Finite Model Theory and Graph Isomorphism. II.

Anuj Dawar

University of Cambridge Computer Laboratory
visiting RWTH Aachen

Beroun, 13 December 2013

Recapitulation

Finite Model Theory aims to study the expressive power of logic on finite structures.

The *expressiblity* of classes of finite structures is closely related to their *computational complexity*.

To prove that properties are not definable in a logic, we seek examples of graphs that are *distinguished* by the property but not by the logic.

Recapitulation. II.

This leads to an exploration of notions of *indistinguishability* that *stratify* the graph isomorphism relation.

We looked at two stratifications, in terms of *quantifier rank* ($\equiv_p$) and *number of variables* ($\equiv^k$).

These have characterisations in terms of two-player *games*.

Deciding Graph Isomorphism

Graph Isomorphism: Given graphs G, H, decide whether $G \cong H$ is

- not known to be in P
- not expected to be NP-complete.

In practice and *on average*, graph isomorphism is efficiently decidable.

Tractable Approximations of Isomorphism

A *tractable approximation* of graph isomorphism is a *polynomial-time decidable* equivalence $\equiv$ on graphs such that:

$$G \cong H \quad \Rightarrow \quad G \equiv H.$$

Practical algorithms for testing graph isomorphism typically decide such an approximation.

If this fails to distinguish a pair of graphs G and H, more discriminating tests are deployed.

Vertex Classification

The following problem is easily seen to be computationally equivalent to graph isomorphism:

> *Given a graph G and a pair of vertices u and v, decide if there is an automorphism of G that takes u to v.*

Given G and H, let $G + u$ denote the graph extending G with a new vertex u adjacent to *all* vertices in G, and similarly for $H + v$.

Then, $G \cong H$ *if, and only if,* in the graph $(G + u) \oplus (H + v)$, there is an automorphism taking u to v.

Equivalence Relations

The algorithms we study aim to decide equivalence relations on *vertices* (or tuples of vertices) that approximate the *orbits* of the automorphism group.

For such an equivalence relation $\equiv$, we also write $G \equiv H$ to indicate that G and H are not distinguished by the corresponding isomorphism test.

> *For connected graphs, this means that for every u in G, there is a v in H so that $u \equiv v$ in the disjoint union of G and H.*

Partition Refinement

For a pair of k-tuples $\mathbf{a}, \mathbf{b} \in V(G)^k$, we write $\mathbf{a} \equiv^k \mathbf{b}$ to denote that there is no formula of L^k that distinguishes the two tuples.

The equivalence relation $\equiv^k$ on $V(G)^k$ can be obtained through a series of *refinements*:

$$\equiv^k_0 \ \supseteq \ \equiv^k_1 \ \supseteq \cdots \supseteq \ \equiv^k_i \quad \cdots$$

where $\mathbf{a} \equiv^k_0 \mathbf{b}$ iff the map $\mathbf{a} \mapsto \mathbf{b}$ is a *partial isomorphism* and

$\mathbf{a} \equiv^k_{i+1} \mathbf{b}$ iff for each $j (1 \leq j \leq k)$ and each $u \in V(G)$, there is a $v \in V(G)$ such that

$$\mathbf{a}[u/a_j] \equiv^k_i \mathbf{b}[v/b_j]$$

and *vice versa*.

Computing Partition Refinements

$\mathbf{a} \equiv_i^k \mathbf{b}$ iff *Duplicator* has a strategy for i moves of the k-pebble game starting from position $\mathbf{a}, \mathbf{b}$.

We obtain the relation $\equiv^k$ by starting with the classsification of k-tuples given by $\equiv_0^k$ and *iteratively* refining it.

Each step requires $n^{O(k)}$ work and there are at most n^k steps of refinement.

Thus, $\equiv^k$ is decidable in time $n^{O(k)}$.

Is There a Logic for P?

The question of whether or not there is a logic expressing exactly the P properties of *(unordered) relational structures* is the central problem in *Descriptive Complexity*.

If we assume structures are *ordered*, then FP, the extension of first-order logic with least fixed points suffices. **(Immerman; Vardi 1982)**

In the absence of order FP fails to express simple cardinality properties such as *evenness*.

Fixed-point Logic with Counting

Immerman had proposed FPC—the extension of FP with a mechanism for *counting*

Two sorts of variables:

- $x_1, x_2, \ldots$ range over $|A|$—the domain of the structure;
- $\nu_1, \nu_2, \ldots$ which range over *numbers* in the range $0, \ldots, |A|$

If $\varphi(x)$ is a formula with free variable x, then $\nu = \#x\varphi$ denotes that ν is the number of elements of A that satisfy the formula φ.

We also have the order $\nu_1 < \nu_2$, which allows us (using recursion) to define arithmetic operations.

Expressive Power of FPC

Most *"obviously"* polynomial-time algorithms can be expressed in FPC.

Many non-trivial polynomial-time algorithms can be expressed in FPC:

- FPC captures all of P over any *proper minor-closed class of graphs* **(Grohe 2012)**
- FPC can express *linear programming* problems; *max-flow* and *maximum matching* on graphs. **(Anderson, D., Holm 2013)**

But some cannot be expressed. How do we prove this?

Counting Quantifiers

C^k is the logic obtained from *first-order logic* by allowing:

- *counting quantifiers*: $\exists^i x\, \varphi$; and
- only the variables $x_1, \ldots. x_k$.

Every formula of C^k is equivalent to a formula of first-order logic, albeit one with more variables.

For every sentence φ of FPC, there is a k such that if $G \equiv^{C^k} H$, then

$$G \models \varphi \quad \text{if, and only if,} \quad H \models \varphi.$$

Counting Game

Immerman and Lander (1990) defined a *pebble game* for C^k.
This is again played by *Spoiler* and *Duplicator* using k pairs of pebbles $\{(a_1, b_1), \ldots, (a_k, b_k)\}$.

At each move, Spoiler picks i and a set of vertices of one graph (say $X \subseteq V(H)$)

Duplicator responds with a set of vertices of the other graph (say $Y \subseteq V(G)$) of the same size.

Spoiler then places a_i on an element of Y and Duplicator must place b_i on an element of X.

Spoiler wins at any stage if the partial map from G to H defined by the pebble pairs is not a partial isomorphism

If Duplicator has a winning strategy for p moves, then G and H agree on all sentences of C^k of quantifier rank at most p.

Bijection Games

$\equiv^{C^k}$ is also characterised by a k-pebble *bijection game*. **(Hella 96)**.
The game is played on graphs G and H with pebbles $a_1, \ldots, a_k$ on G and $b_1, \ldots, b_k$ on H.

- *Spoiler* chooses a pair of pebbles a_i and b_i;
- *Duplicator* chooses a bijection $h : V(G) \to V(H)$ such that for pebbles a_j and $b_j (j \neq i)$, $h(a_j) = b_j$;
- *Spoiler* chooses $a \in V(G)$ and places a_i on a and b_i on $h(a)$.

Duplicator loses if the partial map $a_i \mapsto b_i$ is not a partial isomorphism.
Duplicator has a strategy to play forever if, and only if, $G \equiv^{C^k} H$.

Equivalence of Games

It is easy to see that a winning strategy for *Duplicator* in the bijection game yields a winning strategy in the counting game:

> *Respond to a set $X \subseteq V(G)$ (or $Y \subseteq V(H)$) with $h(X)$ ($h^{-1}(Y)$, respectively).*

For the other direction, consider the partition induced by the equivalence relation

$$\{(a, a') \mid (G, \mathbf{a}[a/a_i]) \equiv^{C^k} (G, \mathbf{a}[a'/a_i])\}$$

and for each of the parts X, take the response Y of *Duplicator* to a move where *Spoiler* would choose X.
Stitch these together to give the bijection h.

Counting Tuples of Elements

We could consider extending the counting logic with quantifiers that count *tuples* of elements.

This does not add further expressive power.

$$\exists^i \overline{xy}\ \varphi$$

is equivalent to

$$\bigvee_{f \in F} \bigwedge_{j \in \mathrm{dom}(f)} \exists^{f(j)} x\ \exists^j y\ \varphi$$

where F is the set of finite partial functions f on $\mathbb{N}$ such that $(\sum_{j \in \mathrm{dom}(f)} jf(j)) = i$.

Thus, there is no strengthening to the game if we allow *Spoiler* to move more than one pebble in a move (with *Duplicator* giving a bijection between sets of tuples.)

Vertex Classification Algorithms

We return to *vertex classification algorithms* for *graph ismorphism*.

Recall,

> *The algorithms we study aim to decide equivalence relations on vertices (or tuples of vertices) that approximate the orbits of the automorphism group.*

For such an equivalence relation $\equiv$, we also write $G \equiv H$ to indicate that G and H are not distinguished by the corresponding isomorphism test.

> *For connected graphs, this means that for every u in G, there is a v in H so that $u \equiv v$ in the disjoint union of G and H.*

Equitable Partitions

An equivalence relation $\equiv$ on the vertices of a graph $G = (V, E)$ induces an *equitable partition* if

for all $u, v \in V$ *with* $u \equiv v$ *and each* $\equiv$*-equivalence class* S,

$$|\{w \in S \mid (u, w) \in E\}| = |\{w \in S \mid (v, w) \in E\}|.$$

The *naive vertex classification* algorithm finds the *coarsest* equitable partition of the vertices of G.

Colour Refinement

Define, on a graph $G = (V, E)$, a series of equivalence relations:

$$\equiv_0 \;\supseteq\; \equiv_1 \;\supseteq \cdots \supseteq\; \equiv_i \quad \cdots$$

where $u \equiv_{i+1} v$ if they have the same number of neighbours in each $\equiv_i$-equivalence class.

This converges to the coarsest equitable partition of G.

The coarsest equitable partition can be computed in *quadratic time*.

Almost All Graphs

Naive vertex classification provides a simple test for isomorphism that works on *almost all graphs*:

> *For graphs G on n vertices with vertices u and v, the probability that $u \equiv v$ goes to 0 as $n \to \infty$.*

But the test fails miserably on *regular graphs*.

Weisfeiler-Lehman Algorithms

The *k-dimensional Weisfeiler-Lehman* test for isomorphism (as described by **Babai**), generalises naive vertex classification to k-tuples.

For a graph G, let $\equiv^{WL^k}$ be the coarsest equivalence relation on k-tuples of vertices so that for k-tuples $\mathbf{u}$ and $\mathbf{v}$, if $\mathbf{u} \equiv^{WL^k} \mathbf{v}$, then:

$\mathbf{u}$ *and* $\mathbf{v}$ *induce isomorphic subgraphs*

and for each k-tuple $\alpha_1, \ldots, \alpha_k$ of $\equiv^{WL^k}$-classes,

$$|\{u \mid \bigwedge_j \mathbf{u}[u/u_j] \in \alpha_j\}| = |\{v \mid \bigwedge_j \mathbf{v}[v/v_j] \in \alpha_j\}|$$

Induced Partitions

In other words,

Given an equivalence relation $\equiv$ on V^k, each k-tuple $\mathbf{u}$ induces a *labelled partition* of V.
The labels of the partition are k-tuples $\alpha_1, \ldots, \alpha_k$ of $\equiv$-equivalence classes, and the corresponding part is the set:

$$\{u \mid \bigwedge_j \mathbf{u}[u/u_j] \in \alpha_j\}.$$

Define $\equiv'$ to be the equivalence relation where $\mathbf{u} \equiv' \mathbf{v}$ if, in the partitions they induce, the corresponding parts *have the same cardinality*.
Then, $\equiv^{WL^k}$ is the limit of the sequence:

$$\equiv_0 \ \supseteq \ \equiv_1 \ \supseteq \cdots \supseteq \ \equiv_i \quad \cdots$$

where $\mathbf{u} \equiv_0 \mathbf{v}$ if, and only if, they induce isomorphic subgraphs and $\equiv_{i+1}$ is $\equiv_i'$.

Weisfeiler-Lehman Algorithms

If G, H are n-vertex graphs and $k < n$, we have:

$$G \cong H \quad \Leftrightarrow \quad G \equiv^{WL^n} H \quad \Rightarrow \quad G \equiv^{WL^{k+1}} H \quad \Rightarrow \quad G \equiv^{WL^k} H.$$

$G \equiv^{WL^k} H$ is decidable in time $n^{O(k)}$.

The equivalence relations $\equiv^{WL^k}$ form a *family* of tractable approximations of graph isomorphism.

It is not difficult to show that $G \equiv^{C^{k+1}} H$ if, and only if, $G \equiv^{WL^k} H$.

Graph Isomorphism Integer Program

Yet another way of approximating the *graph isomorphism relation* is obtained by considering it as a *0/1 linear program*.

If A and B are adjacency matrices of graphs G and H, then $G \cong H$ if, and only if, there is a *permutation matrix* P such that:

$$PAP^{-1} = B \quad \text{or, equivalently} \quad PA = BP$$

Introducing a variable x_{ij} for each entry of P and adding the constraints:

$$\sum_i x_{ij} = 1 \quad \text{and} \quad \sum_j x_{ij} = 1$$

we get a system of equations that has a *0-1 solution* if, and only if, G and H are isomorphic.