On the Complexity of Learning and Testing Hyperfinite Graphs

Krzysztof Onak*
CMU
konak@cs.cmu.edu

April 20, 2012

Abstract

We show a simple algorithm for learning families of hyperfinite graphs. This gives a simple proof of the main theorem of Newman and Sohler (STOC 2011) that every property of hyperfinite graphs is testable with a constant number of queries.

For minor-free graphs, the query complexity of learning and testing is $2^{(1/\varepsilon)^{O(1)}}$, where ε is the proximity parameter. For minor-closed families of graphs that do not exclude trees, we give a query lower bound of $2^{\Omega(1/\varepsilon)}$.

It is known due to a result of Czumaj, Shapira, and Sohler (SICOMP 2009) that all hereditary properties of hyperfinite graphs are testable with a constant number of queries. We show that one-sided testing of hereditary properties may require an arbitrarily fast growing number of queries as a function of the proximity parameter. We also show a trivial property that requires $\Omega(n)$ queries in the case of one-sided testing.

1 Introduction

Hyperfinite graphs can be partitioned into constant size components by cutting an arbitrarily small fraction of edges. Important examples of bounded-degree hyperfinite families of graphs include bounded-degree graphs with an excluded minor [AST90] (for instance, bounded-degree planar graphs, bounded-degree graphs with constant tree-width), graphs of subexponential growth[1] [Ele08], and the family of non-expanding bounded-degree graphs considered by Czumaj, Shapira, and Sohler [CSS09]. We defer a formal definition of hyperfiniteness to the next section.

In *property testing*, the goal is to design efficient algorithms that accept inputs that have a specific property and reject inputs that are significantly different from any input with the property. These algorithms can err with small constant probability. In this paper we focus on the number of queries to the input graph that such algorithms have to make, and we ignore other computational issues. In particular, some of our properties may not be decidable, but even then, there exist non-uniform testing algorithms for them.

A recent paper of Newman and Sohler [NS11] shows that every property of constant-degree hyperfinite graphs can be tested with a number of queries that is independent of the size of the graph. In an earlier paper, Czumaj, Shapira, and Sohler [CSS09] prove that every hereditary

*Supported by a Simons Postdoctoral Fellowship.

[1] The *growth* of a graph or a family of graphs is a function $g : \mathbb{Z}_+ \to \mathbb{Z}_+$ such that $g(d)$ equals the maximum number of vertices at distance at most d from any vertex d.

property (which includes a number of important properties such as k-colorability, H-minor-freeness, perfectness, etc.) can be tested with a constant number of queries and one-sided error, i.e., a graph that has the property is always accepted with probability 1.

1.1 Overview of Results

Learning and Testing. Our main result is a simple proof of the theorem of Newman and Sohler [NS11] that every property of constant-degree hyperfinite graphs can be tested with a number of queries that does not depend on the size of the input graph. We use partitioning oracles of Hassidim et al. [HKNO09] to show that such graphs can in fact be learned up to an arbitrarily small fraction of edges. A partitioning oracle provides access to a good partition of the input graph into constant-size components. By sampling we learn how vertices are allocated to different types of constant-size components, which allows to almost recover the original graph. This result appears in Section 3.

Let us note that this approach in fact suffices to obtain other types of sublinear algorithms from the paper of Newman and Sohler [NS11] such as a tester for graph isomorphism and approximation algorithms for graph parameters. See [NS11] for more details.

Exponential Lower Bound for Testing. Our main result implies that any property of constant-degree minor-free graphs can be tested with $2^{\mathrm{poly}(1/\varepsilon)}$ queries. We match that bound by showing that any class of constant-degree hyperfinite graphs that does not exclude forests of degree bounded by 3, requires $2^{\Omega(1/(\varepsilon))}$ queries for some specific property. In particular this implies that any minor-free class of graphs that does not exclude a cycle-free minor requires $2^{\Omega(1/(\varepsilon))}$ queries for testing a specific property and for learning. This result is presented in Section 4.

Separation between One-Sided and Two-Sided Testing. Even though we give a query upper bound on the number of queries necessary to test any property of a specific class of constant-degree hyperfinite graphs, one-sided error testing (i.e., always accepting inputs with a specific property) turns out to be much more difficult in many cases. First, we show a trivial property that requires $\Omega(n)$ queries for one sided testing. Second, recall that Czumaj et al. [CSS09] show that all hereditary properties are testable with one-sided error. Despite that we show a hereditary properties that require arbitrarily quickly growing numbers of samples as a function of the proximity parameter. These results appear in Section 5.

2 Preliminaries

2.1 Graph Definitions

We start with a definition of hyperfiniteness.

Definition 1 (Hyperfiniteness)

- *Let $G = (V, E)$ be a graph. G is* (ε, k)-hyperfinite *if it is possible to remove $\varepsilon|V|$ edges of the graph such that the remaining graph has connected components of size at most k.*
- *Let ρ be a function from $\mathbb{R}_+$ to $\mathbb{R}_+$. A graph G is* ρ-hyperfinite *if for every $\varepsilon > 0$, G is $(\varepsilon, \rho(\varepsilon))$-hyperfinite.*

- *Let $\mathcal{C}$ be a family of graphs. $\mathcal{C}$ is* ρ-hyperfinite *if every graph in $\mathcal{C}$ is ρ-hyperfinite.*

Throughout the paper we focus on *graph properties*. Graph properties are closed under permutation (or in other words, relabeling) of vertices. Graph properties that are closed under vertex removal are called *hereditary*.

2.2 Model

Throughout the paper we assume that the number of vertices in the input graph is n and that the maximum degree in the graph is bounded by $d \geq 2$.

We use the bounded-degree model [GR02], in which an algorithm has access to the number of vertices in the graph, vertices from the graph selected uniformly at random, the degree of each vertex, and the list of neighbors of each vertex. In the last case, the algorithm specifies in its query a vertex v and an integer i between 1 and $\deg(v)$ and obtains the label of the i-th neighbor of v.

2.3 Distance

We now provide a number of definitions related to the distance between graphs and the distance of a graph to a property.

Definition 2 (Distance between graphs) *Let G and G' be two graphs on n vertices of maximum degree bounded by d. Let T be the minimum number of edges that have to be modified in G (either inserted into G or removed from G) so that G and G' become isomorphic. The* distance *between G and G' is T/dn.*

Definition 3 (Distance from a property) *Let G be a graph on n vertices. Let $\mathcal{P}$ be a graph property, i.e., a family of graphs closed under the permutation of vertices. Let $\mathcal{P}_n$ be the subfamily of G consisting of graphs on n vertices. The* distance of G to $\mathcal{P}$ *is the minimum distance between G and a graph in $\mathcal{P}_n$ if $\mathcal{P}_n$ is non-empty, and 2 if $\mathcal{P}_n = \emptyset$.*

Definition 4 (ε-far and ε-close) *We say that a graph G is* ε-far *from a graph property $\mathcal{P}$ if the distance of G from $\mathcal{P}$ is at least ε. Analogously, we say that G is* ε-close *to $\mathcal{P}$ if the distance of G from $\mathcal{P}$ is less than ε.*

2.4 Learning and Testing Algorithms

We now provide definitions of learning and testing algorithms.

Definition 5 (Learning algorithm) *We say that an algorithm is an* ε-learning algorithm *if given access to an input graph G, it outputs a graph G' such that with probability $2/3$, the distance between G and G' is ε.*

Definition 6 (Testing algorithm) *We say that an algorithm is an* ε-testing algorithm *(or an* ε-tester*) for a graph property $\mathcal{P}$ if*

- *it accepts every input that has the property $\mathcal{P}$ with probability $2/3$,*
- *it rejects every input that is ε-far from $\mathcal{P}$ with probability $2/3$.*

Moreover, if the testing algorithm never rejects an input that has the property $\mathcal{P}$, we say that it test with one-sided error *or that it is a* one-sided tester*.*

3 Efficient Learning and Testing

3.1 Preliminaries

We now define the main tool that we use in this section. A partitioning oracle provides query access to a global partition of the graph into small components.

Definition 7 *We say that $\mathcal{O}$ is an* (ε, k)-partitioning oracle *for a family $\mathcal{C}$ of graphs if given query access to a graph $G = (V, E)$ in the adjacency-list model, it provides query access to a partition P of V. For a query about $v \in V$, $\mathcal{O}$ returns $P[v]$. The partition has the following properties:*

- *P is a function of the graph and random bits of the oracle. In particular, it does not depend on the order of queries to $\mathcal{O}$.*
- *For every $v \in V$, $|P[v]| \leq k$ and $P[v]$ induces a connected graph in G.*
- *If G belongs to $\mathcal{C}$, then $|\{(v, w) \in E : P[v] \neq P[w]\}| \leq \varepsilon|V|$ with probability $9/10$.*

Let $\mathcal{D}_1$ and $\mathcal{D}_2$ be two distributions on a finite support X. Let $p_1(x)$ and $p_2(x)$ denote the probabilities of an event $x \in X$ in $\mathcal{D}_1$ and $\mathcal{D}_2$, respectively. We write $|\mathcal{D}_1 - \mathcal{D}_2|_1$ to denote $\sum_{x \in X} |p_1(x) - p_2(x)|$.

Another important fact that we use is that by collecting a specific number of independent samples from an unknown distribution $\mathcal{D}$, we obtain a distribution that is very likely to be close $\mathcal{D}$. This result is easy to prove via the Chernoff bound.

Fact 8 (folklore) *Let $\mathcal{D}$ be a distribution on a support of size n. Let $\mathcal{D}'$ be an empirical distribution resulting from $O(n/\varepsilon^2)$ independent samples from $\mathcal{D}$, where the constant hidden by the big-Oh notation is sufficiently large. With probability $99/100$, $|\mathcal{D} - \mathcal{D}'|_1 \leq \varepsilon$, where $|\mathcal{D} - \mathcal{D}'|_1$.*

3.2 Learning Graphs

The following theorem shows that there is a learning algorithm that uses a partitioning oracle and makes a limited number of queries to the oracle and the input graph.

Theorem 9 *If there is an $(\varepsilon d/2, k)$-partitioning oracle $\mathcal{O}$ for a class $\mathcal{C}$ of graphs, then there is an ε-learning algorithm for $\mathcal{C}$. The algorithm makes $O(2^{k^2}/\varepsilon^2)$ queries to $\mathcal{O}$ and additionally $O(k^2 \cdot 2^{k^2}/\varepsilon^2)$ queries to the input graph.*

Proof The oracle $\mathcal{O}$ provides query access to a partition of the input graph. Every component in the partition has at most k vertices. Let Q_k be the number of different unlabeled graphs on at most k vertices. It is easy to show that $Q_k \leq 2^{k^2}$. Each vertex in the partition belongs to one of Q_k different components. Let $\mathcal{D}$ be the distribution of vertices over different component types. It follows from Fact 8 that with probability $99/100$, we can learn a distribution $\mathcal{D}'$ such that $|\mathcal{D} - \mathcal{D}'|_1 \leq \varepsilon/4$ by sampling $O(Q_k/\varepsilon^2)$ vertices and learning their components. The exploration of each component takes at most $O(k^2)$, so the total number of queries necessary to learn $\mathcal{D}'$ is bounded by $O(k^2 \cdot 2^{k^2}/\varepsilon^2)$.

Next the algorithm searches for a graph $G_\star$ on n vertices with a distribution $\mathcal{D}''$ of vertices over different component types such that $|\mathcal{D}'' - \mathcal{D}'|_1 \leq \varepsilon/4$. If $\mathcal{D}'$ has been computed correctly,

such a distribution $\mathcal{D}''$ exists and moreover, $|\mathcal{D}'' - \mathcal{D}|_1 \leq \varepsilon/2$. We now bound the distance between the partition provided by $\mathcal{O}$ and $G_\star$. The total number of vertices which have to be moved to components of different type is bounded by $\varepsilon n/4$. Their neighbor lists are the only ones that we have to modified to turn one of the graphs into the other. The total number of edge insertions and deletions is bounded by $\varepsilon dn/2$, which bounds the distance between the graphs by $\varepsilon/2$.

Now observe that the oracle $\mathcal{O}$ removes at most $\varepsilon dn/2$ edges with probability 9/10. In this case the distance between the partition and the original graph is bounded by $\varepsilon/2$. By the triangle inequality, the distance between the input graph and $G_\star$ is bounded by ε provided all the specified events hold. The probability that any of them does not hold is bounded by $1/100 + 1/10 \leq 1/3$. Summarizing, our algorithm is an ε-learning algorithms. ■

3.3 Ramifications for Testing

In this section we present implications of the learning algorithm for property testing by plugging in different partitioning oracles. We start with a partitioning oracle for all hyperfinite graphs.

Theorem 10 (partitioning oracle for hyperfinite graphs, [HKNO09], see also [Ona10]) *Let G be an $(\varepsilon, \rho(\varepsilon))$-hyperfinite graph with degree bounded by $d \geq 2$. There is an $(\varepsilon d, \rho(\varepsilon^3/3456000))$-partitioning oracle with the following properties. The oracle answers every query, using $2^{d^{O(\rho(\varepsilon^3/3456000))}}/\varepsilon$ queries to the graph.*

We plug in the oracle from Theorem 10 into Theorem 9 to obtain the following result. The testing result follows by using the learning result with ε set to $\varepsilon/3$. This way if the learning algorithm does not fail, we learn the graph sufficiently well to distinguish graphs with a given property from graphs ε-far from the property.

Corollary 11 (Learning and testing complexity for hyperfinite graphs) *There is an ε-learning algorithm for a ρ-hyperfinite family of graphs that makes $2^{d^{O(\rho(\varepsilon^3/27648000))}}/\varepsilon^3$ queries to the input graph. There is an ε-testing algorithm for a ρ-hyperfinite family of graphs that makes $2^{d^{O(\rho(\varepsilon^3/2239488000))}}/\varepsilon^3$ queries to the input graph.*

And now we move on to minor-free graphs and an oracle for them.

Theorem 12 (partitioning oracle for minor-free graphs, [HKNO09], see also [Ona10]) *Let H be a fixed minor. For every H-minor-free graph G with degree bounded by $d \geq 2$, there is an $(\varepsilon d, C/\varepsilon^2)$-partitioning oracle, where C is a constant that only depends on H. The oracle makes $d^{\mathrm{poly}(1/(\varepsilon \cdot d))}$ queries to the input graph per query.*

This time we obtain the following corollary.

Corollary 13 *Let H be a fixed minor. There is an ε-learning algorithm for H-minor-free graphs that makes $d^{1/\mathrm{poly}(\varepsilon \cdot d)}/\varepsilon^{O}(1)$ queries to the input graph. There is an ε-testing algorithm for H-minor-free graphs that makes $d^{1/\mathrm{poly}(\varepsilon \cdot d)}/\varepsilon^{O}(1)$ queries to the input graph.*

4 Lower Bound for Testing Properties of Minor-Free Graphs

We omit an easy proof of the following lemma. It is easy to prove it by showing how to encode exponentially large numbers using graphs on n vertices. One simple approach is to create a path and attach additional vertices to it to encode $\Omega(n)$ different bits.

Lemma 14 *The number of different unlabeled trees on n vertices with maximum degree bounded by 3 is of order $2^{\Omega(n)}$.*

We also use the following fact from distribution testing, which easily follows from the birthday paradox.

Fact 15 (folklore, see [GR11, BFF$^+$01]) *Distinguishing between the uniform distribution on $[n]$ and the uniform distribution on the random half of $[n]$ with probability $101/200$ requires $\Omega(\sqrt{(n)})$ independent samples.*

Theorem 16 *Let $\mathcal{C}$ be a hyperfinite family of graphs with the maximum degree bounded by $d \geq 3$ such that all trees of maximum degree bounded by 3 belong to $\mathcal{C}$. There is a property $\mathcal{P}$ that requires $2^{\Omega(1/(\varepsilon \cdot d))}$ queries to be tested on $\mathcal{C}$.*

Proof For each k, let T_k be the number of different unlabeled trees with maximum degree bounded by 3. A graph has the property $\mathcal{P}$ if it consists of disjoint trees on k vertices for some k, and each of the T_k trees has the same number of occurrences. (In particular this implies that $k \cdot T_k$ divides n.)

Let $\varepsilon = 1/(2dk)$ for some large k. For some large n, consider two distributions over graphs. $\mathcal{D}_{\text{accept}}$ is simply a random permutation of the graph containing the same number of disjoint copies of each tree on k vertices. $\mathcal{D}_{\text{reject}}$ is generated by taking a random half S' of the set of all different trees of size k, and then applying a random permutation of vertices to the graph consisting of the same number of copies of each tree in S'.

What is the distance of each graph in $\mathcal{D}_{\text{reject}}$ from $\mathcal{P}$? How does one turn it into a graph that has the property $\mathcal{P}$. One option is to change the size of each tree. This would require at least $n/(2k)$ edge modification, which gives the distance of at least $1/(2dk) = \varepsilon$. Another option is to modify at least half the copies of trees so that each tree is represented the same number of times. This would require at least one edge deletion in half the trees, which again gives the distance of at least $1/(2dk) = \varepsilon$.

What is the query complexity for distinguishing $\mathcal{D}_{\text{reject}}$ and $\mathcal{D}_{\text{accept}}$? If we had a tester with better query complexity than $O(\sqrt{T_k}) = 2^{\Omega(k)} = 2^{\Omega(1/(d\varepsilon))}$, we could use it to construct a sampling algorithm that would contradict the lower bound in Fact 15. ■

5 Separation between One-Sided and Two-Sided Testing

In this section, we show that one-sided testing may require a much higher number of queries than two-sided testing.

5.1 $\Omega(n)$ Lower Bound

First, we exhibit a trivial property that has no constant-query one-sided tester on almost any minor-closed family of graphs.

Observation 17 *Let $\mathcal{F}_1$ be the family of all graphs with maximum degree bounded by 1. Let $\mathcal{C}$ be a hyperfinite family of graphs such that $\mathcal{F} \subseteq \mathcal{C}$. Let $\mathcal{P}$ be the property of having at most $n/4$ edges. A one-sided $1/(8d)$-tester for $\mathcal{P}$ has to make at least $\Omega(n)$ queries.*

Proof Consider a graph G on $n \geq 2$ vertices that consists of $\lfloor n/2 \rfloor$ independent edges. To turn G into a graph with the property $\mathcal{P}$, we have to remove from G at least $\lfloor n/2 \rfloor - \lfloor n/4 \rfloor \geq n/4 - 1/4 \geq n/8$ edges. This implies that the distance of G from $\mathcal{P}$ is at least $1/(8d)$.

Before we pass G to the tester, we randomly permute its vertices. Now to ensure with probability 1 that the input graph is G, not a graph consisting of $\lfloor n/4 \rfloor$ independent edges, the tester has to make at least $\lceil n/4 \rceil$ queries. ■

5.2 Lower Bound for Hereditary Properties

The query complexity of two-sided testing for a specific family of hyperfinite graphs can be bounded by a function that depends only on the proximity parameter ε, and in particular does not depend on the property being tested. Czumaj, Shapira, and Sohler [CSS09] show that the query complexity of one-sided testing a specific hereditary property for a specific family hyperfinite graphs can be bounded by a function of only the proximity parameter. We now show that the complexity of one-sided testing can grow arbitrarily fast as a function of the proximity parameter.

Theorem 18 *Let $\mathcal{F}_2$ be the family of all graphs with maximum degree bounded by 2. Let $\mathcal{C}$ be a hereditary family of graphs such that $\mathcal{F}_2 \subseteq \mathcal{C}$. Let $q : \mathbb{N}_+ \to \mathbb{N}_+$ be an arbitrary function. There is a hereditary property $\mathcal{P}$ such that for $k \geq 3$, one-sided $1/(3dk)$-testing of graphs from $\mathcal{F}_2$ for $\mathcal{P}$ requires at least $q(k)$ queries.*

Proof Every hereditary property can be specified by a list of forbidden induced subgraphs. We define a property $\mathcal{P}$ by adding to this list the graph consisting of $q(k)$ disjoint cycles of length k for each $k \geq 3$.

Fix $k \geq 0$ and consider two sequences of graphs G_n and H_n on n vertices:

- G_n consists of $\min\{q(k) - 1, \lfloor n/k \rfloor\}$ disjoint cycles of length k and the remaining vertices are isolated.
- H_n consists of $\lfloor n/k \rfloor$ disjoint cycles of length k and the remaining vertices are isolated.

Observe that each G_n has the property $\mathcal{P}$. On the other hand, for $n \geq k \cdot q(k)$, H_n does not have the property. The number of cycles in H_n divided by n goes to $1/k$ as $n \to \infty$. To make H_n have the property $\mathcal{P}$, we have to modify all but at most $q(k) - 1$ cycles. Since each edge touches at most two cycles, this requires a number of edge modifications that divided by n can be bounded from by below by a quantity that converges to to $1/(2k)$. This implies that the distance of H_n from $\mathcal{P}$ is bounded from below by a quantity that converges to $1/(2dk)$ as $n \to \infty$. Therefore, for sufficiently large n, H_n is $1/(3dk)$-far from $\mathcal{P}$.

For sufficiently large n, if we pass H_n to a one-sided $1/(3dk)$-tester with randomly permuted vertices, it has to make at least $q(k)$ queries to touch $q(k)$ disjoint cycles in order to ensure that it does not reject G_k with randomly permuted vertices. ■

6 Open Questions

At least two interesting questions are left open:

- **What properties of hereditary graphs can be tested with one-sided error?** The work of Alon and Shapira [AS08] for dense graphs may serve here as an inspiration. Note that in this case there are properties that are significantly different from hereditary properties, but can still be tested with one-sided error by oblivious testers[2]. As an example consider the property in which no induced K_3 is allowed, unless it belongs to an induced K_4.
- **Is there a hereditary property $\mathcal{P}$ such that two-sided testing of planar graphs for $\mathcal{P}$ requires $2^{(1/\varepsilon)^{\Omega(1)}}$ queries?** In particular it would be interesting to see a hereditary property with the list of forbidden induced subgraphs reduced to graphs consisting of single connected components.

References

[AS08] Noga Alon and Asaf Shapira. A characterization of the (natural) graph properties testable with one-sided error. *SIAM J. Comput.*, 37(6):1703–1727, 2008.

[AST90] Noga Alon, Paul D. Seymour, and Robin Thomas. A separator theorem for graphs with an excluded minor and its applications. In *STOC*, pages 293–299, 1990.

[BFF+01] Tugkan Batu, Lance Fortnow, Eldar Fischer, Ravi Kumar, Ronitt Rubinfeld, and Patrick White. Testing random variables for independence and identity. In *FOCS*, pages 442–451, 2001.

[CSS09] Artur Czumaj, Asaf Shapira, and Christian Sohler. Testing hereditary properties of nonexpanding bounded-degree graphs. *SIAM J. Comput.*, 38(6):2499–2510, 2009.

[Ele08] Gábor Elek. L^2-spectral invariants and convergent sequences of finite graphs. *Journal of Functional Analysis*, 254(10):2667 – 2689, 2008.

[GR02] Oded Goldreich and Dana Ron. Property testing in bounded degree graphs. *Algorithmica*, 32(2):302–343, 2002.

[GR11] Oded Goldreich and Dana Ron. On testing expansion in bounded-degree graphs. In *Studies in Complexity and Cryptography*, pages 68–75. 2011.

[HKNO09] Avinatan Hassidim, Jonathan A. Kelner, Huy N. Nguyen, and Krzysztof Onak. Local graph partitions for approximation and testing. In *FOCS*, 2009.

[NS11] Ilan Newman and Christian Sohler. Every property of hyperfinite graphs is testable. In *STOC*, pages 675–684, 2011.

[Ona10] Krzysztof Onak. *New Sublinear Methods in the Struggle Against Classical Problems.* PhD thesis, Massachusetts Institute of Technology, 2010.

[2] A tester is oblivious if it does not know the number of vertices in the input graph.