Combinatorial Games in Finite Model Theory

Phokion G. Kolaitis

Computer Science Department

University of California, Santa Cruz

Course Outline

Topic 1: Ehrenfeucht-Fraïssé Games for First-Order Logic

Topic 2: Ehrenfeucht-Fraïssé Games for Existential Second-Order Logic

Topic 3: k-Pebble Games, Finite Variable Logics, and Fixed-Point Logics.

Topic 4: Existential k-Pebble Games, Datalog, and Constraint Satisfaction

Topic 5: Computational complexity issues.

Finite Model Theory

The study of logics on classes of finite structures

- *Examples of Logics*
  - First-order logic
  - Second-order logic (and its fragments)
  - Logics with fixed-point operators
  - Infinitary logics
  - Logics with generalized quantifiers.

- *Examples of Classes of Finite Structures*
  - The class of all finite graphs
  - The class of all finite ordered graphs
  - The class of all finite planar graphs
  - The class of all finite strings
  - The class of all finite trees.

Reasons for Developing Finite Model Theory

- *Connections and applications to* CS

  Interactions with several areas of CS, including

  – database theory

  – computational complexity

  – computer-aided verification.

- *Study finite model theory in its own right*

  – Traditional focus of mathematical logic has been on fixed infinite structures or on classes of finite and infinite structures.

  – New phenomena emerge, when one focuses on classes of finite structures.

Areas of Research in Finite Model Theory

- Expressive Power of Logics
- Logic and Computational Complexity
- Logic and Asymptotic Probabilities

Fact: Combinatorial games play a prominent role in these areas, especially in the first and the third.

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

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

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

Queries

Definitions:

- *Class $\mathcal{C}$ of σ-structures:* a collection of 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 $h : \mathbf{A} \to \mathbf{B}$ is an isomorphism, then

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

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

  a mapping $Q : \mathcal{C} \to \{0,1\}$ 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

- Transitive Closure TC:

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

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

- Articulation Point AP:

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

  $AP(\mathbf{G}) \ = \ \{a \in V\colon\ a$ is an articulation point of $\mathbf{G}\}$.

- Connectivity CN:

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

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

- Acyclicity, k-Colorability, $k \geq 2$, Hamiltonicity, ...

Definability of Queries

Let L be a logic and $\mathcal{C}$ a class of σ-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 $\mathbf{A} \in \mathcal{C}$

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

- $L[\mathcal{C}]$ denotes the collection of all L-definable queries on $\mathcal{C}$.

First-Order Logic

Syntax of 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 $\exists x$, $\forall x$, $\exists y$, $\forall y$, ... that range over the nodes of the graph.

Examples of first-order definable queries:

- "node x has at least two distinct neighbors"

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

- "each node has at least two distinct neighbors"

$$(\forall x)(\exists y)(\exists z)(\neg(y = z) \wedge E(x, y) \wedge E(x, z))$$

- "there is a path of length 3 from x to y"

$$(\exists z_1)(\exists z_2)(E(x, z_1) \wedge E(z_1, z_2) \wedge E(z_2, y))$$

Limitations of First-Order Logic

Fact: Let $\mathcal{G}$ be the class of finite graphs. None of the following queries is first-order definable on $\mathcal{G}$:

- Transitive Closure
- Articulation Point
- Connectivity
- Acyclicity
- Planarity
- Eulerian
- k-Colorability, for each fixed $k \geq 2$
- Hamiltonicity
- your favorite algorithmically interesting property of graphs

Upper and Lower Bounds for Definability

Let L be a logic, $\mathcal{C}$ a class of structures and Q a query on $\mathcal{C}$.

- To prove that Q is L-definable on $\mathcal{C}$, it suffices to find an L-formula φ that defines Q on $\mathcal{C}$.

  Analogous to finding an algorithm that solves the problem and also obeys certain performance guarantees.

- To prove that Q is not L-definable on $\mathcal{C}$, one has to show that there is *no* L-formula that defines Q on $\mathcal{C}$.

  Analogous to showing that there is *no* algorithm that solves the problem and also obeys certain performance guarantees.

Lower Bounds for Definability - Methods

Question: How does one establish that a query Q is *not* definable in a certain logic L?

In particular, how does one prove *lower bounds* for definability in first-order logic?

Answer: Three methods for establishing lower bounds for definability in first-order logic:

- *Compactness Theorem*
- *Ultraproducts*
- *Ehrenfeucht-Fraïssé Games* (E-F Games)

The Compactness Theorem

Theorem: Let Σ be set of first-order sentences. If every finite subset of Σ has a model, then Σ has a model.

Proposition: CONNECTIVITY is not FO-definable.

Proof: Assume ψ is a first-order sentence defining CONNECTIVITY on all graphs. Let

- $\mathbf{c}$, $\mathbf{d}$: two new constant symbols
- p_n: there is no path of length $\leq n$ from $\mathbf{c}$ to $\mathbf{d}$.
- $\Sigma = \{\psi\} \cup \{p_n : n \geq 1\}$ set of FO-sentences.

Then

- Every finite subset of Σ has a model. Hence,
- Σ has a model $G = (V, E, c, d)$.
  - $G \models \psi$, so G is connected.
  - $G \models p_n$, for every $n \geq 1$.
  - Hence, there is *no* path from c to d. ∎

Lower Bounds for Expressibility - Methods

Note:

- The model guaranteed to exist by the compactness theorem may very well be infinite.

- The preceding proof does *not* establish that CONNECTIVITY is not first-order definable on the class of all finite graphs.

Fact:

- E-F Games is the main method used to establish lower bound for definability in first-order logic on classes of finite structures.

- Moreover, it is a flexible and extendible method: variants of E-F games can be used to obtain lower bounds for definability in stronger logics for which the compactness theorem fails.

Ehrenfeucht-Fraïssé Game with r-Moves

– *Spoiler* and *Duplicator* play on two σ-structures **A** and **B**.

– Each run of the game has r moves. In each move,

- *Spoiler* picks an element from A or from B.
- *Duplicator* picks an element from B or from A.

A typical run could look as follows:

$$\begin{array}{llllll}
\textit{Spoiler} & a_1 \in A & b_2 \in B & b_3 \in B & \dots & a_r \in A \\
 & \updownarrow & \updownarrow & \updownarrow & \dots & \updownarrow \\
\textit{Duplicator} & b_1 \in B & a_2 \in A & a_3 \in A & \dots & b_r \in B
\end{array}$$

- *Duplicator* wins the run if the mapping

  $$a_1 \mapsto b_1, a_2 \mapsto b_2, \dots, a_r \mapsto b_r$$

  is an isomorphism between the σ-structures induced by $\{a_1, \dots, a_r\}$ and $\{b_1, \dots, b_r\}$.

- *Spoiler* wins the run otherwise.

Ehrenfeucht-Fraïssé Game with r-moves

Definition: Let **A** and **B** be two σ-structures.

- The *Duplicator wins* the r-move E-F game on **A** and **B** if the Duplicator can win every run of the game, i.e., if (s)he has a *winning strategy* for the Ehrenfeucht-Fraïssé game.

- The *Spoiler wins* the game, otherwise.

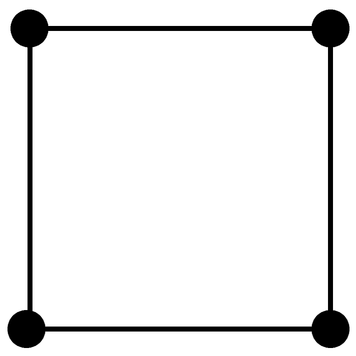
Notation:

$\mathbf{A} \sim_r \mathbf{B}$ denotes that the *Duplicator wins* the r-move Ehrenfeucht-Fraïssé game on **A** and **B**.

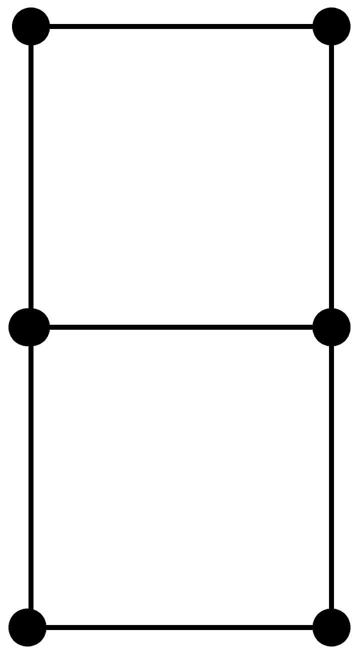
Fact:

$\sim_r$ is an equivalence relation on the class $\mathcal{S}$ of all σ-structures.

Example



A



B

- *Duplic.* wins the 2-move E-F game on $\mathbf{A}$, $\mathbf{B}$.

$$\mathbf{A} \sim_2 \mathbf{B}$$

- *Spoiler* wins the 3-move E-F game on $\mathbf{A}$, $\mathbf{B}$.

$$\mathbf{A} \not\sim_3 \mathbf{B}$$

Winning Strategies in the E-F Game

Definition: A *winning strategy* for the *Duplicator* in the r-move E-F game on $\mathbf{A}$ and $\mathbf{B}$ is a sequence $I_0, I_1, \ldots, I_r$ of non-empty sets of partial isomorphims from $\mathbf{A}$ to $\mathbf{B}$ such that

- $I_0, I_1, \ldots, I_r$ has the *forth property*:

  For every $i < r$, every $f \in I_i$, and every $a \in A$, there is a $g \in I_{i+1}$ such that $a \in \mathrm{dom}(g)$ and $f \subseteq g$.

- $I_0, I_1, \ldots, I_r$ has the *back property*:

  For every $i < r$, every $f \in I_i$, and every $b \in B$, there is a $g \in I_{i+1}$ such that $b \in \mathrm{rng}(g)$ and $f \subseteq g$.

Uses of Ehrenfeucht-Fraïssé Games

Question:

What are Ehrenfeucht-Fraïssé games good for?

Answer:

- E-F games capture the combinatorial/algebraic content of first-order quantification.
- E-F games can be used to characterize and study definability in first-order logic on an arbitrary class of σ-structures.

Quantifier Rank

Definition: Let φ be a first-order formula over σ. The *quantifier rank* of φ is the depth of quantifier nesting in φ.

Example:

- φ is $(\forall x)(\forall y)(\exists z)\theta$, where θ is quantifier-free.

$$\mathrm{qr}(\varphi) = 3.$$

- φ is $(\exists x)E(x,x) \vee (\exists y)(\forall z)\neg E(y,z)$

$$\mathrm{qr}(\varphi) = 2.$$

Definition: Let r be a positive integer, and let $\mathbf{A}$ and $\mathbf{B}$ be two σ-structures.

$\mathbf{A} \equiv_r \mathbf{B}$ if $\mathbf{A}$ and $\mathbf{B}$ satisfy the same first-order sentences of quantifier rank r.

Fact: $\equiv_r$ is an equivalence relation on the class $\mathcal{S}$ of all σ-structures.

E-F Games and First-Order Logic

Theorem: Fraïssé (1954), Ehrenfeucht (1961)

For every r, $\mathbf{A}$, and $\mathbf{B}$, the following are equivalent:

- $\mathbf{A} \equiv_r \mathbf{B}$, i.e., $\mathbf{A}$ and $\mathbf{B}$ satisfy the same first-order sentences of quantifier rank r.
- $\mathbf{A} \sim_r \mathbf{B}$, i.e., *Duplicator* wins the r-move E-F game on $\mathbf{A}$ and $\mathbf{B}$.

Moreover, the following statements are true:

- $\equiv_r$ has finitely many equivalence classes.
- Each $\equiv_r$-equivalence class is FO-definable.

Remark: This provides a purely combinatorial (algebraic) characterization of first-order logic, since

- $\equiv_r$ is defined in terms of logic only;
- $\sim_r$ is defined in terms of games (partial isomorphisms) only.

A Brief History

- Fraïssé – 1954

  Characterized $\equiv_r$ in terms of the existence of sets of partial isomorphisms with the back and forth property.

  Algebraic characterization of $\equiv_r$.

- Ehrenfeucht – 1961

  Characterized $\equiv_r$ in terms of the existence of a winning strategy for the Duplicator in the r-move E-F game.

  Combinatorial characterization of $\equiv_r$.

Theorem: If $\mathbf{A} \sim_r \mathbf{B}$, then $\mathbf{A} \equiv_r \mathbf{B}$.

Proof: By example for $r = 3$. Assume that there is a quantifier-free formula $\theta(x, y, z)$ such that

- $\mathbf{A} \models (\forall x)(\forall y)(\exists z)\theta(x, y, z)$, but
- $\mathbf{B} \models (\exists x)(\exists y)(\forall z)\neg\theta(x, y, z)$.

Then, the Spoiler can win the following run:

$$\begin{array}{llll} \textit{Spoiler} & b_1 \in B & b_2 \in B & a_3 \in A \\ & \updownarrow & \updownarrow & \updownarrow \\ \textit{Duplicator} & a_1 \in A & a_2 \in A & b_3 \in B \end{array}$$

where b_1, b_2, a_3 have been chosen so that

- $\mathbf{B} \models (\forall z)\neg\theta(b_1, b_2, z)$
- $\mathbf{A} \models \theta(a_1, a_2, a_3)$;

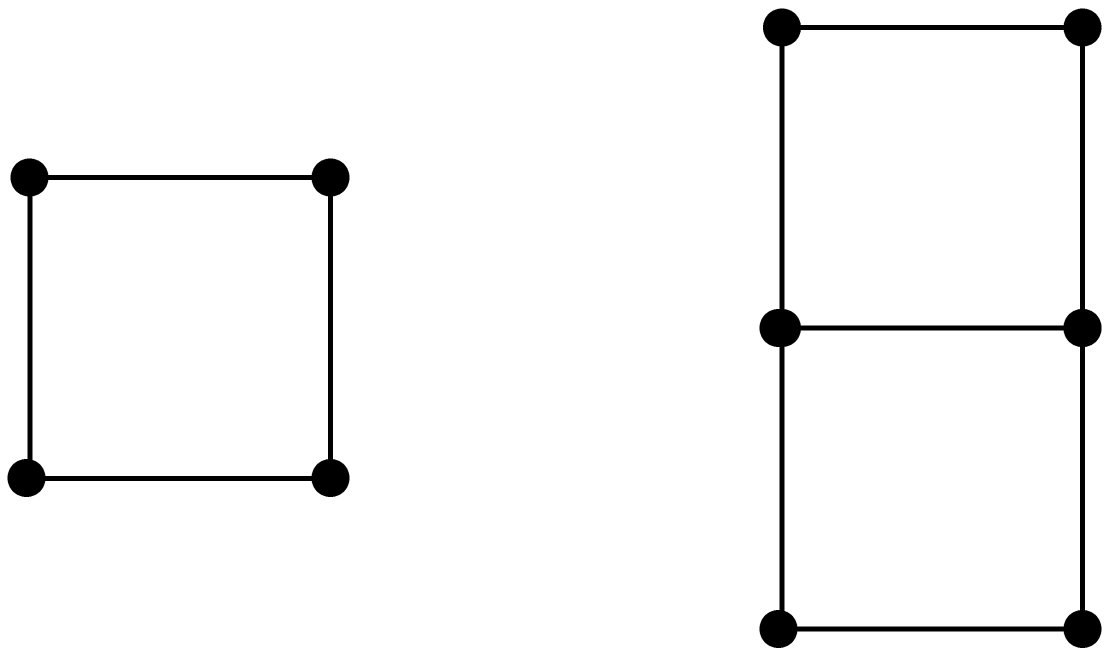
this is possible, because $\mathbf{A} \models (\exists z)\theta(a_1, a_2, z)$.

Thus, $\mathbf{A} \models \theta(a_1, a_2, a_3)$ and $\mathbf{B} \models \neg\theta(b_1, b_2, b_3)$. As a result,

$$a_1 \mapsto b_1, a_2 \mapsto b_2, a_3 \mapsto b_3$$

is *not* an isomorphism and, hence, $\mathbf{A} \not\sim_3 \mathbf{B}$. ∎

Example – Revisited



A **B**

- *Spoiler* wins the 3-move E-F game on **A**, **B**.
- The theorem predicts that $\mathbf{A} \not\equiv_3 \mathbf{B}$. Indeed,

$$\mathbf{A} \models \varphi \text{ and } \mathbf{B} \models \neg\varphi,$$

where φ is the first-order sentence

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

E-F Games and First-Order Definability

Corollary: Let Q be a Boolean query on $\mathcal{C}$.

The following statements are equivalent:

- Q is first-order definable on $\mathcal{C}$.
- There is an r such that for every $\mathbf{A}$, $\mathbf{B}$ in $\mathcal{C}$ if $\mathbf{A} \models Q$ and $\mathbf{B} \not\models Q$, then the *Spoiler* wins the r-move E-F game on $\mathbf{A}$, $\mathbf{B}$. ∎

A Methodology for First-Order Definability

Let Q be a Boolean query on $\mathcal{C}$.

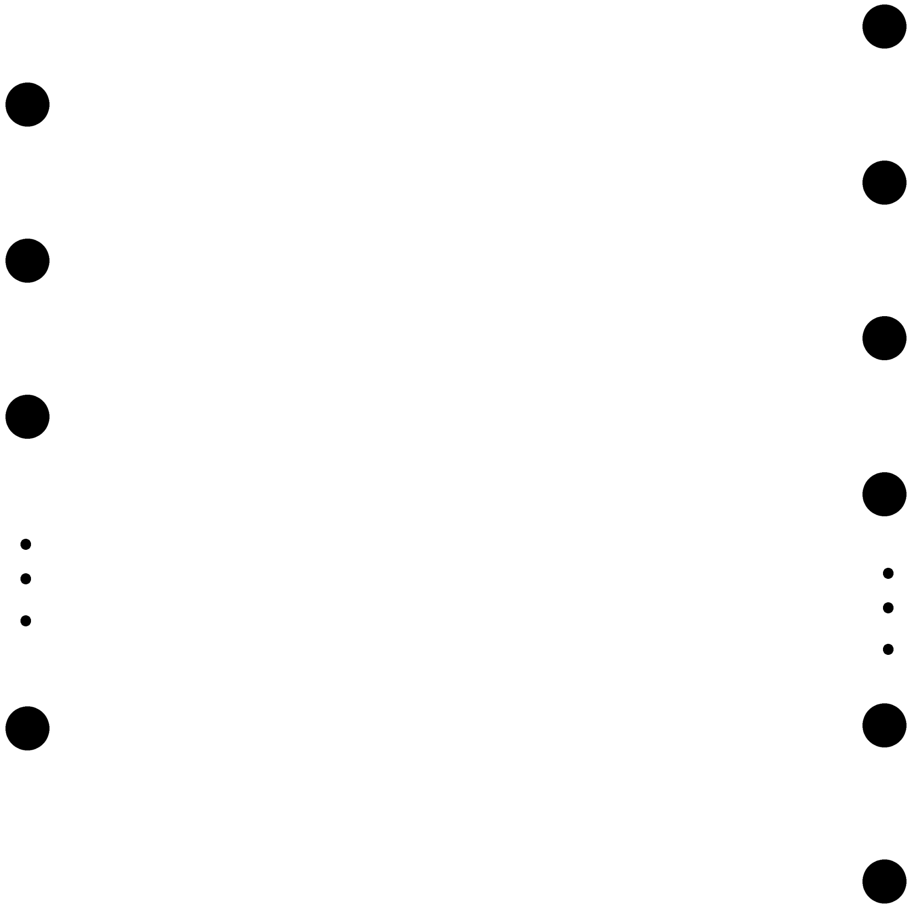
Soundness: To show that Q is *not* first-order definable on $\mathcal{C}$, suffices to show that for every r there are structures $\mathbf{A}_r$ and $\mathbf{B}_r$ in $\mathcal{C}$ such that

- $\mathbf{A}_r \models Q$ and $\mathbf{B}_r \not\models Q$.
- *Duplicat.* wins the r-move E-F game on $\mathbf{A}$, $\mathbf{B}$.

Completeness: This method is also *complete*, i.e., if Q is *not* first-order definable on $\mathcal{C}$, then this can be shown using E-F games.

Applications

Even Cardinality Query:



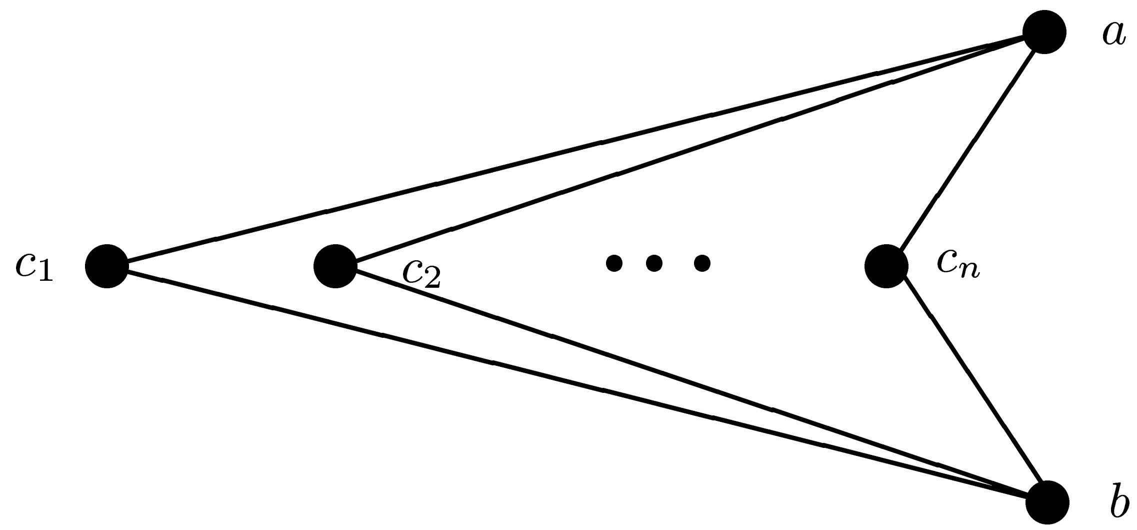
$\overline{K_m}$ $\qquad\qquad$ $\overline{K_n}$

Fact: If $m \geq r$ and $n \geq r$, then $\overline{K}_m \sim_r \overline{K}_n$.

Corollary: Even Cardinality is *not* first-order definable on finite graphs.

Applications

EULERIAN GRAPHS:



A_n

Fact:

- If $n \geq r$, then $\mathbf{A}_{n+1} \sim_r \mathbf{A}_n$.
- $\mathbf{A}_n$ is EULERIAN $\iff$ n is even.

Corollary: EULERIAN is *not* first-order definable on finite graphs.

Methodology for First-Order Definability:

To show that a Boolean query Q is *not* first-order definable on $\mathcal{C}$, suffices to show that for every r there are structures $\mathbf{A}_r$ and $\mathbf{B}_r$ in $\mathcal{C}$ such that

- $\mathbf{A}_r \models Q$ and $\mathbf{B}_r \not\models Q$.
- *Duplicat.* wins the r-move E-F game on $\mathbf{A}$, $\mathbf{B}$.

Implementation Difficulties:

I. How does one find such $\mathbf{A}_r$ and $\mathbf{B}_r$, $r \geq 1$?

No general technique for this task exists.

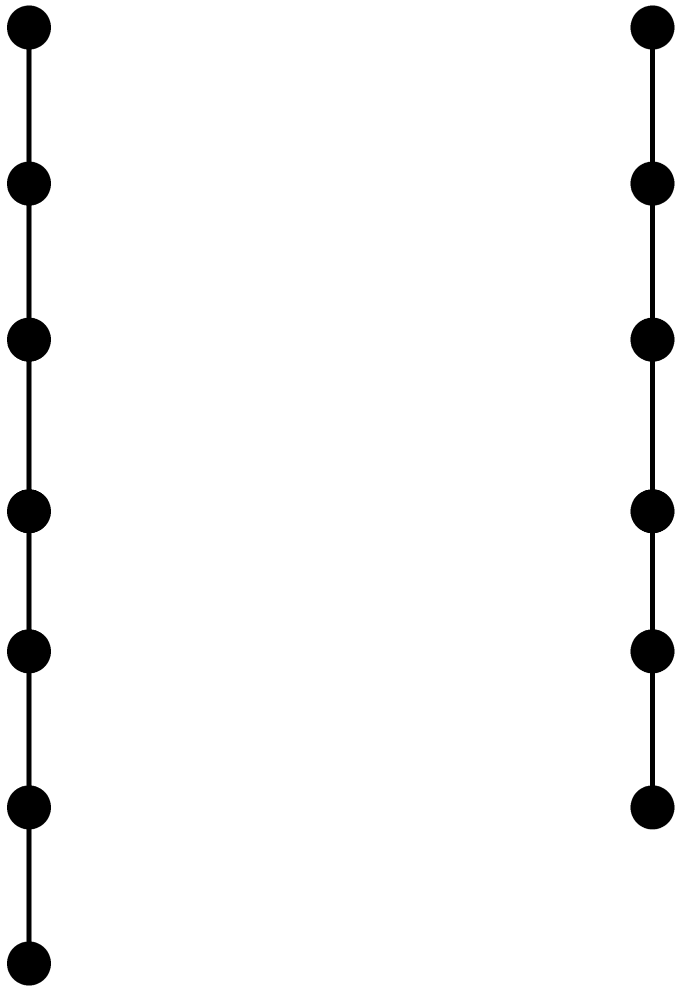
II. After candidates $\mathbf{A}_r$ and $\mathbf{B}_r$ have been found, how does one show that $\mathbf{A}_r \sim_r \mathbf{B}_r$?

In general, it is quite difficult to describe rigorously the winning strategy of the *Duplicator.* However,

- In a few cases, it is possible to give explicit descriptions of the $\sim_r$-equivalence classes.

  In particular, we can do so for *Linear Orders.*
- It is possible to build a "library" of winning strategies.

Example

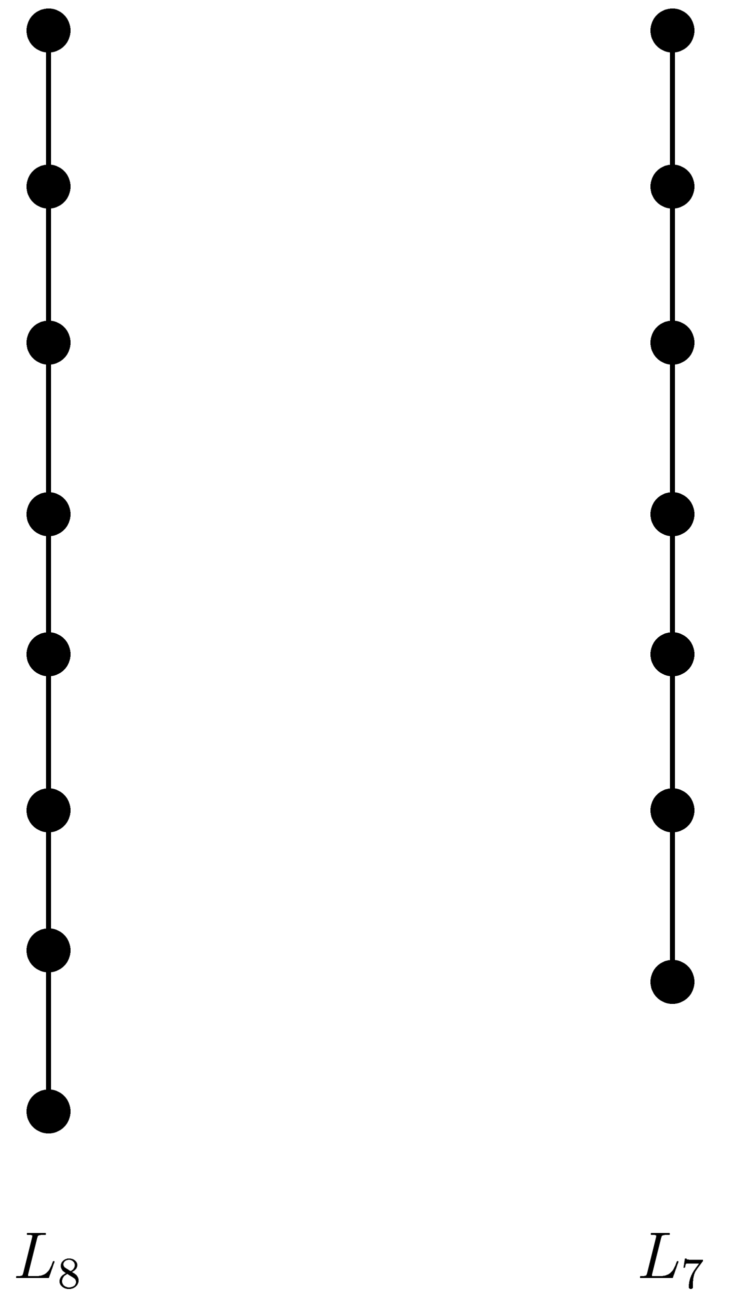


L_7 $\qquad\qquad$ L_6

$$L_7 \not\sim_3 L_6$$

Spoiler wins the 3-move E-F game on L_7 and L_6

Example



$$L_8 \sim_3 L_7$$

Duplicat. wins the 3-move E-F game on L_8 and L_7

Analysis of $\sim_r$ on Linear Orders

Theorem: The following are equivalent:

- $L_m \sim_r L_n$
- $(m = n)$ or $(m \geq 2^r - 1$ and $n \geq 2^r - 1)$

Proof: By induction on $\min(m, n)$, using

Lemma: The following are equivalent:

- $L_m \sim_{r+1} L_n$
- for every $a \in L_m$ there is $b \in L_n$ such that

  $$L_m^{>a} \sim_r L_n^{>b} \text{ and } L_m^{<a} \sim_r L_n^{<b}; \text{ and}$$

  for every $b \in L_n$ there is $a \in L_m$ such that

  $$L_m^{>a} \sim_r L_n^{>b} \text{ and } L_m^{<a} \sim_r L_n^{<b}. \ \blacksquare$$

Corollary: Even Cardinality is *not* first-order definable logic on finite linear orders.

Proof: $L_m \sim_r L_{m+1}$, where $m \geq 2^r - 1$.

Towards a Library of Winning Strategies

Goal: To identify general sufficient conditions for the *Duplicator* to win E-F games.

Some History:

- Gaifman (1982):

  First-order logic can express *local* properties only.

- Fagin, Stockmeyer, Vardi (1993):

  Building on earlier work by Hanf (1965), they developed "off-the-shelf" winning strategies for the *Duplicator* in the E-F game.

- Schwentick (1994), Arora and Fagin (1997):

  Additonal "off-the-shelf" winning strategies.

Neighborhoods and Types

$\mathbf{A} = (A, R_1, \ldots, R_m)$ a σ-structure, $a \in A$, $d \geq 1$.

• The *Gaifman* graph $(A, E_{\mathbf{A}})$ of $\mathbf{A}$ is defined by

$$E_{\mathbf{A}} = \{(b,c) : \text{there exist } R_i \text{ and } (t_1, \ldots, t_s) \in R_i \text{ such that } b \text{ and } c \text{ are among } t_1, \ldots, t_s\}.$$

• $N(a, d)$ = *Neighborhood* of a of radius d.

It consists of a and all points in $\mathbf{A}$ of distance less than d from a in the Gaifman graph $E_{\mathbf{A}}$.

• The *d-type of a* is the isomorphism type of the substructure of $\mathbf{A}$ induced by $N(a, d)$ with a as a distinguished node.

• a and b *have the same d-type* if $N(a, d) \cong N(b, d)$ via an isomorphism h such that $h(a) = b$.

Examples

- *Linear Order* L_n, $a \in L_n$, $d \geq 2$

$$N(a,d) \;=\; L_n$$

- *Clique* K_n, $a \in K_n$, $d \geq 2$

$$N(a,d) \;=\; K_n$$

- *Complement of Clique* $\overline{K}_n$, $a \in K_n$, $d \geq 2$

$$N(a,d) \;=\; \{a\}$$

- *Cycle* C_n, $a \in C_n$, d such that $2d \leq n$

$$\begin{aligned} N(a,d) \;&=\; \text{path of length } 2d-1 \\ &\qquad \text{with } a \text{ as midpoint} \end{aligned}$$

Winning Strategies for the Duplicator

Definition: $\mathbf{A}$ and $\mathbf{B}$ two σ-structures, $d \geq 1$.

$\mathbf{A}$ and $\mathbf{B}$ are *d-equivalent* if for each d-type they have the same # of neighborhoods of that type.

Theorem: Fagin, Stockmeyer, Vardi (1993):

For every $r \geq 1$ there is a $d \geq 1$ such that if $\mathbf{A}$ and $\mathbf{B}$ are d-equivalent, then $\mathbf{A} \sim_r \mathbf{B}$.

In fact, every $d \geq 3^r$ has this property.

Hint of Proof: *Duplicator* can win the r-move E-F game by maintaining the following stronger condition in every round $a_1, b_1, \ldots, a_r, b_r$:

$$\bigcup_{i \leq j} N(a_i, 3^{r-j}) \cong \bigcup_{i \leq j} N(b_i, 3^{r-j})$$

via an isomorphism such that $a_i \mapsto b_i$, $i \leq j \leq r$. ∎

A Simpler Methodology

Corollary:

To show that a Boolean query Q is *not* first-order definable on $\mathcal{C}$, suffices to show that for every r there are structures $\mathbf{A}_r$ and $\mathbf{B}_r$ in such that

- $\mathbf{A}_r \models Q$ and $\mathbf{B}_r \not\models Q$
- $\mathbf{A}_r$ is d-equivalent to $\mathbf{B}_r$ for some $d \geq 3^r$.

Remarks:

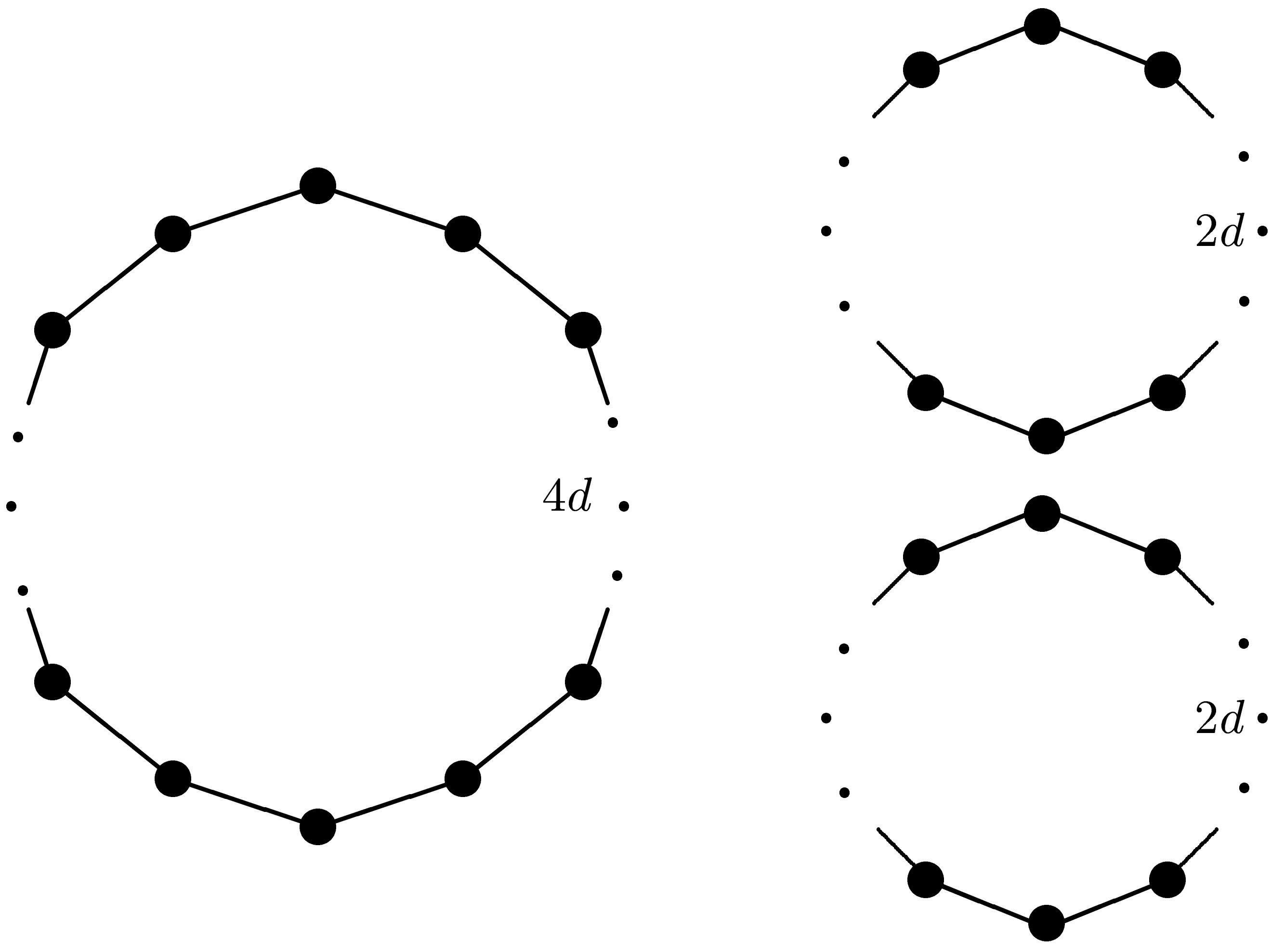
- This method is *sound*, but it is not *complete.*
- It can not be used for first-order definability on linear orders, since for every $d \geq 2$

$$L_m \text{ is } d\text{-equivalent to } L_n \iff m = n.$$

- This method, however, simplifies the combinatorial arguments. Moreover, it provides a clue for finding candidate structures $\mathbf{A}_r$ and $\mathbf{B}_r$.

Theorem: CONNECTIVITY is *not* first-order definable on finite graphs.

Proof: By picture, where d is any number $\geq 3^r$.

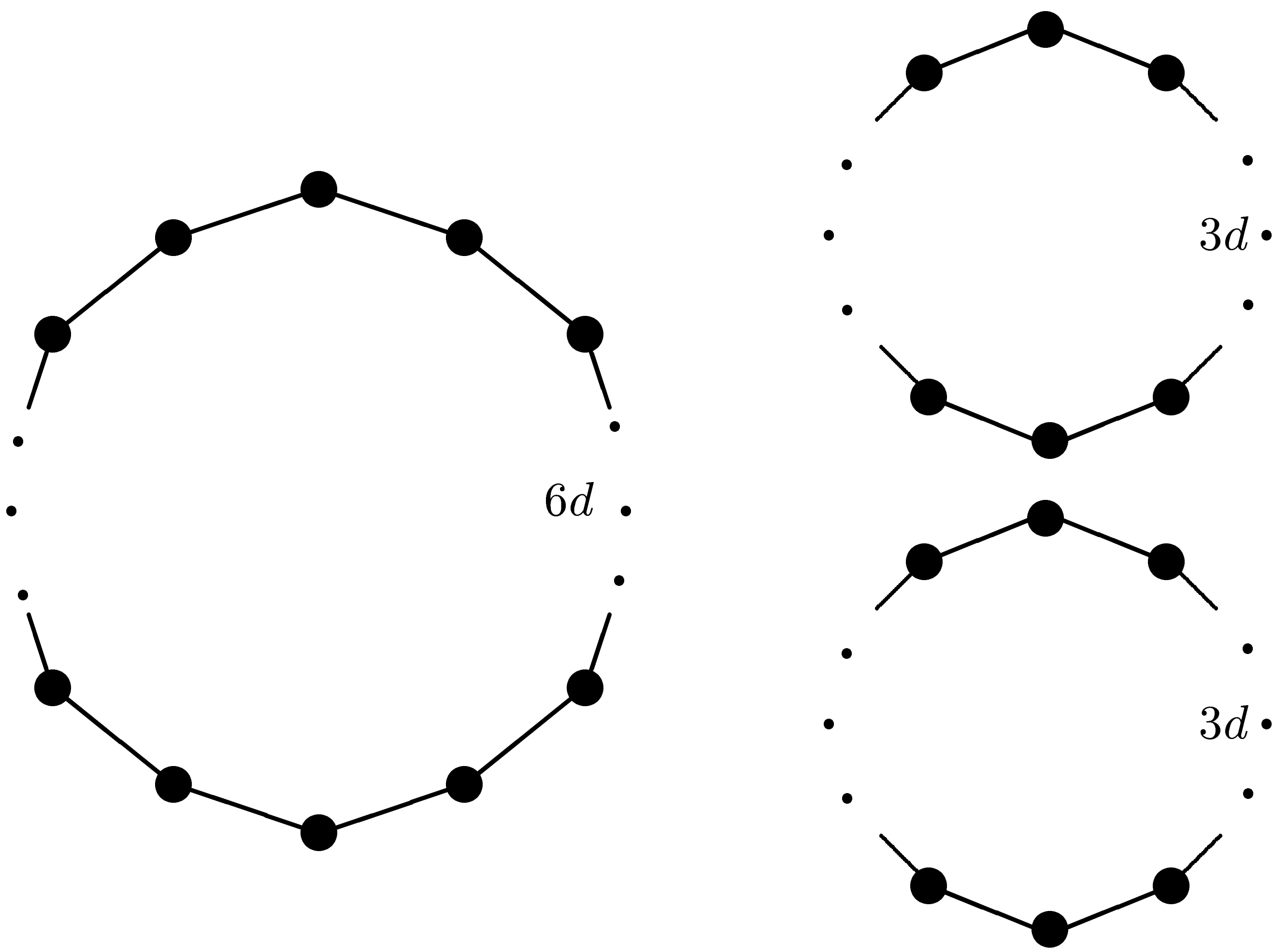


$\mathbf{A}_r$ $\mathbf{B}_r$

- Each d-type is a path of length $2d - 1$
- $\mathbf{A}_r$ is d-equivalent to $\mathbf{B}_r$

Theorem: 2-COLORABILITY is *not* first-order definable on finite graphs.

Proof: By picture, where $d = 3^r$ (an odd number).

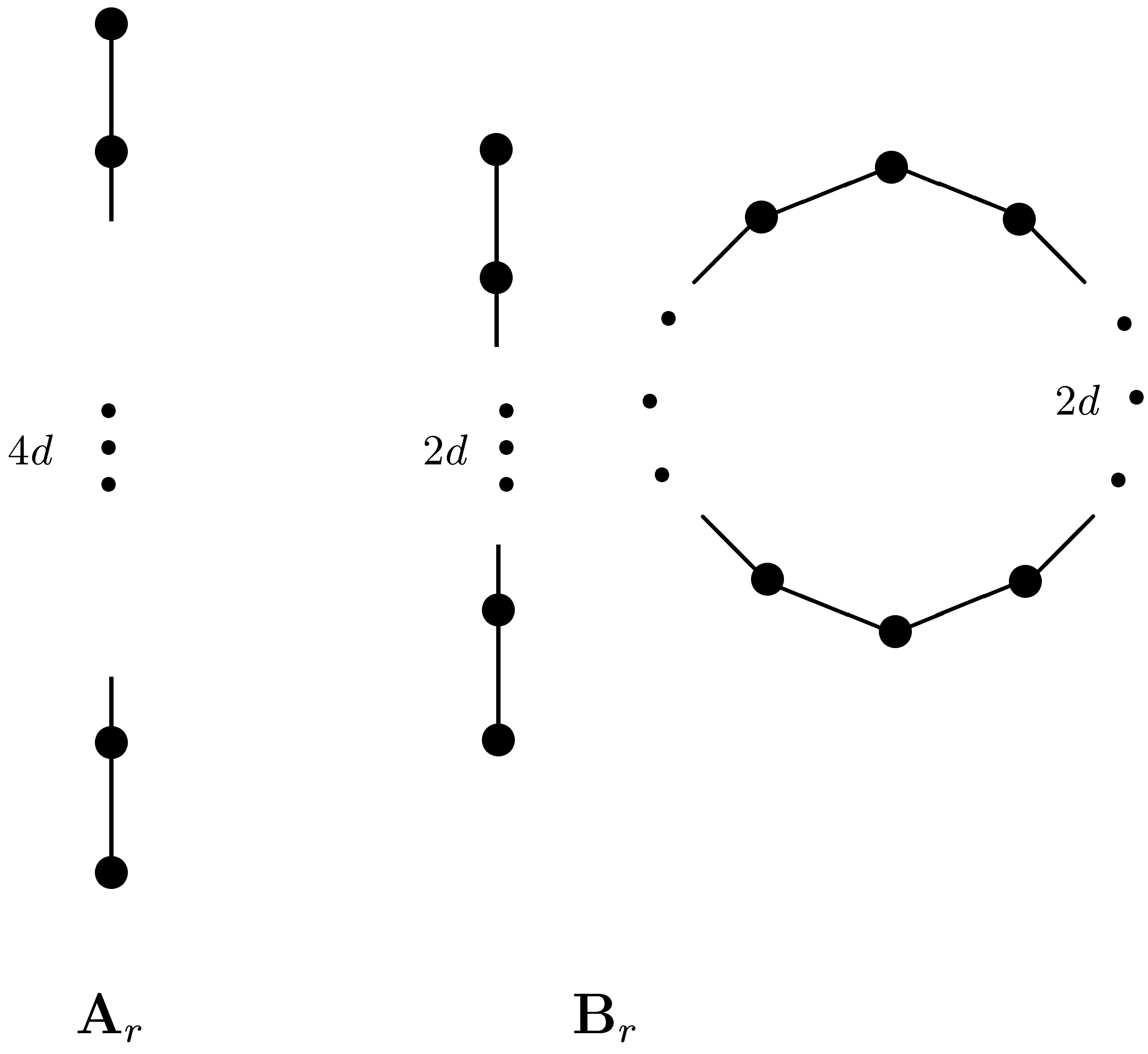


$\mathbf{A}_r$ $\mathbf{B}_r$

- Each d-type is a path of length $2d - 1$
- $\mathbf{A}_r$ is d-equivalent to $\mathbf{B}_r$

Theorem: Acyclicity is *not* first-order definable on finite graphs.

Proof: Again by picture, where $d \geq 3^r$.



- $\mathbf{A}_r$ is d-equivalent to $\mathbf{B}_r$

Homework Assignment #1

Show that the following graph properties are *not* first-order expressible:

- k-Colorability, for each fixed $k \geq 3$
- Planarity
- Rigidity (no non-trivial automorphims)
- Connectivity on linearly ordered graphs

Hint:

Use pictures for the first three properties; use the analysis of $\sim_r$ on linear orders for the last one.

Solution to Homework Assignment #1

Fact: CONNECTIVITY is *not* first-order expressible on lineary ordered graphs.

Proof: Turn each linear order L_n into a graph G_n by defining an edge relation E_n as follows:

$$E_n(a,b) \iff (b = \text{“}a+2\text{”}) \vee (a = n \wedge b = 1)$$

Then,

$$G_n \text{ is connected} \iff n \text{ is even.}$$

Note that E_n is first-order definable from $<$ and recall that EVEN CARDINALITY is not first-order on linear orders. ∎

Computational Complexity

Computational Complexity:

- *The quantitative study of solvability*

  Y. Hartmanis – 1993
- Solvable decision problems are placed into classes according to the time/space resources required to solve them.

Major Complexity Classes:

L	logarithmic space
P	polynomial time
NP	nondeterministic polynomial time
PSPACE	polynomial space
EXPTIME	exponential time
NEXPTIME	nondeterministic exponential time

Complexity Classes

Fact: The following containments hold:

$L \subseteq P \subseteq NP \subseteq PSPACE \subseteq EXPTIME \subseteq NEXPTIME$

Open Problem: Are these containment proper?

Fact: The following containments are proper:

- $L \subsetneqq PSPACE$
- $P \subsetneqq EXPTIME$
- $NP \subsetneqq NEXPTIME$.

Proof: Use time/space *hierarchy* theorems. ∎

Note: Each of the classes L, P, and PSPACE is closed under complements.

Open Problem:

- Is NP closed under complements? In other words, is NP = coNP?
- Is NEXPTIME closed under complements?

Complete Problems

Definition: Let C be a complexity class and Q a decision problem. We say that Q is C-*complete* if

1. $Q \in C$.
2. Q is C-*hard*, i.e., for every $Q' \in C$, there a "suitable" many-one reduction f of Q' to Q:

$$x \in Q' \Longleftrightarrow f(x) \in Q.$$

- For P: "suitable" means that f is computable in logarithmic space.
- For NP and beyond: "suitable" means that f is computable in polynomial time.

Example: The following problems are NP-complete:

SAT, 3-COLORABILITY, HAMILTONICITY, ...

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 Q_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 $Q_{\mathbf{A}}$, 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 Q_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 Q_ψ is C-complete.

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

The Complexity of First-Order Logic

Theorem: The following hold for first-order logic:

- The data complexity of FO is in L.
- 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 Ehrenfeucht-Fraïssé Games

Proposition: Let r be a fixed positive integer.

The following problem is in L (hence, also in P):

Given two finite structures **A** and **B**, does the Duplicator win the r-move E-F game on **A** and **B**?

Proof:

- Each $\equiv_r$-equivalence class is FO-definable.
- The data complexity of FO is in L. ∎

Theorem: Pezzoli – 1998

The following problem is PSPACE-complete:

Given $r \geq 1$ and two finite structures **A** and **B**, does the Duplicator win the r-move E-F game on **A** and **B**?

Hint of Proof: Difficult reduction from QBF. ∎

Beyond First-Order Logic

Limitations of First-Order Logic

Fact: First-order logic FO has severely limited expressive power on the class $\mathcal{G}$ of all finite graphs.

Recall that none of the following PTIME-queries is FO-definable on $\mathcal{G}$:

Connectivity, Acyclicity, 2-Colorability, Planarity, Eulerian, Even Cardinality, ...

Question: How can we increase the expressive power of first-order logic?

Answer: Augment FO with one of the following:

- Second-order quantification
- Infinitary connectives
- Generalized quantifiers
- Fixed-point operators

Second-Order Logic and its Fragments

- *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 $\varphi(\overline{x}, S_1, \ldots, S_m)$ is first-order.

- *Universal Second-Order Logic* USO:

  $$(\forall S_1) \cdots (\forall S_m) \varphi(\overline{x}, S_1, \ldots, S_m),$$

  where $\varphi(\overline{x}, S_1, \ldots, S_m)$ is first-order.

- *Monadic Second-Order Logic* MSO:

  All second-order quantifiers involve unary relation variables, i.e., they range over subsets of the universe of structures.

Second-Order Definable Queries

Monadic Existential Second Order Logic

- Disconnectivity of graphs $\mathbf{G} = (V, E)$:

$$(\exists S)((\exists x)S(x) \land (\exists y)\neg S(y) \land$$
$$(\forall z)(\forall w)(S(z) \land \neg S(w) \rightarrow \neg E(z, w))).$$

- 2-Colorability of graphs $\mathbf{G} = (V, E)$:

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

- k-Colorability, $k \geq 3$, of graphs $\mathbf{G} = (V, E)$:

Monadic Universal Second-Order Logic

- Well-Founded partial orders $\mathbf{A} = (A, \preceq)$:

$$(\forall S)((\exists x)S(x) \rightarrow (\exists y)(S(y) \land (\forall z)(S(z) \rightarrow y \preceq z))).$$

Second-Order Definable Queries

Existential Second-Order Logic

- HAMILTONICITY of graphs $\mathbf{G} = (V, E)$:

  $$(\exists S)((\text{“}S \text{ is a linear order on } V\text{”})\wedge$$

  $$(\forall x)(\forall y)(\text{“}y = x + 1\text{”} \rightarrow E(x, y))),$$

  where S is a binary relation symbol.

Universal Second-Order Logic

- RIGIDITY of graphs $\mathbf{G} = (V, E)$:

  $$(\forall S)(\text{“}S \text{ encodes an auto. of } \mathbf{G}\text{”} \rightarrow (\forall x)S(x, x)),$$

  where S is a binary relation symbol.

Universal Second-Order Logic

Note: Circumscription is a formalism of commonsense reasoning introduced by McCarthy in 1980.

The main focus is on *minimal models*, since they have as few "exceptions" as possible; thus, they embody commonsense.

Definition: Let $\varphi(S)$ be a first-order formula in which S is a k-ary relation symbol.

The *circumscription* of $\varphi(S)$ is the universal second-order formula below asserting that S is a *minimal relation* satisfying $\varphi(S)$

$$\varphi(S) \wedge (\forall T)(T \subsetneq S \rightarrow \neg\varphi(T)).$$

The Complexity of ESO

Theorem: 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. ∎

Note: Actually, a much stronger result holds ...

Descriptive Complexity

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, $\mathrm{NP}[\mathcal{F}] = \mathrm{ESO}[\mathcal{F}]$. ∎

Remarks: Features of Fagin's Theorem:

- Reveals that ESO *captures* NP on finite graphs.
- Provides a logic-based and machine-independent characterization of NP.
- Constitutes the prototypical result of *descriptive complexity*.

NP and Existential Second-Order Logic

Problem: Is NP closed under complements?

(If not, then P $\neq$ NP.)

Corollary: The following are equivalent:

- NP is closed under complements.
- ESO on finite structures is closed under complements.
- 3-COLORABILITY is USO-definable.
- HAMILTONICITY is USO-definable.

Proof:

Use Fagin's Theorem and the NP-completeness of 3-COLORABILITY and HAMILTONICITY. ∎

Combinatorial Games and ESO Logic

Questions:

- Can definability in ESO be characterized in terms of combinatorial games?

- If so, can these games be used to obtain lower bounds for definability in ESO and ultimately resolve the NP $\stackrel{?}{=}$ coNP problem?

Answers:

- Yes, definability in ESO can be characterized using variants of Ehrenfeucht-Fraïssé games.

- So far, there has been very limited success in this direction:

  Only lower bounds for definability in monadic ESO have been obtained using these games.

Ehrenfeucht-Fraïssé Games for ESO Logic

Definition: Let $s_1, \ldots, s_k, r$ be positive integers and let $\mathbf{A}$, $\mathbf{B}$ be two σ-structures.

The $(\langle s_1, \ldots, s_k \rangle, r)$-*Ehrenfeucht-Fraïssé-game on* $\mathbf{A}$ *and* $\mathbf{B}$ is played as follows:

- The *Spoiler* picks relations $S_1, \ldots, S_k$ of arities $s_1, \ldots, s_k$ on A.
- The *Duplicator* picks relations $S'_1, \ldots, S'_k$ of arities $s_1, \ldots, s_k$ on B.
- After this, the two players play an r-move E-F game on $(\mathbf{A}, S_1, \ldots, S_k)$ and $(\mathbf{B}, S'_1, \ldots, S'_k)$.

Proposition: Let Ψ be an ESO-sentence of the form $(\exists P_1) \cdots (\exists P_k)\psi$, where each P_i is a relation symbol of arity s_i and ψ is a first-order sentence of quantifier rank r.

If $\mathbf{A} \models \Psi$ and the Duplicator wins the $(\langle s_1, \ldots, s_k \rangle, r)$-E-F game on $\mathbf{A}$ and $\mathbf{B}$, then $\mathbf{B} \models \Psi$. ∎

Ehrenfeucht-Fraïssé Games for ESO Logic

A Methodology for ESO-Definability

Let Q be a Boolean query on a class $\mathcal{C}$.

Soundness: To show that Q is *not* ESO-definable on $\mathcal{C}$, suffices to show that for every $s_1, \ldots, s_k, r$, there are structures $\mathbf{A}$ and $\mathbf{B}$ in $\mathcal{C}$ such that

- $\mathbf{A} \models Q$ and $\mathbf{B} \not\models Q$.
- *Duplicator* wins the $(\langle s_1, \ldots, s_k \rangle, r)$-E-F game on $\mathbf{A}$, $\mathbf{B}$.

Completeness: This method is also *complete*, i.e., if Q is *not* ESO-definable on $\mathcal{C}$, then this can be shown using these games.

NP vs. coNP via Combinatorial Games

Corollary: The following are equivalent.

- NP $\neq$ coNP.
- For every $s_1, \ldots, s_k, r$, there are finite graphs **G** and **H** such that
  - **G** is not 3-Colorable, **H** is 3-Colorable.
  - The *Duplicator* wins the $(\langle s_1, \ldots, s_k \rangle, r)$-E-F game on **G** and **H**. ∎

Remark:

- Formidable combinatorial difficulties arise when one of the s_i's is bigger than 1.
- Nonetheless, this method has made it possible to obtain lower bounds for definability in monadic ESO (each $s_i = 1$, $1 \leq i \leq k$).

Ajtai-Fagin Games

Note:

- In applying the method of E-F games or one of their variants, one can view the selection of the structures $\mathbf{A}$ and $\mathbf{B}$ as being part of the Duplicator's moves.

- A main difficulty with the $(\langle s_1, \ldots, s_k \rangle, r)$-E-F games is that in effect the Duplicator has to select the structure $\mathbf{B}$ *before* the Spoiler has picked relations $S_1, \ldots, S_k$ on $\mathbf{A}$.

- In 1990, Ajtai and Fagin introduced a variant of these games in which the Duplicator selects the structure $\mathbf{B}$ *after* the Spoiler has picked relations $S_1, \ldots, S_k$ on $\mathbf{A}$.

  In general, this makes the task of the Duplicator easier.

Ajtai-Fagin Games for Monadic ESO Logic

Definition: $\mathcal{C}$ a class of σ-structures, Q a Boolean query on $\mathcal{C}$, and positive integers k, r.

The (k, r)-*Ajtai-Fagin game for* Q *on* $\mathcal{C}$ is played as follows:

- *Duplicator* picks a structure $\mathbf{A} \in \mathcal{C}$ such that $\mathbf{A} \models Q$.
- *Spoiler* picks k-unary relations $S_1, \ldots, S_k$ on $\mathbf{A}$ (i.e., $S_1, \ldots, S_k$ are subsets of A).
- *Duplicator* first picks a structure $\mathbf{B} \in \mathcal{C}$ such that $\mathbf{B} \not\models Q$, and then picks k unary relations $S'_1, \ldots, S'_k$ on $\mathbf{B}$.
- After this, the two players play an r-move E-F game on $(\mathbf{A}, S_1, \ldots, S_k)$ and $(\mathbf{B}, S'_1, \ldots, S'_k)$.

Ajtai-Fagin Games for Monadic ESO

A Methodology for Monadic ESO-Definability

Let Q be a Boolean query on a class $\mathcal{C}$.

Soundness: To show that Q is *not* monadic ESO-definable on $\mathcal{C}$, suffices to show that for every k and every r, the *Duplicator* wins the (k, r)-Ajtai-Fagin game on $\mathcal{C}$.

Completeness: This method is also *complete*, i.e., if Q is *not* monadic ESO-definable on $\mathcal{C}$, then this can be shown using this method.

Lower Bounds for Monadic ESO-Definability

Theorem: Fagin – 1975

CONNECTIVITY is *not* monadic ESO definable on finite graphs

Proof:

- Original proof used E-F games for monadic ESO.
- Fagin, Stockmeyer and Vardi (1993) obtained a simpler proof using Ajtai-Fagin games for monadic ESO and d-equivalence. ∎

Corollary: Monadic ESO is *not* closed under complements on finite graphs. ∎

Remark: No implications for the NP $\stackrel{?}{=}$ coNP problem can be derived from this result.

Lower Bounds for Monadic ESO-Definability

Theorem: Connectivity is *not* monadic ESO definable on finite graphs

Proof: Let k be a positive integer, $\mathbf{A} = (A, E)$ a cycle, and $S_1, \ldots, S_k$ unary relations on A.

- For $b \in A$, the *color* of b is defined as the Boolean vector $c(b) = (c_1, \ldots, c_k)$ such that if $b \in S_i$, then $c_i = 1$; otherwise, $c_i = 0$.

- The number of colors depends only on k.

- For every $d \geq 1$ and every $a \in A$, the neighborhood $N(a, d)$ of a in $(\mathbf{A}, S_1, \ldots, S_k)$ consists of $2d - 1$ points whose distance from a in $\mathbf{A}$ is at most d, and is completely determined by the colors of these points.

- The number of different d-types depends only on k and d (and not on $|A|$).

The *Duplicator* wins the (k, r)-Ajtai-Fagin game for CONNECTIVITY on finite graphs as follows:

- *Duplicator* picks a large cycle $\mathbf{A}$ so that for every $S_1, \ldots, S_k$ on A there are $\geq 4d$ points with the same d-type in $(\mathbf{A}, S_1, \ldots, S_k)$, where $d = 3^{r-1}$.
- After *Spoiler* picks $S_1, \ldots, S_k$, there are two nodes a_p and a_q in A having the same d-type and $N(a_p, 2d) \cap N(a_q, 2d) = \emptyset$.
- *Duplicator* constructs $\mathbf{B} = \mathbf{B}_0 \oplus \mathbf{B}_1$ consisting of two disjoint cycles $\mathbf{B}_0$ and $\mathbf{B}_1$:
  - Disconnects $\mathbf{A}$ by "pinching" it at a_p, a_q.
  - Creates $\mathbf{B}_0$ by joining a_p and a_{q+1} in the first component.
  - Creates $\mathbf{B}_1$ by joining a_{p+1} and a_q in the second component.
- *Duplicator* picks the same sets $S_1, \ldots, S_k$ in $\mathbf{B}$.
- $\mathbf{A}$ and $\mathbf{B}$ are d-equivalent, as each node in $\mathbf{A}$ has the same d-type as its clone in $\mathbf{B}$. ∎

Lower Bounds for Monadic ESO-Definability

Theorem: de Rougemont – 1983

CONNECTIVITY is not monadic ESO-definable on finite graphs $\mathbf{G} = (V, E, \mathrm{Suc})$, where Suc is the graph of a *successor* function on V. ∎

Theorem: Schwentick – 1994

CONNECTIVITY is not monadic ESO-definable on *ordered* finite graphs $\mathbf{G} = (V, E, \leq)$, where $\leq$ is a linear order on V. ∎

Open Problem: Show that there is a Boolean query Q on finite graphs such that

- Q is in NP (hence, Q is ESO-definable).
- Q is not binary ESO-definable, i.e., Q is not definable by any ESO-sentence $(\exists P_1) \cdots (\exists P_k)\psi$, where each P_i is a binary relation symbol.

Fixed-Point Logics

Finite Variable Logics

and

Pebble Games

Limitations of First-Order Logic

Fact: First-order logic FO has severely limited expressive power on the class $\mathcal{G}$ of all finite graphs.

Recall that none of the following PTIME-queries is FO-definable on $\mathcal{G}$:

Connectivity, Acyclicity, 2-Colorability, Planarity, Eulerian, Even Cardinality, ...

Question: How can we increase the expressive power of first-order logic?

Answer: Augment FO with one of the following:

- Second-order quantification
- Infinitary connectives
- Generalized quantifiers
- Fixed-point operators

Fixed-Point Operators

Note: In effect, FO on $\mathcal{G}$ has limited expressive power because it does not embody *recursion*.

In general, recursion can be introduced via fixed-points of suitable operators.

Example: *Factorial* function

$$\left|\begin{array}{lcl} f(o) & = & 1 \\ f(n+1) & = & (n+1)f(n) \end{array}\right.$$

But also the factorial function is a fixed-point of the recursive specification

$$f \quad = \quad \lambda n.\,(n = 0 \to 1 \,\Box\, (n+1)f(n))\,.$$

Idea: Add recursion to first-order logic by augmenting it with fixed-points of first-order formulas.

Fixed-Points of First-Order Formulas

Let $\varphi(x_1, \ldots, x_k, S)$ be first-order over $\sigma \cup \{S\}$, where S is a k-ary relation symbol not in σ.

- On every σ-structure $\mathbf{A}$, it gives rise to an operator $\Phi : \mathcal{P}(A^k) \to \mathcal{P}(A^k)$, where

  $$\Phi(P) \;=\; \{(a_1, \ldots, a_k) : \mathbf{A} \models \varphi(a_1, \ldots, a_k, P)\}$$

- P is a *fixed-point* of Φ (and of φ) if $P = \Phi(P)$.
- In general, φ may have no fixed points, e.g.

  $$\varphi(x, S) \;\equiv\; \neg S(x).$$

- We say that Φ is *monotone* if

  $$S_1 \subseteq S_2 \;\Longrightarrow\; \Phi(S_1) \subseteq \Phi(S_2).$$

- If $\varphi(x_1, \ldots, x_k, S)$ is *positive* in S, then Φ is monotone.

Least Fixed-Points of Monotone Operators

Theorem: Tarski–Knaster

Every monotone operator Φ has a *least* fixed-point.

Moreover, its least fixed-point can be obtained by iterating Φ.

Definition: The *stages* of Φ are defined by

$$\left| \begin{array}{lcl} \Phi^1 & = & \Phi(\emptyset) \\ \Phi^{n+1} & = & \Phi(\Phi^n) \end{array} \right.$$

Fact: If Φ is monotone, then

- $\Phi^1 \subseteq \Phi^2 \subseteq \Phi^n \subseteq \Phi^{n+1} \subseteq \ldots$.
- On the class $\mathcal{F}$ of all finite σ-structures the least-fixed point of Φ is equal to $\bigcup_{n=1}^{\infty} \Phi^n$.
- Moreover, for every finite σ-structure $\mathbf{A}$ there is an integer $s \leq |A|^k$ such that

$$\mathbf{A} \models \Phi^s = \Phi^{s+1} = \bigcup_{n=1}^{\infty} \Phi^n.$$

Least Fixed-Points of Positive Formulas

Notation: Let $\varphi(x_1, \ldots, x_k, S)$ be a positive first-order formula.

- $\varphi^n(x_1, \ldots, x_k)$ is a first-order formula defining the n-th stage Φ^n, $n \geq 1$, of the operator Φ.
- $\varphi^\infty(x_1 \ldots, x_k)$ is a new formula with semantics given by the least fixed-point of $\varphi(x_1, \ldots, x_k, S)$.

Example: Transitive Closure

$$\begin{aligned}
\varphi(x, y, S) &\equiv E(x, y) \vee (\exists z)(E(x, z) \wedge S(z, y)) \\
\varphi^n(x, y) &\equiv \text{there is a path of length} \\
& \text{at most } n \text{ from } x \text{ to } y \\
\varphi^\infty(x, y) &\equiv \text{there is a path from } x \text{ to } y
\end{aligned}$$

Least Fixed-Point Logic

Fact:

By simultaneous induction, all concepts extend to *systems* of first-order formulas

$$\varphi_1(\overline{x}_1, S_1, \ldots, S_m), \ldots, \varphi_m(\overline{x}_m, S_1, \ldots, S_m)$$

provided $\text{arity}(\overline{x}_i) = \text{arity}(S_i)$, $1 \leq i \leq m$.

In particular, every system of positive first-order formulas has a least fixed-point $(\varphi_1^\infty, \ldots, \varphi_m^\infty)$.

Definition: Least Fixpoint Logic LFP

All formulas φ_m^∞ for some system $(\varphi_1, \ldots, \varphi_m)$ of positive first-order formulas.

Fact:

LFP can be equivalently defined as the smallest collection of formulas containing every FO-formula, φ^∞ for every positive first-order formulas φ, and closed under $\wedge$, $\vee$, $\exists$, and $\forall$.

Examples

- CONNECTIVITY of finite graphs $\mathbf{G} = (V, E)$.

$$\begin{aligned} \varphi(x, y, S) &\equiv E(x, y) \vee (\exists z)(E(x, z) \wedge S(z, y)) \\ \varphi^{\infty}(x, y) &\equiv \text{there is a path from } x \text{ to } y \end{aligned}$$

Corollary: CONNECTIVITY is LFP-definable.

$$\mathbf{G} \text{ is connected} \iff \mathbf{G} \models (\forall x)(\forall y)\varphi^{\infty}(x, y).$$

- ACYCLICITY of finite graphs $\mathbf{G} = (V, E)$.

$$\begin{aligned} \psi(x, S) &\equiv (\forall y)(E(y, x) \to S(y)) \\ \psi^{1}(x) &\equiv \text{indegree}(x) = 0 \\ \psi^{2}(x) &\equiv (\forall y)(E(y, x) \to \text{indegree}(y) = 0) \\ &\vdots \\ \psi^{\infty}(x) &\equiv \text{no path through } x \text{ leads to a cycle} \end{aligned}$$

Corollary: ACYCLICITY is LFP-definable.

$$\mathbf{G} \text{ is acyclic} \iff \mathbf{G} \models (\forall x)\psi^{\infty}(x)$$

More Examples

- GEOGRAPHY: Game on finite graphs $\mathbf{G} = (V, E)$

$$\begin{array}{c}\texttt{HELSINKI}\\ \swarrow\\ \texttt{ISTANBUL}\\ \swarrow\\ \texttt{LISBON}\\ \swarrow\\ ?\end{array}$$

$$\begin{array}{rcl}\chi(x, S) & \equiv & (\forall y)\neg E(x, y) \vee \\ & & (\forall y)(E(x, y) \rightarrow (\exists z)(E(y, z) \wedge S(z))) \\ \chi^{\infty}(x) & \equiv & x \text{ is a winning position for Player I}\end{array}$$

Note: GEOGRAPHY is a P-complete problem.

The Complexity of LFP

Theorem: Data complexity of LFP is P-complete.

Proof:

- Fix a positive FO-formula $\varphi(x_1, \ldots, x_k, S)$.
  Given a finite $\mathbf{A}$, the least fixed-point φ^∞ is polynomial-time computable on $\mathbf{A}$, because
  - There are at most $|A|^k$ distinct stages φ^n.
  - φ^{n+1} can be computed from φ^n in polynomial time (data complexity of FO is in P).
- GEOGRAPHY is P-complete and LFP-definable.

Theorem: The combined complexity of LFP and the expression complexity of LFP is EXPTIME-complete.

LFP vs. Polynomial Time

Fact: If $\mathcal{C}$ is a class of finite σ-structures, then

$$\mathrm{FO}[\mathcal{C}] \ \subseteq \ \mathrm{LFP}[\mathcal{C}] \ \subseteq \ \mathrm{PTIME}[\mathcal{C}],$$

where $\mathrm{PTIME}[\mathcal{C}]$ is the collection of all polynomial-time computable queries on $\mathcal{C}$.

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

$$\mathrm{FO}[\mathcal{G}] \ \subsetneq \ \mathrm{LFP}[\mathcal{G}] \ \subsetneq \ \mathrm{PTIME}[\mathcal{G}].$$

Proof:

- CONNECTIVITY $\in$ $\mathrm{LFP}[\mathcal{G}] - \mathrm{FO}[\mathcal{G}]$.
- EVEN CARDINALITY $\in$ $\mathrm{PTIME}[\mathcal{G}] - \mathrm{LFP}[\mathcal{G}]$ (to be proved later). ∎

Note: LFP does *not* capture P on $\mathcal{G}$. However,

Descriptive Complexity Theory

Notation: $\mathcal{O}$ is the class class of all *ordered* finite σ-structures $\mathbf{A} = (A, R_1, \ldots, R_m, <)$, where $<$ is a linear order on A.

Theorem: Immerman (1982), Vardi(1982)

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

- Q is in P.
- Q is LFP-definable on $\mathcal{O}$.

In other words, PTIME[$\mathcal{O}$] $=$ LFP[$\mathcal{O}$]. ∎

Note: In Fagin's Theorem, we can "guess" a linear order using an existential quantifier $(\exists T)$, where T is a binary relation symbol.

Beyond Least Fixed-Point Logic

Note:

LFP is based on the fact that every positive formula has a least fixed-point.

Questions:

- Can meaningful fixed-point semantics be given to recursive specifications that involve arbitrary first-order formulas?
- If fixed-points of arbitrary first-order definable operators are considered, is there a gain in expressive power?

Answers:

- The answer to the first question is “yes”.
  PFP: Partial Fixed-Point Logic.
- The answer to the second question is more complicated ...

Partial Fixed-Points

$\varphi(x_1, \ldots, x_k, S)$ an arbitrary first-order formula

- Recall the operator $\Phi : \mathcal{P}(A^k) \to \mathcal{P}(A^k)$, where

  $\Phi(P) \;=\; \{(a_1, \ldots, a_k) : \mathbf{A} \models \varphi(a_1, \ldots, a_k, P)\}.$

- Recall also the *stages* of Φ

$$\left|\begin{array}{lcl} \Phi^1 & = & \Phi(\emptyset) \\ \Phi^{n+1} & = & \Phi(\Phi^n) \end{array}\right.$$

- $\Phi^1, \Phi^2, \ldots$ may *not* converge to a fixed-point.

Definition: Abiteboul and Vianu (1989)

The *partial fixed-point* of $\varphi(x_1, \ldots, x_k, S)$

$$\varphi^\infty = \begin{cases} \Phi^s & \text{if } \mathbf{A} \models \Phi^s = \Phi^{s+1} \text{ for some } s \\ \emptyset & \text{otherwise} \end{cases}$$

Partial Fixed-Point Logic

Definition: Partial Fixed-Point Logic PFP:

All formulas φ_m^∞, for some system $(\varphi_1, \ldots, \varphi_m)$, with partial fixed-point semantics.

Example: Consider the system

$$\left|\begin{array}{lcl} \varphi(x,y,S,T) & \equiv & ((\exists z)(E(x,z) \wedge E(z,y)) \wedge \\ & & \qquad (\forall u)(\forall v)\neg S(u,v)) \vee \\ & & \qquad (\exists z)(S(x,z) \wedge S(z,y)) \\ \psi(x,y,S,T) & \equiv & S(x,y) \vee T(x,y) \end{array}\right.$$

Then

- $\varphi^n(a,b) \iff$

  there is a path of length 2^n from a to b.

- $\psi^\infty(a,b) \iff$

  there is a path of length a power of 2 from a to b.

Example

- Consider finite structures of the form $\mathbf{S} = (F, A, R)$, where $A \subseteq F$ and $R \subseteq F^3$.

  Intuitively, F is a set of "formulas", A is a set of axioms, and R is a "rule of inference".

- Let

  $$\theta(x, T) \;\equiv\; A(x) \lor (\exists y)(\exists z)(T(y) \land \neg T(z) \land R(x, y, z))$$

- NONMONOTONIC PATH SYSTEMS query:

  Given $\mathbf{S} = (F, A, R)$ and $s \in F$, is $s \in \varphi^{\infty}$?

Theorem: Grohe (1994)

NONMONOTONIC PATH SYSTEMS is PSPACE-complete.

The Complexity of PFP

Theorem: Data complexity of PFP is PSPACE-complete.

Proof:

- Fix a first-order formula $\varphi(x_1, \ldots, x_k, S)$.

  Given a finite $\mathbf{A}$, we can compute the partial fixed-point φ^∞ in polynomial space as follows:
  - Compute φ^{n+1} from φ^n using polynomial space;
  - Maintain a counter that counts the iterations in binary up to $2^{|A|^k}$ to determine whether the sequence of stages converges to a fixed-point on $\mathbf{A}$.

- NONMONOTONIC PATH SYSTEMS is PSPACE-complete and PFP-definable. ∎

Theorem: The combined complexity of LFP and the expression complexity of LFP is EXPSPACE-complete.

PFP vs. Polynomial Space

Fact: If $\mathcal{C}$ is a class of finite σ-structures, then

$$\mathrm{LFP}[\mathcal{C}] \subseteq \mathrm{PFP}[\mathcal{C}] \subseteq \mathrm{PSPACE}[\mathcal{C}],$$

where PSPACE[$\mathcal{C}$] is the collection of all polynomial-space computable queries on $\mathcal{C}$.

Fact: On the class $\mathcal{F}$ of all finite σ-structures

$$\mathrm{LFP}[\mathcal{F}] \subseteq \mathrm{PFP}[\mathcal{F}] \subsetneqq \mathrm{PSPACE}[\mathcal{F}].$$

Proof:

EVEN CARDINALITY $\in$ PSPACE[$\mathcal{F}$] − PFP[$\mathcal{F}$] (to be proved later). ∎

Note: PFP does *not* capture P on $\mathcal{F}$. However,

Descriptive Complexity Theory

Theorem: Vardi (1982), Abiteboul-Vianu (1989)

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

- Q is in PSPACE
- Q is PFP-definable on $\mathcal{O}$.

In other words, PSPACE$[\mathcal{O}]$ $=$ PFP$[\mathcal{O}]$. ∎

Question:

How do LFP and PFP compare on the class $\mathcal{F}$ of all finite σ-structures?

Problem: Chandra–Harel (1982)

Show that on the class $\mathcal{F}$ of all finite σ-structures

$$\text{LFP}[\mathcal{F}] \neq \text{PFP}[\mathcal{F}].$$

Limitations of Fixed-Point Logics

Question:

- What tools can we use to analyze the expressive power of logics with fixpoint operators?
- How can we establish negative results for definability in least fixed-point logic LFP or in partial fixed-point logic PFP?

  (e.g., $\textsc{Even Cardinality} \notin \mathrm{PFP}(\mathcal{G})$)

Answer:

- Logics with fixed-point operators can be viewed as fragments of *infinitary logics* with *finitely many* variables.
- The expressive power of infinitary logics with finitely many variables can be characterized in terms of *pebble games*.

The Infinitary Logic $\mathrm{L}^{\omega}_{\infty\omega}$

Definition: $\mathrm{L}_{\infty\omega}$ is the extension of first-order logic with arbitrary disjunctions and conjunctions

Fact: $\mathrm{L}_{\infty\omega}$ is *too* powerful on finite structures, because *every* query on $\mathcal{F}$ is $\mathrm{L}_{\infty\omega}$-definable, where $\mathcal{F}$ is the class of all finite σ-structures

Definition: Barwise (1977)

- *Infinitary logic with k variables, $k \geq 1$.*

  $\mathrm{L}^{k}_{\infty\omega}$ is the collection of $\mathrm{L}_{\infty\omega}$-formulas with at most k distinct variables.

- *Infinitary logic with finitely many variables*

$$\mathrm{L}^{\omega}_{\infty\omega} \quad = \quad \bigcup_{k\geq 1} \mathrm{L}^{k}_{\infty\omega}.$$

Notation:

$$\mathrm{L}^{k}_{\omega\omega} \quad = \quad \text{First-order logic with } k \text{ variables.}$$

Cardinalities of Linear Orders

Fact: Let θ_m be a first-order sentence such that

$$\theta_m \quad \equiv \quad \text{there are at least } m \text{ elements.}$$

On the class $\mathcal{G}$ of all finite graphs, θ_m is *not* equivalent to any first-order sentence with fewer than m variables.

Fact:

On the class $\mathcal{L}$ of finite linear orders, each $\theta_m,\ m \geq 1$, is equivalent to a sentence of $\mathrm{L}^2_{\omega\omega}$.

Proof: For $m = 4$

$$\theta_4 \quad \equiv \quad (\exists x)(\exists y)[y < x \wedge (\exists x)(x < y \wedge (\exists y)(y < x))] \ \blacksquare$$

Fact:

$\mathrm{L}^2_{\infty\omega}$ can define *arbitrary* cardinalities on $\mathcal{L}$.

Indeed, for every set S of integers

$$n \in S \quad \Longleftrightarrow \quad \mathrm{L}_n \models \bigvee_{m \in S} (\theta_m \wedge \neg\theta_{m+1})$$

Transitive Closure - Revisited

For every $m \geq 1$, let

$$\begin{aligned} p_m(x,y) &\equiv \text{there is a path of length } m \text{ from } x \text{ to } y \\ p_m(x,y) &\equiv (\exists z_1)\ldots(\exists z_{m-1})(E(x,z_1) \wedge \ldots \\ &\qquad\qquad \wedge E(z_{m-1},y)) \end{aligned}$$

Fact: Each $p_m(x,y)$, $m \geq 1$, is $L^3_{\omega\omega}$-definable.

Proof: By induction, reusing variables

$$\begin{aligned} \psi_1(x,y) &\equiv E(x,y) \\ \psi_m(x,y) &\equiv (\exists z)[E(x,z) \wedge (\exists x)(z = x \wedge \psi_{m-1}(x,y))] \end{aligned}$$

Corollary: CONNECTIVITY is $L^3_{\infty\omega}$-definable

$$(\forall x)(\forall y)(\bigvee_{m \geq 1} \psi^m(x,y)).$$

Fixed-Point Logics vs. Finite Variable Logics

Theorem: If $\mathcal{C}$ is a class of finite structures, then

$$\mathrm{LFP}[\mathcal{C}] \subseteq \mathrm{PFP}[\mathcal{C}] \subseteq \mathrm{L}^{\omega}_{\infty\omega}[\mathcal{C}].$$

Remarks:

- $\mathrm{L}^{\omega}_{\infty\omega}$ can be used as a tool in studying fixed-point logics. In particular,
  - Certain structural properties of $\mathrm{L}^{\omega}_{\infty\omega}$ are inherited by fixed-point logics.
  - Lower bounds for definability in $\mathrm{L}^{\omega}_{\infty\omega}$ yield similar results for definability in fixed-point logics.
- Advantage of $\mathrm{L}^{\omega}_{\infty\omega}$ over LFP and PFP:
  - Definability in $\mathrm{L}^{k}_{\infty\omega}$ can be characterized in terms of combinatorial *k-pebble games*.

k-Pebble Games

Spoiler and *Duplicator* play on two σ-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 either $\mathbf{A}$ or $\mathbf{B}$.
- *Duplicator* tries to duplicate the move on the other structure.

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

- *Spoiler* wins if at some point the mapping

$$a_1 \mapsto b_1, a_2 \mapsto b_2, \dots, a_l \mapsto b_l$$

  is *not* a partial isomorphism.

- *Duplicator* wins if the above never happens.

  Note that if the *Duplicator* wins, then the k-pebble game can go "forever" ...

Winning Strategies in the k-Pebble Game

Definition: A *winning strategy* for the *Duplicator* in the k-pebble game is a non-empty family I of partial isomorphisms from $\mathbf{A}$ to $\mathbf{B}$ 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 $a \in \mathrm{dom}(g)$.

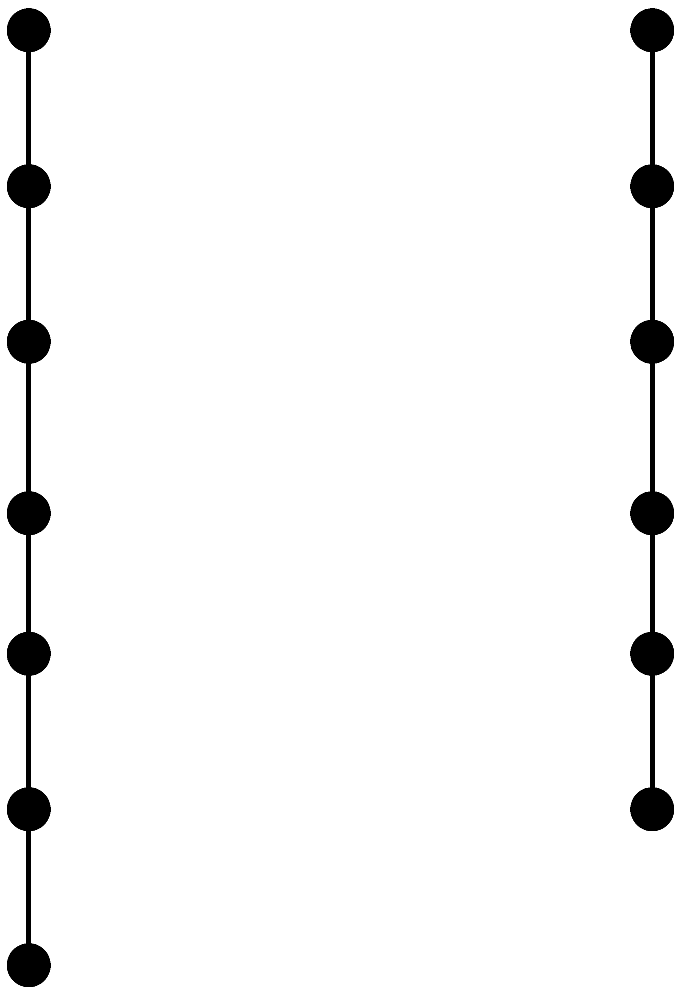
  (I has the *forth property up to k*)

- If $f \in I$ and $|f| < k$, then for every $b \in B$, there is $g \in I$ so that $f \subseteq g$ and $b \in \mathrm{rng}(g)$.

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

Example

Linear Orders of Different Size



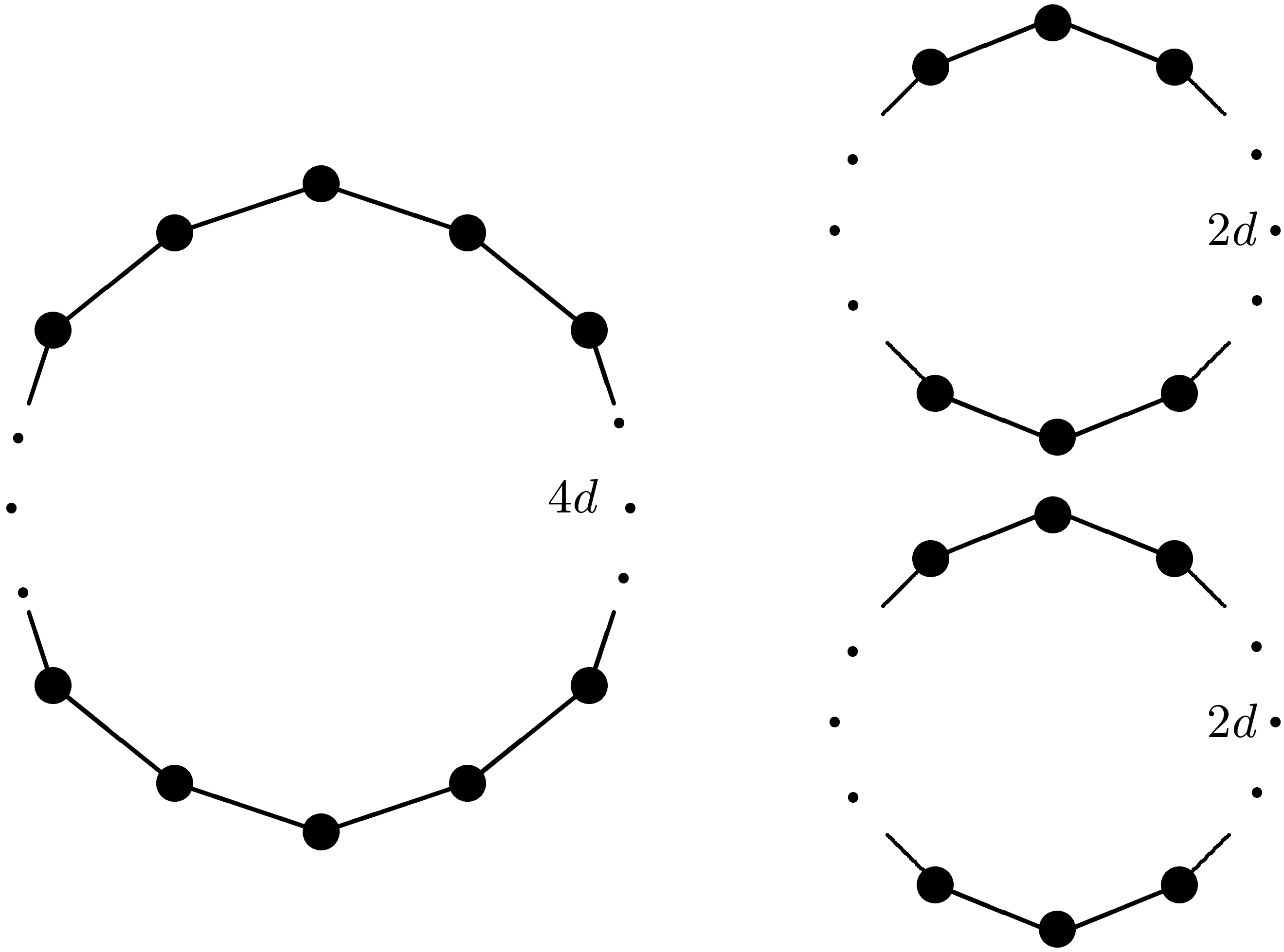
L_m L_n

– *Spoiler* wins the 2-pebble game on L_m and L_n.

– Contrast with the Ehrenfeucht–Fraïssé game!

Example

CONNECTIVITY – revisited



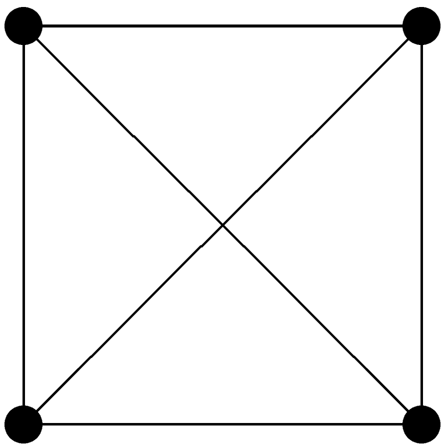
$\mathbf{A}_r$ $\mathbf{B}_r$

Fact:

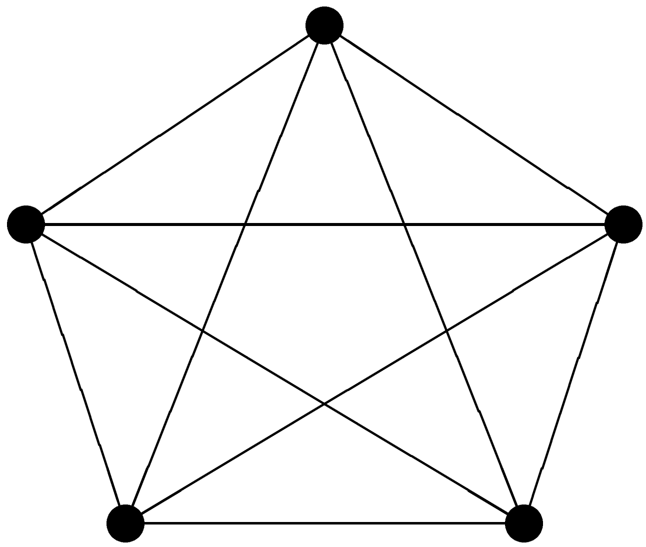
Spoiler wins the 3-pebble game on $\mathbf{A}_r$ and $\mathbf{B}_r$.

Example

Cliques of Different Size



$\mathbf{K}_4$



$\mathbf{K}_5$

Fact: Let $\mathbf{K}_k$ be the *complete* graph with k nodes.

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

$L^k_{\infty\omega}$-Equivalence and k-Pebble Games

Definition: Let $\mathbf{A}$ and $\mathbf{B}$ be two σ-structures.

- $\mathbf{A} \equiv^k_{\infty\omega} \mathbf{B}$, if $\mathbf{A}$ and $\mathbf{B}$ satisfy the same $L^k_{\infty\omega}$-sentences.
- $\mathbf{A} \equiv^k_{\omega\omega} \mathbf{B}$, if $\mathbf{A}$ and $\mathbf{B}$ satisfy the same $L^k_{\omega\omega}$-sentences.

Theorem: Barwise (1977), Immerman (1982)

Let $\mathbf{A}$ and $\mathbf{B}$ be two σ-structures. The following statements are equivalent:

- $\mathbf{A} \equiv^k_{\infty\omega} \mathbf{B}$.
- *Duplic.* wins the k-pebble game on $\mathbf{A}$ and $\mathbf{B}$.

Moreover, if $\mathbf{A}$ and $\mathbf{B}$ are finite, then the above statements are also equivalent to

- $\mathbf{A} \equiv^k_{\omega\omega} \mathbf{B}$.

$\mathrm{L}^k_{\infty\omega}$-Definability and k-Pebble Games

Corollary: Let Q be a Boolean query on $\mathcal{C}$. The following statements are equivalent:

- Q is $\mathrm{L}^\omega_{\infty\omega}$-definable on $\mathcal{C}$.
- There is a k such that for every $\mathbf{A}, \mathbf{B}$ in $\mathcal{C}$, if $\mathbf{A} \models Q$ and the *Duplicator* wins the k-pebble game on $\mathbf{A}$ and $\mathbf{B}$, then $\mathbf{B} \models Q$.

A Methodology for $\mathrm{L}^\omega_{\infty\omega}$-Definability

Let Q be a Boolean query on $\mathcal{C}$.

Soundness: To show that Q is *not* $\mathrm{L}^\omega_{\infty\omega}$-definable on $\mathcal{C}$, suffices to show that for every k there are structures $\mathbf{A}_k$ and $\mathbf{B}_k$ in $\mathcal{C}$ such that

- $\mathbf{A}_k \models Q$ and $\mathbf{B}_k \not\models Q$.
- *Duplic.* wins the k-pebble game on $\mathbf{A}$, $\mathbf{B}$.

Completeness: This method is also *complete*.

Applications

Corollary: Let $\mathcal{G}$ be the class of all finite graphs.

- Even Cardinal. is *not* $\mathrm{L}^{\omega}_{\infty\omega}$-definable on $\mathcal{G}$.
- $(k+1)$-Clique is *not* $\mathrm{L}^{k}_{\infty\omega}$-definable on $\mathcal{G}$.

Proof:

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

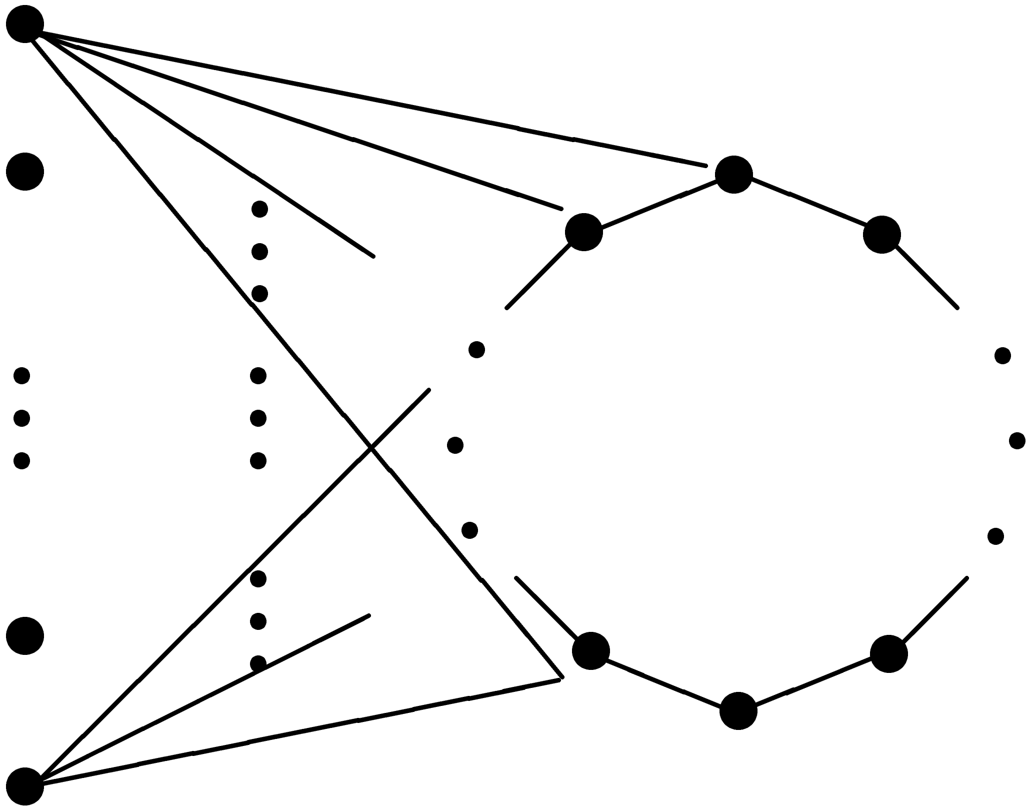
Corollary: Let $\mathcal{G}$ be the class of all finite graphs.

- Even Cardinal. is *not* PFP-definable on $\mathcal{G}$.
- Even Cardinal. is *not* LFP-definable on $\mathcal{G}$.

Theorem: HAMILTONICITY is *not* $L^{\omega}_{\infty\omega}$-definable.

Proof: (de Rougemont - 1983)

$\overline{K_m} \quad \times \quad C_n$



- $\overline{\mathbf{K}}_m \times \mathbf{C}_n$ is HAMILTONIAN $\iff$ $m \leq n$
- *Duplicator* wins the k-pebble game on $\overline{\mathbf{K}}_k \times \mathbf{C}_k$ and $\overline{\mathbf{K}}_{k+1} \times \mathbf{C}_k$.

Homework Assignment #2

- Show that 2-Colorability is LFP-definable on finite graphs.

- Show that the following query Q is LFP-definable on finite graphs:

  Given a graph G and nodes s, t_1, t_2, are there node disjoint simple paths from s to t_1 and from s to t_2?

- Show that there is no positive first-order formula $\psi(x, y, S)$ such that $\psi^\infty(x, y)$ defines the complement $\overline{TC}$ of the Transitive Closure query on finite graphs.

Homework Assignment #3

- Show that PERFECT MATCHING is *not* $L^{\omega}_{\infty\omega}$-definable on finite undirected graphs.

- Show that 3-COLORABILITY in *not* $L^{\omega}_{\infty\omega}$-definable on finite undirected graphs.

  Note: This is hard (Dawar - 1993).

Pebble Games

and

$\equiv^k_{\infty\omega}$-Equivalence

Other Uses of k-Pebble Games

- *Negative Results*

  So far, k-pebble games have been used to obtain lower bounds for definability in $L^{\omega}_{\infty\omega}$ and *a fortiori* for definability in PFP and LFP.

- *Positive Results*

  - k-pebble games can be used to establish structural properties of $L^{\omega}_{\infty\omega}$, which in many cases are inherited by PFP and LFP.
  - k-pebble games can be used to unveil strong connections between LFP and $L^{\omega}_{\infty\omega}$.

- *The Role of* $\equiv^{k}_{\infty\omega}$

  These positive results entail an in-depth study of $\equiv^{k}_{\infty\omega}$ using k-pebble games.

Logic & Asymptotic Probabilities

Q :	Boolean query on a class $\mathcal{C}$
$\mathcal{C}_n$:	Class of structures in $\mathcal{C}$ of cardinality n
μ_n :	Probability measure on $\mathcal{C}_n$, $n \geq 1$.
$\mu_n(Q)$:	Probability of Q on $\mathcal{C}_n$ w.r.t. μ_n, $n \geq 1$.

Definition: *Asymptotic probability of query Q*

$$\mu(Q) \quad = \quad \lim_{n \to \infty} \mu_n(Q), \quad \text{provided the limit exists}$$

Example: *Uniform Measure* on $\mathcal{G}$

$\mu_n(\mathcal{C})$: Fraction of graphs with n nodes in $\mathcal{C}$

- $\mu(\text{4-REGULAR}) = 0$
- $\mu(\text{2-COLORABLE}) = 0$
- $\mu(\text{HAMILTONIAN PATH}) = 1$
- $\mu(\text{EVEN CARDINALITY})$ does *not* exist

0-1 Laws

Problem: Is there a connection between the *definability* of a query Q in some logic $\mathcal{L}$ and its *asymptotic probability* w.r.t. a given measure?

Definition: $\mathcal{L}$ a logic, $\mathcal{C}$ a class of structures.

$\mathcal{L}$ has a *0-1 law on* $\mathcal{C}$ with respect to μ_n, $n \geq 1$, if

$$\mu(Q) = 0 \quad \text{or} \quad \mu(Q) = 1$$

for every $\mathcal{L}$-definable query Q on $\mathcal{C}$.

Problem: Which logics have a 0-1 law w.r.t. the uniform measure on the class $\mathcal{F}$ of all finite σ-structures?

$\boxed{\textbf{0-1 Laws}}$

Theorem: w.r.t. the uniform measure on $\mathcal{F}$

- First-Order Logic has a 0-1 law.

  Glebskii et al. – 1969, Fagin – 1972

- Least Fixed-Point Logic LFP has a 0-1 law.

  Blass, Gurevich, Kozen – 1985

- Partial Fixed-Point Logic PFP has a 0-1 law.

  K ... & Vardi – 1987

Note: Recall that

$$\mathrm{FO}[\mathcal{F}] \subsetneq \mathrm{LFP}[\mathcal{F}] \subseteq \mathrm{PFP}[\mathcal{F}] \subsetneq \mathrm{L}^{\omega}_{\infty\omega}[\mathcal{F}].$$

0-1 Laws for $L^{\omega}_{\infty\omega}$

Proposition: K ... & Vardi – 1992

The following two statements are equivalent:

1. $L^{k}_{\infty\omega}$ has a 0-1 law w.r.t. $\{\mu_n : n \geq 1\}$ on $\mathcal{C}$.

2. There is an equivalence class D of $\equiv^{k}_{\infty\omega}$ on $\mathcal{C}$ such that $\mu(D) = 1$.

Theorem: K ... & Vardi – 1992

$L^{\omega}_{\infty\omega}$ has a 0-1 law w.r.t. the uniform measure on the class $\mathcal{F}$ of all finite σ-structures

Hint of Proof:

For every $k \geq 1$, let $\mathbf{A}_k$ be a model of all *extension axioms* for σ with at most k variables. Use k-pebble games to show that

$$\mu([\mathbf{A}_k]_{\equiv^{k}_{\infty\omega}}) = 1. \blacksquare$$

More on 0-1 Laws

Corollary:

- LFP and PFP have a 0-1 law w.r.t. the uniform measure on $\mathcal{F}$.
- Even Cardinality is not $L^{\omega}_{\infty\omega}$-definable.

Theorem: Let $\mathcal{G}$ be the class of all finite undirected graphs and let α be a number $0 < \alpha < 1$.

$L^{\omega}_{\infty\omega}$ does *not* have a 0-1 law w.r.t. non-uniform measure $p(n) = n^{-\alpha}$ on $\mathcal{G}$.

- Shelah & Spencer – 1987, for rational α
- McArthur – 1996, for irrational α.

Normal Form for $\mathrm{L}^k_{\infty\omega}$-Definability

Proposition: K ... & Vardi – 1992

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

- Q is $\mathrm{L}^k_{\infty\omega}$-definable on $\mathcal{F}$.
- Q (viewed as a class of finite structures) is the union of $\equiv^k_{\infty\omega}$-equivalence classes on $\mathcal{F}$.

Theorem: Dawar–Lindell-Weinstein (1995)

Let $\mathcal{F}$ be the class of all finite σ-structures.

- Each $\equiv^k_{\infty\omega}$-equivalence class on $\mathcal{F}$ is $\mathrm{L}^k_{\omega\omega}$-definable.
- For every $\mathrm{L}^k_{\infty\omega}$-sentence ψ, there are $\mathrm{L}^k_{\omega\omega}$-sentences ψ_m, $m \geq 1$, such that for every finite $\mathbf{A}$

$$\mathbf{A} \models \psi \iff \mathbf{A} \models \bigvee_{i=1}^{\infty} \psi_m.$$

Note: Finiteness is of the essence in these results.

Definability and Complexity of $\equiv^k_{\infty\omega}$

Definition: For $k \geq 2$, let E_k be the following query on $\mathcal{F}$:

Given two finite structures $\mathbf{A}$ and $\mathbf{B}$, is $\mathbf{A} \equiv^k_{\infty\omega} \mathbf{B}$?

Proposition: (KV92, DLW95)

For each $k \geq 2$, the query E_k is LFP-definable.

Proof:

The query "Does the *Spoiler* win the k-pebble game on $\mathbf{A}$ and $\mathbf{B}$?" is LFP-definable. ∎

Corollary: For each $k \geq 2$, the query E_k is in P.

Definability and Complexity of $\equiv^k_{\infty\omega}$

Theorem: Grohe (1996)

For each $k \geq 2$, the query E_k is P-complete.

Hint of Proof Difficult reduction from the MONOTONE CIRCUIT VALUE Problem. ∎

Note:

Sharp contrast with $\equiv_r$ and r-move EF-game, since, for each $r \geq 2$, deciding whether $\mathbf{A} \equiv_r \mathbf{B}$ is in L (i.e., solvable in logarithmic space).

Open Problem: Show that the following decision problem is EXPTIME-complete:

Given a positive integer k and two finite structures $\mathbf{A}$ and $\mathbf{B}$, is $\mathbf{A} \equiv^k_{\infty\omega} \mathbf{B}$?

k-Types on Finite Structures

Definition: Assume that $k \geq 2$, $\mathbf{A}$ is a finite σ-structure, $\overline{a} \in A^k$, and $\overline{b} \in A^k$.

- The *k-type* of $\overline{a}$ is the collection of all $\mathrm{L}^k_{\infty\omega}$-formulas $\varphi(\overline{x})$ such that $\mathbf{A} \models \varphi(\overline{a})$.
- $\overline{a} \equiv^k_{\infty\omega} \overline{b}$, if $\overline{a}$ and $\overline{b}$ have the same k-type.

Note:

- It is *not* possible to define in LFP a linear order between elements of all finite σ-structures.

  However, for every $k \geq 2$:

- It is possible to define in LFP a linear order between k-types of elements of finite σ-structures.

Ordering the k-Types

Theorem: Abiteboul-Vianu (1991)

Let $\mathcal{F}$ be the class of all finite σ-structures.

For every $k \geq 2$, there is a LFP-definable query P_k on $\mathcal{F}$ that *linearly orders the k-types*, i.e.,

- $\mathrm{arity}(P_k) = 2k$;
- for every $\mathbf{A} \in \mathcal{F}$, $P_k(\mathbf{A})$ is a linear pre-order on A^k whose induced equivalence relation coincides with $\equiv^k_{\infty\omega}$ on $\mathbf{A}$.

Hint of Proof: Dawar-Lindell-Weinstein (1995)

- Use k-pebble games to design a *color-refinement* algorithm that inductively orders the k-types.
- Show that this algorithm can be expressed in LFP. ∎

LFP vs. PFP

Recall the following problem.

Problem: Chandra–Harel (1982)

Show that on the class $\mathcal{F}$ of all finite σ-structures

$$\mathrm{LFP}[\mathcal{F}] \neq \mathrm{PFP}[\mathcal{F}].$$

It has a surprising resolution:

Theorem: Abiteboul–Vianu (1991)

The following statements are equivalent:

- $\mathrm{LFP}[\mathcal{F}] = \mathrm{PFP}[\mathcal{F}]$.
- PTIME $=$ PSPACE.

Proof of the Abiteboul–Vianu Theorem

Theorem: Abiteboul–Vianu (1991)

The following statements are equivalent:

- $\mathrm{LFP}[\mathcal{F}] = \mathrm{PFP}[\mathcal{F}]$.
- PTIME = PSPACE.

Proof:

Top to bottom is easy:

Assume that $\mathrm{LFP}[\mathcal{F}] = \mathrm{PFP}[\mathcal{F}]$. It follows that PTIME = PSPACE, because

- $\mathrm{PFP}[\mathcal{F}]$ contains PSPACE-complete problems.
- $\mathrm{LFP}[\mathcal{F}] \subseteq$ PTIME.

Bottom to top is much harder and uses essentially all the machinery developed so far.

Proof of the Abiteboul–Vianu Theorem

If PTIME = PSPACE, then LFP[$\mathcal{F}$] = PFP[$\mathcal{F}$].

Hint of Proof: Dawar–Lindell-Weinstein (1995)

- Use *transfer properties* between $\mathcal{F}$ and

$$\mathcal{F}/\equiv^k_{\infty\omega} \quad = \quad \{\mathbf{A}/\equiv^k_{\infty\omega}\colon\ \mathbf{A} \in \mathcal{F}\}.$$

- $\mathcal{F}/\equiv^k_{\infty\omega}$ admits a LFP-definable linear order. Consequently, on $\mathcal{F}/\equiv^k_{\infty\omega}$

$$\begin{aligned} \text{LFP} &= \text{PTIME} \\ \text{PFP} &= \text{PSPACE}. \end{aligned}$$

- Assume that Q is a query on $\mathcal{F}$ definable by a PFP-formula φ with k distinct variables.

$$\begin{array}{ccc} \varphi \in \text{PFP}[\mathcal{F}] & \equiv & \psi' \in \text{LFP}[\mathcal{F}] \\ \downarrow & & \uparrow \\ \varphi^* \in \text{PFP}[\mathcal{F}/\equiv^k_{\infty\omega}] & \equiv & \psi \in \text{LFP}[\mathcal{F}/\equiv^k_{\infty\omega}] \end{array} \blacksquare$$

Concluding Remarks

Research in finite model theory has branched into three directions:

- Expressive Power of Logics
- Logic and Computational Complexity
- Logic and Asymptotic Probabilities

Finite variable logics and pebble games constitute a meeting point that brings these areas together.

Datalog

and

Existential k-Pebble Games

Datalog

Definition:

- Datalog is the data sublanguage of logic programming
- A Datalog program is a finite set of function-free, $\neq$-free and negation-free rules of the form:

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

Example 1: Transitive Closure

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

Example 2: Transitive Closure

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

Datalog and Existential LFP

Proposition: Chandra–Harel (1985)

The following are equivalent for a query Q.

- Q is definable by a Datalog program.
- Q is definable by φ_m^∞ for some system $(\varphi_1, \ldots, \varphi_m)$ of positive existential formulas in which neither $\neq$, nor negated atomic formulas occur.

Proof: Each rule gives rise to a disjunct that is a positive existential formula, and vice versa. ▮

Example: Transitive Closure

- Datalog Program

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

- Existential Positive Formula

$$\varphi(x,y,S) \;\equiv\; E(x,y) \vee (\exists z)(E(x,z) \wedge S(z,y)).$$

Example

- 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.

- Datalog program for Path Systems:

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

- Existential formula $\psi(x, T)$ for Path Systems

$$A(x) \vee (\exists y)(\exists z)(T(y) \wedge T(z) \wedge R(x,y,z))$$

$$\psi^\infty(x) \quad \equiv \quad x \text{ is a theorem of } \mathbf{S}$$

Theorem: Cook (1974)

Path Systems is a P-complete problem.

The Complexity of Datalog

Proposition:

The data complexity of Datalog is P-complete.

Proof:

- Datalog is a fragment of LFP, hence its data complexity is in P.
- Path Systems is P-complete and definable by a Datalog program. ∎

$\boxed{\textbf{Datalog}(\neq)}$

Definition:

Datalog($\neq$) is the extension of Datalog in which $\neq$ are allowed in the rules.

Example: w-Avoiding Paths:

Let Q be the query: given $\mathbf{G} = (V, E)$ and nodes x, y, w, is there a path from x to y avoiding w?

- Q is definable by the Datalog($\neq$) program:

$$\begin{aligned} T(x,y,w) &\quad :- \quad E(x,y) \wedge w \neq x \wedge w \neq y \\ T(x,y,w) &\quad :- \quad E(x,z) \wedge T(z,y,w) \wedge w \neq x. \end{aligned}$$

- Q is definable by the least fixed-point $\varphi^\infty(x,y,w)$ of the formula $\varphi(x,y,w,T)$

$$(E(x,y) \wedge w \neq x \wedge w \neq y) \vee$$

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

Even Simple Path

Definition: Let Even Simple Path be the query: given a graph $\mathbf{G} = (V, E)$ and two nodes x, y, is there a simple path of even length from x to y?

Fact: Even Simple Path is NP-complete.

(Fortune, Hopcroft, Wiley - 1980)

Fact: Even Simple Path on undirected graphs is definable by a Datalog($\neq$) program (so, is in P).

(Yannakakis - 1990)

Datalog($\neq$)-Program for Even Simple Path

$$
\begin{array}{lcl}
T(x,y,w) & :- & E(x,y) \wedge w \neq x \wedge w \neq y \\
T(x,y,w) & :- & E(x,z) \wedge T(z,y,w) \wedge w \neq x \\
P(x,y) & :- & E(x,y) \\
P(x,y) & :- & Q(x,w), E(w,y), T(x,w,y) \\
Q(x,y) & :- & P(x,w), P(w,y), T(x,w,y).
\end{array}
$$

Fact:

- T defines w-AVOIDING PATH;
- P defines ODD SIMPLE PATH
- Q defines EVEN SIMPLE PATH

Proof: Exercise – Use *Menger's Theorem.* ∎

Datalog vs. LFP

Fact: *Preservation Properties* of Datalog.

- Datalog queries are preserved under *homomorphisms*.
- Datalog($\neq$) queries are preserverd under *one-to-one homomorphisms*.

Proof: Exercise - use preservation properties of positive existential formulas. ∎

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

$$\text{Datalog}[\mathcal{G}] \subsetneq \text{Datalog}(\neq)[\mathcal{G}] \subsetneq \text{LFP}[\mathcal{G}].$$

Proof:

- w-Avoiding Path $\notin$ Datalog[$\mathcal{G}$].
- Acyclicity $\notin$ Datalog($\neq$)[$\mathcal{G}$]. ∎

Existential Positive Fragments of $L^{\omega}_{\infty\omega}$

Definition: K ... & Vardi – 1995

- $\exists L^{k}_{\infty\omega}$ is the fragment of $L^{k}_{\infty\omega}$ that contains all atomic formulas, $=$, and is closed under $(\exists x)$ and infinitary disjunctions and conjunctions.
- $\exists L^{\omega}_{\infty\omega} = \bigcup_{k=1}^{\infty} \exists L^{k}_{\infty\omega}$.
- $\exists L^{\omega}_{\infty\omega}(\neq)$ is the fragment of $L^{k}_{\infty\omega}$ that contains all atomic formulas, $=$, $\neq$, and is closed under $(\exists x)$ and infinitary disjunctions and conjunctions.
- $\exists L^{\omega}_{\infty\omega}(\neq) = \bigcup_{k=1}^{\infty} \exists L^{k}_{\infty\omega}(\neq)$.

Theorem: If $\mathcal{C}$ is a class of finite structures, then

- $\text{Datalog}[\mathcal{C}] \subseteq \exists L^{\omega}_{\infty\omega}[\mathcal{C}]$.
- $\text{Datalog}(\neq)[\mathcal{C}] \subseteq \exists L^{\omega}_{\infty\omega}(\neq)[\mathcal{C}]$.

Existential k-Pebble Games

Asymmetric variants of the k-pebble game:

Spoiler plays on $\mathbf{A}$, *Duplicator* plays on $\mathbf{B}$.

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

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

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 $\mathbf{A}$ to $\mathbf{B}$ such that

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

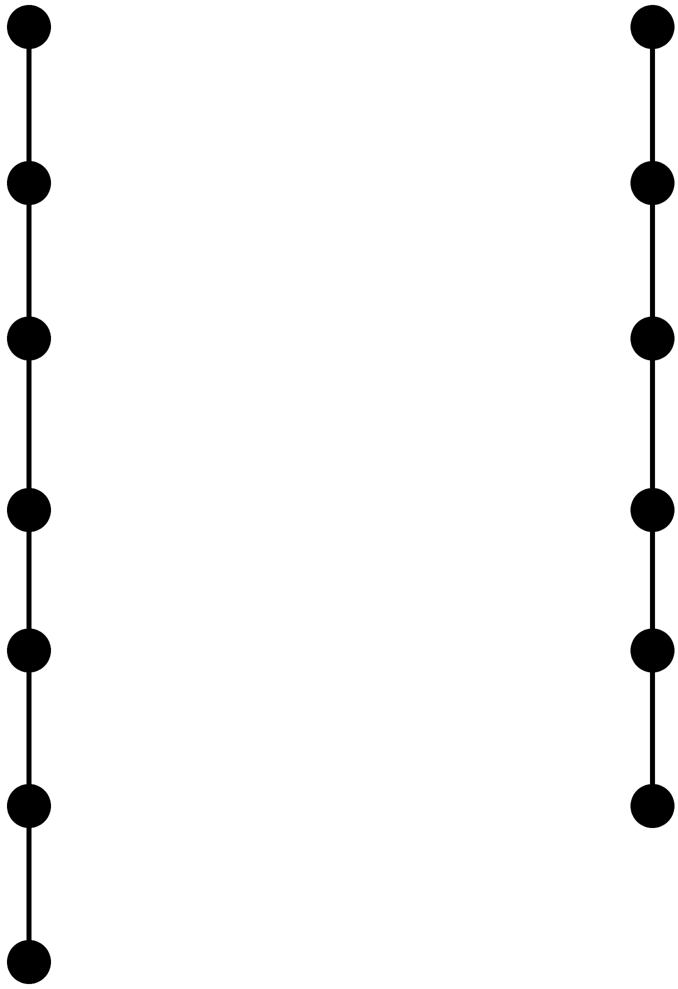
  (I is *closed under subfunctions*).

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

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

Note: Similar definition of *winning strategy* for the *Duplicator* in the $(\exists, \neq, k)$-pebble game with partial one-to-one homomorphims in place of partial homomorphisms.

Linear Orders of Different Size



L_m L_n

- *Spoiler* wins the $(\exists, \neq, 2)$-pebble game on L_m and L_n
- *Duplic.* wins the $(\exists, \neq, 2)$-pebble game on L_n and L_m

Uses of Existential k-Pebble Games

- Existential k-pebble games can be used to study the expressive power of $\exists L^k_{\infty\omega}$ and $\exists L^k_{\infty\omega}(\neq)$.
- In particular, existential k-pebble games can be used to obtain lower bounds for definability in Datalog and Datalog($\neq$).
- Existential k-pebble games are easier for the *Duplicator* to win, because the *Spoiler* is not allowed to switch between the two structures.

$(\exists, k)$-Pebble Games and $\exists L^k_{\infty\omega}$-Definability

Definition: Let **A**, **B** be two σ-structures.

$\mathbf{A} \preceq^k_{\infty\omega} \mathbf{B}$, if every $\exists L^k_{\infty\omega}$-sentence that is true on **A** is also true on **B**.

Theorem: K . . . & Vardi – 1995

The following statements are equivalent.

- $\mathbf{A} \preceq^k_{\infty\omega} \mathbf{B}$.
- *Duplicator* wins the $(\exists, k)$-pebble game on **A** and **B**.

Remark:

An analogous result holds for $(\exists, \neq, k)$-pebble games and $\exists L^k_{\infty\omega}(\neq)$.

Lower Bounds for $\exists \mathrm{L}^k_{\infty\omega}$-Definability

Corollary: Let Q be a Boolean query on $\mathcal{C}$. The following statements are equivalent:

- Q is $\exists \mathrm{L}^\omega_{\infty\omega}$-definable on $\mathcal{C}$.
- There is a k such that for every $\mathbf{A}, \mathbf{B}$ in $\mathcal{C}$, if $\mathbf{A} \models Q$ and $\mathbf{B} \not\models Q$, then the *Spoiler* wins the $(\exists, k)$-pebble game on $\mathbf{A}$, $\mathbf{B}$.

A Methodology for $\exists \mathrm{L}^\omega_{\infty\omega}$-Definability

To show that a Boolean query Q is *not* expressible in $\exists \mathrm{L}^\omega_{\infty\omega}$, suffices to show that for every k there are structures $\mathbf{A}_k$ and $\mathbf{B}_k$ such that

- $\mathbf{A}_k \models Q$ and $\mathbf{B}_k \not\models Q$.
- *Duplicator* wins the $(\exists, k)$-pebble game on $\mathbf{A}$, $\mathbf{B}$.

Fact: This method is also *complete*.

Similar method for $\exists \mathrm{L}^\omega_{\infty\omega}(\neq)$-definability.

FIxed Subgraph Homeomorphism (FISH)

Definition: Let H and G be directed graphs.

Homeomorphism $h : H \rightsquigarrow G$

- h maps $1-1$ the nodes of H to nodes of G;
- h maps the edges of H to pairwise node-disjoint simple paths of G.

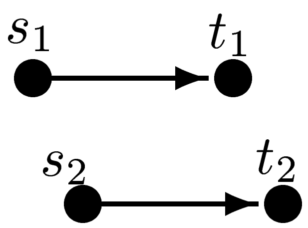
Definition: Let H be a fixed directed graph.

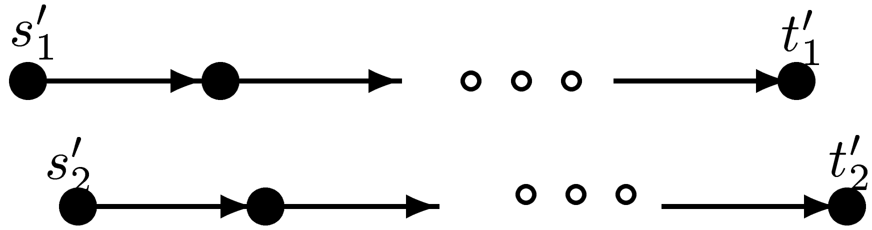
FISH Query with Pattern H asks:

- Given a directed graph G and a $1-1$ mapping from the nodes of H to the nodes of G, is there a homeomorphism $h : H \rightsquigarrow G$ extending this mapping?

Two Disjoint Paths Query

Pattern H



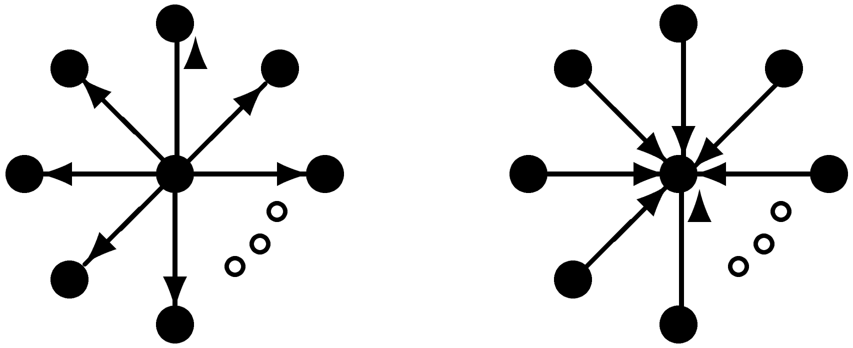


FISH Query with Pattern H:

Does G contain two node-disjoint simple paths from s'_1 to t'_1 and from s'_2 to t'_2?

Computational Complexity of FISH Queries

Star Graphs: single *source* or a single *sink*



Hopcroft, Fortune, Wiley (1980)

Dichotomy Theorem:

• If H is a star graph, then the FISH query with pattern H is in P.

• If H is *not* a star graph, then the FISH query with pattern H is NP-complete.

Note:

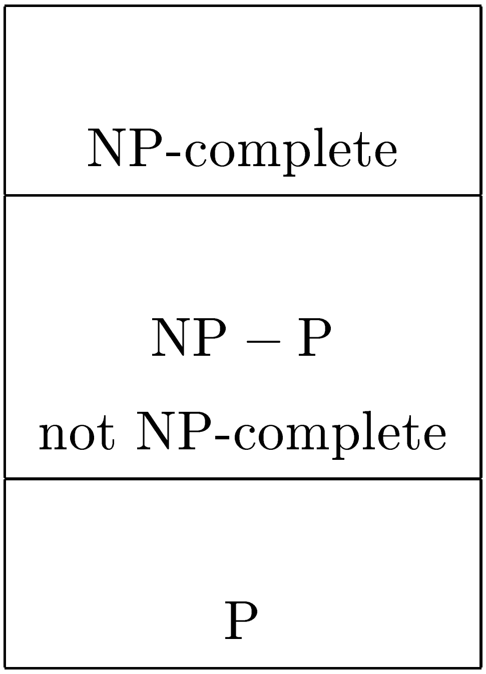
This dichotomy is *proper* only if P $\neq$ NP.

The Fine Structure of NP

Theorem: Ladner - 1975

If P $\neq$ NP, then there is a problem L such that

- $L \in \text{NP} - \text{P}$;
- L is *not* NP-complete. $\blacksquare$



Remark: Thus, a family of decision problems in NP may contain problems of such intermediate complexity.

This possibility cannot be ruled out *a priori.*

Dichotomy Theorem for FISH Queries

Theorem: K … & Vardi – 1995

- If H is a star graph, then the FISH query with pattern H is $\exists \mathrm{L}^{\omega}_{\infty\omega}(\neq)$-definable; in fact, it is definable in Datalog($\neq$).

- If H is *not* a star graph, then the FISH query with pattern H is *not* $\exists \mathrm{L}^{\omega}_{\infty\omega}(\neq)$-definable.

Note:

This dichotomy is indeed *proper*.

No complexity-theoretic assumptions are made.

Dichotomy Theorem for FISH Queries

Hint of Proof:

Two Disjoint Paths query $2DP$ is the key case of the negative results.

Show that for every $k \geq 1$ there are directed graphs $\mathbf{A}_k$ and $\mathbf{B}_k$ such that

- $\mathbf{A}_k \models 2DP$ and $\mathbf{B}_k \not\models 2DP$
- Duplic. wins the $(\exists, \neq, k)$-pebble game on $\mathbf{A}$, $\mathbf{B}$.

How do we find such directed graphs $\mathbf{A}_k$ and $\mathbf{B}_k$?

- $\mathbf{A}_k$ consists of two disjoint *long* paths;
- $\mathbf{B}_k$ is extracted from the NP-completeness proof.

Duplicator's winning strategy is quite complicated.

Moral:

NP-completeness proofs can also yield interesting structures to play combinatorial games.

Open Problems

Open Problem 1: Show that
TWO DISJOINT PATHS is *not* $L^{\omega}_{\infty\omega}$-definable.

Theorem:

Every FISH query on undirected graphs is in P.

(Uses Robertson-Seymour results on graph minors).

Open Problem 2: Analyze the definability of FISH queries on undirected graphs.

Constraint Satisfaction

and

Existential k-Pebble Games

Constraint Satisfaction Problem (CSP)

Fundamental & ubiquitous problem in AI and CS:

- Boolean Satisfiability
- Database Query Processing and Optimization
- Planning and Scheduling
- Belief Maintenance
- Machine Vision
- $\cdots$
- Computational Linguistics

Constraint Satisfaction Problem (CSP)

Input: (V, D, C)

- A finite set V of *variables*
- A finite *domain* D of values for the variables
- A set C of *constraints* (t, R) restricting the values that tuples of variables can take.
  - t: a tuple $t = (x_1, \ldots, x_m)$ of variables
  - R: a relation on D of arity $|t| = m$

Solution: A mapping $h : V \to D$ such that

$$h(t) = (h(x_1), \ldots, h(x_m)) \in R,$$

for every constraint $(t, R) \in C$.

Question: Does (V, D, C) have a solution?

Is there an *assignment* of values to the variables such that all constraints are satisfied?

Example

- 3-COLORABILITY:

  Given a graph $\mathbf{H} = (V, E)$, is it 3-colorable?

- 3-COLORABILITY as a CSP problem:
  - The variables are the nodes in V
  - The domain is the set $D = \{R, G, B\}$ of three colors.
  - The constraints are the pairs $((u, v), Q)$, where $(u, v) \in E$ and Q is the relation

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

Examples

- 3-SAT:

  Given a 3CNF-formula φ with variables $v_1, \ldots, v_n$ and clauses $c_1, \ldots, c_m$, is φ satisfiable?

- 3-SAT as a CSP problem

  – Variables $v_1, \ldots, v_n$

  – Domain $D = \{0, 1\}$

  – Constraints $((x, y, z), R_i)$, $i = 0, 1, 2, 3$

Clause	Relation
$(x \vee y \vee z)$	$R_0 = \{0,1\}^3 - \{(0,0,0)\}$
$(\neg x \vee y \vee z)$	$R_1 = \{0,1\}^3 - \{(1,0,0)\}$
$(\neg x \vee \neg y \vee z)$	$R_2 = \{0,1\}^3 - \{(1,1,0)\}$
$(\neg x \vee \neg y \vee \neg z)$	$R_3 = \{0,1\}^3 - \{(1,1,1)\}$

CSP vs. the Homomorphism Problem

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

Fact: Feder & Vardi – 1993

CSP can be identified with the Homomorphism Problem.

Example

- 3-Colorability as a CSP problem:

  – The variables are the nodes in V

  – The domain is the set $D = \{R, G, B\}$ of three colors.

  – The constraints are the pairs $((u, v), Q)$, where $(u, v) \in E$ and Q is the relation

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

- 3-Colorability as a Homomorphism Problem

An undirected graph $\mathbf{H} = (V, E)$ is 3-Colorable

$$\Longleftrightarrow$$

there is a homomorphism $h : \mathbf{H} \to \mathbf{K}_3$, where $\mathbf{K}_3$ is the 3-clique.

From CSP to the Homomorphism Problem

Given a CSP-instance (V, D, C), let $R_1, \ldots, R_m$ be the distinct relations on D occurring in C.

Define the instance $(\mathbf{A}, \mathbf{B})$ of the HOMOMORPHISM PROBLEM, where

$\mathbf{A} = (V, \{t : (t, R_1) \in C\}, \ldots, \{t : (t, R_m) \in C\})$

and

$\mathbf{B} = (D, R_1, \ldots, R_m)$.

Fact: The CSP instance (V, D, C) has a solution

$$\Longleftrightarrow$$

there is a homomorphism from $\mathbf{A}$ to $\mathbf{B}$.

From the Homomorphism Problem to CSP

Given an instance $\mathbf{A} = (A, R_1^{\mathbf{A}}, \ldots, R_m^{\mathbf{A}})$ and $\mathbf{B} = (B, R_1^{\mathbf{B}}, \ldots, R_m^{\mathbf{B}})$ of the HOMOMORPHISM PROBLEM, define the instance (V, D, C) of CSP, where

- $V = A$ (i.e., the elements of A are variables);
- $D = B$ (i.e., the elements of B are values);
- $C = \{(t, R_i^{\mathbf{B}}) : t \in R_i^{\mathbf{A}}, 1 \leq i \leq m\}$.

Fact: There is a homomorphism from **A** to **B**

$$\Longleftrightarrow$$

the CSP-instance (V, D, C) has a solution.

Uniform and Non-Uniform CSP

- Uniform CSP:

  $$\{(\mathbf{A}, \mathbf{B}) : \exists \text{ homomorphism } h : \mathbf{A} \to \mathbf{B}\}$$

  Uniform CSP is NP-complete

- Coping with NP-completeness: Identify tractable cases by restricting input structures.

- Let $\mathcal{A}$, $\mathcal{B}$ be two classes of structures

  $$\mathrm{CSP}(\mathcal{A}, \mathcal{B}) = \{(\mathbf{A}, \mathbf{B}) : \mathbf{A} \in \mathcal{A}, \mathbf{B} \in \mathcal{B} \text{ and } \exists \text{ homomorphism } h : \mathbf{A} \to \mathbf{B}\}$$

- Non-Uniform CSP: Fix a structure $\mathbf{B}$

  $\mathrm{CSP}(\mathbf{B}) = \{\mathbf{A} : \exists \text{ homomorphism } h : \mathbf{A} \to \mathbf{B}\}$

  In other words, $\mathrm{CSP}(\mathbf{B}) = \mathrm{CSP}(\mathcal{F}, \{\mathbf{B}\})$.

Complexity of Non-Uniform CSP

- Complexity of non-uniform CSP depends on $\mathbf{B}$
  - CSP($\mathbf{K}_3$) is NP-complete.
    (3-COLORABILITY)
  - CSP($\mathbf{K}_2$) is in P.
    (2-COLORABILITY)
- Research Program:
  Identify the tractable cases of non-uniform CSP

Dichotomy Conjecture: Feder & Vardi –1993

For every structure $\mathbf{B}$, one of the following holds:

- CSP($\mathbf{B}$) is in P.
- CSP($\mathbf{B}$) is NP-complete.

(Recall Ladner's Theorem)

Evidence for the Dichotomy Conjecture

Theorem: Schaefer – 1978

GENERALIZED SATISFIABILITY PROBLEMS

If **B** is a Boolean structure (i.e., $B = \{0, 1\}$), then either CSP(**B**) is in P or CSP(**B**) is NP-complete.

Theorem: Hell & Nešetril – 1990

Let **B** be an *undirected* graph.

- If **B** is 2-Colorable, then CSP(**B**) is in P.
- If **B** not 2-Colorable, then CSP(**B**) is NP-complete.

Open Problems:

- Dichotomy for CSP(**B**), where $B = \{0, 1, 2\}$.
- Dichotomy for CSP(**B**), where B is a directed graph.

Non-Uniform CSP and Monadic ES0

Let $\mathbf{B} = (B, R_1^{\mathbf{B}}, \ldots, R_m^{\mathbf{B}})$ be a finite σ-structure .

Fact: CSP($\mathbf{B}$) is definable by a *monadic, strict, monotone* ESO-sentence, i.e., by an ESO-sentence of the form: $(\exists S_1) \cdots (\exists S_k) \forall \overline{x} \theta$, where

- $S_1, \ldots, S_k$ are unary relation symbols.
- θ is a quantifier-free and $\neq$-free first-order formula in which all occurrences of the relation symbols $R_1, \ldots, R_m$ are negative (after negation symbols have been "pushed" inside).

Proof: Use one S_i for each element of B. ▮

Theorem: Feder & Vardi -1993

Every query definable by a monadic, strict, monotone ESO-sentence is polynomial-time equivalent to a non-uniform CSP problem via a randomized Turing reduction.

Tractable Cases of Non-Uniform CSP

Extensive pursuit of tractable cases of CSP

First Stage: Numerous particular results.

Second Stage: Broad sufficient conditions for tractability of non-uniform CSP.

Unifying explanations for many tractability results:

- Feder & Vardi – 1993
  $\overline{\mathrm{CSP}(\mathbf{B})}$ is expressible in Datalog, where

  $$\overline{\mathrm{CSP}(\mathbf{B})} \;=\; \{\mathbf{A} : \mathbf{A} \notin \mathrm{CSP}(\mathbf{B})\}.$$

  Example: Non 2-Colorability

  $$\begin{aligned} O(X,Y) &\;:-\; E(X,Y) \\ O(X,Y) &\;:-\; O(X,Z), E(Z,W), E(W,Y) \\ Q &\;:-\; O(X,X) \end{aligned}$$

- Feder & Vardi – 1993, Jeavons et al. – 199X
  Algebraic *closure properties* of constraints
  Connections with universal algebra

From Non-Uniform CSP to Uniform CSP

Fact: Many applications in AI and in database systems involve uniform CSP problems (for instance, conjunctive query containment and evaluation).

Question:

Can we obtain tractability results for uniform CSP from tractability results for non-uniform CSP?

Note: Let $\mathcal{A}$ and $\mathcal{B}$ be two classes of σ-structures.

It is possible that CSP$(\mathcal{A}, \{\mathbf{B}\})$ is in P, for every $\mathbf{B}$ in $\mathcal{B}$, but CSP$(\mathcal{A}, \mathcal{B})$ is NP-complete.

Example: $\mathcal{K}$ = all *cliques* and $\mathcal{G}$ = all graphs:

- CSP$(\mathcal{K}, G)$ is in P, for every graph G;
- CSP$(\mathcal{K}, \mathcal{G})$ in NP-complete.
  (the CLIQUE problem) ∎

From Non-Uniform CSP to Uniform CSP

So, non-uniform tractability does *not* always yield uniform tractability.

However, certain non-uniform tractable cases *do* uniformize!

Theorem: K . . . & Vardi – 1998

Expressibility of non-uniform CSP problems in k-variable Datalog uniformizes.

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 in the head t_0 and at most k distinct variable in the body $t_1, \ldots, t_m$.

Example: NON 2-COLORABILITY revisited

$$\begin{aligned} O(X,Y) & \; :- \quad E(X,Y) \\ O(X,Y) & \; :- \quad O(X,Z), E(Z,W), E(W,Y) \\ Q & \; :- \quad O(X,X). \end{aligned}$$

Therefore,

NON 2-COLORABILITY is expressible in 4-Datalog.

k-Datalog and CSP

Definition: For every positive integer k, let

$$D_k \quad = \quad \{\mathbf{B} : \overline{\mathrm{CSP}(\mathbf{B})} \text{ is expressible in } k\text{-Datalog}\}.$$

Fact: If $\mathbf{B} \in \mathcal{D}_k$, then CSP($\mathbf{B}$) is in P.

Proof: The data complexity of Datalog is in P. ∎

Theorem: K … & Vardi – 1998

CSP($\mathcal{F}, \mathcal{D}_k$) is in P, for every positive integer k.

In words, there is a polynomial-time algorithm for uniform CSP, when restricted to inputs $\mathbf{A}$ and $\mathbf{B}$ such that $\mathbf{A}$ is an arbitrary finite σ-structure and $\mathbf{B}$ is a finite σ-structure such that $\overline{\mathrm{CSP}(\mathbf{B})}$ is expressible in k-Datalog.

Main Ingredient of Proof:

Properties of the $(\exists, k)$-pebble games. ∎

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

Theorem 1: Let k be a positive integer.

- k-Datalog$[\mathcal{F}] \subset \exists L^k_{\infty\omega}[\mathcal{F}]$.
- There is a polynomial-time algorithm to decide whether, given two finite σ-structures $\mathbf{A}$ and $\mathbf{B}$, the *Duplicator* wins the $(\exists, k)$-pebble game on $\mathbf{A}$ and $\mathbf{B}$.
- For each finite σ-structure $\mathbf{B}$, there is a k-Datalog program $\pi_{\mathbf{B}}$ that expresses the query: given a finite σ-structure $\mathbf{A}$, does the *Duplicator* win the $(\exists, k)$-pebble game on $\mathbf{A}$ and $\mathbf{B}$?

Theorem 2: TFAE for $k \geq 1$ and a structure $\mathbf{B}$:

- $\overline{\mathrm{CSP}(\mathbf{B})}$ is expressible in k-Datalog
- $\overline{\mathrm{CSP}(\mathbf{B})}$ is expressible in $\exists L^k_{\infty\omega}$
- $\mathrm{CSP}(\mathbf{B}) = \{\mathbf{A} : \textit{Duplicator} \text{ wins the } (\exists, k)\text{-pebble game on } \mathbf{A} \text{ and } \mathbf{B}\}$.

Corollary: $\mathrm{CSP}(\mathcal{F}, D_k)$ is in P, for every $k \geq 1$.

Coping with the Intractability of CSP

Pursuit of tractable cases of CSP:

- Expressibility of CSP in Datalog
  Feder & Vardi – 1993

- Algebraic closure properties of constraints
  Feder & Vardi – 1993
  Jeavons et al. – 1995 to present

- Local and global *consistency properties*
  (constraint propagation)
  Freuder (1978), Dechter (1992), ...

Consistency Properties & Pebble Games

- It turns out that there are tight connections between consistency properties and $(\exists, k)$-pebble games.

- These connections provide a fresh perspective on consistency properties and help identify a new tractable case of CSP.

CSP and Consistency Properties

Definition: An instance (V, D, C) of CSP is

- *i-consistent* if for every partial solution on $i-1$ variables $v_1, \ldots, v_{i-1}$ and for every variable v_i, there is some partial solution on $v_1, \ldots, v_{i-1}, v_i$ extending that solution.

- *strongly k-consistent* if for every $i \leq k$, it is i-consistent.

- *globally consistent* if every partial solution can be extended to a solution.

Note:

Strong k-consistency is a key concept in designing heuristics for CSP and identifying tractable cases.

Consistency Properties and Pebble Games

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

- $\mathcal{I}$ is closed under subfunctions.
- $\mathcal{I}$ has has the *k-forth property:*
  For every $f \in \mathcal{I}$ with $|f| < k$ and every $a \in A$, there is $g \in \mathcal{I}$ such that $f \subseteq g$ and $a \in \mathrm{dom}(g)$.

Observation: Let $\mathbf{A}, \mathbf{B}$ be the instance of the Homomorphism Problem associated with (V, D, C). Then the following are equivalent:

- (V, D, C) is strongly k-consistent.
- The family of all k-partial homomorphisms from $\mathbf{A}$ to $\mathbf{B}$ is a winning strategy for the *Duplicator* in the $(\exists, k)$-pebble game on $\mathbf{A}$ and $\mathbf{B}$.

Establishing Strong k-Consistency

Definition: Let (V, D, C) be an instance of CSP.

Establishing strong k-consistency for (V, D, C) means finding a strongly k-consistent instance (V', D', C') of CSP such that

- $V = V'$ and $D = D'$;
- (V', D', C') is at least as constrained as (V, D, C);
- (V, D, C) and (V', D', C') have the same space of solutions.

Note: Establishing strong k-consistency is a key step in constraint propagation and in identifying tractable cases in which a solution can be found using a backtrack-free search.

When can strong k-consistency be established?

Theorem: K … & Vardi – 2000

The following are equivalent for a CSP-instance (V, D, C).

- It is possible to establish strong k-consistency for (V, D, C).
- The *Duplicator* wins the $(\exists, k)$-pebble game on $\mathbf{A}$ and $\mathbf{B}$, where $\mathbf{A}, \mathbf{B}$ form the instance of the Homomorphism Problem associated with (V, D, C).

Furthermore , when this happens, the set $\mathcal{W}^k(\mathbf{A}, \mathbf{B})$ of all *winning configurations* for the *Duplicator* gives rise to the largest (and, hence, least constrained) CSP-instance that establishes strong k-consistency for (V, D, C).

Intuition: Computing winning strategies for the *Duplicator* amounts to constraint propagation.

Consistency Properties and Datalog

Corollary: Let $\mathbf{B}$ be a fixed finite σ-structure. Then the following are equivalent.

- $\overline{\mathrm{CSP}(\mathbf{B})}$ is expressible in k-Datalog

- For every finite σ-structure $\mathbf{A}$, establishing strong k-consistency for $\mathbf{A}$ and $\mathbf{B}$ implies that there is a homomorphism from $\mathbf{A}$ to $\mathbf{B}$.

Note: Moreover, in this case $\mathrm{CSP}(\mathbf{B})$ is solvable in polynomial-time.

Tractable Cases of CSP

CSP($\mathbf{B}$) is solvable in polynomial time, if

- Establishing strong k-consistency for $\mathbf{A}$ and $\mathbf{B}$ implies that that there is a homomorphism from $\mathbf{A}$ to $\mathbf{B}$.

  ($\overline{\mathrm{CSP}(\mathbf{B})}$ is expressible in k-Datalog)

  - **Advantage:** Wide applicability
  - **Disadvantage:** No information on how to construct a homomorphism

- Establishing strong k-consistency for $\mathbf{A}$ and $\mathbf{B}$ implies that every partial homomorphism from $\mathbf{A}$ to $\mathbf{B}$ can be extended to a homomorphism from $\mathbf{A}$ to $\mathbf{B}$.

  (CSP($\mathbf{B}$) is globally k-consistent)

  - **Disadvantage** Limited applicability
  - **Advantage** A homomorphism can be constructed in a backtrack-free manner.

New Sufficient Condition for Tractability

Definition: CSP($\mathbf{B}$) is *k-local* if $\overline{\mathrm{CSP}(\mathbf{B}^*)}$ is in k-Datalog, where $\mathbf{B}^*$ is the expansion of $\mathbf{B}$ with all elements of B as constants (each element of B is encoded by a unary predicate).

Theorem: Let $\mathbf{B}$ be a finite σ-structure.

- If CSP($\mathbf{B}$) is k-local, then there is a polynomial-time algorith such that, given $\mathbf{A}$, constructs a homomorphism from $\mathbf{A}$ to $\mathbf{B}$ or tells that no homomorphism exists.

- k-locality is in between the previous two tractable cases of CSP.

  In particular, Horn k-Sat is k-local, but is *not* globally m-consistent for any m.

Concluding Remarks

- Pebble games provide new insights to consistency properties for CSP.

- Pebble games shed light on the connections between the different approaches to CSP; in particular, they link consistency properties and expressibility in Datalog.

- Pebble games help identify new tractable cases of CSP.

Nonetheless, several problems remain open ...

Open Problems

Theorem: Feder & Vardi, Jeavons et al.

For every fixed k, there is a polynomial-time algorithm to decide whether establishing strong k-consistency implies global consistency.

Problem: Is expressibility in k-Datalog a decidable property? In other words:

Is there an algorithm that decides, given $\mathbf{B}$, whether $\overline{\mathrm{CSP}(\mathbf{B})}$ is expressible in k-Datalog?

Problem: Is k-locality a decidable property? In other words:

Is there an algorithm that decides, given $\mathbf{B}$, whether $\overline{\mathrm{CSP}(\mathbf{B}^*)}$ is expressible in k-Datalog?

The Complexity of the $(\exists, k)$-Pebble Game

Theorem: For each fixed $k \geq 2$, the following problem is P-complete:

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

Proof:

- Membership in P: K ... & Vardi – 1995
- P-hardness: Implicit in Grohe – 1996; direct proof by K ... & Panttaja – 2001. ∎

Conjecture: K ... & Vardi – 2000

The following problem is EXPTIME-complete:

Given k and structures $\mathbf{A}$, $\mathbf{B}$, does the *Duplicator* win the existential k-pebble game on $\mathbf{A}$ and $\mathbf{B}$?

Note: If this conjecture is true, it will explain why all known algorithms for establishing strong k-consistency are exponential in k.