YALE UNIVERSITY
Department of Computer Science

AN OPTIMAL LOWER BOUND ON THE NUMBER OF VARIABLES FOR GRAPH IDENTIFICATION

Jin-Yi Cai and Neil Immerman

YALEU/DCS/TR-626
May 1988
Revised October 1988

An Optimal Lower Bound on the Number of Variables for Graph Identification

Neil Immerman*and Jin-yi Cai†

Computer Science Dept.
Yale University
New Haven, CT 06520

Abstract

In this paper we show that $\Omega[n]$ variables are needed for First-Order logic with counting to distinguish a sequence of pairs of graphs G_n and H_n. These graphs have n vertices each, have color class size 4, and admit a linear time canonization algorithm. This counterexample disposes of several conjectures concerning the sufficiency of first-order logic with counting and v variables, or equivalently the stable colorings of $(v-1)$-tuples of vertices for identifying simple classes of graphs. Our proof shows that the number of variables needed to identify a class of graphs in first-order logic with counting is almost exactly determined by the size of separators for these graphs. We thus determine tight lower bounds on the number of variables needed to identify various classes of graphs in first-order logic with or without counting.

1 Introduction

In this paper we show that $\Omega[n]$ variables are needed for First-Order logic with counting to distinguish a sequence of pairs of graphs G_n and H_n. These graphs have n vertices each, have color class size 4, and admit a linear time canonization algorithm. This contrasts sharply with results in [13] where it is shown that two variables suffice to identify all trees and almost all graphs, and that three variables suffice even without counting to identify all graphs of color class size 3 or less.

Our result disposes of several conjectures concerning the sufficiency of first-order logic with counting and v variables, or equivalently the stable colorings of $(v-1)$-tuples of vertices

*Research supported by NSF grants DCR-8603346 and CCR-8806308.

†Research supported by NSF grant CCR-8709818.

for identifying simple classes of graphs. (We note that Fürer has independently proved a similar counter-example concerning stable colorings [5].) The linear lower bound allows us to precisely determine how many variables are needed to identify many classes of graphs in first-order logic, with or without counting.

In the next section we provide some background material and we precisely state our results. In Section 3 we prove the linear lower bound. Section 4 describes some corollaries and extensions of this work.

2 Background

In [8,9,10,11] one of us has pursued an alternate view of complexity theory in which the complexity of a problem is characterized in terms of the complexity of the simplest first-order sentences expressing the problem. For example, it is shown in [8] that the polynomial-time properties are exactly the properties expressible by first-order sentences iterated polynomially many times:

Fact 2.1 *[8]*

$$P \;=\; \bigcup_{k=1}^{\infty} FO(\leq)[n^k]$$

The notation $\mathrm{FO}(\leq)[n^k]$ denotes the set of properties describable by a very uniform sequence of sentences $\{\varphi_n\}$ such that each sentence φ_n has length $O[n^k]$ and has a bounded number of variables independent of n.[1] The symbol $\leq$ is included to emphasize the presence of a total ordering on the universe of the input structures. In [9] and in [16] it is also shown that this uniform sequence of formulas can be represented by a least fixed point operator (LFP) applied to a single formula. Thus,

$$P \;=\; \mathrm{FO}(\leq) + \mathrm{LFP} \;=\; \bigcup_{k=1}^{\infty} \mathrm{FO}(\leq)[n^k]\,.$$

Fact 2.1 gives a natural language expressing exactly the polynomial-time properties of ordered graphs. Let a ***graph property*** be an order independent property of ordered graphs. One can ask the question,

Question 2.2 *Is there a natural language for the polynomial-time graph properties?*

Since the notion of "natural" is not well defined, some readers may prefer the more precise question:

[1]In [8] the notation $\mathrm{Var\&Sz}[O[1], n^k]$ instead of $\mathrm{FO}[n^k]$ was used.

Question 2.3 *Is there a recursively enumerable listing of a set of Turing machines that accept exactly all the polynomial-time graph properties?*

We remark that should it be the case that graph canonization (i.e. given a graph return a canonical form such that two graphs are isomorphic iff their canonical forms are equal) is in polynomial time, then the answer to Question 2.3 is, "Yes." Thus a negative answer would imply that P is not equal to NP.

Previous to this paper, the only polynomial-time graph properties known not to be expressible in FO + LFP (without ordering) were "counting problems". For example, that a graph has an even number of edges is not expressible in FO + LFP. In [11] the appropriately defined class "FO + LFP + counting" was proposed as an answer to Question 2.2. We show here that this language fails badly on certain linear time properties of graphs.

In [13] and [12] the exact number of variables needed to identify various classes of trees with and without counting, respectively, is determined. (Without counting this number increases linearly with the arity of the trees; with counting two variables suffice.) The question of how many variables are needed to identify various classes of graphs is interesting in its own right, and also has applications to temporal logic. We will show that the number of variables is determined by the separator size of the graphs.

In the remainder of this section we explain some of the logical background we need, including a description of the pebble game used to prove the lower bound. All the relevant background material that we only sketch here may be found in [13].

2.1 First-Order Logic

For our purposes, a *graph* will be defined as a finite first-order structure, $G = \langle V, E\rangle$. V is the universe, (the vertices); and E is a binary relation on V, (the edges).

The *first-order language* of graph theory is built up in the usual way from the variables, $x_1, x_2, \ldots$, the relations symbols, E and $=$, the logical connectives, $\wedge, \vee, \neg, \rightarrow$, and the quantifiers, $\forall$ and $\exists$. The quantifiers range over the vertices of the graph in question. For example consider the following first-order sentence:

$$\varphi \equiv \forall x \forall y [E(x,y) \rightarrow \; E(y,x) \wedge x \neq y]$$

φ says that G is undirected and loop free. We will only consider graphs that satisfy φ, in symbols: $G \models \varphi$.

It is useful to consider a slightly more general set of structures. The *first-order language of colored graphs* consists of the addition of a countable set of unary relations $\{C_1, C_2, \ldots\}$ to the first-order language of graphs.[2] Define a *colored graph* to be a graph that interprets

[2]Coloring relations are a clean tool for restricting the automorphisms of graphs. However, all the coloring relations in this paper could be replaced by simple gadgets in the graphs, without changing any of the results.

these new unary relations so that all but finitely many of the predicates are false at each vertex. These unary relations may be thought of as colorings of the vertices.

Definition 2.4 For a given language $\mathcal{L}$ we say that the graphs G and H are *$\mathcal{L}$-equivalent* ($G \equiv_{\mathcal{L}} H$) iff for all sentences $\varphi \in \mathcal{L}$,

$$G \models \varphi \quad \Leftrightarrow \quad H \models \varphi .$$

We say that $\mathcal{L}$ *identifies* the graph G iff for all graphs H, if $G \equiv_{\mathcal{L}} H$ then G and H are isomorphic. $\mathcal{L}$ *identifies* a set of graphs S if it identifies every element of S.

Of course the First-Order Language of Colored Graphs identifies all colored graphs. From a computational viewpoint it is interesting to consider weaker languages admitting much faster equivalence testing algorithms.

2.2 The Languages $\mathcal{L}_k$ and $\mathcal{C}_k$

Define $\mathcal{L}_k$ to be the set of first-order formulas, φ, such that the quantified variables in φ are a subset of $x_1, x_2, \ldots, x_k$. Note that variables in first-order formulas are similar to variables in programs: they can be reused (i.e. requantified).

Define a *color class* to be the set of vertices which satisfy a particular set of color relations. The *color class size* of a graph is the cardinality of its largest color class. In [13] it is shown that $\mathcal{L}_3$ identifies the set of graphs of color class size 3.

As noted above, the languages $\mathcal{L}_k$ are too weak to count, or even to express the parity of the number of edges. It is thus natural to strengthen these languages by adding *counting quantifiers* to the languages $\mathcal{L}_k$, thus obtaining the new languages $\mathcal{C}_k$. For each positive integer, i, we include the quantifier, $(\exists i\, x)$. The meaning of "$(\exists 17\, x_1)\varphi(x_1)$", for example, is that there exist at least 17 vertices such that φ.

Note that every sentence in $\mathcal{C}_k$ is equivalent to an ordinary first-order sentence with perhaps many more variables and quantifiers. In [13] it is shown that testing $\mathcal{C}_k$ equivalence corresponds to the stable coloring of $k-1$-tuples of vertices. It thus follows that the language $\mathcal{C}_2$ identifies all trees and almost all graphs. In [13], TIME$[n^k \log n]$ algorithms are presented for testing $\mathcal{L}_k$ or $\mathcal{C}_k$ equivalence of graphs on n vertices.

2.3 Pebbling Games

We next describe two pebbling games that are equivalent to testing $\mathcal{L}_k$ and $\mathcal{C}_k$ equivalence, respectively. These games are variants of the games of Ehrenfeucht and Fraisse, [3,4]. Our lower bounds could be proved by induction on the complexity of the sentences in question; but, we find that the games offer more intuitive arguments.

Let G and H be two graphs, and let k be a natural number. Define the $\mathcal{L}_k$ *game* on G and H as follows. There are two players, and there are k pairs of pebbles, $g_1, h_1, \ldots, g_k, h_k$. On each move, Player I picks up any of the pebbles and he places it on a vertex of one of the graphs. (Say he picks up g_i. He must then place it on a vertex from G.[3]) Player II then picks up the corresponding pebble, (If Player I chose g_i then she must choose h_i), and places it on a vertex of the appropriate graph, (H in this case).

Let $p_i(r)$ be the vertex on which pebble p_i is sitting just after move r. Then we say *Player I wins the game at move r* if the map that takes $g_i(r)$ to $h_i(r)$, $i = 1, \ldots, k$, is not an isomorphism of the induced k vertex subgraphs. Note that if the graphs are colored then an isomorphism must preserve color as well as edges. Thus Player II has a winning strategy for the $\mathcal{L}_k$ game just if she can always find matching points to preserve the isomorphism. Player I is trying to point out a difference between the two graphs and Player II is trying to keep them looking the same. The relevant theorem concerning the relationship between this game and the matter at hand is:

Fact 2.5 *[8] Player II has a winning strategy for the $\mathcal{L}_k$ game on G, H if and only if $G \equiv_{\mathcal{L}_k} H$.*

A modification of the $\mathcal{L}_k$ game provides a combinatorial tool for analyzing the expressive power of $\mathcal{C}_k$. Given a pair of graphs define the $\mathcal{C}_k$ game on G and H as follows: Just like the $\mathcal{L}_k$ game we have two players and k pairs of pebbles. The difference is that each move now has two steps.

1. Player I picks up a pebble (say g_i). He then chooses a set, A, of vertices from one of the graphs, (in this case G). Now Player II answers with a set, B, of vertices from the other graph. B must have the same cardinality as A.
2. Player I places h_i on some vertex $b \in B$. Player II answers by placing g_i on some $a \in A$.

The definition for winning is as before. What is going on in the two step move is that Player I asserts that there exist $|A|$ vertices in G with a certain property. Player II answers with the same number of such vertices in H. Player I challenges one of the vertices in B and Player II replies with an equivalent vertex from A. This game captures expressibility in $\mathcal{C}_k$:

Fact 2.6 *[13] Player II has a winning strategy for the $\mathcal{C}_k$ game on G, H if and only if $G \equiv_{\mathcal{C}_k} H$.*

[3]To make the play of the games easier to follow we will use masculine pronouns for Player I and feminine pronouns for Player II.

3 Construction

We construct our counterexample graphs by starting with low degree graphs having only linear size separators. We replace each vertex v of degree k in such a graph by the graph X_k, defined as follows: $X_k = (V_k, E_k)$, where

$$\begin{aligned} V_k &= A_k \cup B_k \cup M_k \text{ where } A_k = \{a_i \mid 1 \le i \le k\},\ B_k = \{b_i \mid 1 \le i \le k\} \\ &\quad \text{and } M_k = \{m_S \mid S \subseteq \{1,\ldots,k\}, |S| \text{ is even}\} \\ E_k &= \{(m_S, a_i) \mid i \in S\} \cup \{(m_S, b_i) \mid i \notin S\} \end{aligned}$$

Thus X_k consists of a set of 2^{k-1} vertices in the middle each connected to one vertex from each of the pairs $\{a_i, b_i\}$, $1 \le i \le k$. Furthermore, each of the middle vertices is connected to an even number of a_i's. (We will assume that the middle vertices (M_k) of X_k have a different color, say magenta, from the others ($A_k \cup B_k$). Furthermore, the pairs a_i and b_i should be able to recognize their mates. If necessary, add vertices c_i colored chartreuse, with edges to a_i and b_i.) The following lemma describes the relevant property of the graph X_k. The proof is immediate.

Lemma 3.1 *Suppose that we color the vertices a_i and b_i of graph X_k with the color i. (Thus all automorphisms of X_k must fix the sets $\{a_1, b_1\}, \ldots, \{a_k, b_k\}$.) Then there are exactly 2^{k-1} automorphisms of X_k. Each is determined by interchanging a_i and b_i for each i in some subset S of $\{1, \ldots, n\}$ of even cardinality.*

Let G be a finite, connected, undirected graph such that every vertex of G has degree at least two. Define the graph $X(G)$ ("X of G") as follows. For each vertex v of G, we replace v by a copy of X_k, call it $X(v)$, where k is the degree of v. To each edge (v, w) of v we associate one of the pairs $\{a_i, b_i\}$ from $X(v)$, call this pair $a(v, w)$ and $b(v, w)$. Finally, we connect the a vertices and the b vertices at each end of each edge, that is we draw the edges $(a(u, v), a(v, u))$ and $(b(u, v), b(v, u))$. If G is a colored graph, then each vertex in $X(v)$ should inherit the color of v. Next, define the graph $\tilde{X}(G)$ ("X twist of G") as follows: In the above construction of $X(G)$ arbitrarily choose one edge (v, w) and twist it, that is reverse the connections, drawing edges $(a(u, v), b(v, u))$ and $(b(u, v), a(v, u))$. In the next lemma we show some relevant properties of $X(G)$ and $\tilde{X}(G)$, including the fact that $\tilde{X}(G)$ is well defined.

Lemma 3.2 *Let G be any finite, connected graph such that every vertex of G has degree at least two. Let $X(G)$ and $\tilde{X}(G)$ be as above. Let $\hat{X}(G)$ be constructed like $X(G)$, but with exactly t of its edges twisted. Then $\hat{X}(G)$ is isomorphic to $X(G)$ iff t is even and $\hat{X}(G)$ is isomorphic to $\tilde{X}(G)$ iff t is odd.*

Proof First observe the following fact about $\hat{X}(G)$. Let v be any vertex of G, and let $(x, v), (y, v)$ be any two edges incident at v. If in $\hat{X}(G)$ we twist both (x, v) and (y, v), then the resulting graph is isomorphic to $\hat{X}(G)$. (This is immediate from Lemma 3.1.)

Now suppose that the number of twists in t is greater than or equal to two. The above observation lets us move the twists towards each other until they overlap and cancel each other out. Thus if t is even then $\hat{X}(G)$ is isomorphic to $X(G)$, otherwise it is isomorphic to $\tilde{X}(G)$.

It remains to show that $X(G)$ is not isomorphic to $\tilde{X}(G)$. Assume for the sake of a contradiction that φ is an isomorphism from $X(G)$ to $\tilde{X}(G)$. Consider the action of φ on any pair $\{a(v, w), b(v, w)\} \subset X(v)$, for (v, w) an edge of G. Because of the colorings in the definition of X_k, φ must map the pair $\{a(v, w), b(v, w)\}$ to some $\{a(v', w'), b(v', w')\}$ in $\tilde{X}(G)$, and thus φ also maps $\{a(w, v), b(w, v)\}$ to $\{a(w', v'), b(w', v')\}$ Define $\oplus\varphi$ to be the sum mod 2 over all such pairs in $X(G)$ of the number of times φ maps an a to a b. Clearly if we consider the two pairs corresponding to every edge (x, y) in G, the number of such switches is either zero or two, except for the unique edge chosen in the construction of $\tilde{X}(G)$, when the number is one. Hence $\oplus\varphi$ is one mod 2. Now let's consider the mod 2 sum in another way, namely in terms of each copy of X_k in $X(G)$. By Lemma 3.1, it is immediate that $\oplus\varphi$ is zero mod 2. This contradiction proves the lemma. ∎

A *separator* of a graph $G = (V, E)$ is a subset $S \subset V$ such that the induced subgraph on $V - S$ has no connected component with more than $|V|/2$ vertices. We now prove our main theorem:

Theorem 3.3 *Let T be a graph such that every separator of T has at least $s + 1$ vertices. Then*

$$X(T) \equiv_{C_s} \tilde{X}(T) .$$

Proof By Fact 2.6, it suffices to give a winning strategy for Player II in the s pebble game on $X(T)$ and $\tilde{X}(T)$. We know by Lemma 3.2 that if we add a twist to any edge of $X(T)$, then the resulting graph is isomorphic to $\tilde{X}(T)$. After the k^{th} move of the game, let R_k be the largest connected component in $T - P_k$ where P_k is the set of vertices $v \in T$ such that just after the k^{th} move there is a pebble on a vertex of $X(v)$ in $X(T)$. Since T has no s separator, we know that R_k contains over half the vertices of T. Player II's winning strategy will be to maintain the following property:

(*) For each vertex $v \in R_k$, let $X^v(T)$ be $X(T)$ with an edge adjacent to v twisted. Then there exists an isomorphism $\alpha_{k,v}$ from $X^v(T)$ to $\tilde{X}(T)$, such that for all $i \leq s$, $\alpha_{k,v}$ maps the vertex under pebble i in $X(T)$ to the vertex under pebble i in $\tilde{X}(T)$.

Clearly if Player II can maintain (*), then the map from the pebbled points in $X(T)$ to the corresponding pebbled points in $\tilde{X}(T)$ is a partial isomorphism, and she wins. We show by induction on k, that Player II can maintain ($*$). First let us make a remark about Player I's moves. As is shown in [13], it always suffices for Player I to restrict himself to choosing a set of monochromatic points at each move. Furthermore, if Player I chooses a (magenta) vertex in the middle of an $X(u)$, then all the other vertices in that $X(u)$ are determined. Therefore, it suffices for Player I to play in a single $M(u)$ per move. Furthermore, since one point in $M(u)$ determines all of $M(u)$, it suffices for Player I to choose only a single point at a time. (Thus counting does not help at all in distinguishing $X(T)$ from $\tilde{X}(T)$!)

Player II's inductive strategy can now be stated. Assume ($*$) holds, and suppose that on move $k+1$ Player I picks up pebble i and puts it down on a vertex in $M(w)$. Note that a new largest component R_{k+1} is determined. Let v be a vertex in $R_k \cap R_{k+1}$. Player II's response is to answer Player I's move according to the isomorphism $\alpha_{k,v}$. To maintain ($*$), let $\alpha_{k+1,v} = \alpha_{k,v}$. Since there is a pebble-free path from v to every other vertex in R_{k+1}, the proof of Lemma 3.2 shows us how to define all the other isomorphisms, $\alpha_{k+1,u}$, $u \in R_{k+1}$. ∎

Corollary 3.4 *There exists a sequence of pairs of graphs $\{G_n, H_n\}$, $n \in \mathbf{N}$ admitting a linear time canonization algorithm and having the following additional properties:*

1. *G_n and H_n have $O[n]$ vertices and color class size four.*
2. *There exists a constant $k > 0$ such that $G_n \equiv_{C_{kn}} H_n$.*
3. *G_n is not isomorphic to H_n.*

Proof This follows immediately from Theorem 3.3 when we let $G_n = X(T_n)$ and $H_n = \tilde{X}(T_n)$ where the T_n's are a sequence of degree three graphs admitting only linear separators, with each vertex of T_n colored a unique color. These graphs are well known to exist, see for example [1]. ∎

4 Corollaries

A long time ago, one of us showed that there is a polynomial-time property of graphs that requires $\Omega[2^{\sqrt{\log n}}]$ quantifiers to express in first-order logic without ordering. That proof also used the graphs $X(D_n)$ and $\tilde{X}(D_n)$, for a certain sequence of degree three graphs $\{D_n\}$ [7, Theorem 7]. Now, Corollary 3.4 improves that lower bound to $\Omega[n]$ variables.[4] It also shows graphically that if we exclude the ordering relation from inductive first-order logic, then the addition of counting does not suffice to express all polynomial-time graph properties. In particular, we have the following:

[4]This is a major improvement because n is much bigger than $2^{\sqrt{\log n}}$, and because a sentence with q quantifiers can make use of at most q variables, but a sentence with v variables can make use of 2^{n^v} quantifiers.

Corollary 4.1 *Let Γ be the set of all graphs of the form $X(G)$, or $\tilde{X}(G)$, for any G a graph of degree at most three and color class size one. Then the isomorphism problems for graphs in Γ is expressible in first-order logic with ordering and sum mod 2, but it is not expressible by any sequence of first-order sentences from $C_{r(n)}$, where $r(n) = o[n]$. In particular, inductive logic with counting, but without ordering does not contain all the graph properties in the low level complexity class AC^0 plus Parity gates, cf. [2].*

Proof We have already seen the lower bound in Corollary 3.4. We must only show that we can distinguish $X(G)$ from $\tilde{X}(G)$ in first-order logic with ordering and sum mod 2. This is easy. The ordering gives us a way to mark each of the pairs $a(u,v)$ and $b(u,v)$ in the graphs. Let $a(u,v)$ be the first of the pair, and $b(u,v)$ the second. (Note that since the vertices in $M(u)$ and $M(v)$ inherit unique colors from u and v, we are given as part of the input which pair of vertices is $a(u,v), b(u,v)$.) Now, given this assignment of a's and b's, a simple first-order sentence asserts that $X(u)$ is straight (i.e. isomorphic to X_3) or twisted (i.e. each vertex in $M(u)$ is adjacent to an odd number of a's). Now, the graph is isomorphic to $X(G)$ iff the sum mod 2 of the number of twisted vertices and edges is 0, and it's isomorphic to $\tilde{X}(G)$ iff the sum mod 2 is 1. ∎

The next result proves a straight forward upper bound that nearly matches our lower bound on the number of variables needed to identify a class of graphs Δ as a function of the separator size of Δ.

Proposition 4.2 *Let Δ be a set of graphs closed under induced subgraph, such that every graph $G \in \Delta$ has a separator of size at most $s(n)$, where n is the number of vertices of G. Then Δ is identified by $C_{V(n)}$ where*

$$V(n) = 3 + \sum_{i=0}^{\lfloor \log n \rfloor} s(\lfloor n2^{-i} \rfloor) \ .$$

(In particular, $V(n) \leq s(n) \log n$, and if $s(n) = n^{\alpha}$, then $V(n) = O[s(n)]$.)

Proof By induction on n, the number of vertices of G. Given G, we can first say that there exist vertices $x_1, \ldots, x_{s(n)}$ such that every connected component of $G - \{x_i | 1 \leq i \leq s(n)\}$ has size at most $\lfloor n/2 \rfloor$. This is expressible in $s(n) + 3$ variables. Next we assert how many connected components of each isomorphism type there are. This requires $V(\lfloor n/2 \rfloor)$ variables, in addition to the $s(n)$ that we leave on $x_1, \ldots, x_{s(n)}$. ∎

5 A Question of Poizat

In [14], Poizat asks whether a complete L_n theory that has at least two non-isomorphic finite models must have infinitely many. This question was answered in the negative by

Thomas [15]. His construction involved graphs with an additional relation of arity n. It is interesting to note that a slight generalization of our construction above produces complete C_n and L_n theories of graphs with any desired number of non-isomorphic models (Theorem 5.3).

First note that the graph X_k defined above embeds the group $(\mathbf{Z}/2\mathbf{Z})$ into any vertex of degree k. A similar construction can be carried out for any finite abelian group, G. For simplicity we present the construction just for $G = (\mathbf{Z}/n\mathbf{Z})$ and $k = 3$. Define the directed graph $X^n = (V^n, E^n)$ as follows:

$$\begin{aligned} V^n &= M^n \cup \bigcup_{i=1}^{3} (\mathbf{Z}/n\mathbf{Z}) \times \{i\}, \\ &\qquad\qquad \text{where } M^n = \{m \in (\mathbf{Z}/n\mathbf{Z})^3 \mid \sum_{i=1}^{3} m_i = 0\} \\ E^n &= \{(m, \langle m_i, i\rangle) \mid 1 \le i \le 3\} \cup \{(\langle a, i\rangle, \langle a+1, i\rangle) \mid a \in (\mathbf{Z}/n\mathbf{Z}), 1 \le i \le 3\} \end{aligned}$$

Thus X^n consists of a set of n^2 vertices in the middle, each connected to one vertex from each of the 3 copies of $(\mathbf{Z}/n\mathbf{Z})$. Furthermore, for each middle vertex m, the sum in $(\mathbf{Z}/n\mathbf{Z})$ of the vertices that m is connected to is equal to 0. Of course, $X^2 = X_3$. The following generalization of Lemma 3.1 is immediate.

Lemma 5.1 *Suppose that we color the vertices $(\mathbf{Z}/n\mathbf{Z}) \times \{i\}$ of graph X^n with the color i. (Thus all automorphisms of X^n must fix the sets $(\mathbf{Z}/n\mathbf{Z}) \times \{i\}$.) Then there are exactly n^2 automorphisms of X^n. Each is determined by performing an m_i step rotation of $(\mathbf{Z}/n\mathbf{Z}) \times \{i\}$ for some $m \in M^n$.*

In particular, Let R be any finite, connected, regular graph of degree three and color class size one. Define the graph $X^n(R)$ to be R with each vertex replaced by X^n. Furthermore, for $0 \le i < n$, let $X_i^n(R)$ be the same graph, with one of the connections twisted by i positions. The following generalization of Lemma 3.2 and Theorem 3.3 is immediate.

Lemma 5.2 *Let R be as above. Let $\hat{X}^n(R)$ be constructed like $X^n(R)$, but with the sum mod n of all its edge twists equal to t. Then $\hat{X}^n(R)$ is isomorphic to $X_i^n(R)$ iff $i = t$. Furthermore, if every separator of R has at least $s + 1$ vertices then*

$$\hat{X}^n(R) \equiv_{C_s} X^n(R)\,.$$

We thus get another solution to Poizat's problem:

Theorem 5.3 *Let $n \ge 2$, and $s \ge 4$. Let R be a color class one, connected, degree three graph with no separator of less than or equal to s vertices. Then the L_s theory and the C_s theory of $X^n(R)$ each admit exactly n models, all of them finite.*

Proof This follows from Lemma 5.2. In L_4 we can say that every pair of vertices, each from a different copy of $(\mathbf{Z}/n\mathbf{Z})$ in any fixed $X^n(r)$, is connected to a unique middle vertex, which in turn is connected to a unique vertex in the third copy of $(\mathbf{Z}/n\mathbf{Z})$. Furthermore, for any pair of vertices m, m' from the same middle component, the sum of the distances from m's neighbors to m''s neighbors in the three copies of $(\mathbf{Z}/n\mathbf{Z})$ is a multiple of n. This last fact is expressible in L_4 because we can name the corresponding neighbors in any of the copies of $(\mathbf{Z}/n\mathbf{Z})$, and then express the distance from one to the other. Thus, the L_4 theory determines that we have one of the $X_i^n(R)$'s; and by Theorem 5.3 the C_s theory can say no more. ∎

6 Conclusions and Open Questions

1. We redirect the reader's attention to Questions 2.2 and 2.3. We have shown in Corollary 3.4 that first-order logic plus counting and least fixed point, but without ordering, fails badly. The question,"What besides counting must be added to $\mathrm{FO} + \mathrm{LFP}$ to get all polynomial-time graph problems?" is worthy of much study, cf. [13,6].

2. Planer graphs have separators of size $O[\sqrt{n}]$, and thus by Proposition 4.2 they can be identified in $C_{\sqrt{n}}$. However, Theorem 3.3 does not give a matching lower bound because even if G is planer, the graph $X(G)$ need not be. We would like to know if $\Omega[\sqrt{n}]$ variables are necessary to identify planer graphs.

Acknowledgements: Thanks to Sandeep Bhatt who improved our results by pointing out that the essential property of the counterexample graphs we were using was that their separators are large. Thanks to Alan Woods for pointing out that our construction solves Poizat's problem.

References

[1] M. Ajtai, "Recursive Construction for 3-Regular Expanders," *28th IEEE FOCS Symp.* (1987), 295-304

[2] David Mix Barrington, Neil Immerman, and Howard Straubing, "On Uniformity Within NC^1," *Third Annual Structure in Complexity Theory Symp.* (1988).

[3] A. Ehrenfeucht, "An Application of Games to the Completeness Problem for Formalized Theories," *Fund. Math.* 49 (1961), 129-141.

[4] R. Fraissé, "Sur les Classifications des Systems de Relations," *Publ. Sci. Univ. Alger* I (1954).

[5] Martin Fürer, "A Counterexample In Graph Isomorphism Testing – Extended Abstract," manuscript (October, 1987).

[6] Yuri Gurevich, "Logic and the Challenge of Computer Science," in *Current Trends in Theoretical Computer Science,* ed. Egon Börger, Computer Science Press.

[7] Neil Immerman, "Number of Quantifiers is Better than Number of Tape Cells," *JCSS* **22**, No. 3, June 1981, 65-72.

[8] Neil Immerman, "Upper and Lower Bounds for First Order Expressibility," *JCSS* **25**, No. 1 (1982), 76-98.

[9] Neil Immerman, "Relational Queries Computable in Polynomial Time," *14th ACM STOC Symp.,* (1982), 147-152. Also appeared in revised form in *Information and Control,* 68 (1986), 86-104.

[10] Neil Immerman, "Languages That Capture Complexity Classes," *SIAM J. Comput.* **16**, No. 4 (1987), 760-778. A preliminary version of this paper appeared in *15th ACM STOC Symp.,* (1983) 347-354.

[11] Neil Immerman, "Expressibility as a Complexity Measure: Results and Directions," *Second Structure in Complexity Theory Conf.* (1987), 194-202.

[12] Neil Immerman and Dexter Kozen, "Definablitity with Bounded Number of Bound Variables," *Second LICS Symp.* (1987).

[13] Neil Immerman and Eric S. Lander, "Describing Graphs: A First-Order Approach to Graph Canonization," Tech Report 605, Yale University Department of Computer Science (1988).

[14] B. Poizat, "Deux ou trois chose que je sais de Ln" *JSL* **47** (1982), 641-658.

[15] Simon Thomas, "Theories With Finitely Many Models," *J. Symbolic Logic,* **51**, No. 2 (1986), 374-376.

[16] M. Vardi, "Complexity of Relational Query Languages," *14th Symposium on Theory of Computation,* 1982, (137-146).