The quest for a logic for PTIME

Benedikt Pago [1]

ESSLLI 2025, Bochum

[1]University of Cambridge

UNIVERSITY OF CAMBRIDGE
Department of Computer Science and Technology

Comprehensive texts:

- **Lectures 1-3:** textbook *Finite Model Theory and its Applications* by Grädel et al. (mainly this chapter: https://www.logic.rwth-aachen.de/pub/graedel/FMTbook-Chapter3.pdf).
- **Lectures 4-5:** Survey article *A Logic for P: Are we nearly there yet?* by Dawar and P. in ACM SIGLOG News 11.2 (https://dl.acm.org/doi/10.1145/3665453.3665459).

See also list of references at the end of each set of slides.

Course overview

1. What is a logic for polynomial time?
2. Fixed-point logics
3. The Cai-Fürer-Immerman construction
4. Linear-algebraic logics
5. Choiceless Polynomial Time

Let τ be a relational vocabulary. Inductive definition of FO[τ]:

- **Atomic formulas:**
  - $Rx_1 \dots x_r$, where $R \in \tau$ is an r-ary relation symbol, and $x_1, \dots x_r$ are first-order variables.
  - $x = y$ for first-order variables x, y.
- **Logical connectives:** If φ, ψ are formulas, then so are $\varphi \wedge \psi$, $\varphi \vee \psi$ and $\neg\varphi$.
- **First-order quantifiers:** If $\varphi(x)$ is a formula and x a free first-order variable, then $\exists x \varphi(x)$ and $\forall x \varphi(x)$ are formulas.

A τ**-structure** $\mathfrak{A} = (A, R_1^{\mathfrak{A}}, \dots, R_m^{\mathfrak{A}})$ consists of a (finite) universe A and relations $R_i^{\mathfrak{A}} \subseteq A^{\mathrm{ar}\, R_i}$.

Write $\mathfrak{A} \models \psi$ if a sentence $\psi \in$ FO[τ] holds in $\mathfrak{A}$.

Why descriptive complexity?

- **Big goal in TCS:** Algorithmic complexity lower bounds, e.g. showing a problem is not in LOGSPACE, not in PTIME, etc.
- **Problem:** Lower bounds against Turing machines hard to prove.
- **Solution:** Characterise the complexity class $\mathcal{C}$ by a logic $\mathcal{L}$ and prove lower bounds against $\mathcal{L}$.

Proving a problem is not in a complexity class $\mathcal{C}$:

- Suppose there is a logic $\mathcal{L}$ that defines precisely those classes of finite structures that are decidable in $\mathcal{C}$.
- (Ideally) there is a *model-comparison game* for $\mathcal{L}$ such that:

$$\mathfrak{A} \equiv_{\mathcal{L}} \mathfrak{B} \iff \text{Duplicator has a winning strategy in the game played on } (\mathfrak{A}, \mathfrak{B}).$$

- Proving that a class $\mathcal{K}$ of structures is not in $\mathcal{C}$: Define $\mathfrak{A}$ and $\mathfrak{B}$ such that $\mathfrak{A} \equiv_{\mathcal{L}} \mathfrak{B}$ and $\mathfrak{A} \in \mathcal{K}$ but $\mathfrak{B} \notin \mathcal{K}$.

Example: The Ehrenfeucht-Fraïssé Game

Definition

Let $\mathfrak{A}, \mathfrak{B}$ two structures, $k \in \mathbb{N}$ the number of rounds.
The *position* after round $r \leq k$ is $(\bar{a} \in A^r, \bar{b} \in B^r)$. In each round r,

- Spoiler places a pebble $a_r \in A$ or $b_r \in B$.
- Duplicator places the r-th pebble on an element of the other structure.
- If $\bar{a} \to \bar{b}$ does *not* define a *local isomorphism* $\mathfrak{A}[\bar{a}] \to \mathfrak{B}[\bar{b}]$, then Spoiler wins.

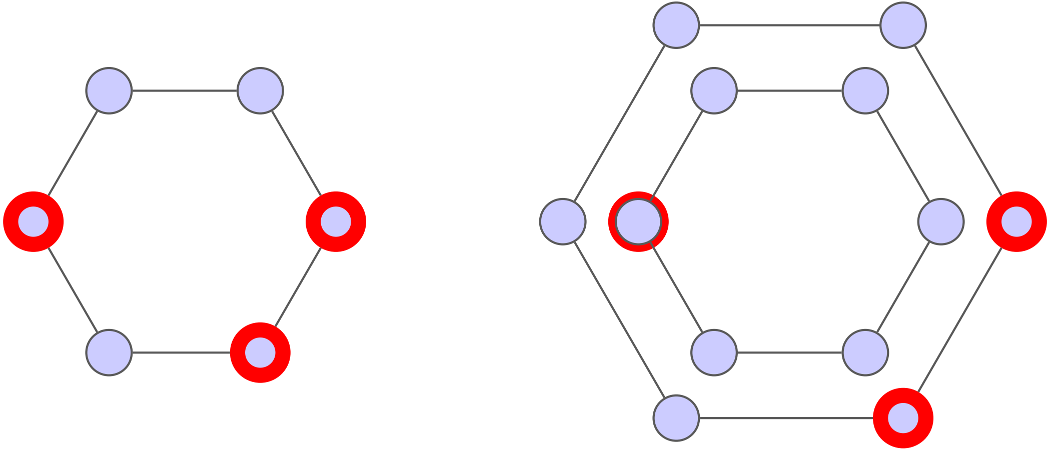
Duplicator wins if Spoiler has not won after k rounds.

Theorem

Duplicator wins the k-round EF-game on $(\mathfrak{A}, \mathfrak{B})$ if and only if no FO-sentence with quantifier rank $\leq k$ distinguishes $\mathfrak{A}$ and $\mathfrak{B}$.

Theorem

There is no FO-sentence that expresses whether a graph is connected.



Suppose we agree on what a "logic" is.

Definition (Capturing complexity classes with logics)

A logic $\mathcal{L}$ captures a complexity class $\mathcal{C}$ if:

1. For every sentence $\psi \in \mathcal{L}$, the model-checking problem (on finite structures) $\mathcal{MC}_\psi$ is in $\mathcal{C}$.
2. For every isomorphism-closed class $\mathcal{K}$ of finite τ-structures whose membership problem is in $\mathcal{C}$, there is a sentence $\psi_\mathcal{K} \in \mathcal{L}$ such that

$$\mathcal{K} = \{\mathfrak{A} \text{ a finite structure with vocabulary } \tau \mid \mathfrak{A} \models \psi_\mathcal{K}\}.$$

What is a logic?

Consider the following "logic" $\mathcal{L} = \{(M, p) \mid M$ a deterministic TM with time bound $p(n)\}$.

Say $\mathfrak{A} \models (M, p) \iff M$ accepts $\mathfrak{A}$ $\mathfrak{A} \models (M, p) \iff M$ accepts code($\mathfrak{A}$).

Problem: "*M accepts* $\mathfrak{A}$" is not well-defined because we cannot input $\mathfrak{A}$ itself into a TM.

Problem: code($\mathfrak{A}$) is not well-defined (e.g. a graph has up to $n!$ different adjacency matrices), and the acceptance behaviour of M depends on code($\mathfrak{A}$), rather than on $\mathfrak{A}$.

Definition

A **logic** $\mathcal{L}$ is a set of sentences with a satisfaction relation $\models$ such that:

1. $\mathcal{L}$ is *isomorphism-invariant*: Whenever $\mathfrak{A} \cong \mathfrak{B}$, then $\mathfrak{A} \models \psi \iff \mathfrak{B} \models \psi$ for all $\psi \in \mathcal{L}$.
2. $\mathcal{L}$ is *decidable*.

This is not restrictive enough to meaningfully talk about $\mathcal{L}$ capturing a complexity class such as PTIME. For example, consider $\mathcal{L} = \mathbb{N}$ with

$$\mathfrak{A} \models n$$
$$\iff \mathfrak{A} \text{ is in the } n\text{-th isomorphism-closed polynomial time decidable class of finite structures.}$$

Gurevich's definition

Definition (Gurevich, 1988)

A **logic capturing PTIME** is a set $\mathcal{L}$ of sentences with a satisfaction relation $\models$ such that:

1. $\mathcal{L}$ is *isomorphism-invariant*.
2. $\mathcal{L}$ is *decidable*.
3. $\mathcal{L}$ is *effective*: There exists a TM M that takes as input $\psi \in \mathcal{L}$ and produces $(M_\psi, p(n))$ such that the machine M_ψ, time-bounded by the polynomial $p(n)$, decides $\mathcal{MC}_\psi$.
4. $\mathcal{L}$ defines precisely the isomorphism-closed classes of finite structures that are PTIME-decidable.

We might also want: $\mathcal{L}$ should admit a useful *tool for proving inexpressibility* results.

Gurevich's conjecture: There is *no logic* that captures PTIME.

Is isomorphism-invariant PTIME syntactic?

- Informally, a complexity class $\mathcal{C}$ is called **syntactic** if it is *recursively enumerable*.
- There is a logic capturing PTIME if and only if the isomorphism-invariant fragment of PTIME is syntactic.

Theorem (Dawar, 1995)

There is a logic capturing PTIME if and only if PTIME has a complete problem under FO-reductions.

Consequences of the (non-)existence of a logic for PTIME

If there is a logic for PTIME, then

- There is a universal algorithm for all problems in PTIME, up to very simple symmetry-preserving preprocessing.
- Both $P \neq NP$ and $P = NP$ are possible.

If there is no logic for PTIME, then

- PTIME is not enumerable and no algorithmic technique solves all problems in P.
- $P \neq NP$, because NP is captured by a logic.

A Logic for NP

The start of descriptive complexity: Fagin's Theorem

Theorem (Fagin, 1974)

An isomorphism-closed class of finite structures $\mathcal{K}$ is in NP if and only if $\mathcal{K}$ is definable by an existential second-order sentence.

Existential Second-Order Logic

The set $\exists$**SO** of existential second-order formulas over a relational vocabulary τ consists of all formulas of the form $\exists X_1 \dots \exists X_m \varphi(X_1, \dots, X_m)$, where $\varphi \in \mathrm{FO}[\tau \cup \{X_1, \dots, X_m\}]$.

Note:

1. $\exists$SO is *isomorphism-invariant*.
2. $\exists$SO has a *decidable* syntax.
3. If P $=$ NP, then $\exists$SO would be *effective* for P in the sense that $\exists$SO-sentences could be compiled into polynomial time Turing machines.

$\Longrightarrow$ If P $=$ NP, then $\exists$SO is a logic capturing P.

Example (3-colourability)

A graph $\mathcal{G}$ is 3-colourable if and only if

$$\begin{aligned}\mathcal{G} \models \exists X \exists Y \exists Z (&\forall x (X\,x \lor Y\,x \lor Z\,x)\\ &\land \forall x (\neg(X\,x \land Y\,x) \land \neg(X\,x \land Z\,x) \land \neg(Y\,x \land Z\,x))\\ &\land \forall x \forall y (Exy \to (\neg(X\,x \land X\,y) \land \neg(Y\,x \land Y\,y) \land \neg(Z\,x \land Z\,y))\end{aligned}$$

Proving Fagin's Theorem: The easy direction.

Theorem (Fagin, 1974)

An isomorphism-closed class of finite structures $\mathcal{K}$ is in NP if and only if $\mathcal{K}$ is definable by an $\exists$SO-sentence.

$\mathcal{K} \in \exists\text{SO} \implies \mathcal{K} \in \text{NP}$:

Lemma

For every sentence $\psi \in \exists\text{SO}$, the model-checking problem $\mathcal{MC}_\psi$ is in NP.

Proof.

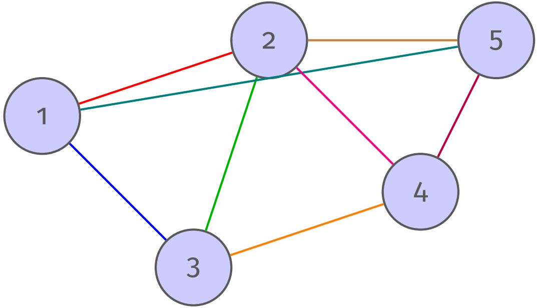
Let $\psi = \exists X_1 \dots \exists X_m \varphi(X_1, \dots, X_m)$. Given a structure $\mathfrak{A}$, guess relations $R_1, \dots, R_m$ over A. Check whether $\mathfrak{A} \models \varphi(R_1, \dots, R_m)$ in deterministic polynomial time.

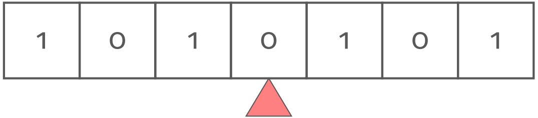
Proving Fagin's Theorem: The hard direction.

Ingredients for proving $\mathcal{K} \in \mathsf{NP} \implies \mathcal{K} \in \exists\mathsf{SO}$:

1. **If a structure $\mathfrak{A}$ has a relation $<$ that defines a linear order on its universe, then an encoding** $\mathrm{code}(\mathfrak{A}, <) \in \{0,1\}^*$ **is FO-definable in $\mathfrak{A}$.**
2. Given $\mathrm{code}(\mathfrak{A}, <)$, $\exists$SO can simulate the run of a non-deterministic polynomial-time TM.
3. Use second-order existential quantifiers to guess a linear order in $\exists$SO.

Inputting a structure to a Turing machine





$(\mathfrak{A}, <) \Longrightarrow \text{code}(\mathfrak{A}, <)$

Logically definable string encodings

- In a linearly ordered τ-structure $(\mathfrak{A}, <)$ with $|A| = n$, $<$ induces a lexicographic order on A^k, for every $k \in \mathbb{N}$. Hence there is a canonical bijection $\iota_\# : A^k \to \{0, \dots, n^k - 1\}$.
- For every r-ary $R \in \tau$, the relation $R^\mathfrak{A} \subseteq A^r$ can be encoded with the string $\chi(R^\mathfrak{A}) = b_1 \dots b_{n^r}$ where:

$$b_i = \begin{cases} 1 & \text{the tuple } \bar{a} \in A^r \text{ with } \iota_\#(\bar{a}) = i \text{ is in } R^\mathfrak{A} \\ 0 & \text{else} \end{cases}$$

- For each $R \in \tau$, $\chi(R^\mathfrak{A})$ is FO-definable in $(\mathfrak{A}, <)$: For every $\sigma \in \{0, 1\}$, there is a formula $\beta^R_\sigma(x_1, \dots, x_r)$ such that

$$(\mathfrak{A}, <) \models \beta^R_\sigma(\bar{a}) \iff \chi(R^\mathfrak{A}) \text{ has } \sigma \text{ at position } \iota_\#(\bar{a}).$$

Logically definable string encodings

- $\text{code}(\mathfrak{A}, <) := 1^{|A|}\chi(R_1^{\mathfrak{A}})\dots\chi(R_m^{\mathfrak{A}})$.
- The *word structure* $(\{1, \dots, \text{len}(\text{code}(\mathfrak{A}, <))\}, P_0, P_1, <)$ that represents the string "$\text{code}(\mathfrak{A}, <)$" is *FO-interpretable* in $(\mathfrak{A}, <)$.

Definition (FO-interpretation)

A σ-structure $\mathfrak{B}$ is FO-interpretable in a τ-structure $\mathfrak{A}$ if there exist formulas $\varphi_\delta(\bar{x}), (\varphi_R)_{R\in\sigma}$ and a $k \in \mathbb{N}$ such that

- $B = \{\bar{a} \in A^k \mid \mathfrak{A} \models \varphi_\delta(\bar{a})\}$
- For each r-ary $R \in \sigma$, $R^{\mathfrak{B}} = \{(\bar{a}_1, \dots, \bar{a}_r) \mid \mathfrak{A} \models \varphi_R(\bar{a}_1, \dots, \bar{a}_r)\}$.

Proving Fagin's Theorem: The hard direction.

Ingredients for proving $\mathcal{K} \in \mathsf{NP} \implies \mathcal{K} \in \exists\mathsf{SO}$:

1. If a structure $\mathfrak{A}$ has a relation $<$ that defines a linear order on its universe, then an encoding code($\mathfrak{A}$), $<$) $\in \{0, 1\}^*$ is FO-definable in $\mathfrak{A}$. ✔
2. **Given** code($\mathfrak{A}, <$)**,** $\exists$SO **can simulate the run of a non-deterministic polynomial-time TM.**
3. Use second-order existential quantifiers to guess a linear order in $\exists$SO.

- **Goal:** Given a polynomial time NTM M, define a sentence $\psi_M \in \exists\text{SO}$ such that

$$\text{code}(\mathfrak{A}, <) \models \psi_M \iff M \text{ has an accepting run on the input string } \text{code}(\mathfrak{A}, <).$$

- **Subgoal:** Define a sentence $\varphi_M(\overline{X})$ such that, for any run of M, encoded as a tuple $\overline{X}$ of relations,

$$(\text{code}(\mathfrak{A}, <), \overline{X}) \models \varphi_M(\overline{X}) \iff \text{the run encoded by } \overline{X} \text{ is accepting.}$$

- Then $\psi_M := \exists \overline{X} \varphi(\overline{X})$.

There is a $k \in \mathbb{N}$ such that every run of M on an input string of length n takes *at most n^k steps*.
Encode a run using relations over $\text{code}(\mathfrak{A}, <)$:

- For every **state** q of M, a relation

$$X_q := \{\bar{t} \in [n]^k \mid \text{ in step } \bar{t}, M \text{ is in state } q\}.$$

- For each **tape symbol** $\sigma \in \{0, 1\}$, a relation

$$Y_\sigma := \{(\bar{t}, \bar{a}) \in [n]^k \times [n]^k \mid \text{ in step } \bar{t}, \text{ the } \bar{a}\text{-th tape cells contains symbol } \sigma\}.$$

- The **head position** relation

$$Z := \{(\bar{t}, \bar{a}) \in [n]^k \times [n]^k \mid \text{ in step } \bar{t}, \text{ the head is at the } \bar{a}\text{-th tape cell}\}.$$

Simulating Turing machines in ∃SO

- **Goal:** Given a polynomial time NTM M, define a sentence $\psi_M \in \exists\mathrm{SO}$ such that

$$\mathrm{code}(\mathfrak{A}, <) \models \psi_M \iff M \text{ has an accepting run on the input string } \mathrm{code}(\mathfrak{A}, <).$$

- **Subgoal:** Define a sentence $\varphi_M(\overline{X})$ such that, for any run of M, encoded as a tuple $\overline{X}$ of relations,

$$(\mathrm{code}(\mathfrak{A}, <), \overline{X}) \models \varphi_M(\overline{X}) \iff \text{the run encoded by } \overline{X} \text{ is accepting.}$$

- Then $\psi_M := \exists \overline{X} \varphi(\overline{X})$.
- **Final step:** ψ_M is evaluated in code($\mathfrak{A}$, … in $(\mathfrak{A}, <)$ to obtain a sentence ψ'_M to be…

φ_M expresses that the relations $\overline{X}$ encode a valid run. The order $<$ is used to compare each time step and its successor.

Proving Fagin's Theorem: The hard direction.

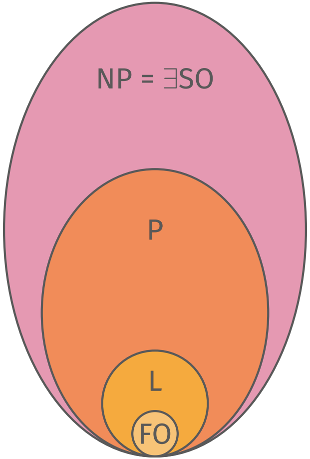
Ingredients for proving $\mathcal{K} \in \mathsf{NP} \implies \mathcal{K} \in \exists\mathsf{SO}$:

1. If a structure $\mathfrak{A}$ has a relation $<$ that defines a linear order on its universe, then an encoding code$(\mathfrak{A}), <) \in \{0,1\}^*$ is FO-definable in $\mathfrak{A}$. ✔
2. Given code$(\mathfrak{A}, <)$, ∃SO can simulate the run of a non-deterministic polynomial-time TM. ✔
3. **Use second-order existential quantifiers to guess a linear order in** ∃SO. ✔

Theorem (Fagin, 1974)

An isomorphism-closed class of finite structures $\mathcal{K}$ is in NP *if and only if $\mathcal{K}$ is definable by an $\exists$SO-sentence.*

This requires **guessing a linear order** $<$ on the input structure $\mathfrak{A}$ so that $\text{code}(\mathfrak{A}, <) \in \{0,1\}^*$ becomes definable in $(\mathfrak{A}, <)$.


NP = ∃SO
P
L
FO

[1] Ashok Chandra and David Harel. **“Structure and complexity of relational queries”.** In: 21st Annual Symposium on Foundations of Computer Science (sfcs 1980). IEEE. 1980, pp. 333–347. 10.1109/SFCS.1980.41.

[2] Anuj Dawar. **“Generalized Quantifiers and Logical Reducibilities”.** In: Journal of Logic and Computation 5.2 (Apr. 1995), pp. 213–226. ISSN: 0955-792X. 10.1093/logcom/5.2.213. https://academic.oup.com/logcom/article-pdf/5/2/213/6244716/5-2-213.pdf. https://doi.org/10.1093/logcom/5.2.213.

[3] Anuj Dawar and Benedikt Pago. **“A Logic for P: Are we Nearly There Yet?”** In: ACM SIGLOG News 11.2 (2024), pp. 35–60. 10.1145/3665453.3665459. https://doi.org/10.1145/3665453.3665459.

[4] Heinz-Dieter Ebbinghaus and Jörg Flum. **Finite Model Theory.** Springer, 1999.

[5] Ronald Fagin. **“Generalized first-order spectra and polynomial-time recognizable sets”.** In: Complexity of computation 7 (1974), pp. 43–73.

[6] Erich Grädel et al. **Finite Model Theory and Its Applications.** Texts in Theoretical Computer Science. An EATCS Series. Springer, 2007. ISBN: 978-3-540-00428-8. 10.1007/3-540-68804-8. https://doi.org/10.1007/3-540-68804-8.

[7] Martin Grohe. **"The quest for a logic capturing PTIME".** In: 2008 23rd Annual IEEE Symposium on Logic in Computer Science. IEEE. 2008, pp. 267–271. 10.1109/LICS.2008.11.

[8] Yuri Gurevich. **"From invariants to canonization".** In: Bulletin of the European Association of Theoretical Computer Science (BEATCS) 63 (1997), pp. 115–119.

[9] Yuri Gurevich. **"Logic and the Challenge of Computer Science".** In: Current Trends in Theoretical Computer Science. Computer Science Press, 1988.

[10] Neil Immerman. **Descriptive Complexity.** Springer Science, 2012.

[11] Leonid Libkin. **Elements of Finite Model Theory.** Springer, 2004.