PTIME Queries Revisited

Alan Nash, Mathematics and CSE Departments
Jeff Remmel, Mathematics Department
Victor Vianu, Computer Science and Engineering Department

University of California, San Diego
Outline

- Languages and Evaluators.
- Two flavors of: “Is there a logic for PTIME?”
- The Blass-Gurevich language.
- Separations of the two flavors.
- Sets of PTIME properties with no logic.
- P-faithful evaluators.
- Finitely generated languages.
- Set Finitely generated languages.
Preliminaries

- Structures can be encoded as graphs.
- Encoding graphs: by encoding adjacency matrix.
- Closure under isomorphism.
- Property of graphs: set of graphs closed under isomorphism.
- $\mathcal{P}_G$: languages in P encoding properties of graphs.
- Language:
  1. Syntax: computable set of strings (programs).
  2. Semantics: mapping to properties of graphs.

$$\varphi \mapsto \{G : G \models \varphi\}$$

- Evaluator: on input $\langle \varphi, G \rangle$, evaluates $G \models \varphi$. 
Evaluators

- Computable: obvious.
- P-bounded: for every fixed program $e$, runs in polynomial time on input $\langle e, G \rangle$. (I.e., data rather than query complexity.)
- Effectively P-bounded: there exists a computable mapping that produces, for every program $e$, a number $k$ such that $E$ runs in time $|G|^k$ on input $\langle e, G \rangle$. 
Languages, Sets of Properties

- A language is computable or (effectively) P-bounded if it has a computable or (effectively) P-bounded evaluator.

- A set of properties is computable or (effectively) P-bounded if it has a computable or (effectively) P-bounded language.
Logic for PTIME

“Is there a logic for PTIME?” corresponds to one of:

1. Is \( P_G \) P-bounded?
2. Is \( P_G \) effectively P-bounded?

- The consensus seems to be that (2) is the correct formulation, but both have been considered.
- Dawar’s result on the equivalence of (2) with “is there a complete complete language for \( P_G \) under FO reductions?” does not go through for (1).
- \( \mathsf{NP}_G \) is effectively NP-bounded as witnessed by \( \exists \mathsf{SO} \) (Fagin’s Theorem).
- We know that \( P_G \) is computable.
The BG Language $L_Y$

- $\varphi$ is $n$-invariant if $(\forall G : |G| \leq n) \forall \leq, \leq'$:

$$(G, \leq) \models \varphi \text{ iff } (G, \leq') \models \varphi.$$  

- $G \models^* \varphi$ iff $(G, \leq) \models \varphi$ and $\varphi$ is $|G|$-invariant.
- Syntax of $L_Y$: FO+LFP sentences over $\{E, \leq\}$.
- Semantics of $L_Y$: given by $\models^*$.
- $L_Y$ is computable and $\text{coNP}$-bounded.
- Since $L_Y$ gives precisely $P_G$, $P_G$ is computable.
- Is $L_Y$ (effectively) $\text{P}$-bounded? We do not know.
Languages

**Theorem 1.** Every computable set of properties $\mathcal{P}$ that includes all finite properties has a computable language $L$ which is not $\mathcal{P}$-bounded.

**Corollary 1.** $P_G$ has a computable language that is not $P$-bounded.

**Theorem 2.** Every $P$-bounded set of properties $\mathcal{P}$ that includes all finite properties has a $P$-bounded language $L$ which is not effectively $P$-bounded.

**Corollary 2.** If $P_G$ is $P$-bounded, then it has a $P$-bounded language that is not effectively $P$-bounded.
Properties

**Theorem 3.** For every PTIME-recognizable set of graphs $G_0$ with infinite complement there exists a computable set of graph properties $\mathcal{H} \subset P_G$ that is not $P$-bounded and includes all PTIME properties of $G_0$.

**Theorem 4.** For every PTIME-recognizable set of graphs $G_0$ with infinite complement such that its set of PTIME properties is $P$-bounded, there exists a $P$-bounded set of graph properties $\mathcal{H} \subset P_G$ that is not effectively $P$-bounded and includes all PTIME properties of $G_0$. 
Faithfulness

Assume $P \neq NP$.

We know $\exists SO$ captures $NP_G$, so it gives $P_G$ and the additional properties in $NP_G - P_G$.

Is there an evaluator for $\exists SO$ which evaluates every PTIME property in PTIME?

Short of an actual language for $P_G$, this would seem like a tempting alternative.

Unfortunately, this is just as hard as finding a logic for PTIME.
Faithfulness

Let $L$ be a language expressing all properties in $P_G$. An evaluator $E$ for $L$ is

- **P-faithful** if $E(e, G)$ runs in polynomial time with respect to $G$ for every fixed $e$ such that $L(e) \in P_G$.

- **effectively P-faithful** if, furthermore, there exists a computable mapping that produces, for every $e$ for which $L(e) \in P_G$, a number $k$ such that $E(e, G)$ runs in time $|G|^k$.

**Theorem 5.** *If there is an (effectively) P-faithful language $L$ for $P_G$, then there is an (effectively) P-bounded language $K$ for $P_G$.***
Finitely-Generated Languages

A finitely-generated language $L$ is defined from finitely many “building blocks” using a finite set of constructors. The classical example of an FGL is FO.

- **Syntax:** terms given by a finite set $C$ of constant symbols and a finite set $F$ of function symbols.
- **Semantics:**
  - to each $c \in C$ we associate a property $K_c$ (a “building block”) defined by a polynomial-time Turing machine $M_c$ and
  - to each $f \in F$ of arity $k$, we associate a polynomial-time Turing machine $M_f$ (a “constructor”) with access to $k$ oracles.
Finitely-Generated Languages

Evaluator $E$ for FGL $L$:

- If $t \in C$, then $E$ on input $\langle t, G \rangle$ runs $M_t$ on input $G$.

- If $t = f(t_1, \ldots, t_k)$ then $E$ on input $\langle t, G \rangle$ runs $M_f$ on input $G$ with oracles $E(t_1, -), \ldots, E(t_k, -)$.

Remark 1. Every FGL is effectively P-bounded.
FGLs and PTIME

Lemma 1. On ordered structures:

(i) Each FGL is included in $\text{FO}+\text{LFP}^r$ for some $r$.

(ii) For every $r$, $\text{FO}+\text{LFP}^r$ is included in some FGL.

Theorem 6.

(i) If there exists an FGL expressing PTIME on ordered structures then $\text{PTIME} \neq \text{PSPACE}$.

(ii) If no FGL expresses PTIME on ordered structures then $\text{LOGSPACE} \neq \text{PTIME}$.

Also open for unordered structures.
Set FGLs

- Set FGLs are a restricted kind of FGLs.
- Set FGLs include FO(\(Q\)), first-order logic with finitely-many Lindström quantifiers.
- It is known from [Dawar & Hella 1995] that FO(\(Q\)) does not capture PTIME.
- Set FGLs operate on hereditarily finite (HF) sets under some restrictions.
- Complex objects can be simulated with HF sets.
Sets and Atoms

- For finite $A$, $HF(A)$ is the smallest superset of $A$ containing the empty set and closed under pairing $\{x, y\}$ and binary union $x \cup y$.
- $tc(x)$, is the smallest set $y$ satisfying
  \[ x \subseteq y \text{ and } \forall u, v (u \in v \in y \rightarrow u \in y). \]
- $\|x\| := |tc(x)|$.
- Given an order of the atoms $A$, we can encode $x$ as a string of length $\|x\|^2$.
- $atoms(x) := tc(x) \cap A$
Rank and Automorphisms

- \( \text{rank}(x) := 0 \) for \( x \in A \cup \{\emptyset\} \) and \( \max(1 + \text{rank}(y) : y \in x) \) otherwise.
- Every permutation \( \sigma \) of \( A \) induces an automorphism \( \sigma \) of \( \text{HF}(A) \).
- \( x \cong y \) if there is a bijection \( \sigma : \text{atoms}(x) \rightarrow \text{atoms}(y) \) such that \( \sigma(x) = y \).
Set FGLs

- Inputs: (encodings of) sets $x \in \text{HF}(\text{atoms}(x))$.
- Only sets can be passed as inputs to oracles, in an isomorphism-invariant way.
- Inputs to oracle calls:
  1. have no new atoms,
  2. are bounded in size in terms of atoms, and
  3. are bounded in rank independently of size.
Set FGLs

- Inputs: (encodings of) sets $x \in \text{HF}(\text{atoms}(x))$.
- $\exists$ number $m$ and function $g$
  $\forall$ terms $t = f(t_1, \ldots, t_k) \, \forall$ inputs $q$ to oracle calls by $M_f$:
  1. $\text{atoms}(q) \subseteq \text{atoms}(x)$
  2. $\|q\| \in O(\text{atoms}(x)^m)$, and
  3. $\text{rank}(q) \leq g(t, \text{rank}(x))$.

- The set $Q^i_t(x)$ consisting of all input sets $q$ to calls to $t_i$ made by $M_f$ in the evaluation of $t$ on input $x$ is independent of the encoding of $x$.

- Closure under isomorphism:
  if $x \cong y$, then for all terms $t$, $x \models t$ iff $y \models t$. 

Simulating FO(Q)

For signature $\tau$ we need:

- one constant $c_R$ for every relation symbol of $\tau$,
- function symbols $f_\neg$, $f_\lor$, $f_\land$ of arity 2,3,3,
- one function symbol $f_Q$ of arity 2 for every Lindström quantifier $Q$,
- a constant $c_\emptyset$ corresponding to $\emptyset$, and
- function symbols $f_p$ and $f_u$ of arity 2 corresponding to pairing and union.

The term $t_\varphi$ providing the simulation mimics the structure of $\varphi \in \text{FO}(Q)$: each logical operator corresponds to a constructor, which makes calls to its oracles on inputs $\mathcal{A}\bar{a}$ consisting of a structure $\mathcal{A}$ over $\tau$ extended with a tuple $\bar{a}$. 
Simulating FO(Q)

We write $t_{\bar{v}}$ for the term that accepts precisely the set representing the tuple of variables $\bar{v}$.

- If $\varphi$ is an atomic formula $R\bar{x}$, then $t_\varphi := c_R$.
- If $\varphi(\bar{z})$ is $\alpha(\bar{x}) \land \beta(\bar{y})$, then $t_\varphi := f_\land(t_\alpha, t_\beta, t_{\langle \bar{x}, \bar{y} \rangle})$ (similarly for $\lor$ and $\neg$).
- If $\varphi(\bar{z})$ is $Q\bar{x}\alpha(\bar{x}\bar{z})$, then $t_\varphi := t_{f_Q}(t_\alpha, t_{\bar{x}})$.

To illustrate, consider the simulation of a conjunction $\alpha(\bar{x}) \land \beta(\bar{y})$. On input $A\bar{a}$, $M_{f_\land}$ first queries its last oracle on atomless sets in their canonical order until some set $s$ is accepted. If $s$ does not encode appropriate tuples of variables, the constructor rejects. Otherwise, $M_{f_\land}$ uses $\bar{x}$ and $\bar{y}$ to obtain from $\bar{a}$ the tuples on which to issue queries to $t_\alpha$ and $t_\beta$. 
Main Result on Set FGLs

Theorem 7. *There is no set FGL that expresses all PTIME properties of graphs.*

- Non-trivial extension of the proof that $\text{FO}(Q)$ does not capture PTIME.
- Instead of equality types of tuples we use isomorphism types of sets.
- Bound on rank gives bound on number of isomorphism types independent of input size.
- We use “naked” sets (sets of atoms) as inputs.
- We can decide $x \models t$ in time $O(||x||^b)$ for fixed $b$.
- The result follows by a straightforward adaptation of the Time Hierarchy theorem.
Supports

- $S \subseteq A$ $A$-supports $x$ if every permutation $\sigma$ of $A$ which fixes $S$ pointwise fixes $x$.

- $\text{supp}_A(x) := S$ where $S \subseteq A$ is the smallest set $S$ which $A$-supports $x$ if there is such $S$ satisfying $|S| < |A|/2$. $\text{supp}_A(x) := A$ otherwise.

- $\text{supp}(x) := \text{supp}_{\text{atoms}}(x)(x)$. 

Main Result on Set FGLs

Proposition 1. If every building block of a set FGL runs in time $O(n^{t_c})$ and every constructor runs in time $O(n^{t_f})$ and has arity at most $k$, then for every term $t$, for every fixed $r, s, m$ and for every $x$ satisfying

- $|\text{supp}(x)| \leq s$,
- $\|x\| \in O(|\text{atoms}(x)|^m)$, and
- $\text{rank}(x) \leq r$,

we can decide $x \models t$ in time $O(|\text{atoms}(x)|^b)$, where $b := m \cdot \max(t_c, t_f, 4)$. 
Proof Outline

• By induction on term depth $d$. For the inductive step we use the following simulation to evaluate term $t = f(t_1, \ldots, t_j)$.
  
  1. We compute as $M_f$ does on input $x$, except for each query $q$ to oracle $i$ we first look for $q'$ isomorphic to $q$ within table $T_i$.
  2. If such $q'$ is found, we do not issue the query and instead use the answer we obtained for $q'$.
  3. Otherwise, we issue the query and add $q$ and the result of the query to the table $T_i$.
• Running time consists of time spent in (1) the body of $M_f$, (2) table lookup, (3) queries.
• We show all three are $O(|\text{atoms}(x)|^b)$. 
Proof Outline 1

We set $n_x = \|x\|$, $a_x = |\text{atoms}(x)|$, $r_x = \text{rank}(x)$, $s_x = |\text{supp}(x)|$, $s_q := s + b$, and $r_q := g(t, r)$.

The time spent in the body of $M_f$ is

$O(n_x^t) \subseteq O(a_x^{mt}) \subseteq O(a_x^b)$. 
Proof Outline 2

To do the table lookup for a query $q$, we first compute its support, which we can do in time $O(a_x^2 n_x^2)$. To check for isomorphism against $q'$ in the table, we try all possible bijections $\sigma : \text{supp}(q) \rightarrow \text{supp}(q')$. By Lemma 2 below we know that $|\text{supp}(q)| \leq s_q$ for large enough $x$, so this adds a factor of $s_q!$. Finally, also by Lemma 2 below we know that the number of isomorphism classes of $q$ depends only on $r_q$ and $s_q$. Since $\|x\| \in O(|\text{atoms}(x)|^m)$, we can do the table lookup in time $O(a_x^{2+2m}) \subseteq O(a_x^b)$. 
Proof Outline 3

We know from above that the total number of queries we need to make depends on \( k, r_q, \) and \( s_q, \) but not on \( n_x. \) We can show that \( |\text{atoms}(q)| \leq s_q \) or \( |\text{atoms}(q)| \geq a_x - s_q. \) If the former holds, \( \|q\| \) is bounded by a constant depending only on \( r_q \) and \( s_q. \) If the latter holds we have \( O(a_x^m) = O(|\text{atoms}(q)|^m). \) Either way, \( \|q\| \in O(|\text{atoms}(q)|^m) \) so we can apply the induction hypothesis using \( r_q \) and \( s_q \) in place of \( r \) and \( s. \) Therefore, we can answer all queries in time \( O(|\text{atoms}(q)|^b) \subseteq O(a_x^b). \)
Support Lemma

Lemma 2. If every constructor runs in time $O(n_x^{b/m})$, then for fixed $s$ and large enough $x$ satisfying 1, 2, and 3 of Proposition 1 every query $q \in Q^t_{x}$ must satisfy $|\text{supp}_A(q)| \leq s + b$ where $A = \text{atoms}(x)$ and therefore also $|\text{supp}(q)| \leq s + b$. The number of isomorphism classes in $Q^t_{x}$ depends only on $g(t,r)$ and $s + b$.

Lemma 2 is an extension of Theorem 24 in [Blass, Gurevich, and Shelah 1999]
Conclusions

- A logic captures $P_G$ iff $P_G$ is eff. P-bounded.
- P-bounded, effectively P-bounded: not the same. Separated on languages and on sets of properties.
- Proper subsets of $P_G$ can be extended so they are not (effectively) P-bounded.
- Faithfulness does not help.
- Whether some FGL captures PTIME on ordered structures depends on hard complexity problems.
- Set FGLs do not capture $P_G$.
- Known candidates for a logic for $P_G$:
  2. Choiceless polynomial time with counting [Blass, Gurevich, and Shelah 2002].