexity of* L *is in* C if for each L-sentence ψ, the problem P_ψ is in C.

- *The data complexity of* L *is* C*-complete* if it is in C and there is at least one L-sentence ψ such that P_ψ is C-complete.

- *The expression complexity of* L *is in* C if for each finite structure $\mathbf{A}$, the problem $P_{\mathbf{A}}$ is in C.

- *The expression complexity of* L *is* C*-complete* if it is in C and there is at least one finite structure $\mathbf{A}$ such that $P_{\mathbf{A}}$ is C-complete.

The Complexity of First-Order Logic

Theorem: The following hold for first-order logic:

- The data complexity of FO is in LOGSPACE
- The expression complexity of FO is PSPACE-complete
- The combined complexity of FO is PSPACE-complete.

Proof:

- Fix a first-order sentence ψ. Given finite $\mathbf{A}$:
  Cycle through all possible instantiations of the quantifiers of ψ in $\mathbf{A}$, keeping track of the number of them using a counter in binary.
- QBF is PSPACE-complete (Stockmeyer - 1976).
  QBF is the expression complexity of FO on a structure with two distinct elements. ∎

The Complexity of ESO

Theorem: The data complexity of ESO is NP-complete.

Proof:

- Let Ψ be an ESO-sentence of the form

$$\exists S_1 \cdots \exists S_m \varphi.$$

  Given a finite structure $\mathbf{A}$, to test that $\mathbf{A} \models \Psi$,

  1. "Guess" relations $S'_1, \ldots, S'_m$ on A;
  2. Verify that $(\mathbf{A}, S'_1, \ldots, S'_m) \models \varphi$, using the fact that the data complexity of FO is in P.

- 3-Colorability is definable by an ESO-sentence and is NP-complete. ∎

Theorem Both the expression complexity and the combined complexity of ESO are NEXPTIME-complete.

Descriptive Complexity

Note: Actually, a much stronger result holds for the data complexity of ESO:

Theorem: Fagin – 1972

The following are equivalent for a Boolean query Q on the class $\mathcal{F}$ of all finite σ-structures.

- Q is in NP.

- Q is ESO-definable on $\mathcal{F}$.

In other words, NP $=$ ESO on $\mathcal{F}$. ∎

Tutorial Outline

Part I: Queries and Logics

- Queries & Definability of Queries
- First-Order Logic, Existential Second Logic
- Combined, Expression, and Data Complexity

Part II: Logic and CSP Problems

- Conjunctive Queries
- The Chandra-Merlin Theorem
- MMSNP & its extensions.

Part III: Logic and Tractability of CSP

- First-Order Logic and CSP
- Datalog
- Finite-Variable Logics and Pebble Games

Fragments of First-Order Logic

- First-order logic FO has high expression and combined complexity (PSPACE-complete).
- However, there are interesting *fragments* of FO such that:
  1. they have lower expression and combined complexity;
  2. they have been extensively studied in *database theory*;
  3. they are intimately connected to *constraint satisfaction*.

Conjunctive Queries

Definition: A *conjunctive query* is a query definable by a FO-formula in prenex normal form built from atomic formulas, $\wedge$, and $\exists$ only.

$$(\exists z_1 \ldots \exists z_m)\psi(x_1, \ldots, x_k, z_1, \ldots, z_m),$$

where ψ is a conjunction of atomic formulas.

Note: CQs can also be written as a *rule:*

$$Q(x_1, \ldots, x_k) \ :- \ R(y_2, x_3, x_1), S(x_1, y_3), \ldots, S(y_7, x_2)$$

Examples:

- Path of Length 2 (Binary query)

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

$$P2(x_1, x_2) \ :- \ E(x_1, z), E(z, x_2)$$

- Cycle of Length 3 (Boolean query)

$$(\exists x_1 \exists x_2 \exists x_3)(E(x_1, x_2) \wedge E(x_2, x_3) \wedge E(x_3, x_1))$$

$$Q \ :- \ E(x_1, x_2), E(x_2, x_3), E(x_3, x_1)$$

Conjunctive Queries & Databases

- Relational Joins

  Database relations $R_1(A, B, C), R_2(B, C, D)$.

  By definition,

  $$R_1 \bowtie R_2 \;=\; \{(a, b, c, d) : R_1(a, b, c) \text{ and } R_2(b, c, d)\}.$$

  Clearly,

  $$R_1 \bowtie R_2(x, y, z, w) \quad :- \quad R_1(x, y, z), R_2(y, z, w)$$

- Relational joins are precisely the CQs without existential quantification.

- Conjunctive Queries are the most frequently asked queries in databases (a.k.a. SPJ queries)

- The main construct of SQL expresses conjunctive queries

  `SELECT` $R_1.A, R_2.D$

  `FROM` R_1, R_2

  `WHERE` $R_1.B = R_2.B$ `AND` $R_1.C = R_2.C$

Conjunctive Query Evaluation

A fundamental problem about conjunctive queries

Definition: CONJUNCTIVE QUERY EVALUATION

- Given a CQ Q and a structure $\mathbf{A}$, find

$$Q(\mathbf{A}) = \{(a_1, \ldots a_k) : \mathbf{A} \models Q(a_1, \ldots, a_k)\}$$

- For Boolean queries Q, this becomes:
  Given Q and $\mathbf{A}$, does $\mathbf{A} \models Q$? (is $Q(\mathbf{A}) = 1$?)

- Same problem as the
  combined complexity of conjunctive queries

Examples:

- Given a graph H, find all pairs of nodes connected by a path of length 4.

- Given a graph H, does it contain a triangle?

Conjunctive Query Containment

A fundamental problem about conjunctive queries

Definition: CONJUNCTIVE QUERY CONTAINMENT

- Given two k-ary CQs Q_1 and Q_2, is it true that for every structure $\mathbf{A}$,

$$Q_1(\mathbf{A}) \subseteq Q_2(\mathbf{A})?$$

- For Boolean queries, this becomes:

  Given two Boolean queries Q_1 and Q_2, does $Q_1 \models Q_2$? (does Q_1 logically imply Q_2?)

Examples:

- Is it true that if two nodes of a graph $\mathbf{H}$ are connected by a path of length 4, then they are also connected by a path of length 3?

- It is true that if a graph $\mathbf{H}$ contains a $\mathbf{K}_4$, then it also contains a $\mathbf{K}_3$?

Conjunctive Queries and Homomorphisms

- Chandra and Merlin (1977) showed that

  CONJUNCTIVE QUERY EVALUATION

  and

  CONJUNCTIVE QUERY CONTAINMENT

  are the *same* problem.

- The link is the

  HOMOMORPHISM PROBLEM

Homomorphisms

Definition: Consider two relational structures $\mathbf{A} = (A, R_1^{\mathbf{A}}, \ldots, R_m^{\mathbf{A}})$ and $\mathbf{B} = (B, R_1^{\mathbf{B}}, \ldots, R_m^{\mathbf{B}})$.

$h : \mathbf{A} \to \mathbf{B}$ is a *homomorphism* if for every $i \leq m$ and every tuple $(a_1, \ldots, a_n) \in A^n$,

$$R_i^{\mathbf{A}}(a_1, \ldots, a_n) \implies R_i^{\mathbf{B}}(h(a_1), \ldots, h(a_n)).$$

Definition: The Homomorphism Problem

Given two relational structures $\mathbf{A}$ and $\mathbf{B}$, is there a homomorphism $h : \mathbf{A} \to \mathbf{B}$?

In symbols, does $\mathbf{A} \to \mathbf{B}$?

Example: A graph $\mathbf{H} = (V, E)$ is 3-colorable

$$\iff$$

there is a homomorphism $h : \mathbf{H} \to \mathbf{K}_3$, where $\mathbf{K}_3$ is the 3-clique, i.e., $\mathbf{K}_3 = (\{R, G, B\}, E_3)$, where

$$E_3 = \{(R,G), (G,R), (R,B), (B,R), (B,G), (G,B)\}.$$

Canonical CQs and Canonical Structures

Definition: *Canonical Conjunctive Query*

Given $\mathbf{A} = (A, R_1^{\mathbf{A}}, \ldots, R_m^{\mathbf{A}})$, the *canonical CQ of* $\mathbf{A}$ is the Boolean CQ $Q^{\mathbf{A}}$ with the elements of A as variables and the "facts" of $\mathbf{A}$ as conjuncts:

$$Q^{\mathbf{A}} \;:-\; \bigwedge_{i=1}^{m} \bigwedge_{\mathbf{t}} R_i^{\mathbf{A}}(\mathbf{t})$$

Definition: *Canonical Structure*

Given a Boolean conjunctive query Q, let $\mathbf{A}^Q$ be the structure with the variables of Q as elements and the conjuncts of Q as "facts".

Example:

- $\mathbf{A} = (\{a, b, c\}, \{(a, b), (b, c), (c, a)\}$

$$Q^{\mathbf{A}} \;:-\; E(x, y) \wedge E(y, z) \wedge E(z, x)$$

- $Q \;:-\; E(x, y) \wedge E(x, z)$

$$\mathbf{A}^Q = (\{a, b, c), \{(a, b), (a, c)\})$$

Homomorphisms, CQC and CQE

Theorem: Chandra & Merlin – 1977

For relational structures $\mathbf{A}$ and $\mathbf{B}$, TFAE

- There is a homomorphism $h : \mathbf{A} \to \mathbf{B}$
- $\mathbf{B} \models Q^{\mathbf{A}}$ (i.e., $Q^{\mathbf{A}}(\mathbf{B}) = 1$)
- $Q^{\mathbf{B}} \subseteq Q^{\mathbf{A}}$

Alternatively,

For conjunctive queries Q_1 and Q_2, TFAE

- $Q_1 \subseteq Q_2$
- There is a homomorphism $h : \mathbf{A}^{Q_2} \to \mathbf{A}^{Q_1}$
- $\mathbf{A}^{Q_1} \models Q_2$ (i.e., $Q_2(\mathbf{A}^{Q_1}) = 1$)

Illustration: 3-COLORABILITY

For a graph $\mathbf{H}$, the following are equivalent:

1. There is a homomorphism $h : \mathbf{H} \to \mathbf{K_3}$
2. $\mathbf{K}_3 \models Q^{\mathbf{H}}$
3. $Q^{\mathbf{K}_3} \subseteq Q^{\mathbf{H}}$

Proof:

(1) $\Longrightarrow$ (2): A hom. $h : \mathbf{H} \to \mathbf{K_3}$ provides witnesses in $\mathbf{K}_3$ for the existential quantifiers in $Q^{\mathbf{H}}$.

(2) $\Longrightarrow$ (3): If $\mathbf{K}_3 \models Q^{\mathbf{H}}$ and $\mathbf{A} \models Q^{\mathbf{K}_3}$, then there are witness functions $h : \mathbf{H} \to \mathbf{K_3}$ and $h^* : \mathbf{K_3} \to \mathbf{A}$.

The composition $h^* \circ h : \mathbf{H} \to \mathbf{A}$ provides witnesses in $\mathbf{A}$ for the existential quantifiers in $Q^{\mathbf{H}}$.

(3) $\Longrightarrow$ (1): Since $\mathbf{K}_3 \models Q^{\mathbf{K}_3}$, we have $\mathbf{K}_3 \models Q^{\mathbf{H}}$. The witnesses to the existential quantifiers give a homomorphism from $\mathbf{H}$ to $\mathbf{K}_3$. ∎

Illustration: 3-SAT

Let φ be a 3-CNF formula with variables $x_1, \ldots, x_n$:

- $\mathbf{A}^\varphi = (\{x_1, \ldots, x_n\}, R_0^\varphi, R_1^\varphi, R_2^\varphi, R_3^\varphi)$, where

$$\begin{aligned} R_0^\varphi &= \{(x,y,z) : (x \vee y \vee z) \text{ is a clause of } \varphi\} \\ R_1^\varphi &= \{(x,y,z) : (\neg x \vee y \vee z) \text{ is a clause of } \varphi\} \\ R_2^\varphi &= \{(x,y,z) : (\neg x \vee \neg y \vee z) \text{ is a clause of } \varphi\} \\ R_3^\varphi &= \{(x,y,z) : (\neg x \vee \neg y \vee \neg z) \text{ is a clause of } \varphi\} \end{aligned}$$

- $\mathbf{B} = (\{0,1\}, R_0, R_1, R_2, R_3)$, where

$$R_0 = \{0,1\}^3 - \{(0,0,0)\} \quad R_1 = \{0,1\}^3 - \{(1,0,0)\}$$

$$R_2 = \{0,1\}^3 - \{(1,1,0)\} \quad R_3 = \{0,1\}^3 - \{(1,1,1)\}$$

Corollary: The following are equivalent:

- φ is satisfiable.
- $\mathbf{A}^\varphi \rightarrow \mathbf{B}$
- $\mathbf{B} \models Q^{\mathbf{A}^\varphi}$
- $Q^{\mathbf{B}} \subseteq Q^{\mathbf{A}^\varphi}$

CSP and Conjunctive Queries

Conclusion 1:

- Constraint Satisfaction
- The Homomorphism Problem
- Conjunctive Query Evaluation
- Conjunctive Query Containment

are the *same* problem.

Conclusion 2:

Both the combined complexity and the expression complexity of conjunctive query evaluation are NP-complete (contrast with FO-logic).

The Feder-Vardi Dichotomy Conjecture

Definition: $\mathrm{CSP}(\mathbf{B}) = \{A : A \rightarrow B\}$

Conjecture: Feder-Vardi, 1993

If $\mathbf{B}$ is a finite structure, then $\mathrm{CSP}(\mathbf{B})$ is in P or it is NP-complete.

$\mathrm{CSP}(\mathbf{B})$ $\nearrow$ $\searrow$

NP-complete
NP − P, not NP-complete
P

Note: This amounts to a dichotomy conjecture about the expression complexity of conjunctive queries

$$\begin{aligned} \mathrm{CSP}(\mathbf{B}) &= \{\mathbf{A} : \mathbf{B} \models Q^{\mathbf{A}}\} \\ &= \{Q : Q \text{ is a conjunctive query and } \mathbf{B} \models Q\} \end{aligned}$$

CSP and Data Complexity

- We saw that CSP(**B**) is the same problem as the expression complexity of conjunctive queries.
- The data complexity of conjunctive queries is in LOGSPACE, so CSP(**B**) cannot be captured by the data complexity of conjunctive queries.
- However, CSP(**B**) is intimately connected to the data complexity of a fragment of existential second-order logic, called *monadic monotone strict* NP, and denoted by MMSNP.

Existential Monadic Second-Order Logic

Definition: Existential Monadic SO-Logic

(also known as Monadic NP)

$$\exists S_1 \exists S_2 \cdots \exists S_m \psi,$$

where $S_1, \ldots, S_m$ are set variables and ψ is FO.

Fact: If $\mathbf{B} = (B, R_1, \ldots, R_m)$ is a finite structure, then CSP($\mathbf{B}$) is definable by a sentence of existential monadic second-order logic with a universal first-order part, i.e., by a sentence of the form

$$\exists S_1 \cdots \exists S_n \forall y_1 \cdots \forall y_s \theta,$$

where θ is quantifier-free.

Proof: Use one S_i for each element of $B = \{1, \ldots, n\}$, so that S_i is the set of all elements of $\mathbf{A}$ that are mapped to i, for $1 \leq i \leq n$.

CSP and Monadic NP

Example: 3-COLORABILITY

$$\exists R \exists G \exists B \forall x \forall y \theta, \quad \text{where } \theta \text{ asserts}$$

- R, B, G form a partition

$$(R(x) \vee B(x) \vee G(x)) \wedge$$

$$\neg(R(x) \wedge B(x)) \wedge \neg(B(x) \wedge G(x)) \wedge \neg(R(x) \wedge G(x)) \wedge$$

- If (x, y) is an edge, then x and y are in different parts.

$$(E(x, y) \rightarrow (R(x) \rightarrow \neg R(y)) \wedge (B(x) \rightarrow \neg B(y)) \wedge (G(x)$$

Characteristics:

- *Monadic*: SO-quantifiers over set variables only;
- *Strict*: only universal FO-quantifiers;
- *Monotone*: all occurrences of E are negated; there are no $\neq$.

MMSNP - Monadic Monotone Strict NP

Definition: Feder-Vardi, 1993

MMSNP is the class of all monadic ESO-formulas

$$(\exists S_1 \cdots \exists S_n)(\forall y_1 \cdots \forall y_s)\theta,$$

such that

- all relations in the vocabulary have only negative occurrences in θ;
- no inequalities $\neq$ occur in θ.

Proposition: Feder-Vardi, 1993

For every structure $\mathbf{B} = (B, R_1, \ldots, R_m)$, there is a MMNSP-formula $\Psi_{\mathbf{B}}$ that defines CSP($\mathbf{B}$).

Thus, each CSP($\mathbf{B}$) is a query about the data complexity of MMSNP.

CSP vs. MMSNP

Question: What is the exact relationship between CSP and MMSNP?

Theorem: Feder-Vardi, 1993

Every MMSNP-query has a randomized polynomial-time Turing reduction to finitely many CSP($\mathbf{B}$) queries.

Theorem: Kun, 2006

The reduction of MMSNP to CSP can be de-randomized.

Corollary:

(1) CSP and MMSNP are polynomially equivalent.

(2) The Dichotomy Conjecture for CSP is the same as a Dichotomy Conjecture for MMSNP.

CSP vs. Monadic NP

Theorem: Feder-Vardi, 1993

Every problem in NP is polynomially equivalent to

- a problem in strict, monotone, ESO;
- a problem in monadic, monotone, strict ESO with $\neq$;
- a problem in monadic, strict, $\neq$-free ESO.

Corollary: Assuming P $\neq$ NP, the Dichotomy Conjecture fails for all extensions of MMSNP.

Summary

- The HOMOMORPHISM PROBLEM is the same as the combined complexity of conjunctive queries (a fragment of first-order logic)

$$\mathbf{A} \to \mathbf{B} \iff \mathbf{B} \models Q^{\mathbf{A}}$$

- CSP($\mathbf{B}$) is the same problem as the expression complexity of conjunctive queries (a fragment of FO-logic):

  Given a structure $\mathbf{A}$, does $\mathbf{B} \models Q^{\mathbf{A}}$?

  $Q^{\mathbf{A}}$ is the canonical conjunctive query of $\mathbf{A}$.

- CSP($\mathbf{B}$) is polynomially equivalent to the data complexity of MMSNP (a fragment of ESO-logic):

  Given a structure $\mathbf{A}$, does $\mathbf{B} \models \Psi_{\mathbf{B}}$?

  $\Psi_{\mathbf{B}}$ is a MMSNP-sentence obtained from $\mathbf{B}$.

Tutorial Outline

Part I: Queries and Logics

- Queries & Definability of Queries
- First-Order Logic, Existential Second Logic
- Combined, Expression, and Data Complexity

Part II: Logic and CSP Problems

- Conjunctive Queries
- The Chandra-Merlin Theorem
- MMSNP & its extensions.

Part III: Logic and Tractability of CSP

- First-Order Logic and CSP
- Datalog
- Finite-Variable Logics and Pebble Games

Complexity of CSP

Uniform CSP: THE HOMOMORPHISM PROBLEM

$$\text{CSP} = \{(\mathbf{A}, \mathbf{B}) : \mathbf{A} \rightarrow \mathbf{B}\}$$

- Combined complexity of conjunctive queries
- NP-complete.

Non-Uniform CSP: For every structure $\mathbf{B}$,

$$\text{CSP}(\mathbf{B}) = \{\mathbf{A} : \mathbf{A} \rightarrow \mathbf{B}\}$$

- Expression complexity of conjunctive queries;
- Data complexity of MMSNP;
- It is in NP; can be NP-complete.

Research Program: Identify *all* tractable cases of CSP.

Islands of Tractability of CSP

Definition: Let $\mathcal{C}$ be a class of pairs $(\mathbf{A}, \mathbf{B})$ of structures.

- $\text{CSP}(\mathcal{C}) \ = \ \{(\mathbf{A}, \mathbf{B}) \in \mathcal{C} : \ \mathbf{A} \rightarrow \mathbf{B}\}$
- We say that $\mathcal{C}$ is an *island of tractability of* CSP if $\text{CSP}(\mathcal{C})$ is in P.

Research Program: Identify *all* islands of tractability of CSP.

Fact: So far, the main focus has been on islands of tractability $\mathcal{C}$ of the form $\mathcal{C} = \mathcal{A} \times \mathcal{B}$, where $\mathcal{A}$ and $\mathcal{B}$ are two classes of finite structures.

$$\text{CSP}(\mathcal{A}, \mathcal{B}) \ = \ \{(\mathbf{A}, \mathbf{B}) \in \mathcal{A} \times \mathcal{B} : \ \mathbf{A} \rightarrow \mathbf{B}\}$$

Note: $\text{CSP}(\mathbf{B}) \ = \ \text{CSP}(All, \{\mathbf{B}\})$

Logic and Tractability of Non-Uniform CSP

Research Program: Identify *all* islands of tractability of non-uniform CSP, that is, all structures $\mathbf{B}$ such that CSP($\mathbf{B}$) is in P.

Approach through Logic:

- Use logics with tractable data complexity to identify tractable cases of non-uniform CSP.
- If L is a logic whose data complexity is in P and if $\mathbf{B}$ is such that CSP($\mathbf{B}$) is definable by an L-formula, then CSP($\mathbf{B}$) is in P.

Case Study: First-Order Logic

- The data complexity of FO is in P (in fact, in LOGSPACE).
- When is CSP($\mathbf{B}$) FO-definable?

First-Order Logic and Non-Uniform CSP

Theorem: Atserias - 2005

The following are equivalent for a structure $\mathbf{B}$:

- CSP($\mathbf{B}$) FO-definable.
- $\overline{\mathrm{CSP}(\mathbf{B})} = \{\mathbf{A} : \mathbf{A} \not\to \mathbf{B}\}$ is definable by a finite union of conjunctive queries.

Note: Follows also from Rossman's Theorem (2005) about preservation under homomorphisms.

Theorem: Larose, Loten, and Tardif - 2006

The problem of deciding, given $\mathbf{B}$, whether CSP($\mathbf{B}$) is FO-definable is NP-complete.

Note: Membership in NP is non-trivial.

Datalog

Note: Recall that CQs can be written as *rules*:

$$P2(x_1, x_2) \ :- \ E(x_1, z), E(z, x_2)$$

Definition:

- Datalog $=$ Conjunctive Queries + Recursion

  Function, negation and $\neq$-free logic programs

- A Datalog program is a finite set of rules given by conjunctive queries

$$T(\overline{x}) \ :- \ S_1(\overline{y}_1), \ldots, S_r(\overline{y}_r).$$

  – Some relation symbols may occur both in the *heads* and the *bodies* of rules.

    These are the *recursive* relation symbols or *intensional database predicates* (IDBs).

  – The remaining relation symbols are the *extensional database predicates* (EDBs).

Datalog Examples

Definition: TRANSITIVE CLOSURE Query TC

Given graph $\mathbf{H} = (V, E)$,

$TC(\mathbf{H}) = \{(a, b) \in V^2 : \text{there is a path from } a \text{ to } b\}$.

Example 1: Datalog program for TC

$$\left| \begin{array}{lcl} S(x,y) & :- & E(x,y) \\ S(x,y) & :- & E(x,z) \wedge S(z,y) \end{array} \right.$$

Example 2: Another Datalog program for TC

$$\left| \begin{array}{lcl} S(x,y) & :- & E(x,y) \\ S(x,y) & :- & S(x,z) \wedge S(z,y) \end{array} \right.$$

- E is the EDB.
- S is the IDB; it defines TC.

Datalog Examples

Definition: S. Cook – 1974

PATH SYSTEMS $\mathbf{S} = (F, A, R)$

Given a finite set of *formulas* F, a set of *axioms* $A \subseteq F$, and a *rule of inference* $R \subseteq F^3$, compute the *theorems* of this system.

Example: Datalog program for PATH SYSTEMS:

$$\begin{array}{|lcl} T(x) & :- & A(x) \\ T(x) & :- & T(y), T(z), R(x, y, z) \end{array}$$

- A and R are the EDBs.
- T is the IDB; it defines the theorems of $\mathbf{S}$.

Theorem: Cook - 1974

PATH SYSTEMS is a P-complete query.

Data Complexity of Datalog

Theorem:

- Every Datalog query is definable by an "effective and uniform" union of conjunctive queries.
- Every Datalog query is in P.
- The data complexity of Datalog is P-complete.

Proof:

- Datalog programs can be evaluated "bottom-up" in a polynomial number of iterations.
- Each iteration is definable by a finite union of conjunctive queries.
- Path Systems is a P-complete problem.

Evaluation of Datalog Programs

Example : Datalog program for TC

$$\left| \begin{array}{lcl} S(x,y) & :- & E(x,y) \\ S(x,y) & :- & E(x,z) \wedge S(z,y) \end{array} \right.$$

Bottom-up Evaluation

$$\left| \begin{array}{lcl} S^0 & = & \emptyset \\ S^{m+1} & = & \{(a,b)) : \exists z(E(a,z) \wedge S^m(z,b))\} \end{array} \right.$$

Fact:

$$\begin{array}{lcl} S^m & = & \{(a.b) : \text{there is a path of length} \leq m \text{ from } a \text{ t} \\ TC & = & \bigcup_m S^m \\ TC & = & S^{|V|}. \end{array}$$

Preservation Properties

Fact: *Preservation Properties* of Datalog.

- Datalog queries are preserved under *homomorphisms*:

  Let Q be a Datalog query. If $\mathbf{A} \models Q$ and $\mathbf{A} \rightarrow \mathbf{B}$, then $\mathbf{B} \models Q$.

- Similarly, Datalog queries are *monotone*, i.e., they query is preserved if new tuples are added to the EDBs.

Reason: Unions of conjunctive queries have these preservation properties.

Datalog and CSP

Fact: Let $\mathbf{B} = (B, R_1^{\mathbf{B}}, \ldots, R_m^{\mathbf{B}})$.

- In general, CSP($\mathbf{B}$) is *not* monotone.
- Hence, CSP($\mathbf{B}$) is *not* expressible in Datalog.

However,

- $\overline{\text{CSP}(\mathbf{B})}$ is monotone, where

$$\overline{\text{CSP}(\mathbf{B})} = \{\mathbf{A} : \; \mathbf{A} \not\rightarrow \mathbf{B}\}.$$

- Hence, it is conceivable that $\overline{\text{CSP}(\mathbf{B})}$ is expressible in Datalog (and, thus, it is in P).

Datalog and CSP

Fact: Feder & Vardi – 1993

Definability of $\overline{\mathrm{CSP}(\mathbf{B})}$ in Datalog is a unifying explanation for many tractability results about $\mathrm{CSP}(\mathbf{B})$.

Example: 2-Colorability $=$ $\mathrm{CSP}(\mathbf{K}_2)$

Datalog program for Non 2-Colorability

$$
\begin{array}{|lcl}
O(X,Y) & :- & E(X,Y) \\
O(X,Y) & :- & O(X,Z), E(Z,W), E(W,Y) \\
Q & :- & O(X,X)
\end{array}
$$

Datalog and CSP

Theorem: Feder & Vardi – 1993

- If $\mathbf{B} = (B, R_1, \ldots, R_k)$ is such that $\mathrm{Pol}(\{R_1, \ldots, R_k\})$ contains a near-unanimity function, then $\overline{\mathrm{CSP}(\mathbf{B})}$ is definable in Datalog.

  Special Case: 2-SAT

- If $\mathbf{B} = (B, R_1, \ldots, R_k)$ is such that $\mathrm{Pol}(\{R_1, \ldots, R_k\})$ contains a semi-lattice function, then $\overline{\mathrm{CSP}(\mathbf{B})}$ is definable in Datalog.

  Special Cases:

  HORN k-SAT, DUAL HORN k-SAT, $k \geq 2$.

- There are affine Boolean structures $\mathbf{B}$ such that $\overline{\mathrm{CSP}(\mathbf{B})}$ is **not** definable in Datalog.

Horn 3-SAT and Datalog

Horn 3-CNF formula φ viewed as a finite structure

$$\mathbf{A}^{\varphi} \;=\; (\{x_1, \ldots, x_n\}), U, P, N), \quad \text{where}$$

- U is the set of unit clauses x
- P is the set of clauses $(\neg x \vee \neg y \vee z)$
- N is the set of clauses $(\neg x \vee \neg y \vee \neg z)$

Datalog program for HORN 3-UNSAT

Unit Propagation Algorithm

$$\left|\begin{array}{lcl} T(z) & :- & U(z) \\ T(z) & :- & P(x,y,z), T(x), T(y) \\ Q & :- & N(x,y,z), T(x), T(y), T(z) \end{array}\right.$$

CSP and Datalog

Fact: Expressibility in Datalog is a unifying explanation for many, but not all, tractability results about CSP($\mathbf{B}$).

Open Problem: Is there an algorithm to decide whether, given $\mathbf{B}$, we have that $\overline{\text{CSP}(\mathbf{B})}$ is expressible in Datalog?

Note: It follows from the work of Larose, Loten, and Tardif that this problem is NP-hard.

Datalog and CSP

Question: Fix $\mathbf{B} = (B, R_1, \ldots, R_m)$.

When is $\overline{\mathrm{CSP}(\mathbf{B})}$ expressible in Datalog?

Answer:

Feder & Vardi – 1993, K ... & Vardi – 1998, 2000

Expressibility of $\overline{\mathrm{CSP}(\mathbf{B})}$ in Datalog can be characterized in terms of

- Finite-Variable Logics
- Pebble Games
- Consistency Properties.

Existential k-Pebble Games

Spoiler and Duplicator play on two structures $\mathbf{A}$ and $\mathbf{B}$. Each player uses k pebbles. In each move,

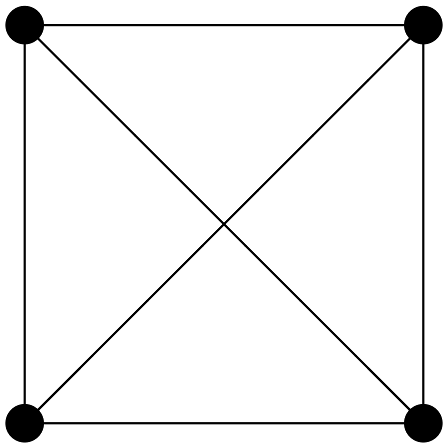
- Spoiler places a pebble on or removes a pebble from an element of $\mathbf{A}$.
- Duplicator tries to duplicate the move on $\mathbf{B}$.

$$\begin{array}{llllll} \mathbf{A}: & a_1 & a_2 & \dots & a_l & \\ & \downarrow & \downarrow & \cdots & \downarrow & \\ \mathbf{B}: & b_1 & b_2 & \dots & b_l & l \leq k \end{array}$$

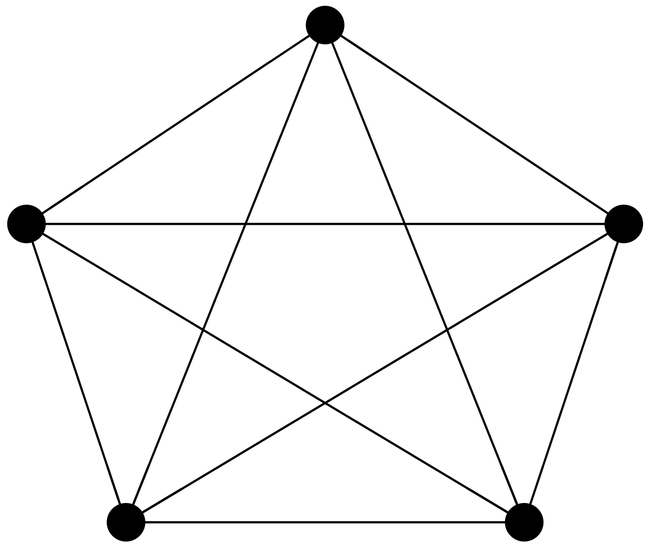
- Spoiler *wins* the $(\exists, k)$-pebble game if at some point the mapping $a_i \mapsto b_i$, $1 \leq i \leq l$, is not a partial homomorphism.
- Duplicator *wins* the $(\exists, k)$-pebble game if the above never happens.

Example

Cliques of Different Size



$\mathbf{K}_4$

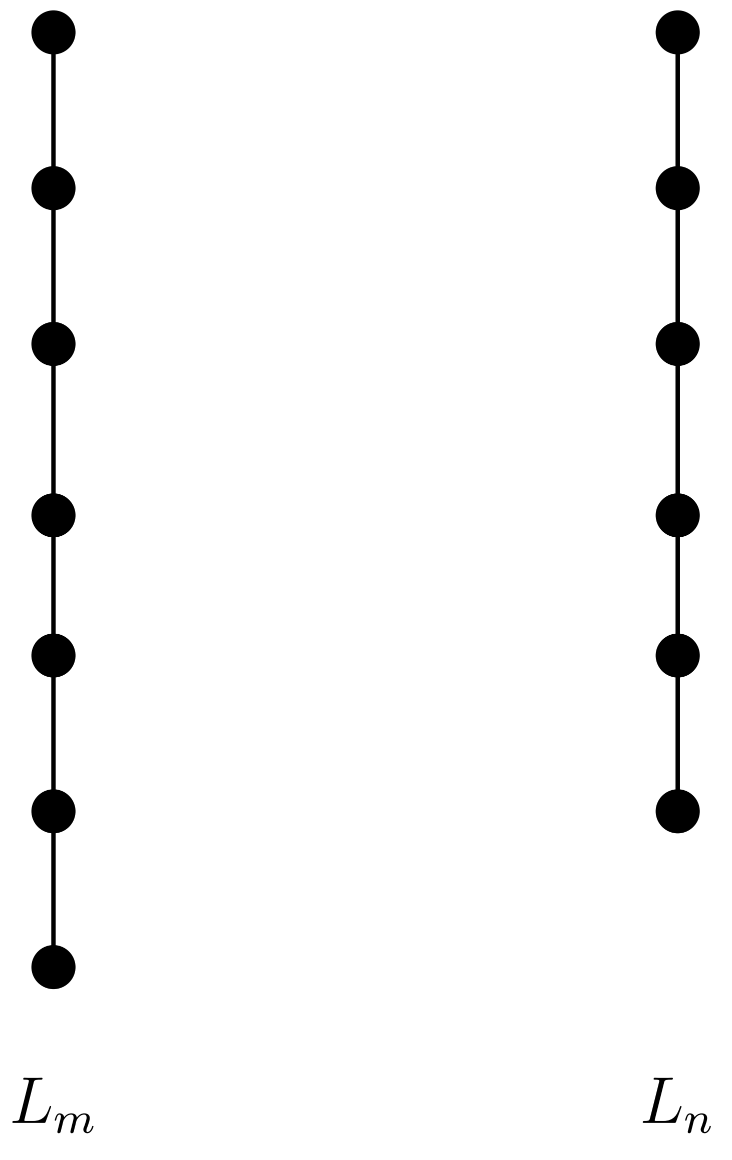


$\mathbf{K}_5$

Fact: Let $\mathbf{K}_k$ be the k-clique

- Duplicator wins the $(\exists, k)$-pebble game on $\mathbf{K}_k$ and $\mathbf{K}_{k+1}$.
- Spoiler wins the $(\exists, k)$-pebble game on $\mathbf{K}_k$ and $\mathbf{K}_{k-1}$.

Paths of Different Size



- Spoiler wins the $(\exists, 3)$-pebble game on L_m and L_n, where $m > n$.
- Duplicator wins the $(\exists, 3)$-pebble game on L_n and L_m, where $m > n$.

Winning Strategies in the $(\exists, k)$-Pebble Game

Definition: A *winning strategy* for the *Duplicator* in the $(\exists, k)$-pebble game is a non-empty family I of partial homomorphisms from **A** to **B** such that

- If $f \in I$ and $h \subseteq f$, then $h \in I$

  (I is *closed under subfunctions*).

- If $f \in I$ and $|f| < k$, then for every $a \in A$, there is $g \in I$ so that $f \subseteq g$ and $a \in \text{dom}(g)$.

  (I has the *forth property up to k*)

Fact: If $\mathbf{A} \rightarrow \mathbf{B}$, then the Duplicator wins the $(\exists, k)$-pebble game on **A** and **B** for every k.

k-Datalog

Definition: A k-Datalog program is a Datalog program in which each rule

$$t_0 \ :- \ t_1, \ldots, t_m$$

has at most k distinct variables.

Example: Non 2-Colorability revisited

$$\begin{array}{|lcl} O(X,Y) & :- & E(X,Y) \\ O(X,Y) & :- & O(X,Z), E(Z,W), E(W,Y) \\ Q & :- & O(X,X) \end{array}$$

Therefore,

Non 2-Colorability is definable in 4-Datalog.

k-Datalog and $(\exists, k)$-Pebble Games

Theorem: K ... & Vardi

- Let Q be a query definable by a k-Datalog program. If $\mathbf{A}$ satisfies Q and the Duplicator wins the $(\exists, k)$-pebble game on $\mathbf{A}$ and $\mathbf{B}$, then also $\mathbf{B}$ satisfies Q.

- There is a polynomial-time algorithm to decide whether, given two finite structures $\mathbf{A}$ and $\mathbf{B}$, the Spoiler or the Duplicator wins the $(\exists, k)$-pebble game on $\mathbf{A}$ and $\mathbf{B}$.

- For every fixed finite structure $\mathbf{B}$, there is a k-Datalog program that expresses the query: given a finite structure $\mathbf{A}$, does the Spoiler win the $(\exists, k)$-game on $\mathbf{A}$ and $\mathbf{B}$?

Datalog and Non-Uniform CSP

Theorem: K ... & Vardi

Let k be a positive integer and $\mathbf{B}$ a finite structure.

Then the following are equivalent:

- $\overline{\mathrm{CSP}(\mathbf{B})}$ is definable in k-Datalog
- $\mathrm{CSP}(\mathbf{B}) = \{\mathbf{A} :$ Duplicator wins the $(\exists, k)$-pebble game on $\mathbf{A}$ and $\mathbf{B}\}$.
- For every finite structure $\mathbf{A}$, establishing strong k-consistency for $\mathbf{A}$ and $\mathbf{B}$ implies that there is a homomorphism from $\mathbf{A}$ to $\mathbf{B}$.

The Complexity of Existential k-Pebble Games

Theorem: K ... and Panttaja - 2003

- (Also implicit in Kasif - 1986)

  For every $k \geq 2$, the following problem is P-complete:

  Given two finite structures **A** and **B**, does the Duplicator win the $(\exists, k)$-pebble game on **A** and **B**?

- The following problem is EXPTIME-complete:

  Given a positive integer k and two finite structures **A** and **B**, does the Duplicator win the $(\exists, k)$-pebble game on **A** and **B**?

Corollary:

The following problem is EXPTIME-complete:

Given a positive integer k and two finite structures **A**, **B**, can strong k-consistency be established for (the CSP instance encoded by) **A** and **B**?

Datalog and Tractability of CSP

Summary:

- Definability of $\overline{\mathrm{CSP}(\mathbf{B})}$ in k-Datalog is a sufficient condition for tractability of $\mathrm{CSP}(\mathbf{B})$.

- Single *canonical* polynomial-time algorithm: determine who wins the $(\exists, k)$-pebble game.

Open Problem:

Fix a positive integer $k \geq 2$. Is there an algorithm to decide whether, given $\mathbf{B}$, we have that $\overline{\mathrm{CSP}(\mathbf{B})}$ is expressible in k-Datalog?

Tractability of Non-Uniform CSP

- Thus far, we have concentrated on tractability results for non-uniform CSP.
- What about tractability results for uniform CSP?
- Does logic help to discover islands of tractability for uniform CSP?

Tractability of Uniform CSP

Recall that if $\mathcal{A}$ and $\mathcal{B}$ are classes of finite structures, then

$$\text{CSP}(\mathcal{A}, \mathcal{B}) \;=\; \{\mathbf{A}, \mathbf{B}) \in \mathcal{A} \times \mathcal{B} : \;\mathbf{A} \rightarrow \mathbf{B}\}$$

Theorem: Dechter & Pearl – 1989

Let σ be a fixed vocabulary, let $k \geq 2$ be a positive integer, and let $\mathcal{T}(k)$ be the class of all σ-structures of *treewidth* less than k.

Then CSP($\mathcal{T}(k)$, *All*) is in P.

Question:

- Can this result be explained in terms of definability in Datalog?
- Can this result be explained in terms of the $(\exists, k)$-pebble game?

Bounded Treewidth & Finite-Variable Logics

Fact: Having $\mathrm{tw}(\mathbf{A}) < k$ turns out to be tightly connected to the canonical query $Q^{\mathbf{A}}$ being definable in a fragment of FO with k variables.

Definition: Fix an integer $k \geq 2$.

- FO^k is the collection of all first-order formulas with k distinct variables.
- CQ^k is the collection of all FO^k-formulas built using atomic formulas, $\wedge$, and $\exists$ only.

Example: Let $\mathbf{C}_n$ be the n-element cycle, $n \geq 3$.

The canonical CQ $Q^{\mathbf{C}_n}$ is expressible in CQ^3.

For instance, $Q^{\mathbf{C}_4}$ is logically equivalent to

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

Bounded Treewidth & Finite-Variable Logics

Question: When is $Q^{\mathbf{A}}$ definable in CQ^k?

Definition: $\mathbf{A}$ and $\mathbf{B}$ are *homomorphically equivalent*, denoted $\mathbf{A} \sim_h \mathbf{B}$, if there are homomorphisms $h : \mathbf{A} \to \mathbf{B}$ and $h' : \mathbf{B} \to \mathbf{A}$.

Theorem: Dalmau, K ..., Vardi - 2002

Fix a k and a finite structure $\mathbf{A}$.

Then the following are equivalent:

- $Q^{\mathbf{A}}$ is definable in CQ^k.
- There is some $\mathbf{B} \in \mathcal{T}(k)$ such that $\mathbf{A} \sim_h \mathbf{B}$.
- $\mathrm{core}(\mathbf{A}) \in \mathcal{T}(k)$.

Cores

Definition: We say that a structure $\mathbf{B}$ is the *core* of a structure $\mathbf{A}$ if

- $\mathbf{B}$ is a submodel of $\mathbf{A}$.
- There is a homomorphism from $\mathbf{A}$ to $\mathbf{B}$ (thus, $\mathbf{A} \equiv_h \mathbf{B}$).
- There is no homomorphism $h : \mathbf{B} \to \mathbf{B}'$ from $\mathbf{B}$ to a proper submodel $\mathbf{B}'$ of $\mathbf{B}$.

Examples:

- $\text{core}(\mathbf{K}_k) = \mathbf{K}_k$
- If $\mathbf{H}$ is 2-colorable, then $\text{core}(\mathbf{H}) = \mathbf{K}_2$.
- If $\mathbf{H}$ is 3-colorable and contains a $\mathbf{K}_3$, then $\text{core}(\mathbf{H}) = \mathbf{K}_3$.

Note: Cores play an important role in database query processing and optimization.

Beyond Bounded Treewidth

Definition: Fix a vocabulary σ and a $k \geq 2$.

$\mathcal{H}(\mathcal{T}(k))$ is the class of all σ-structures that are homomorphically equivalent to a structure in $\mathcal{T}(k)$.

Fact: $\mathcal{H}(\mathcal{T}(k))$ is the class of all σ-structures $\mathbf{A}$ such that core($\mathbf{A}$) has treewidth less than k.

Example: Every 2-colorable graph is in $\mathcal{H}(\mathcal{T}(2))$.

Fact: $\mathcal{T}(k)$ is properly contained in $\mathcal{H}(\mathcal{T}(k))$

Proof: There are 2-colorable graphs of arbitrarily large treewidth (for instance, $m \times m$-grids)

Islands of Tractability of Uniform CSP

Theorem : Dalmau, K ..., Vardi – 2002

Fix a vocabulary σ and an integer $k \geq 2$.

- For every structure $\mathbf{A} \in \mathcal{H}(\mathcal{T}(k))$ and for every structure $\mathbf{B}$, the following are equivalent:
  1. $\mathbf{A} \rightarrow \mathbf{B}$
  2. The Duplicator wins the $(\exists, k)$-pebble game on $\mathbf{A}$ and $\mathbf{B}$.

- If $\mathbf{B}$ is a fixed σ-structure, then $\overline{\text{CSP}(\mathcal{H}(\mathcal{T}(k)), \{\mathbf{B}\})}$ is definable in k-Datalog.

- $\text{CSP}(\mathcal{H}(\mathcal{T}(k)), All)$ is in P.

  Actually, it is definable in least fixed-point logic LFP.

  Algorithm:

  Determine the winner in the $(\exists, k)$-pebble game.

Classification Theorem

Theorem: Grohe – 2003

Assume that FPT $\neq W[1]$.

If $\mathcal{A}$ is a r.e. class of finite structures over some fixed vocabulary σ such that CSP$(\mathcal{A}, All)$ is in P, then there is a $k \geq 2$ such that $\mathcal{A} \subseteq \mathcal{H}(\mathcal{T}(k))$.

Note: FPT $\neq W[1]$ is the analog of P $\neq$ NP for parametrized complexity.

Conclusion: For every fixed vocabulary σ, the classes $\mathcal{H}(\mathcal{T}(k))$ constitute the *largest* islands of tractability of the form CSP$(\mathcal{A}, All)$ among all classes $\mathcal{A}$ of σ-structures.

Summary

- The combinatorial concept of bounded treewidth has a logical reconstruction via definability in finite-variable logics.

- $\mathrm{CSP}(\mathcal{H}(\mathcal{T}(k)), \mathit{All})$, $k \geq 2$, are large islands of tractability of uniform CSP.

- Determining the winner in the $(\exists, k)$-pebble game is a polynomial-time algorithm for $\mathrm{CSP}(\mathcal{H}(\mathcal{T}(k)), \mathit{All})$ (hence, also for $\mathrm{CSP}(\mathcal{T}(k)), \mathit{All})$).

Logic and CSP

- UNIFORM CSP is the same problem as the *combined complexity of conjunctive queries*
- NON-UNIFORM CSP
  - is the same problem as the *expression complexity of conjunctive queries*
  - is polynomially equivalent to the *data complexity of* MMSNP
- Datalog and $(\exists, k)$-pebble games provide a unifying explanation for many, but not all, tractability results for NON-UNIFORM CSP
- $(\exists, k)$-pebble games give rise to large islands of tractability for UNIFORM CSP.

Concluding Remarks

- Constraint Satisfaction is a meeting point of
  - Computational Complexity
  - Database Theory
  - Logic
  - Universal Algebra
  - Graph Theory.

- The quest for islands of tractability of CSP goes on through the synergy and interaction of all these areas.