Property Testing of Graphs and the Role of Neighborhood Distributions

Dissertation

zur Erlangung des Grades eines

Doktors der Naturwissenschaften

der Technischen Universität Dortmund
an der Fakultät für Informatik

von
Hendrik Fichtenberger

Dortmund
2020

Tag der mündlichen Prüfung:	11. Februar 2020
Dekan:	Prof. Dr. Gernot Fink
Gutachter:	Prof. Dr. Christian Sohler
	Prof. Dr. Artur Czumaj

Abstract

Property testing considers decision problems in the regime of sublinear complexity. Most classical decision problems require at least linear time complexity in order to read the whole input. Hence, decision problems are relaxed by introducing a gap between "yes" and "no" instances: A property tester for a property Π (e. g., planarity) is a randomized algorithm with constant error probability that accepts objects that have Π (planar graphs) and that rejects objects that have linear edit distance to any object from Π (graphs with a linear number of crossing edges in every planar embedding). For property testers, locality is a natural and crucial concept because they cannot obtain a global view of their input. In this thesis, we investigate property testing in graphs and how testers leverage the information contained in the neighborhoods of randomly sampled vertices: We provide some structural insights regarding properties with constant testing complexity in graphs with bounded (maximum vertex) degree and a connection between testers with constant complexity for general graphs and testers with logarithmic space complexity for random-order streams. We also present testers for some minor-freeness properties and a tester for conductance in the distributed CONGEST model.

Given a graph, the k-disk of a vertex is the subgraph induced by a BFS of depth k; the histogram of the collection of all k-disks – which summarizes the graph's local neighborhoods – is called k-disk frequency vector. It is known that one can approximate the k-disk frequency vector of any given bounded-degree graph by the k-disk frequency vector of some constant-size graph. We show how to construct such a small approximation graph if the input graph has girth $2k + 2$. Constant-query testers for bounded-degree graphs are closely related to k-disk frequency vectors. It is known that, roughly speaking, all such testers approximate the k-disk frequency vector of the input graph and perform a dictionary lookup to output their decision. We build on this observation and deepen the connection to k-disks by showing that every constant-query testable property contains an infinite hyperfinite subproperty.

For graphs without bounded degree, the number of k-disk isomorphism types depends on the size of the graph at hand, which complicates the analysis of property testers. However, we observe that constant-query testers can only see a constant number of partial k-disk isomorphism types. We define canonical testers for general graphs based on this observation, and we prove that every one-sided error, constant-query tester can be transformed into a logarithmic-space, random-order streaming tester.

Using techniques based on k-disks, one can prove that minor-freeness can be tested with constant query complexity and two-sided error, but not with constant query complexity and one-sided error. We present a partial result on the way to a tight upper bound on one-sided error testing by giving a one-sided error tester for several minor-free properties such as outerplanarity and cactus graphs. Finally, we investigate property testing in the distributed CONGEST model, where every vertex is a processor and communication is only allowed between adjacent vertices. We present a tester for conductance in general graphs.

Contents

Errata. https://www.hendrik-fichtenberger.de/link/phd-thesis-errata

1. Introduction

Over the past decades, our data storage capacities have grown vastly. This has led to vivid developments in areas such as natural sciences, informatics and statistics. Although the speed of information processing hardware has also developed significantly, the data that is accumulated nowadays is often too large to be processed as a whole in order to obtain the desired information. This kind of data is often referred to as *Big Data*, and this term has become a catch phrase in many areas. The main reason of the emerging gap between the amount of available information and the amount of processable information lies in the fact that many classic algorithms' running time is polynomial or exponential in the input size – or worse. Algorithm theory is one of the disciplines of computer science that has answers to this phenomena.

The main impetus of classic approximation algorithms comes as hardness results regarding the computational complexity of a problem. For example, the traveling salesperson problem (TSP) is known to be NP-hard even for metric instances. Christofides' algorithm obtains an approximate solution that is within a factor of $3/2$ of the optimum but it requires only $\mathcal{O}(n^3)$ time.

However, the intractability of large data does not arise solely from the hardness of computations but from the size of the data we seek to obtain answers from. For example, we may not be able to even store all data that we read from the output of physical sensor arrays like the Large Hadron Collider. Or we may have stored the whole data set, but in a distributed fashion that does not allow us to gather all the data in a single place with reasonable effort: For example, one might want to compile some statistics of most frequent words on public pages of the World Wide Web. While all the necessary data is stored *somewhere* and can be accessed electronically, it would require a lot of time and it would generate a lot of traffic to request all this information in order to evaluate it (even if we had a list of all relevant URLs). And even then we would need some algorithm that calculates the desired statistics with only a tiny amount of space available to it compared to the total size of the data it reads.

1.1. A Theoretician's Framework for Large Data

These problems have led to the development of *sublinear* algorithms. Here, sublinear means that the algorithm's requirements of some resource of interest – usually time or space – is asymptotically strictly smaller than the input size. Some classic algorithms that come to one's mind are sublinear but have been invented long before the term *sublinear algorithms* was coined. For example, binary search has logarithmic time complexity, and maintaining a counter for the number of elements in a data structure has logarithmic space complexity. These examples are well known to computer scientists that have never peaked into the area of sublinear algorithms, and they are quite illustrative.

With only small technical modifications, one can count the elements when they are *streamed* instead of given as a whole. In the *streaming model*, the input is a sequence of elements that is

read from its beginning to its end without winding back (i. e., no random access). Counting the fraction of numbers that, e. g., are larger than some threshold Θ boils down to maintaining two counters: one that is incremented for every element that we read, and one that is incremented only for elements that exceed Θ. At any point of time, we may return the fraction of elements larger than Θ in the stream that we have seen so far by dividing one counter by the other. One main insight of algorithms in the streaming model is that a summary of the data set is often enough to compute an (approximate) solution.

As its sublinear time complexity does not allow the algorithm to read the whole data set, the insight of binary search is different: One cannot only formulate the problem using the divide and conquer paradigm but one can always discard one recursion branch per divide step in advance. Sublinear time complexity arises from the fact that the necessary information to solve the problem is not only a small subset of the input but one can also find it without reading the whole input. The algorithm only needs to *query* some part of the input. For binary search, there is even a trail of breadcrumbs that starts at a known position (the center element of the array) and leads to the key very fast. As we will see, the structural properties that sublinear algorithms exploit can be much more subtle – and most algorithms are neither deterministic nor exact. In fact, only the strong promise that the array is sorted enables us to find a key with a deterministic algorithm in the case of binary search.

In this work, we mainly consider sublinear algorithms that read only a small amount of the input. In particular, such an algorithm may read the input by querying an oracle for atomic parts of it one by one. If the input is an array, such a query may, e. g., ask for the i-th element of the array. This mode of access to the input is called *query model* or *oracle access model.* Two popular types of problems that are studied in the query model are approximation problems and property testing problems. In a nutshell, property testing problems are relaxed decision problems. The relaxations of the problem for property testing resemble those of polynomial-time randomized approximation schemes (*PRAS*). A property tester may err with constant probability when it answers, and there is a slack between the *truth* and the *guaranteed quality* of the algorithm's answer. The latter relaxation deserves a more detailed explanation.

The following loosely recited motivation is due to Ilan Newman and captures the essence of property testing in a descriptive and picturesque way. Imagine that you intend to buy a used car, and you want to make sure that it is in working order. Ideally, it is in factory-fresh condition, but that is unlikely as there will be some damages for sure – it is used after all. However, you cannot fully disassemble the car to check everything (even if you had the time, the owner would probably object). Therefore, you want to design some test for used cars that inspects just a few random parts and judges it such that you do not miss the ideal car, you do not buy something that turns out to be a total loss either, and you do not care if it misjudges a car that is neither of these extremes. This is the setting of property testing.

Phrasing this a bit more formally, we have a *universe* of objects and a subset of this universe, which we call a *property*. We aim to design a randomized algorithm that, given a gap parameter $\epsilon \in (0, 1]$ and an object from the universe, decides whether it is in the property or ϵ-far from it; we define that an object is *ϵ-far* from a property if one needs to modify more than an ϵ-fraction of the object's *natural* representation so that it has the property. For example, when we consider a graph with n vertices and m edges, its adjacency list representation has size $\Theta(n+m)$, which may vary between $\Omega(n)$ and $\mathcal{O}(n^2)$ depending on the number of edges. Therefore, we allow $\epsilon(n + m)$ modifications in this model. We do not account for logarithmic factors due to *bit* representation. A two-sided error ϵ-tester may answer incorrectly with probability at most

1/3. If a tester has one-sided error, it must not fail to accept objects that have the property, and therefore it has to present a witness against the property when it rejects. Note that in both cases, the tester may answer arbitrarily when the object neither has the property nor is ϵ-far from the property. We say that objects that are not ϵ-far are *ϵ-close* to the property. Since the main goal of property testing is to study how much of the input has to be read to solve an approximate decision problem, the complexity of a property tester is often measured in terms of queries to the input graph. This is roughly the number of elements of the graph's natural representation the tester queries, but it is not an upper bound on the total running time in general. Sometimes the time complexity, which is obviously at least the query complexity, is also analyzed, and fortunately, it is often asymptotically comparable to the query complexity.

Since the first results in property testing by Blum, Luby, and Rubinfeld [BLR90] (see also [RS92; BLR93]) and Rubinfeld and Sudan [RS96] were published, property testing problems have been formulated and analyzed for a bunch of different types of objects. Examples are (real) functions, sequences, graphs, probability distributions, images and geometric point sets. Property testing started in the traditional complexity setting of sequential computation, but recent developments extended graph property testing to streams or the distributed LOCAL and CONGEST models. The recently published book by Goldreich [Gol17] gives an overview and an introduction to property testing.

1.2. Overview of Results

Note on Joint Work

All the results presented in this thesis would not have been achieved without the joint work with my wonderful coauthors Artur Czumaj, Reut Levi, Pan Peng, Christian Sohler, Maximilian Wötzel and Yadu Vasudev. Detailed acknowledgments can be found in the respective chapters and Appendix A.

Locality is a natural concept in sublinear algorithms: Since, e. g., property testers have sublinear query complexity, they can only explore small parts of the input. Nevertheless, different types of locality arise. We start by exploring a very appealing definition of locality for graphs that is not related to algorithms at all: The *k-disk* of a vertex v (also known as k-hop neighborhood) is defined as the rooted subgraph that is induced by all vertices in the graph that have distance at most k to v. It is obvious that one way to define a notion of *neighborhood* for a vertex is to use its k-disk. But we can also use k-disks to define *locality statistics* of a whole graph: omitting the vertex labels of all k-disks, we can consolidate k-disks that are isomorphic and compute a histogram of k-disk isomorphism types. Since each vertex has exactly one k-disk, this histogram sums ups to n, and we can normalize it to obtain a *k-disk frequency vector*. A bunch of interesting questions arises: What does the k-disk frequency vector tell us about the whole graph? Is there a unique way to stitch the k-disk isomorphism types and obtain the original graph? Are there distributions of k-disks that can only be observed for very large graphs, even approximately? While the first question is very broad, the answer to the second question is no. We observe this while actually investigating the third question in Chapter 2. More precisely, the answer to the third questions is also no, and we will strive to obtain an upper bound on the size such that every k-disk frequency vector can be approximated with a graph of at most this size for k-disks that have no cycles.

Although the concept of k-disks does not require to bring algorithms into play, constant-query property testers for bounded-degree graphs have a very close connection to k-disks. Say that a property tester makes at most q queries to its input graph. There is no way it can explore more than a q-disk – even for a single vertex. One can prove that sampling q many q-disks is enough to serve any property tester with query complexity q: every such property tester can be simulated by querying q random q-disks in advance and answering the original tester from this pool. In a nutshell, this means that constant-query property testers for bounded-degree graphs approximate the k-disk frequency vector of the input graph and base their decision solely on this estimate. Together with the fact that one can tell from the k-disk frequency vector of a graph whether this graph is close to being planar or not, this gives an intuitive understanding why planarity can be tested with constant query complexity (and two-sided error). One interesting property of bounded-degree planar graphs is that they can be decomposed into arbitrarily small connected components by removing an accordingly chosen fraction of their edges. Graphs that have this property, which is not restricted to planar graphs, are called *hyperfinite*. All these results circle around k-disks and k-disk frequency vectors, and in Chapter 3 we distill that *every* infinite constant-query testable property of bounded-degree graphs contains an infinite hyperfinite subproperty. In fact, even constant-query property testers in general graphs can be characterized by k-disks: in Chapter 4, we show that every such tester can be reduced to an algorithm that subsamples random q-disks, where q is the query complexity of the tester. This result arises from the insight that in general graphs, constant-query testers can only perform a random exploration of the neighborhood of vertices with large degree. We relate this result to random-order streams and show that every constant-query tester for general graphs that queries adjacency lists can be transformed into a random-order streaming property tester that uses $\mathcal{O}(\log n)$ bits of space. Roughly speaking, the uniformly random order of edges in random-order streams allows us to collect the neighborhood in a similar way as a query-based tester.

As mentioned above, planarity is constant-query testable with two-sided error. More generally, minor-freeness can be tested with constant query complexity and two-sided error. But how about one-sided error? The answer to this question was unknown for some time after the result for two-sided error was published, and a property tester for general minor-freeness was only published recently. In Chapter 5, we present a one-sided error tester for outerplanarity and some other families of forbidden minors. Our property tester uses a combinatorial approach together with a clustering of the graph – another notion of locality. Clustering asks to group objects that are *close* under some notion of dissimilarity. If the input is a set of points, this is usually some metric, e. g., the ℓ_2-distance. In our case, the notion of dissimilarity is the length of the shortest path between two vertices, and the size of our clusters is sublinear in the size of the graph. We study how these local parts of the graph, i. e., the clusters, interact with each other and show that, roughly, a large edge-cut between two clusters implies a minor.

So far, we have only discussed property testers that probe the neighborhood of a sublinear set of vertices. Again, this is natural because classic property testers have sublinear query complexity. However, one can study other models of computation than the (sequential and centralized) query model. One family of models that gained some popularity recently is distributed graph computation. In the LOCAL and CONGEST models, every vertex of the input graph is a processor that can only communicate with its neighbors, i. e., its adjacent vertices. The complexity of a distributed algorithm in this model is the number of synchronized rounds of communication it requires. An interesting observation is that if an algorithm has round

complexity r, it cannot exchange messages with vertices that have distance greater than r. This gives rise to another notion of locality, namely, a massively parallel locality because all vertices compute and communicate simultaneously. Yet, there is no way for two vertices that are far apart (or even disconnected) to communicate with each other. We investigate the quality of the overall communication flow and develop a distributed property tester for conductance in general graphs in Chapter 6.

1.3. Preliminaries

The definitions and results in this section are the base for all subsequent chapters. However, some formal definitions or statements are only referred to in non-technical sections, and they are given in the following for the sake of completeness only. They are printed in a lighter color.

Hopefully, this thesis is a self-contained work for everyone on the level of a graduate student in computer science. In addition, you may refer to the following books, which reflect standard references or the author's preferences. An algorithm-driven book on randomness in algorithm design was written by Motwani and Raghavan [MR95]; Mitzenmacher and Upfal [MU17] provide an introduction to probability theory for computer scientists and Goldreich [Gol17, Chapter 2] focuses on the most important tools for property testing. An introduction to graph theory was written by Diestel [Die17].

1.3.1. Graphs

This section covers notation and all relevant definitions and results that exceed those covered by a typical introductory course on graph theory, e. g., the excellent book by Diestel [Die17].

Let $G = (V, E)$ be a graph. Unless stated otherwise, all graphs are simple and undirected. If there is no ambiguity, we use n to denote $|V|$ and m to denote $|E|$. Sometimes we refer to the vertices of G as $V(G)$ and to the edges of G as $E(G)$. The *size of a graph* is n. We require that there is a total order on the vertices, and without loss of generality, we assume that $V = [n]$. For the sake of clarification, we may add a subscript to predicates and operators that indicates the graph. If two graphs G and H are isomorphic, we write $G \cong H$. $\cong$

Operators on Vertices. Let $u, v \in V$ be vertices. The neighbors (adjacent vertices) of v are denoted $\Gamma(v) = \{u \mid \{v, u\} \in E\}$. The degree (number of neighbors) of v is denoted $d(v)$. The length of the shortest path between u and v is denoted $\mathrm{dt}(u, v)$. $\Gamma(\bullet), d(\bullet), \mathrm{dt}(\bullet, \bullet)$

Operators on Sets of Vertices and Edges. Let $S, T \subseteq V$ be sets of vertices. The complement of S with respect to the vertex set of the graph at hand is denoted $\bar{S} \coloneqq V \setminus S$. We write $S \times T$ to denote $\{\{u, v\} \mid u \in S \wedge v \in T\}$ in undirected graphs. The symmetric difference of S and T is $S \oplus T$. We let $G[S]$ denote the subgraph induced by S. The *girth* of G (length of the shortest cycle) is denoted $\mathrm{girth}(G)$. The cut between S and T is denoted by $E(S, T) \coloneqq E \cap (S \times T) = \{\{u, v\} \mid u \in \{u, v\} \in E \wedge u \in S \wedge v \in T\}$. $G[\bullet], \bar{\bullet}, \mathrm{girth}(\bullet)$ $E(\bullet, \bullet)$

Classes of Graphs The class of all graphs is called *general graphs*, and graphs that have a constant maximum degree d, i. e., for all vertices v it holds that $d(v) \leq d$, are called *bounded-*

degree graphs. We use the following notation for common graphs: the clique of size k is denoted K_k, the biclique on k, ℓ vertices is denoted $K_{k,\ell}$ and a cycle on k vertices is denoted C_k.

Hyperfinite Graphs (Chapters 2 and 3). A graph G is planar if it can be embedded into the Euclidean plane without crossing edges, and it is outerplanar if it is planar and there exists a planar embedding such that all vertices are incident to the outer face. A graph G is $\mathcal{F}$-minor free for a family of forbidden minors $\mathcal{F}$ if G does not contain H as a minor for any $H \in \mathcal{F}$. By Kuratowski's [Kur30] or Wagner's [Wag37] theorem, a graph is planar if and only if it is $\{K_5, K_{3,3}\}$-minor free, and it is outerplanar if it is $\{K_4, K_{2,3}\}$-minor free [CH67]. All minor-free graphs can be decomposed into connected components of constant size by removing a constant fraction of their vertices. This follows from a result by Alon, Seymour, and Thomas [AST90] that generalizes the Lipton-Tarjan planar separator theorem [LT79]. In bounded-degree graphs, this is also true with regard to the removal of a constant fraction of edges. All graphs that have this property, including graphs that are not minor-free, are said to be hyperfinite.

$\rho(\bullet)$

1.1 Definition (hyperfinite graphs [Ele07b]). Let $\epsilon \in (0, 1]$ and $k \geq 1$. A graph G with maximum degree bounded by d is said to be (ϵ, k)-*hyperfinite* if one can remove at most ϵdn edges from G so that each connected component of the resulting graph has at most k vertices. For a function $\rho : \mathbb{R}^+ \to \mathbb{N}^+$, a graph G is said to be ρ-*hyperfinite* if G is $(\epsilon, \rho(\epsilon))$-hyperfinite for every $\epsilon \in (0, 1]$. A set Π of graphs is called ρ-hyperfinite if every graph in Π is ρ-hyperfinite. ■

Since every finite set Π of graphs is trivially ρ-hyperfinite for $\rho \equiv \max_{(V,E)\in\Pi}|V|$, it is mostly interesting to consider infinite sets. We assume that ρ is a monotonically decreasing function, and for most infinite sets of hyperfinite graphs we would expect that $\lim_{\epsilon\to 0} \rho(\epsilon) = \infty$. If there exists any ρ such that Π is ρ-hyperfinite, we say that Π is hyperfinite.

1.2 Lemma [AST90]. For every family $\mathcal{F}$ of forbidden minors, there exists a function $\rho : (0, 1] \to \mathbb{N}$ such that every $\mathcal{F}$-minor free bounded-degree graph is ρ-hyperfinite. ■

$\mathrm{cond}(\bullet), \Phi(\bullet)$

Expansion and Conductance (Chapters 3 and 6). Let $G = (V, E)$ be a graph, and let $S \subseteq V$. The *volume* of S is the sum of degrees of vertices in S, i. e., $\mathrm{vol}(S) := \sum_{v\in S} d(v)$. For a set $S \subseteq V$ such that $\mathrm{vol}(S) \leq \mathrm{vol}(V \setminus S)$, the *conductance* of S is $\mathrm{cond}(S) = |E(S, \bar{S})|/\mathrm{vol}(S)$. The conductance of G is defined as $\Phi(G) = \min_{S\subset V, \mathrm{vol}(S)\leq\mathrm{vol}(\bar{S})} \mathrm{cond}(S)$. If G is a bounded-degree graph, one usually uses a the more convenient definition $\mathrm{cond}(S) = |E(S, \bar{S})|/(d|S|)$, which approximates the conductance up to a factor of $1/d$ if $\mathrm{vol}(S) \geq 1$. In this case, we also refer to conductance as *expansion*, which is defined as $\Phi(G) := \min_{S\subset V, |S|\leq|V|/2} \phi_G(S)$. We call G a ϕ-*conductor (graph)* or ϕ-*expander (graph)*, respectively, if $\Phi(G) \geq \phi$.

1.3.2. k-Disks

In essence, a k-disk of a vertex is a maximal rooted subgraph of radius k. The k-disks of a graph capture the neighborhoods of its vertices.

$\mathrm{disk}_\bullet(\bullet, \bullet), \cong$

1.3 Definition k-disk. For any $v \in V$, the k-*disk* of v (also known as k-hop neighborhood), denoted by $\mathrm{disk}_k(G, v)$, is defined as the subgraph that is induced by the vertices that are at distance at most k to v and is rooted at v. Two k-disks are isomorphic if and only if there exists a *root-preserving graph isomorphism* between them, i. e., a graph isomorphism that identifies the roots. ■

For any two k-disks Γ' and Γ'', we write $\Gamma' \cong \Gamma''$ if Γ' is isomorphic to Γ'', and write $\Gamma' \ncong \Gamma''$ otherwise. Let R_k be the set of all rooted graphs with radius at most k. The set of equivalence classes of R_k under isomorphism, i. e., the quotient space, is $\mathscr{T}_k := R_k/\cong$. The size of $\mathscr{T}_k$ is denoted D_k. $\mathscr{T}_\cdot, D_\cdot$

1.4 Lemma. The size of a k-disk $\Gamma \in \mathscr{T}_k$ is at most $2d^k$, and $D_k \leq 2^{4d^{2k}}$. ■

Proof. Since G has bounded degree d, the size of each of its k-disks $\Gamma_i \in \mathscr{T}_k$ is bounded by $1 + d + \ldots + d^k = \sum_{i=0}^{k} d^i \leq 2d^k$. The number of edges in a k-disk is at most $4d^{2k}$. We may assume that the vertices are ordered (e. g., the root is the first vertex in this order), and therefore we have $D_k \leq 2^{4d^{2k}}$. □

The *k-disk count vector* $\mathrm{cnt}_k(G)$ of a graph G is a D_k-dimensional vector where the i-th entry counts the number of k-disks in G that are isomorphic to $\Gamma_i \in \mathscr{T}_k$. By Lemma 1.4, D_k is finite. Note that the total number of k-disks in G is exactly $|V(G)|$. Given a k-disk isomorphism type Γ, $\mathrm{cnt}_k(G)_\Gamma$ is defined as the entry in $\mathrm{cnt}_k(G)$ that corresponds to Γ. Given a subset of vertices $S \subseteq V$, let $\mathrm{cnt}_k(S \mid G)$ be the k-disk count vector such that the ith entry counts the number of k-disks of G with root vertex in S that are isomorphic to Γ_i. $\mathrm{cnt}_\cdot(\bullet)$

The *k-disk frequency vector* of G, denoted by $\mathrm{freq}_k(G)$, is the vector where the i-th entry counts the fraction of k-disks in G that are isomorphic to $\Gamma_i \in \mathscr{T}_k$, or equivalently, $\mathrm{freq}_k(G) := \mathrm{cnt}_k(G)/|V(G)|$. We define $\mathrm{freq}_k(S \mid G) := \mathrm{cnt}_k(S \mid G)/|S|$ and $\mathrm{freq}_k(G)_\Gamma := \mathrm{cnt}_k(G)_\Gamma/|V(G)|$ similarly. $\mathrm{freq}_\cdot(\bullet)$

In bounded-degree graphs, inserting or removing an edge changes only a constant number of k-disks.

1.5 Lemma. Let $G = (V, E)$ be a bounded-degree graph and let G' be a graph that is obtained from G by removing an edge $\{u, v\} \in E$. Then it holds that $\|\mathrm{freq}_k(G) - \mathrm{freq}_k(G')\|_1 \leq 4d^k/n$. ■

Proof. First note that the number of vertices whose k-disks may be altered by removing a single edge is the number of k-disks that contain this edge. The number of such k-disks is exactly the number of vertices w such that there exists a path of length at most k from w to u and a path of length at most k from w to v. For an upper bound, it suffices to bound the number of paths to one vertex, say u. This is exactly the size of $\mathrm{disk}_k(G, w)$, which is at most $2d^k$ by Lemma 1.4. Finally, since a vertex that has different k-disks in G and G' can contribute at most $2/n$ to the ℓ_1-norm distance of $\mathrm{freq}_k(G)$ and $\mathrm{freq}_k(G')$, the claim follows. □

The following result by Alon states that for every graph G, there exists a constant-size graph H that approximates the k-disk frequency vector of G.

1.6 Lemma [Alo11]. Let $d, \delta, k > 0$. There exists $M_{d,\delta,k}$ such that for every bounded-degree graph G, there exists a graph H of size at most $M_{d,\delta,k}$ and $\|\mathrm{freq}_k(G) - \mathrm{freq}_k(H)\|_1 < \delta$. ■

In Chapter 2, we discuss this result in more detail, recite an existential proof (see Lemma 2.1) and derive upper bounds on $M_{d,\delta,k}$ for graphs with large girth. It is also an essential ingredient of Theorem 3.2 in Chapter 3. Therefore, it is useful to coin a name for H.

1.7 Definition ((δ, k)-DFP). We call the small graph H obtained from Lemma 1.6 a *(δ, k)-disk frequency preserver* (abbreviated as *(δ, k)-DFP*) of G. ■

1.3.3. Property Testing

This section introduces the concept of property testing and is tailored to sparse graphs. For a more rigorous introduction to property testing see, e. g., the recent book by Goldreich [Gol17].

Graph Models and Properties. Given a universe $\mathcal{U} = \bigcup_{n\in\mathbb{N}} \mathcal{U}_n$ of objects, a property Π of $\mathcal{U}$ is a subset of $\mathcal{U}$. Objects in $\mathcal{U}$ are partitioned into sets of comparable objects $(\mathcal{U}_n)_{n\in\mathbb{N}}$. For each set $\mathcal{U}_n$, objects are represented by functions $f : D_n \to R_n$. In this work, we consider graph property testing, so the universe $\mathcal{U}$ is the set of all graphs or a subset of this set, and $\mathcal{U}_n$ contains all graphs of size n from $\mathcal{U}$. The definition of D_n and R_n depend on the graph model. Roughly speaking, D_n represents elements we may ask questions about (e. g., entries of the adjacency matrix) and R_n represents the answers to these queries (e. g., in the case of adjacency matrices 0 and 1).

$\mathcal{U}_\cdot, \Pi_\cdot, \overline{\Pi}_\cdot$

1.8 Definition (graph model, property). A graph model is a tuple

$$\left(\mathcal{U} = \bigcup_{n\in\mathbb{N}} \mathcal{U}_n, (\delta_n)_{n\in\mathbb{N}}, (D_n)_{n\in\mathbb{N}}, (R_n)_{n\in\mathbb{N}}\right),$$

where $\mathcal{U}_n$ is a subset of the set of all graphs with n vertices, δ_n is a metric on $\mathcal{U}_n$, D_n is a model-specific domain set and R_n is a model-specific codomain set. Graphs in $\mathcal{U}_n$ are represented by functions $f_G : D_n \to R_n$. A property $\Pi = \bigcup_{n\in\mathbb{N}} \Pi_n$ of $\mathcal{U}$ is a subset of $\mathcal{U}$, where $\Pi_n \subseteq \mathcal{U}_n$. In this work (and in most other works on property testing), we assume that Π is invariant under isomorphism, i. e., for every pair of isomorphic graphs G and H it holds that $G \in \Pi \Leftrightarrow H \in \Pi$. The complement of Π is $\overline{\Pi} = \mathcal{U} \setminus \Pi$, and $\overline{\Pi}_n$ contains all graphs in $\overline{\Pi}$ of size n. A subset $\Pi' \subseteq \Pi$ is called a *subproperty* of Π if Π' is a property. ■

In general, a graph property tester for some property Π is a randomized algorithm that has query access to a function f_G representing an input graph $G = (V, E)$, the size n of G, a proximity parameter ϵ and possible other model-specific information as input. With constant error probability over the random coins of the algorithm, it outputs *accept* if $G \in \Pi$ and it outputs *reject* if more than roughly $\epsilon(|V| + |E|)$ values of f_G need to be changed in order to obtain a graph $G' \in \mathcal{U}$. In the latter case, G is said to be *ϵ-far* from Π, otherwise it is *ϵ-close*. The term $|V| + |E|$ should be thought of as the *combinatorial* size of the graph, and its precise quantity depends on the graph model.

$\overline{\Pi}_{n;>\epsilon}, \overline{\Pi}_{>\epsilon}$

1.9 Definition (ϵ-close, ϵ-far). Given a metric δ on $\mathcal{U}$, two graphs $G, H \in \mathcal{U}$ are ϵ-far if $\delta(G, H) > \epsilon$, and they are ϵ-close otherwise. Given a subset $\Pi \subseteq \mathcal{U}$, G is ϵ-far from Π if $\delta(G, \Pi) = \min_{H\in\Pi} \delta(G, H) > \epsilon$, and it is ϵ-close otherwise. Let $\overline{\Pi}_{n;>\epsilon}$ denote the set of all graphs of size n that are ϵ-far from Π_n. Let $\overline{\Pi}_{>\epsilon} \subseteq \overline{\Pi}$ be the set of all graphs that are ϵ-far from Π, i. e., $\overline{\Pi}_{>\epsilon} = \bigcup_{n\geq 1} \overline{\Pi}_{n;>\epsilon}$. ■

The complexity of a property tester is the number of function evaluations of f_G. In other words, a property tester is a probabilistic oracle machine that queries f_G, and the number of queries is its complexity. If the tester may fail for all graphs, it is a two-sided error tester. If it always answers correctly for graphs in Π, it is a one-sided error tester.

1.10 Definition (property tester). Let Π be a graph property. A property tester for Π in graph model $(\mathcal{U}, (\delta_n)_n, (D_n)_n, (R_n)_n)$ is a sequence of probabilistic oracle machines, denoted $\mathcal{T} = (\mathcal{T}_n)_{n\in\mathbb{N}}$. For every $n \in \mathbb{N}$, there exists $t_n \in \mathbb{N}$ such that the running time of $\mathcal{T}_n$ on any input

graph of size n is at most t_n. Given an input graph of size n, the tester $\mathscr{T}_n$ is selected and gets input parameter ϵ and access to G through query access to a function $f_G : D_n \to R_n$ according to the graph model. Its output is a binary decision, which satisfies that

1. if $G \in \Pi$, then it accepts with probability at least $2/3$
2. if $\delta(G, \Pi) > \epsilon$, then it rejects with probability at least $2/3$.

If $\mathscr{T}$ always accepts graphs in Π, $\mathscr{T}$ has *one-sided error*, otherwise it has *two-sided error*. If $\mathscr{T}_n$ is the same algorithm as $\mathscr{T}_{n'}$ for every $n, n' \in \mathbb{N}$, we say that $\mathscr{T}$ is *uniform*, otherwise it is *non-uniform*. To simplify notation, we refer to $\mathscr{T}_n$ as $\mathscr{T}$ when n is defined by the graph at hand. ■

A uniform tester usually gets access to the size of the input graph as an additional input parameter (or through f_G). One reason to consider non-uniform testers is that one can reduce decision problems to graph properties by, e. g., including a graph G in the property if and only if the bit representation of n is in the language of the problem. While most of these properties are pathological, it would complicate the analysis of reductions between different models of property testing (e. g., those presented in Chapter 4; see the discussion in Section 4.1.1).

Since the key idea of property testing is to minimize the access to the input, the computational complexity is often measured in terms of the query complexity, i. e., the number of queries to the function that represents the graph. We note that the error probability of $1/3$ in Definition 1.10 is arbitrary but conventional and can be decreased to any constant 2^{-k} by taking the majority's answer of $\mathcal{O}(k)$ repetitions (cf. *probability amplification* in, e. g., [MR95]), which increases the query complexity of the original tester by a factor of $\mathcal{O}(k)$.

$q_\bullet, q_{\mathscr{T}}(\bullet), q_\Pi(\bullet)$

1.11 Definition (query complexity). The query complexity $q_{\mathscr{T}}$ of a tester $\mathscr{T}$ for a graph property Π is the maximum number of values of f_G that $\mathscr{T}$ queries. The value of $q_{\mathscr{T}}$ may be parameterized by definitions of the testing problem (e. g., the distance parameter ϵ), by parameters of the model (e. g., a general bound on the vertex degrees) or by properties of the input (e. g., n). The optimal query complexity of a property Π, i. e., the minimum query complexity of all testers for Π, is denoted q_Π and is parameterized by the relevant variables, e. g., $q_\Pi(\epsilon)$. ■

Ideally, the query complexity of a property tester does not depend on the input at all. In particular, this means that the query complexity is independent of the size of the input. Of course, this is not always possible, but interestingly, some properties are constant-query testable in some graph models, while they are not constant-query testable in others.

1.12 Definition (constant-query testable). We call a property Π *constant-query testable* if there exists a tester $\mathscr{T}$ for Π such that $q_{\mathscr{T}}$ is independent of the input. ■

The first graph model that was used in property testing is the *dense graph model*. The key assumption is that the input graph has $\Omega(n^2)$ edges. The graph is accessed through queries to its adjacency matrix and the distance of two graphs is measured in terms of the fraction of adjacency matrix entries that have to be changed, which makes this model suitable for graphs with $\Omega(n^2)$ edges only: For any $\epsilon, \delta > 0$, there exists an $n \in \mathbb{N}$ such that any two graphs of size at least n that have at most $n^{2-\delta}$ edges are ϵ-close, rendering the notion of distance in this model meaningless. A classic example of a property that is constant-query testable in the dense graph model is bipartiteness [GGR96; AK02].

1.13 Definition (dense model [GGR96]). In the dense model, the universe $\mathscr{U}$ is the set of all graphs. Let $G = (V, E)$ be a graph. Without loss of generality, assume that $V = [n]$.

The representation domain is the set $D_n = [n] \times [n]$, and the representation codomain is the set $R_n = \{0, 1\}$. Then, f_G maps $\{u, v\}$ to 1 if $\{u, v\} \in E$ and to 0 otherwise for every $\{u, v\} \in V \times V$. The distance of two graphs G and H is defined as

$$\delta(G, H) := \frac{|E_G \oplus E_H|}{n^2}.$$ ■

d

Many graphs from applications are not dense but rather sparse, which led researchers to consider another graph model. In the bounded-degree model, we consider only graphs with maximum vertex degree d for a constant parameter $d \in \mathbb{N}$. Since the adjacency matrix of bounded-degree graphs is very sparse, an algorithm can access the graph by querying its adjacency lists instead. As d is constant, the distance between two graphs is simply measured as the number of edges that need to be inserted or deleted, normalized by nd, which is an upper bound on the total number of edges. Bipartiteness is not constant-query testable in bounded-degree graphs [GR97], some other properties like connectivity are (see Section 3.1.2).

1.14 Definition (bounded-degree model [GR97]). Let $d \in \mathbb{N}$. In the bounded-degree model, the universe $\mathscr{U}$ is the set of all graphs whose maximum vertex degree is bounded by d. Let $G = (V, E)$ be a graph with bounded degree d. Without loss of generality, assume that $V = [n]$.

The representation domain is the set $D_n = \{(i, j) \mid i \in [n] \wedge j \in [d]\}$, and the representation codomain is the set $R_n = [n] \cup \{\star\}$. Then, f_G maps $(v, 1), \dots, (v, d(v))$ bijectively to the neighbors of v and $f_G(v, d(v) + 1), \dots, f_G(v, d) = \star$ for every $v \in V$. The distance of two graphs G and H is defined as

$$\delta(G, H) := \frac{|E_G \oplus E_H|}{nd}.$$ ■

To avoid redundancy, we write *bounded-degree graph* instead of *graph with bounded degree d*. The (constant) bound on the degree often simplifies the design of sublinear property testers and their analysis. However, the bound is also rather artificial and does not reflect graphs from applications that are often sparse, but may contain some vertices with quite large, say linear, degree. For example, planar graphs have a bounded *average* degree, but do not have a bounded degree in general. The so far most versatile model is the general model.

1.15 Definition (general model [PR99]). In the general model, the universe $\mathscr{U}$ is the set of all graphs. The model provides redundant access to the graph by access to adjacency lists and the adjacency matrix. For the sake of simplicity, we express this by two different functions. Let $G = (V, E)$ be a graph. Without loss of generality, assume that $V = [n]$.

For the adjacency list access, the representation domain is the set $D_{L,n} = \{(i, j) \mid i \in [n] \wedge j \in [n]\}$, and the representation codomain is the set $R_{L,n} = [n] \cup \{\star\}$. Then, $f_{L,G}$ maps $(v, 1), \dots, (v, d(v))$ bijectively to the neighbors of v and $f_{L,G}(v, d(v) + 1, \dots, f_{L,G}(v, n) = \star$ for every $v \in V$.

For the adjacency matrix access, the representation domain is the set $D_{M,n} = [n] \times [n]$, and the representation codomain is the set $R_{M,n} = \{0, 1\}$. Then, $f_{M,G}$ maps (u, v) to 1 if $\{u, v\} \in E$ and to 0 otherwise for every $\{u, v\} \in V \times V$.

The distance of two graphs G and H is defined as

$$\delta(G, H) := \frac{|E_G \oplus E_H|}{n + \max(|E_G|, |E_H|)}.$$ ■

1.3.4. Notation and Bounds

If a parameterized concept is parameterized by a universal constant, we may omit the parameter if only its existence is of importance within the current context. For example, if a graph has bounded degree d for some universal constant d, we may just write that it is a bounded-degree graph.

All asymptotic notation may depend implicitly on parameters of the problem at hand that are not given within the asymptotic term, e. g., if $f \in \mathcal{O}(n)$, then it may be that $f = f(n, \epsilon) = n/\epsilon$. If a function does not depend on some parameter and we want to make this explicit, it is printed as a subscript of the asymptotic symbol, e. g., $f' \in \mathcal{O}_\epsilon(n)$. A tilde indicates that polylogarithmic factors have been omitted, e. g., $\tilde{\mathcal{O}}(n) = \mathcal{O}(n \cdot \text{poly}(\log n))$.

The following bounds are commonly used in computer science.

1.16 Lemma (Jensen's inequality [Jen06], cf. [MU17]). For every convex function f, it holds that

$$E(f(X)) \geq f(E([)X]) .$$ ■

1.17 Lemma (Markov's inequality, cf. [MU17]). Let X be a non-negative random variable. For every $a > 0$, it holds that

$$\Pr[X \geq a] \leq \frac{E(X)}{a} .$$ ■

1.18 Lemma (Chernoff bound [Che52], due to Rubin [Bat96], cf. [MU17]). Let $X_1, \dots, X_n$ be independent Poisson trials and let $p_i := \Pr[X_i = 1]$ for all $i \in [n]$. Let $X = \sum_{i=1}^{n} X_i$ and $\mu = E(X)$. Then, for all $\delta > 0$,

$$\Pr[X \geq (1+\delta)\mu] \leq \left(\frac{e^\delta}{(1+\delta)^{1+\delta}} \right) .$$ ■

1.19 Lemma (Hoeffding's inequality [Hoe63], cf. [MU17]). Let $X_1, \dots, X_n$ be independent random variables such that for all $i \in [n]$, $E(X_i) = \mu$ and $\Pr(a \leq X_i \leq b) = 1$. Then, for all $\delta > 0$,

$$\Pr\left[\left| \frac{1}{n} \sum_{i=1}^{n} (X_i - \mu) \right| \geq \delta \right] \leq 2 \exp\left(\frac{-2n\delta^2}{(b-a)^2} \right) .$$ ■

2. Small Graphs That Preserve the Local Neighborhood of Large Graphs

The local neighborhood of a vertex – the k-disk – is a very intuitive concept of locality in a graph. The histogram of all local neighborhoods captures a significant part of the local structure of a graph. Interestingly, there is no local structure of this kind that is exclusive to arbitrarily large graphs, i. e., it cannot be observed in graphs of constant size. In this chapter, we develop an upper bound on the size of the smallest graph that corresponds to a given local structure for graphs without short cycles. The results presented in this chapter are joint work with Pan Peng and Christian Sohler [FPS15].

2.1. Introduction

Consider a sublinear query algorithm for graphs that has constant query complexity. An immediate consequence of the complexity is that for every vertex v that the algorithm queries, it can only see a subset of the neighborhood of v that has constant size. In fact, one can show that a property tester with constant query complexity is actually equivalent to a *canonical tester* that approximates the k-disk distribution of a graph for some constant k and decides solely based on the outcome of this approximation (see Chapter 3). Particularly, k depends on the query complexity of the original tester, and the canonical tester might have larger (but constant) query complexity.

This motivates us to take a closer look at k-disk distributions and the graphs they represent. A natural question is whether there exists a graph for every distribution on k-disks, i. e., a normalized vector from $[0, 1]^{D_k}$. It is easy to understand that this is not true: there is no graph with only one vertex whose k-disk is, say, a d-ary tree, and all other vertices' k-disks are isolated vertices. However, this track leads us to more involved questions.

For example, consider the graph that we obtain from a street map by turning all intersections into vertices and all streets into edges. In the grid-like plans of places like Manhattan, Philadelphia or Portland, the local neighborhood – the k-disk – of almost every intersection looks like a part of a grid. Although these cities are usually quite big, the regularity of grids suggests that this layout could also be applied to small cities (as often, this example from reality cannot capture the asymptotic meaning of *not constant* but only some idea of *big*). On the other hand, the historical development of a city like Hamburg has lead to many local neighborhoods that are relatively unique. One might wonder whether we can squeeze the street layout of Hamburg into a smaller city while preserving the distribution of local neighborhoods, as we can do it with the grid plans. On a higher level, this leads us to the following question: does every k-disk distribution have a *local tangle factor* that can make it arbitrarily hard to construct small graphs with this distribution?

The answers is no. In fact, one can show that for every ϵ, there exists a number M that depends only on ϵ (and k and d) such that for every graph G, there exists a small graph H of

size M whose k-disk distribution differs only by ϵ in total variation (or ℓ_1) distance.

2.1.1. The Formal Problem

Let $G = (V, E)$ be a bounded-degree graph and let $k \in \mathbb{N}$. The preceding discussion might be formalized by the following question: What is the size of the smallest graph H such that $\mathrm{freq}_k(G) = \mathrm{freq}_k(H)$? It turns out that the answer to this particular question does not reflect our intention: it is $|V|$. To see why, assume that G contains only one vertex of degree 2. Therefore, there exists a k-disk Δ such that $\mathrm{freq}_k(G)_\Delta = 1/n$, which can only be the case for graphs of size at least n. The problem becomes much more interesting when we choose some $\epsilon > 0$ and relax the question: What is the size of the smallest graph H such that $\|\mathrm{freq}_k(G) - \mathrm{freq}_k(H)\|_1 < \epsilon$? One might guess that this relaxation does not change the answer much. However, Alon [Alo11] has proved that there exists a bound that depends on d, ϵ and k only.

2.1 Lemma (Lemma 1.6; [Alo11]). Let $d, \epsilon, k > 0$. For every bounded-degree graph G, there exists a graph H whose size only depends on d, ϵ and k such that $\|\mathrm{freq}_k(G) - \mathrm{freq}_k(H)\|_1 < \epsilon$. ■

Proof [Lov12, Proposition 19.10]. We construct an $\epsilon/(2D_k)$-net $N := \{i\epsilon/(2D_k) \mid i \in 0, \dots, 2D_k/\epsilon\}^{D_k}$. Each $x \in N$ is assigned an arbitrary graph $H' := H(x)$ such that $\|\mathrm{freq}_k(H') - x\|_1 < \epsilon/2$ if such an H' exists and $H(x) := \star$ otherwise. Since N is finite, there exists $c := c(d, \epsilon, k)$ such that $\max_{x \in N} |H(x)| \leq c$ (where $|\star| := 0$). Let G be an arbitrary graph, and let $x := \arg\min_{x' \in N} \|\mathrm{freq}_k(G) - x'\|_1$. Since N is an $\epsilon/(2D_k)$-net, we have $\|\mathrm{freq}_k(G) - x\|_1 < \epsilon/2$. By the triangle inequality, $\|\mathrm{freq}_k(G) - \mathrm{freq}_k(H(x))\|_1 < \epsilon$. □

The purely existential nature of the proof of this lemma gives rise to the question for upper bounds on the size of H. In particular, we are interested in an explicit bound on $|H|$.

2.2 Problem [Alo11]. Let $\epsilon, k > 0$. Give an upper bound $M_{d,\epsilon,k}$ such that for all bounded-degree graphs G, there exists a graph H of size at most $M_{d,\epsilon,k}$ such that $\|\mathrm{freq}_k(G) - \mathrm{freq}_k(H)\|_1 < \epsilon$. ■

This problem, as general as stated above, is still open. Some partial progress has been made, though (see Section 2.1.4). In this chapter, we discuss results for graphs with girth at least $2k + 2$.

2.1.2. Results in This Chapter

We present two results in this chapter: a deterministic linear time construction, which mainly serves as a way to obtain an upper bound on $M_{d,\epsilon,k}$ for graphs with girth at least $2k + 2$, and a randomized constant-time construction with a slightly worse bound. The bound of the deterministic construction is as follows.

2.3 Theorem. Let $k \geq 1$ and $\epsilon \in (0, 1)$. Let $G = (V, E)$ be a bounded-degree graph and $\mathrm{girth}(G) \geq 2k + 2$. There is a deterministic algorithm that outputs a graph $H = (V', E')$ such that

$$\|\mathrm{freq}_k(G) - \mathrm{freq}_k(H)\|_1 \leq \epsilon \text{ and } |V'| \leq 45 \frac{d^{3k+2} D_k}{\epsilon}.$$

The algorithm has time and query complexity $\mathcal{O}(|V|)$. ■

Having established the bound on $|V'|$, we can obtain a constant-time construction for H quite easily. First, we approximate $\mathrm{freq}_k(G)$ by sampling vertices and determining their k-disks. Then, we enumerate all graphs of size at most $|V'|$ and output the graph that has the closest

k-disk frequency vector to G in ℓ_1-norm distance. However, the running time of this approach may be exponential in $|V'|$. If we relax the bound on $|V'|$ by a factor of ${D_k}^2/\epsilon$ instead, we can obtain a randomized construction of H with better running time.

2.4 Theorem. Let $k \geq 1$ and $\epsilon, \delta \in (0,1)$. Let $G = (V, E)$ be a bounded-degree graph of size $|V| \geq 10^7 d^{6k+4} {D_k}^6/(\epsilon^4\delta^3)$ and girth$(G) \geq 2k+2$. There is an algorithm that outputs, with probability $1-\delta$, a graph $H = (V', E')$ such that

$$\|\mathrm{freq}_k(G) - \mathrm{freq}_k(H)\|_1 \leq \epsilon \text{ and } |V'| \leq \frac{5000 d^{3k+2} {D_k}^3}{\epsilon^2\delta}.$$

The algorithm has query complexity at most $10^4 \cdot \frac{d^{4k+4}{D_k}^3}{\epsilon^2\delta}$ and time complexity $\mathcal{O}(\mathrm{poly}(|V'|) \cdot d^{2k})$. ■

2.1.3. Connection to Graph Limits

There is a compelling connection between Lemma 1.6 and results from the theory of graph limits. This subject of discrete mathematics deals with the convergence of sequences of graphs and their limit objects. Thinking of an infinite sequence of graphs, the first thing one may observe is that the size of these graphs must go to infinity if every graph appears at most once in this sequence. Therefore, the limit object of this sequence – if it converges – should be an *infinite* graph. We state a few definitions from the limit theory of bounded-degree graphs to make this intuition formally more precise.

Let (X, T) be a Polish space, and denote the Borel σ-algebra of X by $\mathscr{B}(X)$. A bounded-degree graph $G = (X, E)$ is called Borel graph if $E \in \mathscr{B}(X^2)$, i. e., E is Borel in X^2. For this exposition, one may think that $X = [0,1] \subset \mathbb{R}$ and roughly imagine that E is a binary relation $\sim$ on $[0,1]$ such that for every $x \in [0,1]$, the number of y such that $x \sim y$ is at most d.

2.5 Example. Let $G = (X, E)$, where $X = [0,1]$ is equipped with its Borel σ-algebra and for every $x \in X$, $(x/2, x) \in E$. This graph is a Borel graph and has bounded degree 2. It contains an uncountable number of countably infinite paths. ■

If X is equipped with a probability measure λ and it holds that for any two measurable sets $A, B \subseteq X$

$$\int_A |\Gamma(x) \cap B| \,\mathrm{d}\lambda(x) = \int_B |\Gamma(x) \cap A| \,\mathrm{d}\lambda(x),$$

then G is called a *graphing* (the integral is Lebesgue). In other words, the expected number of neighbors of A that lie in B is the same as the expected number of neighbors of B that lie in A.

2.6 Example. The Borel graph G from Example 2.5 equipped with $\lambda(X') = \int_{X'} 1 \,\mathrm{d}x$ is a graphing. The same graph equipped with $\lambda(X') = \int_{X' \cap [0,1/2)} 2 \,\mathrm{d}x$ is not give a graphing. ■

By providing a probability measure λ, the notion of k-disk frequency vectors generalizes to graphings. In particular, $\mathrm{freq}_k(\Delta \mid G)$ is the probability that the k-disk of a random vertex according to λ is isomorphic to Δ: $\mathrm{freq}_k(\Delta \mid G) = \int_X [\![\mathrm{disk}_k(G, x) \cong \Delta]\!] \,\mathrm{d}\lambda(x)$. Note that the number as well as the size of k-disks in graphings is finite because the vertex degree is bounded.

A sequence of bounded-degree graphs $(G_i)_{i \in \mathbb{N}}$ is *locally convergent* if $\mathrm{freq}_k(G \mid \Delta)$ converges for every k and every $\Delta \in \mathscr{T}_k$. One can show that the limit object of a locally convergent

sequence of bounded-degree graphs, its *local limit*, is a graphing.[1] However, it is not known if *every* graphing is the local limit of some locally convergent sequence of graphs. This was raised as an open problem by Aldous and Lyons [AL07] and it is "a central unsolved open problem in the limit theory of bounded-degree graphs"[2], which will establish several other conjectures if it is true[3].

2.7 Conjecture [AL07, Question 10.1]. Every graphing is the local limit of a locally convergent sequence of bounded-degree graphs. ■

Lemma 1.6 can be seen as a finite version of this conjecture. Conjecture 2.7, if it is true, would imply that for every graphing G and every $k \geq 0$, there is a finite graph G whose k-disk distribution is arbitrarily close to that of G. Lemma 1.6 implies that for every arbitrarily large but finite graph G and every $k \geq 0$, there exists a graph of constant size whose neighborhood is arbitrarily close. So, in a sense, *uncountable* (graphing) and *finite* (graph) are exchanged for *finite* (graph) and *constant-size* (graph) in the statement of Lemma 1.6.

2.1.4. More Related Work

In [Fic14], an upper bound on $M_{d,\epsilon,k}$ was given for the special case of hyperfinite graphs by the author. The bound is based on a geometric construction that employs Carathéodory's theorem [Car07; Ste09] on the set of k-disk vectors. It is also known that Conjecture 2.7 holds for some special cases. In particular, Lovász [Lov12, Theorem 21.14] proved that a graphing is hyperfinite if and only if it is the limit of a hyperfinite graph sequence, which implies that Conjecture 2.7 holds for hyperfinite graphings. Elek [Ele10] proved that if a graphing is concentrated on trees, it arises as the limit of a graph sequence with increasing girth. This result was generalized to so-called *treeable* graphings by Hosseini and Mendez [HM16].

Although Lemma 1.6 and Conjecture 2.7 are related, the analysis of upper bounds on $M_{d,\epsilon,k}$ for the known special cases is quite different from the proofs of [Ele10; Lov12], which rely on measure theory. On the other hand, the proofs presented in [Fic14], this chapter and [HM16] use combinatorial arguments.

2.1.5. Open Problems and Future Work

For completeness, recall that Problem 2.2 is still open for general graphs. We discuss other open problems related to Chapters 2 to 4 in Section 4.1.3 to provide a broader, unified view on them.

2.1.6. Overview of the Analysis

Our results are based on the following transformation of a graph G with girth $2k + 2$ that fully preserves the k-disk distribution of G: Let $\{u_1, v_2\}, \{u_2, v_1\} \in E$ be two edges with the properties that (i) the distance from u_1 to v_1 and the distance from v_2 to u_2 in G are large and (ii) the k-disks of u_1 and u_2 are isomorphic and (iii) the k-disks of v_1 and v_2 are isomorphic. Then one can replace the edges $\{u_1, v_2\}, \{u_2, v_1\}$ by $\{u_1, v_1\}, \{u_2, v_2\}$ without changing the k-disk frequency

[1] Benjamini and Schramm [BS01] provide a construction for suitable limit objects, and by results of Aldous and Lyons [AL07] and Elek [Ele07a] these can be turned into graphings (see also [Lov12, Theorem 18.37]).
[2] Lovász [Lov12, p. 357]
[3] Hosseini and Mendez [HM16]

vector of the graph. The reason is that if two instances share the same k-disk distribution, a property tester that has a low query complexity (or round complexity in the case of distributed testers) has the same view on both instances. It turns out that this local transformation is also useful in the context of lower bounds in property testing (see Section 6.4).

Our construction works as follows. We identify a subset $V_1 \subseteq V$ of constant size that has approximately the same distribution of k-disks (with respect to G) as $V_2 = V \setminus V_1$. Then we use our transformation to turn G into a graph G' where V_1 has a small cut (relative to the size of V_1) to V_2 and the k-disk distribution of G is preserved. The graph $G'[V_1]$ has constant size and a similar k-disk distribution as G, which gives us Theorem 2.3. Instead of carefully selecting vertices into V_1, we can pick vertices uniformly at random. Since the result is the same in expectation, we can obtain a constant-time algorithm with a slightly worse bound on $|V_1|$ this way, which gives us Theorem 2.4.

2.2. Preliminaries

In this chapter, all graphs are bounded-degree graphs unless stated otherwise. Let $U \subseteq V$, $k \in \mathbb{N}$ and let $\Gamma \in \mathscr{T}_k$. We denote the set of vertices in U whose k-disk is isomorphic to Γ by U^Γ, i. e., $U^\Gamma := \{v \in U \mid \mathrm{disk}_k(G, v) \cong \Gamma\}$. For any integer k and $\Gamma', \Gamma'' \in \mathscr{T}_k$, we call an edge $\{u, v\} \in E$ a (Γ', Γ'')-*edge* if $\mathrm{disk}_k(G, u) \cong \Gamma'$ and $\mathrm{disk}_k(G, v) \cong \Gamma''$.

$U^\bullet$, $(\bullet, \bullet)$-edge

In this section, we often consider both k-disks and $(k-1)$-disks after fixing k. For the sake of readability, we use Γ to denote k-disks and Δ to denote $(k-1)$-disks in these cases.

For any k-disk $\Gamma \in \mathscr{T}_k$ and $(k-1)$-disk $\Delta \in \mathscr{T}_{k-1}$, Γ is called Δ-*extensive* if the $(k-1)$-disk of the root of Γ is isomorphic to Δ. We denote the set of all Δ-extensive k-disks Γ by $\mathrm{ext}(\Delta)$. Given a k-disk $\Gamma \in \mathscr{T}_k$ with root r and a $(k-1)$-disk $\Delta \in \mathscr{T}_{k-1}$, let $\mathrm{sub}_\Gamma(\Delta)$ be the number of neighbors of r whose $(k-1)$-disk is isomorphic to Δ, i. e.,

Δ-extensive

$\mathrm{sub}_\Gamma(\bullet)$

$$\mathrm{sub}_\Gamma(\Delta) := \left|\{v \mid \{r, v\} \in E(\Gamma) \wedge \mathrm{disk}_{k-1}(\Gamma, v) \cong \Delta\}\right|.$$

For any $\Delta', \Delta'' \in \mathscr{T}_{k-1}$, let $\mathrm{sub}_{\Delta'}(\Delta'')$ be the total number of (Δ', Δ'')-edges starting at the root of any k-disk $\Gamma \in \mathrm{ext}(\Delta')$, i. e.,

$\mathrm{sub}_\Delta(\bullet)$

$$\mathrm{sub}_{\Delta'}(\Delta'') := \sum_{\Gamma \in \mathrm{ext}(\Delta')} \mathrm{sub}_\Gamma(\Delta'').$$

Note that $\mathrm{sub}_{\Delta'}(\Delta'')$ is not necessarily symmetric with respect to Δ' and Δ''. However, the following bound on the number of pairs of $(k-1)$-disks Δ', Δ'' whose roots can be adjacent is symmetric.

2.8 Lemma. It holds that $\sum_{\Delta', \Delta'' \in \mathscr{T}_{k-1}} \mathrm{sub}_{\Delta'}(\Delta'') \leq D_k d$. ■

Proof. Since a k-disk determines the $(k-1)$-disks of the root and of all of the root's at most d neighbors, we have

$$\sum_{\Delta', \Delta'' \in \mathscr{T}_{k-1}} \mathrm{sub}_{\Delta'}(\Delta'') = \sum_{\Delta', \Delta'' \in \mathscr{T}_{k-1}} \sum_{\Gamma \in \mathrm{ext}(\Delta')} \mathrm{sub}_\Gamma(\Delta'') \leq \sum_{\Gamma \in \mathscr{T}_k} d = D_k d. \qquad \square$$

2.3. Locality-Preserving Rewiring of Edge Cuts

We develop the main technical tool that we use to prove Theorems 2.3 and 2.4. In essence, we show that if a graph has girth at least $2k+2$, then the size of the cut between two sets V_1 and V_2 that have similar k-disk frequency vectors can be reduced to a small fraction of $\min(|V_1|, |V_2|)$.

2.3.1. Cuts Between Locally Similar Vertex Sets

We start by examining the cut edges of subsets of vertices. Our first lemma shows that the fraction of (Δ', Δ'')-edges that start in an arbitrary subset $V_1 \subseteq V$ is approximately the same as for another arbitrary subset $V_2 \subseteq V$ if the frequency distributions of the k-disks in V_1 and V_2 are close. Recall that V^Δ denotes the subset of vertices in V whose k-disks are isomorphic to Δ.

2.9 Lemma. Let $G = (V, E)$ be a graph, $k \geq 1$, $\lambda \in [0, 1]$ and let $V_1, V_2 \subseteq V$ be such that $|\mathrm{freq}_k(V_1 \mid G)_\Gamma - \mathrm{freq}_k(V_2 \mid G)_\Gamma| \leq \lambda$ for all k-disks $\Gamma \in \mathscr{T}_k$. Then, for all $(k-1)$-disks $\Delta', \Delta'' \in \mathscr{T}_{k-1}$ such that $\Delta' \not\cong \Delta''$, it holds that

$$\left| \frac{|E(V_1^{\Delta'}, V^{\Delta''})|}{|V_1|} - \frac{|E(V_2^{\Delta'}, V^{\Delta''})|}{|V_2|} \right| \leq \lambda \cdot \mathrm{sub}_{\Delta'}(\Delta'') \,. \qquad \blacksquare$$

Proof. Consider any Δ'-extensive k-disc $\Gamma \in \mathrm{ext}(\Delta')$. Then the number of (Δ', Δ'')-edges in G such that the root of Δ' belongs to V_1 equals

$$\left| E(V_1^{\Delta'}, V^{\Delta''}) \right| = \sum_{\Gamma \in \mathrm{ext}(\Delta')} \mathrm{cnt}_k(V_1 \mid G)_\Gamma \cdot \mathrm{sub}_\Gamma(\Delta'') \,.$$

An analogous equation holds for $|E(V_2^{\Delta'}, V^{\Delta''})|$. Note that since $\Delta' \not\cong \Delta''$, even edges that start and end in V_1 are counted only once in the right-hand side of the equation because $\mathrm{ext}(\Delta') \cap \mathrm{ext}(\Delta'') = \emptyset$. Therefore,

$$\begin{aligned}
&\left| \frac{|E(V_1^{\Delta'}, V^{\Delta''})|}{|V_1|} - \frac{|E(V_2^{\Delta'}, V^{\Delta''})|}{|V_2|} \right| \\
&= \left| \frac{\sum_{\Gamma \in \mathrm{ext}(\Delta')} \mathrm{cnt}_k(V_1 \mid G)_\Gamma \cdot \mathrm{sub}_\Gamma(\Delta'')}{|V_1|} - \frac{\sum_{\Gamma \in \mathrm{ext}(\Delta')} \mathrm{cnt}_k(V_2 \mid G)_\Gamma \cdot \mathrm{sub}_\Gamma(\Delta'')}{|V_2|} \right| \\
&\leq \sum_{\Gamma \in \mathrm{ext}(\Delta')} \left| \mathrm{freq}_k(V_1 \mid G)_\Gamma - \mathrm{freq}_k(V_2 \mid G)_\Gamma \right| \cdot \mathrm{sub}_\Gamma(\Delta'') \\
&\leq \lambda \cdot \sum_{\Gamma \in \mathrm{ext}(\Delta')} \mathrm{sub}_\Gamma(\Delta'') \\
&= \lambda \cdot \mathrm{sub}_{\Delta'}(\Delta'') \,.
\end{aligned}$$

□

If V_1, V_2 is a partitioning of V, the former result can be improved. In particular, we show that if $\mathrm{freq}_k(V_1 \mid G) \approx \mathrm{freq}_k(V_2 \mid G)$, then for almost every (Δ', Δ'')-edge from V_1 to V_2 there is a counterpart, i. e., a (Δ', Δ'')-edge from V_2 to V_1. We will later use this result to reduce the size of the cut without altering the k-disk frequency vector. In particular, we swap the endpoints of edges in the cut so that the new edges lie completely in V_1 and V_2, respectively.

2.10 Lemma. Let $G = (V, E)$ be a graph, $k \geq 1$, $\lambda \in [0, 1]$ and let $V_1 \uplus V_2 = V$ be a partitioning of V such that $|\mathrm{freq}_k(V_1 \mid G)_\Gamma - \mathrm{freq}_k(V_2 \mid G)_\Gamma| \leq \lambda$ for all k-disks $\Gamma \in \mathscr{T}_k$. Then, for all $(k-1)$-disks

$\Delta', \Delta'' \in \mathcal{T}_k k-1$, it holds that

$$\left||E(V_1^{\Delta'}, V_2^{\Delta''})| - |E(V_2^{\Delta'}, V_1^{\Delta''})|\right| \leq \frac{|V_1||V_2|}{|V|} \cdot \lambda \cdot \left[\operatorname{sub}_{\Delta'}(\Delta'') + \operatorname{sub}_{\Delta''}(\Delta')\right]. \qquad \blacksquare$$

Proof. If $\Delta' \cong \Delta''$, the bound holds trivially because $|E(V_1^{\Delta'}, V_2^{\Delta''})| = |E(V_2^{\Delta'}, V_1^{\Delta''})|$. Therefore, assume that $\Delta' \ncong \Delta''$ now. By symmetry it holds that

$$\left|E(V_i^{\Delta'}, V_j^{\Delta''})\right| = \left|E(V_j^{\Delta''}, V_i^{\Delta'})\right| \qquad \forall i, j \in \{1, 2\}. \tag{2.1}$$

Furthermore, since V_1, V_2 is a partitioning of V, we have

$$\left|E(V_i^{\Delta'}, V^{\Delta''})\right| = \left|E(V_i^{\Delta'}, V_1^{\Delta''})\right| + \left|E(V_i^{\Delta'}, V_2^{\Delta''})\right| \qquad \forall i \in \{1, 2\} \tag{2.2}$$

and an analogous equation for $|E(V_i^{\Delta''}, V^{\Delta'})|$. Now,

$$\begin{aligned}
&\left||E(V_1^{\Delta'}, V_2^{\Delta''})| - |E(V_2^{\Delta'}, V_1^{\Delta''})|\right| \\
&= \frac{|V_1||V_2|}{|V|} \cdot \frac{(|V_1| + |V_2|)}{|V_1||V_2|} \cdot \left||E(V_1^{\Delta'}, V_2^{\Delta''})| - |E(V_2^{\Delta'}, V_1^{\Delta''})|\right| \\
&= \frac{|V_1||V_2|}{|V|} \cdot \left|\left(\frac{1}{|V_1|} + \frac{1}{|V_2|}\right) \cdot \left(\left|E(V_1^{\Delta'}, V_2^{\Delta''})\right| - \left|E(V_1^{\Delta''}, V_2^{\Delta'})\right|\right) + 0 - 0\right| \\
&= \frac{|V_1||V_2|}{|V|} \cdot \left|\left(\frac{1}{|V_1|} + \frac{1}{|V_2|}\right) \cdot \left(\left|E(V_1^{\Delta'}, V_2^{\Delta''})\right| - \left|E(V_1^{\Delta''}, V_2^{\Delta'})\right|\right)\right. \\
&\quad \left. + \frac{|E(V_1^{\Delta'}, V_1^{\Delta''})| - |E(V_1^{\Delta''}, V_1^{\Delta'})|}{|V_1|} - \frac{|E(V_2^{\Delta'}, V_2^{\Delta''})| - |E(V_2^{\Delta''}, V_2^{\Delta'})|}{|V_2|}\right| \\
&\overset{(a)}{=} \frac{|V_1||V_2|}{|V|} \cdot \left|\frac{|E(V_1^{\Delta'}, V_1^{\Delta''})| + |E(V_1^{\Delta'}, V_2^{\Delta''})|}{|V_1|} - \frac{|E(V_2^{\Delta'}, V_2^{\Delta''})| + |E(V_2^{\Delta'}, V_1^{\Delta''})|}{|V_2|}\right. \\
&\quad \left. - \frac{|E(V_1^{\Delta''}, V_1^{\Delta'})| + |E(V_1^{\Delta''}, V_2^{\Delta'})|}{|V_1|} + \frac{|E(V_2^{\Delta''}, V_2^{\Delta'})| + |E(V_2^{\Delta''}, V_1^{\Delta'})|}{|V_2|}\right| \\
&\overset{(b)}{=} \frac{|V_1||V_2|}{|V|} \cdot \left|\frac{|E(V_1^{\Delta'}, V^{\Delta''})|}{|V_1|} - \frac{|E(V_2^{\Delta'}, V^{\Delta''})|}{|V_2|} - \frac{|E(V_1^{\Delta''}, V^{\Delta'})|}{|V_1|} + \frac{|E(V_2^{\Delta''}, V^{\Delta'})|}{|V_2|}\right| \\
&\overset{(c)}{\leq} \frac{|V_1||V_2|}{|V|} \cdot \lambda\left[\operatorname{sub}_{\Delta'}(\Delta'') + \operatorname{sub}_{\Delta''}(\Delta')\right],
\end{aligned}$$

where (a) follows from Eq. (2.1), (b) follows from Eq. (2.2) and (c) follows from applying Lemma 2.9. □

2.3.2. Rewiring of Edges

Lemma 2.10 enables us to analyze our main technical tool, i. e., the rewiring of edges. First, we will prove that under some condition we can rewire two (Δ', Δ'')-edges without altering the k-disk frequency vector of the graph or the partitions. This part of the proof shows that there exists, for every vertex $v \in V$, an isomorphism function that maps the k-disk of v in the original graph to the k-disk of v in the rewired graph. We then show that if we cannot find such

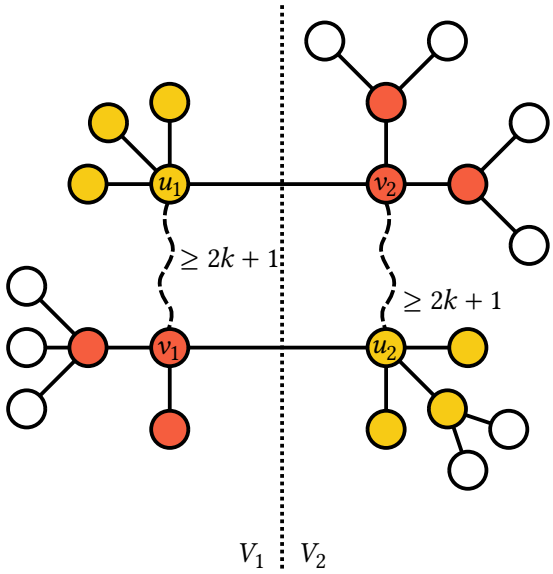


Figure 2.1.: If condition $(\star)$ is satisfied (as here for $k = 2$), it is possible to replace the edges $\{u_1, v_2\}$ and $\{u_2, v_1\}$ by $\{u_1, v_1\}$ and $\{u_2, v_2\}$, respectively, without changing the k-disk vector of G. Otherwise, the cut of V_1 and V_2 is small and removing these edges affects only few k-disks in V_1.

(Δ', Δ'')-edges, the cut of V_1 and V_2 is small. This implies that the removal of the remaining edges changes the k-disk frequency vector of the graph only slightly. Therefore, V_1 and V_2 are proper subsets of V, but their induced graphs have roughly the same k-disk frequency vector as the whole graph.

2.11 Proposition. Let $G = (V, E)$ be a graph with $\operatorname{girth}(G) \geq 2k + 2$, $k \geq 1$, $\lambda \in [0, 1]$ and let $V_1 \uplus V_2 = V$ be a partitioning of V such that $|\operatorname{freq}_k(V_1 \mid G)_\Gamma - \operatorname{freq}_k(V_2 \mid G)_\Gamma| \leq \lambda$ for all k-disks $\Gamma \in \mathscr{T}_k$. Then either there exists a graph $H = (V, F)$ such that

$$\operatorname{girth}(H) \geq 2k + 2 \tag{2.3}$$

$$|F \cap (V_1 \times V_2)| \leq |E \cap (V_1 \times V_2)| - 2 \tag{2.4}$$

$$\operatorname{disk}_k(H, w) \cong \operatorname{disk}_k(G, w) \qquad \forall w \in V \tag{2.5}$$

or the cut between V_1 and V_2 is small:

$$E(V_1, V_2) \leq 6d^{2k+2} D_k + 2\lambda D_k d \cdot \min(|V_1|, |V_2|)\,. \tag{2.6}$$

■

Proof. Consider the following condition (see Fig. 2.1):

$(\star)$ There exist $\{u_1, v_2\} \in (V_1 \times V_2) \cap E$ and $\{u_2, v_1\} \in (V_2 \times V_1) \cap E$ such that $\operatorname{dt}_G(u_1, v_1)$, $\operatorname{dt}_G(v_2, u_2) \geq 2k + 1$, $\operatorname{disk}_{k-1}(G, u_1) \cong \operatorname{disk}_{k-1}(G, u_2)$ and $\operatorname{disk}_{k-1}(G, v_2) \cong \operatorname{disk}_{k-1}(G, v_1)$.

Informally, it states that, for a suitable choice of $(k-1)$-disks Δ' and Δ'', there exists a (Δ', Δ'')-edge from V_1 to V_2 and a (Δ', Δ'')-edge from V_2 to V_1 such that two endpoints of different edges are not too close. We prove in the following that if condition $(\star)$ is satisfied,

then there exists a graph H with the desired properties (see Claim 2.12), and that Eq. (2.6) holds otherwise (see Claim 2.13).

2.12 Claim. If condition ($\star$) is satisfied, there exists a graph $H = (V, F)$ such that Ineq. (2.3) and (2.4) and Expr. (2.5) are satisfied. ■

2.13 Claim. If condition ($\star$) is not satisfied, then Eq. (2.6) holds. ■

We prove the two claims in the subsequent sections. The proposition follows directly. □

Proof of Claim 2.12

Proof. Suppose that condition ($\star$) is satisfied. We define an intermediate graph $G' := (V, E')$ that is obtained by deleting the edges $\{u_1, v_2\}$ and $\{u_2, v_1\}$ from G, i. e., $E' := E \setminus \{\{u_1, v_2\}, \{u_2, v_1\}\}$. We further define $H := (V, F)$ to be the graph that is obtained by adding the edges $\{u_1, v_1\}$ and $\{u_2, v_2\}$ to G', i. e., $F := E' \cup \{\{u_1, v_1\}, \{u_2, v_2\}\}$.

Observe that since $\mathrm{dt}_G(u_1, v_1), \mathrm{dt}_G(u_2, v_2) \geq 2k + 1$, Ineq. (2.3) holds, and by definition of H, Ineq. (2.4) holds. Thus, it remains to prove that Expr. (2.5) also holds, i. e., for any vertex w, the k-disks of w in G and H are isomorphic. In what follows, we carefully construct a root-preserving bijection $f : V(\mathrm{disk}_k(G, w)) \to V(\mathrm{disk}_k(H, w))$ such that for all $x, y \in V(\mathrm{disk}_k(G, w))$, $\{x, y\} \in E(\mathrm{disk}_k(G, w))$ if and only if $\{f(x), f(y)\} \in E(\mathrm{disk}_k(H, w))$ to formally prove this somewhat intuitive observation.

Let $w \in V$. We distinguish between the cases that (i) neither $\{u_1, v_2\}$ nor $\{u_2, v_1\}$, (ii) either one of them, or (iii) both are contained in $\mathrm{disk}_k(G, w)$. First, we will specify two isomorphism functions $g_u : V(\mathrm{disk}_k(G, u_1)) \to V(\mathrm{disk}_k(G, u_2))$ and $g_v : V(\mathrm{disk}_k(G, v_2)) \to V(\mathrm{disk}_k(G, v_1))$ for $\mathrm{disk}_k(G, u_1) \cong \mathrm{disk}_k(G, u_2)$ and $\mathrm{disk}_k(G, v_2) \cong \mathrm{disk}_k(G, v_1)$, respectively. If there is more than one candidate for g_u or g_v, we make an arbitrary choice unless stated otherwise. We will then define f using these two functions g_u and g_v and prove that f is an isomorphism between $\mathrm{disk}_k(G, w)$ and $\mathrm{disk}_k(H, w)$.

Case 1; $\{u_1, v_2\} \notin \mathrm{disk}_k(G, w)$, $\{u_2, v_1\} \notin \mathrm{disk}_k(G, w)$. In this case, we define $f(x) := x$ for all $x \in V(\mathrm{disk}_k(G, w))$. We claim that neither $\{u_1, v_1\}$ nor $\{v_2, u_2\}$ belongs to $E(\mathrm{disk}_k(H, w))$. Without loss of generality assume that $\{u_1, v_1\} \in E(\mathrm{disk}_k(H, w))$. Then $\mathrm{dt}_G(w, u_1), \mathrm{dt}_G(w, v_1) \leq k$, which implies that $\mathrm{dt}_G(u_1, v_1) \leq \mathrm{dt}_G(u_1, w) + \mathrm{dt}_G(w, v_1) \leq 2k$. This is a contradiction to the assumption that $\mathrm{dt}_G(u_1, v_1) \geq 2k + 1$. The same argument shows that $\{u_2, v_2\} \notin E(\mathrm{disk}_k(G, w))$. Therefore, the k-disks $\mathrm{disk}_k(G, w)$ and $\mathrm{disk}_k(H, w)$ do not contain any of the edges $\{u_1, v_2\}$, $\{u_2, v_1\}$, $\{u_1, v_1\}$, $\{u_2, v_2\}$ and thus $\mathrm{disk}_k(G, w) \cong \mathrm{disk}_k(H, w)$ by our definition of H.

Case 2.1; $\{u_1, v_2\} \in \mathrm{disk}_k(G, w)$, $\{u_2, v_1\} \notin \mathrm{disk}_k(G, w)$. In this case it holds that $u_2, v_1 \notin \mathrm{disk}_k(G, w)$, since otherwise, either $\mathrm{dt}_G(u_1, v_1) \leq 2k$ or $\mathrm{dt}_G(v_2, u_2) \leq 2k$, which contradicts condition ($\star$).

Now we observe that since $\mathrm{girth}(G) \geq 2k + 2$, the k-disk $\mathrm{disk}_k(G, w)$ is a tree. This implies that the deletion of the edge $\{u_1, v_2\}$ will partition $\mathrm{disk}_k(G, w)$ into two connected components, say P_{u_1} and P_{v_2}, which represent the set of vertices in $\mathrm{disk}_k(G, w)$ that are connected to u_1 after deleting $\{u_1, v_2\}$ and the set of remaining vertices that are connected to v_2, respectively. Without loss of generality assume that $w \in P_{u_1}$. The case that $w \in P_{v_2}$ can be analyzed similarly.

Let $f(x) := x$ if $x \in P_{u_1}$ and $f(x) := g_v(x)$ if $x \in P_{v_2}$. If there is more than one candidate for g_v, we make an arbitrary choice among all isomorphism functions that map P_{v_2} to (a subset of) $\mathrm{disk}_{k-1}(G', v_1)$, i. e., the $(k-1)$-disk of v_1 after deleting $\{u_2, v_1\}$ (see Fig. 2.2). Since $\mathrm{disk}_{k-1}(G, v_2) \cong \mathrm{disk}_{k-1}(G, v_1)$ by ($\star$), there is always an isomorphism function that satisfies this condition. Moreover, f is a bijection because g_v is a bijection, and the image of

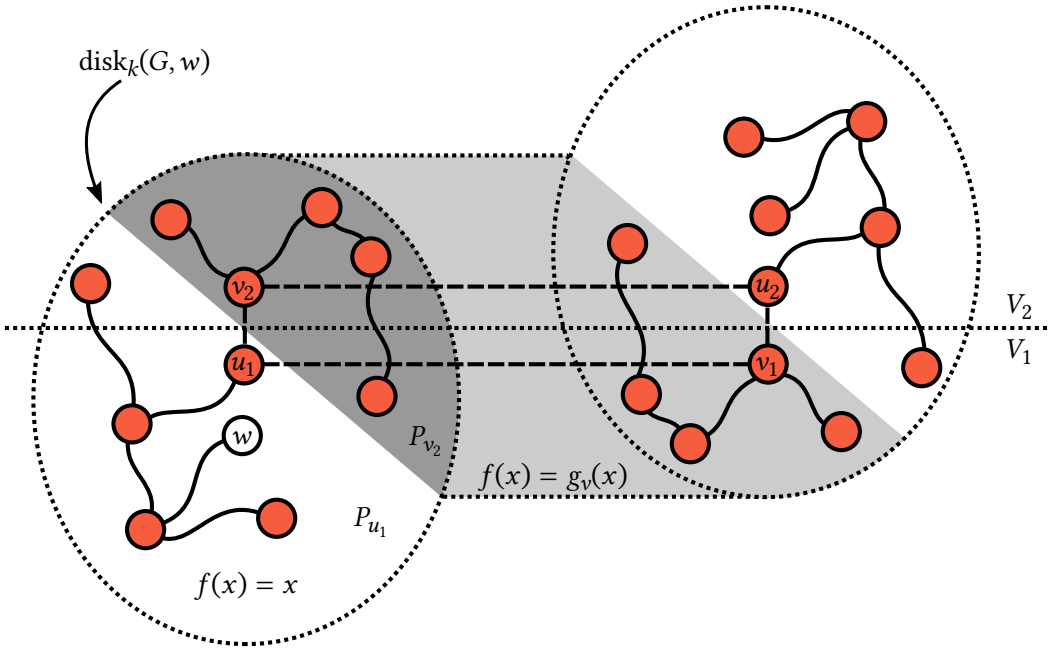


Figure 2.2.: The k-disk of w in G, disk$_k(G, w)$, is partitioned into two parts P_{u_1} (white background) and P_{v_2} (dark gray background) by the edge $\{u_1, v_2\}$. The dashed edges are only present in either G or H but not in G'. If $w \in P_{u_1}$ as in the figure, then $f(x) = x$ for any $x \in P_{u_1}$, and $f(x) = g_v(x)$ for any $x \in P_{v_2}$, where g_v is chosen such that the image of P_{v_2} under g_v is a subset of disk$_k(G', v_1)$ (light gray background).

$V(\text{disk}_k(G, w))$ under f is $V(\text{disk}_k(H, w))$ by the construction of H. We now prove that f is an isomorphism function between disk$_k(G, w)$ and disk$_k(H, w)$.

First note that $f(w) = w$, $f(u_1) = u_1$ and $f(v_2) = g_v(v_2) = v_1$. Now consider any $x, y \in V(\text{disk}_k(G, w))$. If $x, y \in P_{u_1}$ or $x, y \in P_{v_2}$, then $f(x) = x$, $f(y) = y$ or $f(x) = g_v(x)$, $f(y) = g_v(y)$, respectively. Therefore, $\{x, y\} \in E(G)$ if and only if $\{f(x), f(y)\} \in E(H)$.

Now consider the case that $x \in P_{u_1}$ and $y \in P_{v_2}$. If $x = u_1$ and $y = v_2$, then we know that $\{x, y\} \in E(G)$ and also that $\{f(x), f(y)\} = \{u_1, v_1\} \in E(H)$ by the definition of H. Otherwise, either $x \neq u_1$ or $y \neq v_2$. In this case, there is no edge $\{x, y\}$ in G since disk$_k(G, w)$ is a tree and x, y lie on different sides of the edge $\{u_1, v_2\}$. Recall that $f(u_1) = u_1$ and $f(v_2) = g_v(v_2) = v_1$. Since f is a bijection, either $f(x) \neq u_1$ or $f(y) \neq v_1$. Observe that disk$_k(H, w)$ is a tree by Ineq. (2.3). Hence there is no edge between $f(x)$ and $f(y)$ in H as they lie on different sides of the edge $\{u_1, v_1\}$. The case that $x \in P_{v_2}$ and $y \in P_{u_1}$ is symmetric.

Therefore, the function f is a root-preserving isomorphism function between disk$_k(G, w)$ and disk$_k(H, w)$.

Case 2.2; $\{u_1, v_2\} \notin \text{disk}_k(G, w)$, $\{u_2, v_1\} \in \text{disk}_k(G, w)$. This case can be analyzed similarly to the foregoing case.

Case 3; $\{u_1, v_2\} \in \text{disk}_k(G, w), \{u_2, v_1\} \in \text{disk}_k(G, w)$. Note that this case cannot occur because otherwise we would have $\text{dt}_G(u_1, v_1), \text{dt}_G(u_2, v_2) \leq 2k$, which contradicts the assumption that $\text{dt}_G(u_1, v_1), \text{dt}_G(u_2, v_2) \geq 2k{+}1$. This completes the case analysis and the proof of Claim 2.12. □

Proof of Claim 2.13

Proof. Suppose that condition ($\star$) is not satisfied. First, note that we have

$$E(V_1, V_2) = \sum_{\Delta' \in \mathscr{T}_{k-1}} \sum_{\Delta'' \in \mathscr{T}_{k-1}} \left| E(V_1^{\Delta'}, V_2^{\Delta''}) \right| .$$

Now, let $\Delta', \Delta'' \in \mathscr{T}_{k-1}$ be any two $(k-1)$-disk isomorphism types. The key observation is that if $\{u_1, v_2\}$ is a (Δ', Δ'')-edge from V_1 to V_2, then for every (Δ', Δ'')-edge $\{u_2, v_1\}$ from V_2 to V_1 the distance between u_2 and v_2 or the distance between v_1 and u_1 must be smaller than $2k+1$ (otherwise, ($\star$) would be satisfied and the edges could be rewired). Since the graph has bounded degree, this implies an upper bound on the number of possible endpoints u_2, v_1 and thus implies an upper bound on $|E(V_2^{\Delta'}, V_1^{\Delta''})|$. It follows that $|E(V_1^{\Delta'}, V_2^{\Delta''})|$ is also bounded by Lemma 2.10. In case there is no (Δ', Δ'')-edge from V_1 to V_2, the number of (Δ', Δ'')-edges from V_2 to V_1 can be bounded directly by Lemma 2.10.

We proceed to make this precise. For every choice of $\Delta', \Delta'' \in \mathscr{T}_{k-1}$, we distinguish two cases as mentioned before: whether we can find a (Δ', Δ'')-edge from V_1 to V_2 or not.

Case 1; there exist $u_1 \in V_1$ and $v_2 \in V_2$ such that $\{u_1, v_2\} \in E$, $\mathrm{disk}_{k-1}(G, u_1) \cong \Delta'$ and $\mathrm{disk}_{k-1}(G, v_2) \cong \Delta''$, i. e., $|E(V_1^{\Delta'}, V_2^{\Delta''})| > 0$. Since condition ($\star$) is not satisfied, at least one endpoint of every (Δ', Δ'')-edge $\{u_2, v_1\}$ from V_2 to V_1 must have distance less than $2k+1$ to u_1 or v_2. Without loss of generality, fix such an edge with $\mathrm{dt}_G(u_2, v_2) < 2k+1$. The case $\mathrm{dt}_G(u_1, v_1) < 2k+1$ can be analyzed similarly. There are at most $3d^{2k}/2$ vertices with distance less than $2k+1$ to v_2 by Lemma 1.4. Each of these close vertices can be adjacent to at most d vertices in V_1 whose $(k-1)$-disks are isomorphic to Δ''. Taking the symmetric case $\mathrm{dt}_G(u_1, v_2) < 2k+1$ into account, we have $|E(V_2^{\Delta'}, V_1^{\Delta''})| \leq 2 \cdot 3d^{2k}/2 \cdot d \leq 3d^{2k+1}$. Now by Lemma 2.10, it holds that

$$\left|E(V_1^{\Delta'}, V_2^{\Delta''})\right| + \left|E(V_2^{\Delta'}, V_1^{\Delta''})\right| \leq 6d^{2k+1} + \frac{|V_1||V_2|}{|V|} \cdot \lambda \Big[\mathrm{sub}_{\Delta'}(\Delta'') + \mathrm{sub}_{\Delta''}(\Delta') \Big] .$$

Case 2; there do not exist $u_1 \in V_1$ and $v_2 \in V_2$ such that $\{u_1, v_2\} \in E$, $\mathrm{disk}_{k-1}(G, u_1) \cong \Delta'$ and $\mathrm{disk}_{k-1}(G, v_2) \cong \Delta''$, i. e., $|E(V_1^{\Delta'}, V_2^{\Delta''})| = 0$. By Lemma 2.10, we have

$$\left|E(V_1^{\Delta'}, V_2^{\Delta''})\right| + \left|E(V_2^{\Delta'}, V_1^{\Delta''})\right| \leq 0 + \frac{|V_1||V_2|}{|V|} \cdot \lambda \Big[\mathrm{sub}_{\Delta'}(\Delta'') + \mathrm{sub}_{\Delta''}(\Delta') \Big] .$$

This completes the case analysis. By Lemma 2.8, we have $|E(V_1^{\Delta'}, V_2^{\Delta''})| \neq 0$ for at most $D_k d$ pairs of $(k-1)$-disks Δ', Δ''. It follows that

$$\begin{aligned}
& E(V_1, V_2) \\
&= \sum_{\Delta' \in \mathscr{T}_{k-1}} \sum_{\Delta'' \in \mathscr{T}_{k-1}} |E(V_1^{\Delta'}, V_2^{\Delta''})| \\
&\leq D_k d \cdot \max_{\Delta', \Delta'' \in \mathscr{T}_{k-1}} |E(V_1^{\Delta'}, V_2^{\Delta''})| \\
&\leq D_k d \cdot \left(6d^{2k+1} + \frac{\lambda |V_1||V_2|}{|V|} \cdot \max_{\Delta', \Delta'' \in \mathscr{T}_{k-1}} \Big[\mathrm{sub}_{\Delta'}(\Delta'') + \mathrm{sub}_{\Delta''}(\Delta') \Big] \right) \\
&\leq 6d^{2k+2} D_k + \lambda \cdot \min(|V_1|, |V_2|) \cdot 2 D_k d ,
\end{aligned}$$

where the last step follows from Lemma 2.8. □

2.4. Constructions and Quantitative Bounds

In this section, we prove Theorems 2.3 and 2.4.

2.4.1. Linear-Time Construction and Quantitative Bound

2.3 Theorem. Let $k \geq 1$ and $\epsilon \in (0,1)$. Let $G = (V, E)$ be a bounded-degree graph and $\operatorname{girth}(G) \geq 2k+2$. There is a deterministic algorithm that outputs a graph $H = (V', E')$ such that

$$\|\operatorname{freq}_k(G) - \operatorname{freq}_k(H)\|_1 \leq \epsilon \text{ and } |V'| \leq 45\frac{d^{3k+2}D_k}{\epsilon}.$$

The algorithm has time and query complexity $\mathcal{O}(|V|)$. ∎

Proof. Let $\varphi := 15D_k d^{3k+2}/\epsilon$. Without loss of generality assume that $\varphi \leq |V|/3$ (otherwise just output $H := G$ directly). First, we partition V into two parts V_1 and V_2 so that $\varphi \leq |V_1| \leq 2\varphi$ and for any k-disk Γ, $|\operatorname{freq}_k(G)_\Gamma - \operatorname{freq}_k(V_i \mid G)_\Gamma| \leq 1/\varphi$. Such a partition can be constructed as follows: For each k-disk $\Gamma \in \mathscr{T}_k$, we put $\lceil \varphi \cdot \operatorname{freq}_k(G)_\Gamma \rceil$ vertices v with $\operatorname{disk}_k(G, v) \cong \Gamma$ into V_1 and the remaining ones into V_2. Thus, $\varphi \leq |V_1| \leq \varphi + |\mathscr{T}_k| \leq 2\varphi$ and we have

$$|\operatorname{freq}_k(G)_\Gamma - \operatorname{freq}_k(V_1 \mid G)_\Gamma| \leq \left|\frac{\varphi \cdot \operatorname{freq}_k(G)_\Gamma - \lceil \varphi \cdot \operatorname{freq}_k(G)_\Gamma \rceil}{\varphi}\right| \leq \frac{1}{\varphi}.$$

Since $|V_2| = n - 2\varphi \geq \varphi$, we also have $|\operatorname{freq}_k(G)_\Gamma - \operatorname{freq}_k(V_2 \mid G)_\Gamma| \leq 1/\varphi$. By the triangle inequality, the partitions V_1 and V_2 satisfy the prerequisite of Proposition 2.11 with $\lambda = 2/\varphi$. Additionally, it follows that

$$\|\operatorname{freq}_k(G) - \operatorname{freq}_k(V_1 \mid G)\|_1 \leq \frac{D_k}{\varphi}. \tag{2.7}$$

Let $G' = (V, E')$ be a copy of G. As long as G' and the partition V_1, V_2 satisfy the prerequisite of Proposition 2.11 and condition $(\star)$, we *rewire* the edges of G' according to Proposition 2.11 so that G' will satisfy the properties given by Ineq. (2.3) and (2.4) and Expr. (2.5). When G' does not satisfy condition $(\star)$ anymore, we let $H := G'[V_1]$ and we are done. Note that at the end of the process, G' satisfies Eq. (2.6), which implies that the number of edges between V_1 and V_2 in G', i. e., the *boundary* of H, is at most

$$6d^{2k+2}D_k + 2\lambda d D_k \cdot \min(|V_1|, |V_2|) \leq 6d^{2k+2}D_k + \frac{4dD_k}{\varphi} \cdot 2\varphi \leq 7d^{2k+2}D_k.$$

Now note that for any vertex $v \in H$, the k-disk of v in H differs from the k-disk of v in G' only if v is within distance at most k to the boundary of H, which in turn has size at most $7d^{2k+2}D_k$. By Lemma 1.4, we have that the total number of vertices in H with different k-disks in H and G' is at most $2d^k \cdot 7d^{2k+2}D_k \leq 14d^{3k+2}D_k$, which implies that

$$\|\operatorname{freq}_k(V_1 \mid G) - \operatorname{freq}_k(H)\|_1 \leq \frac{14d^{3k+2}D_k}{\varphi}. \tag{2.8}$$

It follows from Eqs. (2.7) and (2.8) and the triangle inequality that

$$\|\mathrm{freq}_k(G) - \mathrm{freq}_k(H)\|_1 \leq \frac{D_k + 14d^{3k+2}D_k}{\varphi} \leq \frac{15d^{3k+2}D_k}{\varphi} \leq \epsilon\,,$$

where the last inequality follows from our choice of φ.

Finally, we note that the graph H can be constructed by the following deterministic linear-time algorithm. We first compute the k-disk frequency vector $\mathrm{freq}_k(G)$ of G. Since all k-disks are trees (because G has girth at least $2k+2$), we can verify isomorphism of two k-disks in time $\mathcal{O}(2d^k) \subseteq \mathcal{O}(1)$ by Lemma 1.4 and [AH74, Theorem 3.3]. In total, we need $\mathcal{O}(|V|)$ time to identify the k-disk of every vertex. Then, we consider all bounded-degree graphs with girth at least $2k+2$ and size at most φ and output the graph H with the k-disk frequency vector $\mathrm{freq}_k(H)$ that is closest to $\mathrm{freq}_k(G)$ in ℓ_1-norm distance, which has time complexity $\mathcal{O}(\Phi^{\Phi-2} \cdot 2d^k) \subseteq \mathcal{O}(1)$ by Cayley's formula [Cay89] and Lemma 1.4. In total, the running time of the algorithm is $\mathcal{O}(|V|)$. □

2.4.2. Randomized Construction in Constant Time

2.4 Theorem. Let $k \geq 1$ and $\epsilon, \delta \in (0,1)$. Let $G = (V,E)$ be a bounded-degree graph of size $|V| \geq 10^7 d^{6k+4} {D_k}^6/(\epsilon^4\delta^3)$ and girth$(G) \geq 2k+2$. There is an algorithm that outputs, with probability $1-\delta$, a graph $H = (V', E')$ such that

$$\|\mathrm{freq}_k(G) - \mathrm{freq}_k(H)\|_1 \leq \epsilon \text{ and } |V'| \leq \frac{5000 d^{3k+2} {D_k}^3}{\epsilon^2\delta}\,.$$

The algorithm has query complexity at most $10^4 \cdot \frac{d^{4k+4}{D_k}^3}{\epsilon^2\delta}$ and time complexity $\mathcal{O}(\mathrm{poly}(|V'|) \cdot d^{2k})$. ■

Algorithm 2.1

```
1:  function PartitionAndRewire(G = (V, E), φ)
2:      V_1 ← sample φ vertices from V uniformly at random
3:      V_2 ← V ∖ V_1
4:      E' ← E, G' ← (V, E')
5:      for all {u_1, v_2} ∈ (V_1 × V_2) ∩ E' and {u_2, v_1} ∈ (V_2 × V_1) ∩ E' do
6:          if disk_{k-1}(G', u_1) ≅ disk_{k-1}(G', u_2) ∧ disk_{k-1}(G', v_2) ≅ disk_{k-1}(G', v_1)
                ∧ dt_{G'}(u_1, v_1) ≥ 2k + 1 ∧ dt_{G'}(u_2, v_2) ≥ 2k + 1 then
7:              E' ← E' ∖ {{u_1, v_2}, {u_2, v_1}} ∪ {{u_1, v_1}, {u_2, v_2}}
8:              G' ← (V, E')
9:              goto line 5
10:         end if
11:     end for
12:     return H := G'[V_1]
13: end function
```

Proof. We prove this theorem along the lines of Algorithm 2.1. Let $\varphi = 5000 d^{3k+2} {D_k}^3/\epsilon^2\delta$. First, we sample φ vertices $V_1 := \{v_1, \ldots, v_\varphi\}$ from G uniformly at random. Let E_1 denote the event

that all the sampled vertices are different. Note that for any i, j such that $1 \le i < j \le \varphi$, the probability that $v_i = v_j$ is at most $1/|V|$, which implies that $\Pr[E_1] \ge 1 - \varphi^2/|V| \ge 1 - \delta/2$ because $|V| \ge 2\varphi^2/\delta$.

Let $V_2 \coloneqq V \setminus V_1$. For each $i \le \varphi$, let $f_i \in \{0,1\}^{D_k}$ denote the random vector that equals the indicator vector 1_Γ if the k-disk of v_i is isomorphic to Γ. Note that $\mathrm{freq}_k(V_1 \mid G) = \sum_i f_i/\varphi$ and that $\Pr[f_i = 1_\Gamma] = \mathrm{freq}_k(G)_\Gamma$, and thus $\mathrm{E}[\mathrm{freq}_k(V_1 \mid G)] = \mathrm{E}[f_i] = \mathrm{freq}_k(G)$. Let $X \coloneqq \|\mathrm{freq}_k(G) - \mathrm{freq}_k(V_1 \mid G)\|_2^2$. We bound the deviation between $\mathrm{freq}_k(G)$ and $\sum_i f_i/\varphi$. It holds that

$$\begin{aligned}
\mathrm{E}[X] = \mathrm{E}\left[\left\|\mathrm{freq}_k(G) - \frac{\sum_{i=1}^{\varphi} f_i}{\varphi}\right\|_2^2\right] &= \mathrm{E}\left[\frac{1}{\varphi^2}\left\|\sum_{i=1}^{\varphi}(\mathrm{freq}_k(G) - f_i)\right\|_2^2\right] \\
&= \frac{1}{\varphi^2}\sum_{j=1}^{D_k}\mathrm{Var}\left[\left(\sum_{i=1}^{\varphi} f_i\right)_j\right] \\
&\overset{(a)}{=} \frac{1}{\varphi^2}\sum_{i=1}^{\varphi}\sum_{j=1}^{D_k}\mathrm{Var}\left[(f_i)_j\right] \\
&= \frac{1}{\varphi^2}\sum_{i=1}^{\varphi}\mathrm{E}\left[\|\mathrm{freq}_k(G) - f_i\|_2^2\right] \\
&= \frac{1}{\varphi}\,\mathrm{E}\left[\|\mathrm{freq}_k(G) - f_1\|_2^2\right] \\
&\overset{(b)}{\le} \frac{1}{\varphi}\,\mathrm{E}\left[\|\mathrm{freq}_k(G) - f_1\|_1\right] \\
&\le \frac{2}{\varphi},
\end{aligned}$$

where (a) follows from the fact that all f_i are independent of each other and (b) follows from the fact that the absolute values of all entries of $\mathrm{freq}_k(G') - f_1$ are at most 1. Now by Markov's inequality,

$$\Pr\left[\left\|\mathrm{freq}_k(G) - \frac{\sum_i f_i}{\varphi}\right\|_2^2 \ge \frac{2}{\delta}\cdot\frac{2}{\varphi}\right] \le \Pr\left[X \ge \frac{2}{\delta}\cdot\mathrm{E}[X]\right] \le \frac{\delta}{2}.$$

Therefore, if we let $\lambda = \frac{\epsilon}{16 D_k d^{k+1}}$, then with probability at least $1 - \delta/2$,

$$\left\|\mathrm{freq}_k(G) - \frac{\sum_i f_i}{\varphi}\right\|_1 \le \sqrt{D_k}\cdot\left\|\mathrm{freq}_k(G) - \frac{\sum_i f_i}{\varphi}\right\|_2 \le \sqrt{D_k}\cdot\sqrt{\frac{4}{\delta\varphi}} \le \frac{\lambda}{2},$$

where the last inequality follows from our choice of

$$\varphi = \frac{5000 d^{3k+2} {D_k}^3}{\epsilon^2\delta} = \frac{5000 d^k D_k}{256\lambda^2\delta} \ge \frac{16 D_k}{\lambda^2\delta}.$$

This further implies that (with probability at least $1 - \delta/2$)

$$\left\|\mathrm{freq}_k(G) - \mathrm{freq}_k(V_1 \mid G)\right\|_1 = \left\|\mathrm{freq}_k(G) - \frac{\sum_i f_i}{\varphi}\right\|_1 \le \frac{\lambda}{2}. \tag{2.9}$$

Let E_2 denote the event that $\|\mathrm{freq}_k(G) - \mathrm{freq}_k(V_1 \mid G)\|_1 \leq \lambda/2$. Thus $\Pr[E_2] \geq 1 - \delta/2$. If E_2 occurs, then $\|\mathrm{freq}_k(G) - \mathrm{freq}_k(V_2 \mid G)\|_1 \leq \lambda/2$ because $|V_2| \geq |V_1|$, and therefore $\|\mathrm{freq}_k(V_1 \mid G) - \mathrm{freq}_k(V_2 \mid G)\|_1 \leq \lambda$.

Conditioning on both events E_1 and E_2, which occur with probability $\Pr[E_1 \cap E_2] \geq 1 - 2 \cdot \delta/2 = 1 - \delta$, we apply Proposition 2.11 with G, λ and partition V_1, V_2 as follows: Let $G' = (V, E')$ be a copy of G. As long as condition $(\star)$ is satisfied, we replace G' by the rewired graph that satisfies Ineq. (2.3) and (2.4) and Expr. (2.5). After rewiring, there remain at most $6d^{2k+2}D_k + 2\lambda D_k d\varphi$ edges in the cut of V_1 and V_2. The output of the algorithm is $H \coloneqq G'[V_1]$. By Lemma 1.5 and Proposition 2.11 it holds that

$$\begin{aligned}\|\mathrm{freq}_k(V_1 \mid G) - \mathrm{freq}_k(H)\|_1 &\leq \frac{4d^k \cdot \left(6d^{2k+2}D_k + 2\lambda D_k d\varphi\right)}{\varphi} \\ &\leq \frac{24d^{3k+2}D_k}{\varphi} + 8D_k d^{k+1}\lambda \leq \frac{3\epsilon}{4}, \end{aligned} \tag{2.10}$$

where the last inequality follows from our choice of $\varphi = 5000d^{3k+2}{D_k}^3/(\epsilon^2\delta)$ and $\lambda = \epsilon/(16D_k d^{k+1})$. It follows from Eqs. (2.9) and (2.10) and the triangle inequality that

$$\|\mathrm{freq}_k(G) - \mathrm{freq}_k(H)\|_1 \leq \frac{\lambda}{2} + \frac{3\epsilon}{4} \leq \epsilon\,.$$

Now we analyze the query and time complexity of the algorithm. Note that the algorithm only needs to sample φ vertices and query all the $(k+2)$-disks of vertices in V_1. In particular, the rewiring step can be performed as follows: we consider all the vertices that are endpoints of some edges leaving V_1 by exploring the neighbors of all vertices in V_1. We want to find $u_1, v_1 \in V_1$ with $\mathrm{dt}_G(u_1, v_1) \geq 2k+1$ such that $\{u_1, v_2\} \in (V_1 \times V_2) \cap E'$ and $\{u_2, v_1\} \in (V_2 \times V_1) \cap E'$. To test if we should rewire the corresponding edges or not, we only need to consider the $(k+1)$-disks of $v_2, u_2 \in V_2$ to determine if $\mathrm{dt}_G(v_2, u_2) \geq 2k+1$. This implies that we only need to query the $(k+2)$-disks of all vertices in V_1. By Lemma 1.4, it follows that the algorithm makes at most $\varphi \cdot 2d^{k+2}$ queries.

The number of rewiring steps as well as the number of edges with at least one end in V_1 is at most φd. Let $V_1' \coloneqq V_1 \cup \{v \in V_2 \mid \{u, v\} \in E \wedge u \in V_1\}$ be the union of V_1 and all adjacent vertices. We need to know which vertices in V_1' have isomorphic k-disks. The time complexity to maintain this information is $\mathcal{O}((\varphi d)^2 D_k \cdot (2d^k)^{d^2})$ for the initialization and $\mathcal{O}(\varphi d D_k \cdot 2d^k)$ per rewiring step, where $\mathcal{O}(2d^k)$ is the time complexity for testing isomorphism of two k-disks without cycles by Lemma 1.4 and [AH74, Theorem 3.3]. All remaining operations are linear-time graph explorations, so the total time complexity is $\mathcal{O}(\mathrm{poly}(\varphi) \cdot d^{2k})$. □

3. Constant-Query Testable Properties and Their Subproperties

It is one of the grand goals of property testing to obtain characterizations of the various classes of query complexity, e. g., $\text{poly}(\epsilon)$, $\mathcal{O}(1)$ and $\mathcal{O}(n^\delta)$ for $\delta \in (0,1)$. In particular, a long line of research is devoted to constant-query testable properties in the dense graph model. In this chapter, we present a structural condition that is satisfied by all constant-query testable properties in the bounded-degree model: We show that every infinite constant-query testable property contains an infinite hyperfinite property as a subset. The results presented in this chapter are joint work with Pan Peng and Christian Sohler [FPS19].

3.1. Introduction

Following the thread of k-disks from Chapter 2 into the land of property testing, it turns out that the k-disk distribution of a graph plays a central role in constant-query testers for bounded-degree graphs. Goldreich and Ron [GR09] (implicitly) observed that such a tester essentially approximates the k-disk distribution and compares this approximation to a dictionary to derive its final decision. This gives a (technical) characterization of constant-query testable properties.

One important problem in computational complexity is the characterization of certain complexity classes. Of course, the definition of a complexity class can also serve as a characterization of its problems (languages). However, most times this definition reflects the pure objective, e. g., the existence of a polynomial-time algorithm, rather than structural properties of these problems that help us to understand *why* a problem admits this objective.

In property testing, the arguably most extreme case of sublinear complexity has received significant attention: to find a less technical, more structural characterization of properties that are testable with constant query complexity. For the dense graph model, a characterization based on regularity was developed by Alon et al. [Alo+09]. However, for the bounded-degree graph model, much less is known. For example, it is known that every hyperfinite property is two-sided error testable with constant query complexity [NS11]. It is easy to understand that not every constant-query testable property is hyperfinite, though: by an early result of Goldreich and Ron [GR97], connectivity is constant-query testable. On the other hand, expanders graphs are connected[1] and not hyperfinite[2] by their definition.

In this chapter, we prove that every infinite constant-query testable property *contains* an

[1] Assume that G is a γ-expander graph but it is not connected. Then there exists a connected component of volume at most $\text{vol}(m/2)$, and this connected component has a cut size of 0.

[2] Given a γ-expander graph G, assume that it is hyperfinite for some function $\rho(\bullet)$. Set $\epsilon = \gamma/2$ and remove ϵdn edges such that the remaining connected components have size at most $\rho(\epsilon)$. By an averaging argument, there exists a connected component $G[C]$ that has cut size $|E \cap (C \times (V \setminus C))|/\text{vol}(C) \leq \epsilon < \gamma$, and the same holds for its complement $G[V \setminus C]$. At least one of these two subgraphs, say $G[C]$, has volume less than $\text{vol}(G)/2$. This is a contradiction to the fact that G is a γ-expander, namely, to $|E \cap (C \times (V \setminus C))|/\text{vol}(C) \geq \gamma$.

infinite hyperfinite *sub*property, i. e., there exists an infinite isomorphism-invariant subset of the property that is hyperfinite. This is not an actual characterization, but to the best knowledge of the author, it is the first non-trivial structural result that applies to all constant-query testable properties of bounded-degree graphs at the time of writing.

3.1.1. Results in This Chapter

The main result in this chapter is that every infinite property of bounded-degree graphs that is constant-query testable contains an infinite hyperfinite subproperty. Along the lines, we obtain that every infinite, *non-trivially* testable complement of such a property also contains an infinite hyperfinite subproperty.

3.1 Definition (non-trivially testable). A graph property Π is non-trivially testable if it is constant-query testable and there exists $\epsilon > 0$ such that the set of graphs that is ϵ-far from Π is infinite. ■

3.2 Theorem. Let Π be a property of bounded-degree graphs that has unbounded size and that can be tested with two-sided error using $q_\Pi(\epsilon)$ queries. Then, there exists a function ρ such that Π contains a subproperty Π' of unbounded size that is ρ-hyperfinite. Furthermore, the complement of Π contains an infinite hyperfinite subproperty if Π is non-trivially testable. ■

This result implies that any intersection of expander graphs and another set of graphs (that yields a non-trivially testable property) is not constant-query testable. An interesting question that arises from this observation is whether a property must contain expander graphs if it is not constant-query testable. It turns out that the structure of constant-query testable properties conforms to more complex rules.

3.3 Theorem. There exists an infinite graph property Π of bounded-degree graphs such that

- Π is testable (with two-sided error) with query complexity $\mathcal{O}(d/\epsilon^2)$,
- Π is not hyperfinite,
- every graph in Π differs in $\Omega(n)$ edges from every connected graph. ■

In the light of results like the characterization of constant-query testable properties in the dense graph model [Alo+09] that are flavored by the regularity lemma, we develop a partitioning theorem for bounded-degree graphs that resembles the regularity lemma. We obtain that every bounded-degree graph can be partitioned into a constant number of vertex sets $(S_i)_i$ and a small remainder T such that for all $i \neq j$, (i) there are no edges between two sets S_i and S_j and (ii) the restricted disk vector of every non-expanding subset of S_i has roughly the same disk frequency vector as S_i.

3.4 Theorem. Let $G = (V, E)$ be a bounded-degree graph. For every $k \geq 0$ and every $\delta \in (0, 1]$ there exists a function $f(\delta, d, k)$ such that V can be partitioned into $r \leq f(\delta, d, k)$ subsets $S_1, \dots, S_r$ and a set T such that

- for every $i \neq j$ there are no edges between S_i and S_j,
- $|T| \leq \delta d|V|$,
- and for every i and every subset X of S_i with $\mathrm{cond}(X) \leq \delta^2$ it holds that

$$\|\mathrm{freq}_k(X \mid G) - \mathrm{freq}_k(S_i \mid G)\|_1 \leq 3\delta.$$ ■

3.1.2. The Characterization Project

As mentioned above, not much is known about constant-query testable properties of bounded-degree graphs in general. However, for dense graphs, there is a characterization related to Szemerédi's regularity lemma, and a couple of properties of bounded-degree graphs, among them some large families of graphs, are known to be constant-query testable.

Dense Graphs. Switching focus to constant-query property testers in dense graphs for a moment, there exists a combinatorial characterization of constant-query testable properties [Alo+09], which is based on Szemerédi's regularity lemma.

3.5 Lemma (Szemerédi's Regularity Lemma). There exists a function f such that for every $\epsilon > 0$ and for every graph $G = (V, E)$, there is a partition of V into $k \in [1/\epsilon, f(\epsilon)]$ sets $V_1, \dots, V_k$ such that

1. for all but ϵk^2 pairs (i, j) with $i \neq j$, for any subsets $V_i' \subseteq V_i, V_j' \subseteq V_j$ such that $|V_i'| > \epsilon|V_i|$ and $|V_j'| > \epsilon|V_j|$, it holds that
$$\left| \frac{|E(V_i, V_j)|}{|V_i||V_j|} - \frac{|E(V_i', V_j')|}{|V_i'||V_j'|} \right| \leq \epsilon,$$
2. for all pairs (i, j), it holds that $\big||V_i| - |V_j|\big| \leq 1$. ■

Roughly, this result states that every (dense) graph can be partitioned into a constant number of vertex sets so that the edges between almost all pairs of vertex sets behave quasirandomly.[3]

However, investigation of constant-query testable properties started earlier (in literature, this line of research is often referred to as the *characterization project*). Goldreich, Goldwasser, and Ron [GGR96] showed that every graph partition problem is constant-query testable with two-sided error. A graph partition problem asks to partition a graph into a constant number of partitions k such that given upper and lower bounds on the relative size of each partition as well as on the density of the pairwise edge cuts are satisfied. Their algorithm has query complexity $\mathcal{O}(\text{poly}(1/\epsilon))$. Later, Goldreich and Trevisan [GT03] proved that a graph partition problem is – essentially – testable with one-sided error if and only if the subgraph induced by each partition contains no edges. Explicitly using the regularity lemma, Alon et al. [Alo+99] showed that every property that can be expressed as a logical expression using one existential quantifier followed by one universal quantifier over constant-size sets of vertices that reduces to a general coloring problem is constant-query testable. This applies to, e. g., subgraph freeness. Subsequent analysis of constant-query testers was simplified by the work of Goldreich and Trevisan [GT03], who showed that every adaptive tester can be transformed into a non-adaptive tester so that the query complexity is roughly squared. Fischer and Newman [FN07] proved that

[3] The property described in Item 1 is called ϵ-regularity, and it basically states that the edges between all but an ϵ-fraction of the pairs V_i, V_j should behave quasirandomly, or in other words, the density of all sufficiently large subsets of V_i, V_j should be close to the cut density of V_i, V_j. It is known that $f(\epsilon)$ is bounded by a tower $2^{2^{2^{\cdots}}}$ of height $1/\epsilon^2$, which is almost optimal [CF12]. Note that $1/\epsilon \leq k$ ensures that the number of edges inside the partitions is at most ϵn^2, which is often negligible (e. g., in property testing they may be removed so that 2ϵ-far graphs are still ϵ-far). There exist numerous equivalent, weaker and stronger variants of the regularity lemma as well as algorithmic versions. See [KS96] and [Lov12, Chapter 9] for an overview.

there exists a tolerant[4] tester for every constant-query testable property. Drawing on [GT03; FN07], the aforementioned characterization of constant-query testable properties was proved by Alon et al. [Alo+09]. Their main result roughly states that a property is constant-query testable if and only if it is characterized by a finite family of regularity instances, i. e., density pairs of an ϵ-regular partition. In other words, property testing of constant-query testable properties reduces to testing whether one of finitely many partitions is ϵ-regular for the input graph. This can be done by sampling a set of vertices, querying all edges and checking whether the subgraph induced by the sample admits such a partition (by brute force).

Bounded-Degree Graphs. Turning back to bounded-degree graphs, much less is known about necessary (or necessary and sufficient) conditions of being constant-query testable. Goldreich and Ron [GR09] characterize properties that can be tested by constant-query proximity-oblivious testers with one-sided error. Proximity-oblivious testers repeatedly perform a test that is uniform (hence *oblivious*) for all choices of the proximity parameter ϵ on the graph, and only the number of executions of this test may depend on ϵ. The authors show that there exists such a tester if and only if the graphs in the property satisfy a generalized subgraph-freeness condition and for any graph that is far from the property, there is a way to eliminate all forbidden subgraphs so that no new forbidden subgraphs are created in a cascading fashion (this possibility arises due to the generalization of subgraph freeness). Their proof uses a *canonical tester*, which samples a set of vertices, explores the disk of these vertices and rejects if and only if it contains one of the forbidden subgraphs that correspond to the property that is being tested. The canonical tester is uniform with respect to the size of the graph. Although it is not explicitly discussed, there is a rather straightforward modification of this proximity-oblivious tester to general constant-query testers in the bounded-degree model, which is given in Section 3.3.1 for the sake of completeness.[5]

A relatively large number of properties is known to be constant-query testable. This includes subgraph-freeness, edge connectivity, degree regularity and being Eulerian (all one-sided error) [GR97], vertex connectivity (one-sided error) [YI08], (s,t)-disconnectivity (one-sided error) [YK12], perfect matching (two-sided error) [YYI09], fullness and edge-connected orientability (both two-sided error) [ITY12], connectivity for supermodular set functions (two-sided error) [TY15], several parameterized problems (error depends on problem) [IY17], subdivision freeness (two-sided error) [KY13], minor freeness (two-sided error) [BSS08; Has+09] and being hyperfinite (two-sided error) [NS11]. Furthermore, the results of Newman and Sohler [NS11] imply that any property of hyperfinite graphs is constant-query testable. Other models of testing (e. g., proximity-oblivious testing [GR09; GS12]) and testers for restricted graph classes (e. g. testing bipartiteness in general planar graphs [Czu+11]) have also been studied.

From the list in the preceding paragraph, hyperfinite graphs turn out to be closely connected to constant-query testers, disk frequency vectors and the results discussed in Chapter 2. One of the first results in this line of research is due to Czumaj, Shapira, and Sohler [CSS09], who proved that every hereditary[6] property of the family of non-expanding graphs, i. e., every

[4] A tolerant tester is a tester that gives guarantees on the acceptance probability of objects that are very close to the property.

[5] This proof is probably one of the examples that is considered a known result among researchers in the respective community, but which was never spelled out explicitly for some time due to the lack of related *novel* results.

[6] A graph property is hereditary if it is invariant under removal of vertices.

induced subgraph of sufficiently large constant size is a poor expander, is constant-query testable. This family includes, e. g., minor-free graphs – in particular, planar graphs – and graphs with bounded genus. Schramm [Sch11] proved that a graph that is hyperfinite can be distinguished from a graph that is far from being hyperfinite by its disk frequency vector (see Lemma 3.7 below), which lead to further progress: Benjamini, Schramm, and Shapira [BSS08] showed that every minor-closed property of bounded-degree graphs is constant-query testable.[7] Their result is based on [Sch11]: since minor-free graphs are hyperfinite, one can test whether the input graph is hyperfinite by approximating its disk frequency vector. If it is far from hyperfinite, the tester can reject immediately. Otherwise, if the graph is ϵ-far from having the property at hand, one can decompose the graph into connected components of constant size by removing at most, say, an $\epsilon/2$-fraction of the edges. The resulting graph is still $\epsilon/2$-far from the property, but it suffices to check only small subgraphs for witnesses (i. e., forbidden minors) against the property. Hassidim et al. [Has+09] improved their query complexity, which is triple-exponential in $1/\epsilon$, to $2^{\mathrm{poly}(\epsilon,d)}$. This is achieved by a local partitioning oracle for hyperfinite graphs, i. e., an algorithm that gives query access to a (black-box) partition of a hyperfinite input graph. Their main technical contribution is a transformation of certain greedy algorithms that process their input in uniformly random order into local algorithms. In particular, they show that for computing the result of a random greedy step it is sufficient to compute only a constant number of previous greedy steps with constant probability. Finally, Kumar, Seshadhri, and Stolman [KSS19] proved that minor-freeness can be tested with query complexity $d \cdot \mathrm{poly}(1/\epsilon)$. Their two-sided error tester is based on their one-sided error tester for minor-freeness [KSS18] (see Section 5.1.2 for more details). Most notably, the tester uses random walks and the analysis draws on spectral graph theory, while previous approaches are more combinatorial.

[BSS08; Has+09]

The results by Benjamini, Schramm, and Shapira [BSS08] and Hassidim et al. [Has+09] were generalized by Newman and Sohler [NS11], who proved that every hyperfinite property of graphs is constant-query testable. Their main observation is that if two graphs G and H have close k-disk frequency vectors (for an appropriately large but constant k), then G and H are ϵ-close. Therefore, one can test any property of hyperfinite graphs by approximating the k-disk frequency vector of the input graph, reconstructing the input graph from this vector up to an $\epsilon/2$-fraction of the edges, and check whether this reconstruction is $\epsilon/2$-close to the property (e. g., by brute force). Using the result of [Sch11], one can extend this statement and prove that every hyperfinite property of bounded-degree graphs is testable. In particular, one can first test whether the input graph is hyperfinite and then proceed as outlined before.

3.1.3. Open Problems and Future Work

We discuss all open problems related to Chapters 2 to 4 in Section 4.1.3.

3.1.4. Overview of the Analysis

Theorem 3.2. We begin by sketching the idea of the proof of Theorem 3.2, i. e., every non-trivially testable property Π of bounded-degree graphs contains an infinite hyperfinite subproperty. The subproperty is constructed by taking a graph $G \in \Pi$ of size n and transforming

[7] Equivalently, by the Robertson–Seymour theorem [RS04], being $\mathcal{F}$-minor free for any family $\mathcal{F}$ of forbidden minors is constant-query testable.

it step by step into a hyperfinite graph $H \in \Pi$ of size n. Since Π has infinite size, this gives us an infinite number of different hyperfinite graphs in Π. As mentioned in Section 3.1.2, it is known that every constant-query tester for bounded-degree graphs can be transformed into a canonical tester that approximates the k-disk frequency vector of its input graph and answers according to a dictionary lookup on this approximated vector. We exploit this behavior while transforming G. In particular, all intermediate graphs of the transformation will be ϵ-close to Π by ensuring that their k-disk frequency vectors do not differ from the original k-disk frequency vector of G by too much (so that a tester for Π accepts these graphs, which means that they cannot be ϵ-far from Π). This implies that, as a final step, we can apply ϵdn edge modifications to obtain a graph in Π. If the transformed graph was hyperfinite before this final step, the final graph is also hyperfinite because only a small number of edges was modified. However, the actual proof is a bit more complicated, and we make the additional steps more precise in Sections 3.3.2 and 3.3.3.

Theorem 3.3. Consider the following graph property Π: half of the vertices are in the same connected component, and half of the vertices are isolated. This property is not hyperfinite because the connected component may be an expander graph, and it is far from being connected because of the isolated vertices. It remains to prove that one can actually test this property with constant query complexity. We show that, roughly speaking, one can test this property by approximating the number of isolated vertices and running a connectivity tester on all non-isolated vertices of a sufficiently large sample. The essence of the analysis is that one can derive a joint criterion for being far from Π from the two properties that all vertices are isolated and connectivity.

Theorem 3.4. We partition the graph $G = (V, E)$ into sets $(A_i)_i$ by iteratively cutting out the smallest vertex sets with conductance at most δ. Thus, the boundary of every set A_i is at most $\delta d|A_i|$, which sums up to at most δdn, and we can afford to move these boundary vertices to T. Note that this way, there are no edges between A_i and A_j for any pair $i \neq j$. Then, we construct $(S_i)_i$ by grouping the sets $(A_j)_j$ by their approximate k-disk frequency vectors. We observe that for every i and every $X \subseteq S_i$ with $\mathrm{cond}(X) \leq \delta^2$, most non-empty sets $X \cap A_j$ are actually equal to A_j. In particular, less than a δ-fraction of vertices in X belongs to those A_i that are not subsets of X. Otherwise, by contradiction, the number of edges between X and $X \setminus V$ would exceed $\delta|X| \cdot \delta d$ because we always cut out the smallest vertex sets with conductance at most δ when constructing $(A_i)_i$. Therefore, $\mathrm{freq}_k(X \mid G)$ is roughly a linear combination of approximately equal k-disk frequency vectors by the construction of $(S_i)_i$.

3.2. Preliminaries

In this chapter, all graphs are bounded-degree graphs unless stated otherwise. We have the following simple lemma on the ℓ_1-norm distance of the disk frequency vectors of two graphs that are ϵ-close to each other.

3.6 Lemma. Let $\epsilon > 0$ and $k \geq 1$. Let G_1, G_2 be bounded-degree graphs such that G_1 is ϵ-close to G_2. Then, $\|\mathrm{freq}_k(G) - \mathrm{freq}_k(G_2)\|_1 < 6\epsilon d^{k+1}$. ■

Proof. Let $F := E(G_1) \oplus E(G_2)$ denote the set of edges that appear only in one of the two graphs G_1, G_2. Since G_1 is ϵ-close to G_2, it holds that $|F| \leq \epsilon dn$. Note that for any $e \in F$, the

total number of vertices that are within distance at most k to either of its endpoint is at most $2(1 + d + d(d-1) + \cdots + d(d-1)^{k-1}) \leq 3d^k$. This further implies that the total number of vertices that may have different k-disk types in G_1 and G_2 is at most $|F| \cdot 3d^k \leq 3\epsilon d^{k+1} n$. Finally, we note that each vertex with different k-disk types in G_1 and G_2 contributes at most $2/n$ to the ℓ_1-norm distance of $\mathrm{freq}_k(G_1)$ and $\mathrm{freq}_k(G_2)$, which implies that

$$\|\mathrm{freq}_k(G_1) - \mathrm{freq}_k(G_2)\|_1 < 3\epsilon d^{k+1} n \cdot \frac{2}{n} = 6\epsilon d^{k+1}\,. \qquad \square$$

The converse to the above lemma is not true in general, i. e., it is not true that the closeness of the disk frequency vectors of two graphs implies the closeness of these two graphs (see, e. g., Proposition 2.11, the construction in the proof of Theorem 2.3 and Theorem 4.7 in [Fic14]). However, Benjamini, Schramm, and Shapira [BSS08] showed that a weak version of the converse still holds for hyperfinite graphs.

3.7 Lemma [BSS08, Theorem 3]. Let $s \geq 1$ and $\epsilon > 0$. Let Λ_1 be the set of (ϵ, s)-hyperfinite bounded-degree graphs, and let Λ_2 be the set of all bounded-degree graphs that are not $(4\epsilon \log(4d/\epsilon), s)$-hyperfinite. Then it holds that for all graphs $G_1 \in \Lambda_1$ and $G_2 \in \Lambda_2$,

$$\|\mathrm{freq}_k(G_1) - \mathrm{freq}_k(G_2)\|_1 > \frac{8\epsilon}{d} \log(4/3)\,, \text{ where } k = \frac{10sd^{2s+1}}{\epsilon}\,. \qquad \blacksquare$$

3.8 Corollary. Let $\epsilon, s > 0$. Let Π be a constant-query testable property of bounded-degree graphs. Suppose there exists a graph $G \in \Pi_n$ that is (ϵ, s)-hyperfinite. Then, every graph $G' \in \Pi_n$ such that $\|\mathrm{freq}_k(G) - \mathrm{freq}_k(G')\|_1 < (8\epsilon/d)\log(4/3)$ is $(4\epsilon \log(4d/\epsilon), s)$-hyperfinite, where $k = 10sd^{2s+1}/\epsilon$. $\blacksquare$

Disk Frequency Preservers and Blow-Up Graphs. We use DFPs (see Definition 1.7) as a building block to construct graphs of size n that have constant-size connected components and approximately preserve the disk frequencies of a given graph G of size n.

3.9 Definition (blow-up graph). Let $\delta, k > 0$, and let G be a bounded-degree graph of size n. Let H be a (δ, k)-DFP of G of size $h \leq M_{d,\delta,k}$. Let H' be the graph of size n that is composed of $\lfloor n/h \rfloor$ disjoint copies of H and $n - h \cdot \lfloor n/h \rfloor$ isolated vertices. We call H' the (δ, k)-blow-up graph of G. $\blacksquare$

Similar to DFPs, blow-up graphs preserve the k-disk frequency vector of the original graph.

3.10 Lemma. Let $\delta, k > 0$. Let $n \geq n_1(\delta, d, k) := 2M_{d,\delta,k}/\delta$. Let G be any bounded-degree graph of size n and let H be the (δ, k)-blow-up graph of G. We have $\|\mathrm{freq}_k(G) - \mathrm{freq}_k(H)\|_1 < 2\delta$. $\blacksquare$ n_1

Proof. Let $V_1 \subseteq V(H)$ be the $n - h \cdot \lfloor n/h \rfloor$ isolated vertices created during the construction of H and let $V_2 = V(H) \setminus V_1$. By Lemma 1.6, $\|\mathrm{freq}_k(G) - \mathrm{freq}_k(H[V_2])\|_1 < \delta$. Since $|V_1| \leq M_{d,\delta,k}$, it holds that $\|\mathrm{freq}_k(G) - \mathrm{freq}_k(H)\|_1 < \delta + 2|V_1|/n < 2\delta$. $\square$

3.3. Hyperfinite Subproperties

In this section, we give the proof of the main theorem, i. e., Theorem 3.2. We first give the necessary tools in Section 3.3.1, and then give the proof of the first part and second part of Theorem 3.2 in Section 3.3.3 and Section 3.3.2, respectively.

3.3.1. Canonical Tester

One of our main tools is the following characterization of constant-query testable properties by so-called *canonical testers*. Since it is hard to derive statements that hold for all constant-query property testers when they do not share any structure, we use an intermediate layer that unifies all constant-query tester by wrapping them into a universal – or *canonical* – property tester. For bounded-degree graphs, such a tester was given by Goldreich and Ron [GR09] for properties that are proximity-obliviously testable. Later, Czumaj, Peng, and Sohler [CPS16] defined a canonical tester for directed graphs in a similar fashion.

In this section, we define a canonical tester that is already implicit in [GR09]. The main difference is that our canonical tester makes decisions based on the disk frequency vectors instead of the forbidden subgraphs as in [GR09].

$\hat{q}_\Pi(\bullet,\bullet), n_2$

3.11 Theorem (canonical tester, bounded-degree). Let $\Pi = (\Pi_n)_{n\in\mathbb{N}}$ be a graph property that can be tested with query complexity $q_\Pi(\epsilon, d)$. Then there exists $t := c \cdot q_\Pi(\epsilon, d)$ for some constant $c > 1$ and some $n_2 = n_2(\epsilon, d)$ such that for any $\epsilon > 0$, $n \geq n_2$, there exists a tester $\mathscr{T}_C$ that

1. accepts any bounded-degree graph G of size n with probability at least $2/3$, if

$$\min_{G'\in\Pi_n} \|\mathrm{freq}_t(G) - \mathrm{freq}_t(G')\|_1 \leq \frac{1}{12t},$$

2. rejects any bounded-degree graph G of size n with probability at least $2/3$, if

$$\min_{G'\in\overline{\Pi}_{n;>\epsilon}} \|\mathrm{freq}_t(G) - \mathrm{freq}_t(G')\|_1 \leq \frac{1}{12t}.$$

This tester, which we call a *canonical tester* for Π, has query complexity $\hat{q}_\Pi(\epsilon, d) \leq t \cdot d^{t+2}$. ■

Proof. The first part of the proof, i. e., the transformations from the original tester $\mathscr{T}$ to the canonical tester $\mathscr{T}_C$, is similar to the proofs in [GR09; CPS16]. The last part, i. e., how the behavior of $\mathscr{T}_C$ relates to the disk frequency vector, differs from previous work and it is tailored to obtain the characterization as stated in the theorem, which in turn will be helpful in our analysis of the structure of constant-query testable properties.

Let $\mathscr{T}$ be a tester for Π_n with (amplified, see discussion after Definition 1.10) error probability at most $1/24$. Let $t := c \cdot q_{\mathscr{T}}(\epsilon, d)$ for some constant $c > 1$. We transform $\mathscr{T}$ into a canonical tester $\mathscr{T}_C$. In particular, we define the following sequence of testers:

$\mathscr{T}_1$ Let $\mathscr{T}_1$ be the tester that first samples a set S of t vertices uniformly at random from the input graph $G = (V, E)$ and then explores, for every $v \in S$, the t-disk of v. This requires at most $t \cdot d^{t+2}$ queries. We mark all vertices in S as roots and denote the union of all subgraphs observed by $H = (V', E')$. We define $\mathscr{T}_1$ to simulate the execution of $\mathscr{T}$ in the following way. Given a neighbor query (v, i) to G, the tester $\mathscr{T}_1$ will select a random, uniformly distributed permutation $\pi : V \to V$ on-the-fly and provide oracle access to the permuted graph $\pi(G) = (V, \pi(E))$, where $\pi(E) := \{(\pi(u), \pi(v) \mid \{u, v\} \in E\}$. Initially, all vertices in S are considered unused. In the simulation, when $\mathscr{T}$ makes a query (v, i), if v has not appeared in any prior query or answer, then the tester $\mathscr{T}_1$ allocates v to an unused vertex u in the sample set S, and we let $\pi(v) := u$ and u will be considered as used; otherwise $\mathscr{T}_1$ uses the allocation $\pi(v)$ determined before. To answer a neighbor query

(v, i), $\mathscr{T}_1$ selects $w := \pi(u)$, where u is the i'th neighbor of v. If w has been selected as the image of some vertex in the permutation π before, then $\mathscr{T}_1$ returns $\pi^{-1}(w)$; otherwise $\mathscr{T}_1$ returns a random unused value (vertex label) x and we let $\pi(x) = w$. If $w \in S$, then w will be considered used. The returned values will then be fed into $\mathscr{T}$. The tester $\mathscr{T}_1$ makes the same decision as the final decision of $\mathscr{T}$ after receiving all the necessary query answers. We note that $\mathscr{T}_1$ makes decisions based on H, the explored t-discs from t sampled vertices, and may depend on the labels and internal randomness by the algorithm. We denote the event that the tester draws the random permutation π by π.

$$\Pr[\mathscr{T}_1 \text{ correctly answers } G] = \sum_{\pi} \Pr[\mathscr{T} \text{ correctly answers } \pi(G)] \Pr[\pi] \geq n! \cdot \frac{11}{12n!}.$$

$\mathscr{T}_2$ We make the tester label-oblivious by defining the tester $\mathscr{T}_2$ to be the one that accepts G with the average probability that $\mathscr{T}_1$ accepts G over all possible labellings of H with corresponding sequences of queries and answers (see the note on the complexity of this computation on the following page). We denote the event that the tester draws the random permutation π' by π'.

$$\Pr[\mathscr{T}_2 \text{ accepts } G \mid \pi] = \sum_{\pi'} \Pr[\mathscr{T}_1 \text{ accepts } G \mid \pi'] \Pr[\pi'] = \Pr[\mathscr{T}_1 \text{ accepts } G].$$

Since it suffices to consider all labellings of H and random coins of $\mathscr{T}$, this does not require any additional queries.

$\mathscr{T}_C$ We make the tester oblivious of the order of S by defining the tester $\mathscr{T}_C$ to be the one that accepts with probability that is equal to the average of all acceptance probabilities of $\mathscr{T}_2$ over all possible relabellings of vertices in H and orders of S. We denote the event that the tester draws the random permutation π by π and we denote the event that $\mathscr{T}_C$ draws the order S by S.

$$\Pr[\mathscr{T}_C \text{ accepts } G \mid S, \pi] = \sum_{S} \Pr[\mathscr{T}_2 \text{ accepts } G \mid S, \pi] \Pr[S \mid \pi] = \Pr[\mathscr{T}_2 \text{ accepts } G \mid \pi].$$

Again, this does not require any additional queries because we only need to consider all permutations of sampled vertices and random coins of $\mathscr{T}$. The total query complexity is $t \cdot d^{t+2}$.

Now if we let $n_2 := 12d^{2t}t^2$, then for any $n \geq n_2$, it holds that with probability at least $1 - d^{2t}t^2/n \geq 1 - 1/12$, none of the t sampled t-disks will intersect. That is, with probability $1 - 1/12$, the decision of the tester $\mathscr{T}_C$ will only depend on the structure (or the isomorphism types) of the explored t *disjoint* t-disks.

Consider the case that the input graph G satisfies $\min_{G' \in \Pi_n} \|\mathrm{freq}_t(G) - \mathrm{freq}_t(G')\|_1 \leq \delta_C$ for some $\delta_C > 0$. Let $G' \in \Pi_n$ denote a graph for which this minimum is attained. By the definition of disk frequency vectors, there exists a bijection $\Phi : V(G) \to V(G')$ such that $\mathrm{disk}_t(G, v) \ncong \mathrm{disk}_t(G', \Phi(v))$ for at most a δ_C-fraction of the vertices $v \in V(G)$. Since every vertex $v \in S$ is sampled independently and uniformly at random, the probability that $\mathrm{disk}_t(G, v) \ncong \mathrm{disk}_t(G', \Phi(v))$ is bounded by the total variation distance of $\mathrm{freq}_t(G)$ and $\mathrm{freq}_t(G')$, which is at most $\delta_C/2$ by our assumption. By the union bound, the probability that there exists

some vertex $v \in S$ with $\text{disk}_t(G, v) \ncong \text{disk}_t(G', \Phi(v))$ is at most $|S| \cdot \delta_C \leq t \cdot 1/12t = 1/12$. Since $\mathscr{T}_C$ rejects G' with probability at most $1/12$ and the probability that there exists a pair among all the t sampled t-discs that intersects is at most $1/12$, $\mathscr{T}_C$ rejects G with probability at most $1/12 + 1/12 + 1/12 = 1/4$.

The case that G satisfies $\min_{G' \in \overline{\Pi}_{n;>\epsilon}} \|\text{freq}_t(G) - \text{freq}_t(G')\|_1 \leq \delta_C$ can be analyzed analogously. In particular, if G satisfies this condition, then $\mathscr{T}_C$ accepts G with probability at most $1/12 + 1/12 + 1/12 = 1/4$.

Therefore, $\mathscr{T}_C$ accepts (rejects) G with probability at least $1 - 1/4 > 2/3$, if $\min_{G' \in \Pi_n} \|\text{freq}_t(G) - \text{freq}_t(G')\|_1 \leq \delta_C$ (if $\min_{G' \in \overline{\Pi}_{n;>\epsilon}} \|\text{freq}_t(G) - \text{freq}_t(G')\|_1 \leq \delta_C$). □

Note on the Complexity. Since the original tester $\mathscr{T}$ is a black-box tester, this may raise questions about computability and complexity of the canonical tester $\mathscr{T}_C$ from Theorem 3.11. By Definition 1.10, there exists an upper bound on the running time for every tester. Therefore, one can simulate $\mathscr{T}_n$ for any fixed input G of size n and all finite prefixes of infinite random bit strings that $\mathscr{T}_n$ may read. This allows to compute an upper bound on the query complexity of $\mathscr{T}_n$ on any input of size n. Furthermore, it allows to compute the probabilities required by the proof of Theorem 3.11 exactly. The *running time* of this canonical tester is possibly superlinear in n. However, the number of its *queries to the input* is bounded by $\hat{q}_\Pi(\epsilon, d)$, which is independent of n because $q_{\mathscr{T}}(\epsilon, d)$ is independent of n.

3.3.2. The Complement of Non-Trivially Testable Properties

First, we prove the second part of Theorem 3.2, i. e., the complement of every non-trivially testable property (see Definition 3.1) contains a hyperfinite subproperty, because the proof is easier and a subset of the first part's proof. Note that for a property that is *not* non-trivially testable, for any $\epsilon > 0$, we can accept all graphs of size $n \geq n_3$, where $n_3 = n_3(\epsilon)$ is a finite number such that there is no graph of size greater or equal to n_3 that is ϵ-far from having the property. For graphs of size smaller than n_3, one can simply read the whole graph to test if the graph satisfies the property or not.

The idea to prove the second part of Theorem 3.2 is to start with a graph $G \in \overline{\Pi}_{>\epsilon}$ (for some suitable ϵ) and to construct a $(q_\Pi, 1/6q_\Pi)$-blow-up graph H of G. By Theorem 3.11, the canonical tester accepts H with probability less than $1/3$ because its disk frequency vector is close to the disk frequency vector of G, which implies $H \in \overline{\Pi}$. Since all connected components of G have constant size and there exists an infinite number of $G \in \overline{\Pi}_{>\epsilon}$ that may be transformed, the claim follows. The graphs that are neither in Π nor ϵ-far from Π act as a kind of *slack* because they are contained in the complement $\overline{\Pi}_n$ and we can show that $H^{(n)} \notin \Pi_n$ when $G_n \in \overline{\Pi}_{n;>\epsilon}$.

3.12 Lemma (second part of Theorem 3.2). The complement of every non-trivially testable property Π of bounded-degree graphs contains an infinite $(0, k)$-hyperfinite subproperty Π', where k depends only on Π. ■

Proof. Since Π is non-trivially testable, by Definition 3.1, there exists $\epsilon > 0$ and an infinite set $N \subseteq \mathbb{N}$ such that for every $n \in N$, $\overline{\Pi}_{n;>\epsilon}$ is non-empty. Let $\epsilon > 0$ be such that $\overline{\Pi}_{>\epsilon}$ contains an infinite number of graphs. Let $\delta = 1/24t$, where $t := q_\Pi(\epsilon, d)$. Let $k = \hat{q}_\Pi(\epsilon, d) = t^{2t}$ the query complexity of a canonical tester of Π. Fix an arbitrary $n \in N$ such that $n \geq n_1$, where $n_1 = n_1(\delta, d, k)$ is the number given in Lemma 3.10. Let $G_n \in \overline{\Pi}_{n;>\epsilon}$ be an arbitrary graph in $\overline{\Pi}_{n;>\epsilon}$. Let $H^{(n)}$ be the (δ, k)-blow-up graph of G_n. Note that $H^{(n)}$ is $(0, k)$-hyperfinite.

Now we claim that $H^{(n)} \notin \Pi$. Assume on the contrary that $H^{(n)} \in \Pi$. By Lemma 3.10, $\|\text{freq}_k(G_n) - \text{freq}_k\big(H^{(n)}\big)\|_1 \leq 2\delta$. Therefore, by Theorem 3.11, the canonical tester for Π accepts G_n with probability at least $2/3$, which is a contradiction to the fact that $G_n \in \overline{\Pi}_{n;>\epsilon}$. The lemma follows by defining the set $\Pi' \coloneqq \{H^{(n)} : n \in \mathbb{N}\}$. □

3.3.3. Constant-Query Testable Properties

We now prove the first part of Theorem 3.2, i. e., every infinite testable property contains an infinite hyperfinite subproperty. The high-level idea is as follows: we start with an arbitrary graph $G_n \in \Pi_n$ and construct a blow-up graph $H^{(n)}$ of it. In contrast to the previous section, where the graphs that are neither in Π nor ϵ-far from Π act as a kind of *slack* because they are contained in the complement $\overline{\Pi}_n$, we need to show that $H^{(n)}$ is an element of Π. By the property of blow-up graphs as guaranteed by Lemma 3.10, the disk frequency vectors of G_n and $H^{(n)}$ are very close. This enables us to show that $H^{(n)} \notin \overline{\Pi}_{n;>\epsilon}$, i. e., $H^{(n)}$ is ϵ-close to Π_n by Theorem 3.11. However, the graph $H^{(n)}$ might not satisfy Π_n, i.e., $H^{(n)} \in \overline{\Pi}_n$. To prove the first part of Theorem 3.2, we would like to modify $H^{(n)}$ so that it becomes a member of Π_n. Since $H^{(n)}$ is ϵ-close to Π, this is possible by modifying at most ϵdn edges, but still, we do not know how this would affect the size of the connected components. Nevertheless, the resulting graph is (ϵ, k)-hyperfinite, which is the first step to prove our claim.

We start by showing that for any *fixed* ϵ and any graph G of a constant-query testable property Π, we can find another graph G' such that G' is (ϵ, s)-hyperfinite for some sufficiently large s and the disk frequency vectors of G and G' are close.

3.13 Lemma. Let $\delta, \epsilon, k > 0$, let $\epsilon' = \min\{\epsilon, \delta/(36d^{k+1})\}$ and let $n \geq n_4$ for $n_4 \coloneqq n_4(\epsilon, \delta, d, k)$ as defined in the proof. Let Π be a testable graph property with query complexity $q_\Pi = q_\Pi(\epsilon, d)$ and let $G \in \Pi_n$. Then, there exists $G' \in \Pi_n$ such that n_4

- G' is $(\epsilon, M_{d,\delta'/3,k'})$-hyperfinite, and
- $\|\text{freq}_k(G) - \text{freq}_k(G')\|_1 < \delta$,

where $\delta' = \min\{\delta, 1/(5c \cdot q_\Pi(\epsilon', d))\}$ and $k' = \max\{k, c \cdot q_\Pi(\epsilon', d)\}$ for some constant $c > 1$. ■

Proof. Let $n_4(\epsilon, \delta, d, k) \coloneqq \max\{n_1(\delta'/12, d, k), n_2(\epsilon', d)\}$, where n_1, n_2 are the numbers in the statements of Lemma 3.10 and Theorem 3.11, respectively. Let $t = c \cdot q_\Pi(\epsilon', d)$ for the constant $c > 1$ from Theorem 3.11. By definition, it holds that $t \leq k'$. Let H be the $(\delta'/12, k')$-blow-up graph of G. By Lemma 3.10 and our assumption that $n \geq n_4$, it holds that $\|\text{freq}_{k'}(G) - \text{freq}_{k'}(H)\|_1 \leq 2\delta'/12$, which implies that

$$\|\text{freq}_t(G) - \text{freq}_t(H)\|_1 \leq \frac{2\delta'}{12} \leq \frac{1}{12t}, \tag{3.1}$$

as t satisfies that $1/5t \geq \delta'$ and $t \leq k'$.

Let $\mathscr{T}_C$ be the canonical tester for Π with parameter ϵ' and corresponding query complexity $t \coloneqq \hat{q}_\Pi(\epsilon', d)$. Then by Theorem 3.11, $\mathscr{T}_C$ will accept H with probability at least $2/3$. This implies that H is ϵ'-close to Π. Let $G' \in \Pi$ such that H is ϵ'-close to G'. We claim that G' is the graph we are looking for.

First, we show that G' is $(\epsilon, M_{d,\delta'/12,k'})$-hyperfinite. Recall that by definition, H is composed of $\lfloor n/h \rfloor$ disjoint copies of a graph of size h and $n - h \cdot \lfloor n/h \rfloor$ isolated vertices, where $h \leq M_{d,\delta'/12,k'}$.

Reminder: Lemma 3.13

$$\delta' = \min\left\{\delta, \frac{1}{5c \cdot q_\Pi(\epsilon', d)}\right\}, \quad \epsilon' = \min\{\epsilon, \frac{\delta}{36d^{k+1}}\}, \quad t = c \cdot q_\Pi(\epsilon', d)$$

This implies that H is $(0, M_{d,\delta'/12,k'})$-hyperfinite. It follows that G' is $(\epsilon, M_{d,\delta'/12,k'})$-hyperfinite because we can remove at most $\epsilon' dn \leq \epsilon dn$ edges from G' to obtain a graph so that all connected components have size at most $M_{d,\delta'/12,k'}$.

Second, we prove that $\|\text{freq}_k(G) - \text{freq}_k(G')\|_1 \leq \delta$. Note that the bound given by inequality (3.1) implies $\|\text{freq}_k(G) - \text{freq}_k(H)\|_1 \leq 2\delta'/12 \leq \delta/6$, as $k \leq k'$ and $\delta \geq \delta'$. Now since H and G' are ϵ'-close to each other, by Lemma 3.6 we have that $\|\text{freq}_k(H) - \text{freq}_k(G')\|_1 < 6\epsilon' d^{k+1} \leq \delta/6$, where the last inequality follows from our setting of parameters. The claim follows by applying the triangle inequality. □

The above lemma only guarantees that for *every fixed* $\epsilon > 0$ and every graph $G \in \Pi_n$, one can find a graph $G_\epsilon \in \Pi_n$ that is $(\epsilon, M_{d,\delta',k'})$-hyperfinite (for δ' and k' as in Lemma 3.13). However, we cannot directly use G_ϵ to construct an infinite hyperfinite subproperty. Recall that a property Π of graphs is called hyperfinite if there exists a function $\rho : (0, 1] \to \mathbb{N}$ such that Π is $(\epsilon, \rho(\epsilon))$-hyperfinite for every $\epsilon > 0$. Now, for any $\epsilon' < \epsilon$, we cannot guarantee that after removing $\epsilon' dn$ edges from G_ϵ, one can obtain a graph that is the union of connected components of constant size. Furthermore, it is not guaranteed that $G_{\epsilon_1} \cong G_{\epsilon_2}$ if $\epsilon_1 \neq \epsilon_2$.

The idea of overcoming this difficulty is to start with the above hyperfinite graph $G_0 := G_\epsilon \in \Pi_n$ for some fixed $\epsilon > 0$; then we construct a sequence of graphs $(G_i \in \Pi_n)_{i \in \mathbb{N}}$ so that the constructed graph G_{i+1} is guaranteed to inherit hyperfinite properties from G_i. The key idea is to employ Lemma 3.7 and maintain the hyperfinite properties of G_i by causing only a small perturbation of its disk frequency vector. Choosing the parameters in this process carefully, we can maintain these hyperfinite properties for the whole sequence of graphs.

3.14 Lemma (first part of Theorem 3.2). Let Π be an infinite bounded-degree graph property that is testable with query complexity $q_\Pi(\epsilon, d)$. Then, there exists $\Pi' \subseteq \Pi$ such that

- Π' is an infinite subproperty of Π, and
- there exists a monotonically decreasing function $\rho : (0, 1] \to \mathbb{N}$ such that Π' is $(\epsilon, \rho(\epsilon))$-hyperfinite for every $\epsilon > 0$. ■

Proof. Let $X := \{|V| \mid G \in \Pi\}$ be the set of graph sizes in Π. Since Π is an infinite graph property, it holds that X is also an infinite set. We show there exists a monotonically decreasing function $\rho : (0, 1] \to \mathbb{N}$ such that for every $n \in X$, we can find a graph $H^{(n)} \in \Pi_n$ that is $(\epsilon, \rho(\epsilon))$-hyperfinite for every $\epsilon > 0$. This implies that the set $\Pi' = \{H^{(n)} : n \in X\}$ is an infinite ρ-hyperfinite property, which proves the lemma.

Let us now fix an arbitrary $n \in X$ and let $G \in \Pi_n$ be an arbitrary graph in Π_n. We let FindHyper$(G, \delta, \epsilon, k, \Pi_n)$ denote the graph G' that is obtained by applying Lemma 3.13 on $G \in \Pi_n$ with parameters δ, ϵ, k. We construct $H^{(n)}$ as follows.

Let $\epsilon_1 = 1/10$, $\delta_1 = 4\epsilon_1/d\log(4/3)$ and let $k_1 = 1$. If $n < n_5(d)$, where $n_5 := n_4(\epsilon_1, \delta_1, d, k_1) = n_4(1/10, 2/(5d\log(4/3)), d, 1)$ and n_4 is the number given in Lemma 3.13, then we simply let $H^{(n)} = G$, which is a finite graph of size at most n_5. In the following, we assume that $n \geq n_5$.

Let $G_0 = G$. We start by applying Lemma 3.13 to G_0 with parameters $\delta = \delta_1$, $\epsilon = \epsilon_1$ and $k = k_1$ to obtain a graph G_1 that is (ϵ_1, s_1)-hyperfinite, where

$$s_1 \coloneqq M_{d,\delta_1'/3,k_1'}, \delta_1' \coloneqq \min\left\{\delta_1, \frac{1}{5c \cdot q_\Pi(\epsilon_1', d)}\right\}, k_1' \coloneqq \max\{k_1, c{\cdot}q_\Pi(\epsilon_1', d)\}, \epsilon_1' \coloneqq \min\left\{\epsilon_1, \frac{\delta_1}{36d^{k_1+1}}\right\}.$$

We now iteratively construct a new graph G_{i+1} of size n from a graph G_i that is (ϵ_i, s_i)-hyperfinite, where $s_i \coloneqq M_{d,\delta_i'/3,k_i'}$. Let

$$\delta_{i+1} \coloneqq \delta_i/2, \qquad \epsilon_{i+1} \coloneqq \epsilon_i/2, \qquad k_{i+1} \coloneqq \max\{k_i, 10s_i d^{2s_i+1}/\epsilon_i\}.$$

We apply Lemma 3.13 to G_i with parameters $\epsilon = \epsilon_{i+1}$, $\delta = \delta_{i+1}$ and $k = k_{i+1}$ to obtain a graph G_{i+1} that is $(\epsilon_{i+1}, s_{i+1})$-hyperfinite, where $s_{i+1} \coloneqq M_{d,\delta_{i+1}'/3,k_{i+1}'}$, and

$$\delta_{i+1}' \coloneqq \min\left\{\delta_{i+1}, \frac{1}{5c \cdot q_\Pi(\epsilon_{i+1}', d)}\right\}, k_{i+1}' \coloneqq \max\{k_{i+1}, c{\cdot}q_\Pi(\epsilon_{i+1}', d)\}, \epsilon_{i+1}' \coloneqq \min\left\{\epsilon_{i+1}, \frac{\delta_{i+1}}{36d^{k_{i+1}+1}}\right\}.$$

Finally, we stop the process after the i'-th iteration such that $\epsilon_{i'} dn < 1$. We set $H^{(n)} = G_{i'}$. The pseudocode of the whole process is given in Algorithm 3.1.

Algorithm 3.1 Construction of $H^{(n)}$

procedure CONSTRUCT(G, Π_n)
  $G_0 \leftarrow G, \epsilon_1 \leftarrow 1/10,\ \delta_1 \leftarrow 4\epsilon_1/d\log(4/3), k_1 \leftarrow 1$
  $G_1 \leftarrow$FINDHYPER($G_0, \delta_1, \epsilon_1, k_1, \Pi_n$) ▷ apply Lemma 3.13
  $s_1 \leftarrow$SETSIZE($\epsilon_1, \delta_1, k_1, \Pi_n$)
  $i \leftarrow 1$
  while $\epsilon_i dn \geq 1$ **do**
    $\epsilon_{i+1} \leftarrow \epsilon_i/2, \delta_{i+1} \leftarrow \delta_i/2, k_{i+1} \leftarrow \max\{k_i, 10s_i d^{2s_i+1}/\epsilon_i\}$
    $G_{i+1} \leftarrow$FINDHYPER($G_i, \delta_{i+1}, \epsilon_{i+1}, k_{i+1}, \Pi_n$)
    $s_{i+1} \leftarrow$SETSIZE($\epsilon_{i+1}, \delta_{i+1}, k_{i+1}, \Pi_n$)
    $i \leftarrow i + 1$
  return $H^{(n)} \leftarrow G_i$
end procedure
procedure SETSIZE($\epsilon, \delta, k, \Pi_n$)
  $\epsilon' \leftarrow \min\left\{\epsilon, \frac{\delta}{36d^{k+1}}\right\},\ \delta' \leftarrow \min\left\{\delta, \frac{1}{5c \cdot q_\Pi(\epsilon', d)}\right\},\ k' \leftarrow \max\{k, c \cdot q_\Pi(\epsilon', d)\}$
  $s \leftarrow M_{d,\frac{\delta'}{3},k'}$
  return s
end procedure

Now we also note that by the construction and Lemma 3.13, it holds that for any $i \geq 0$,

$$\|\mathrm{freq}_{k_{i+1}}(G_{i+1}) - \mathrm{freq}_{k_{i+1}}(G_i)\|_1 < \delta_{i+1}.$$

By noting that $k_j \le k_{i+1}$ for any $j \le i+1$, we have that

$$\|\mathrm{freq}_{k_j}(G_{i+1}) - \mathrm{freq}_{k_j}(G_i)\|_1 \le \|\mathrm{freq}_{k_{i+1}}(G_{i+1}) - \mathrm{freq}_{k_{i+1}}(G_i)\|_1 < \delta_{i+1}.$$

Furthermore, we have the following claim, which is proved after the proof of this lemma.

3.15 Claim. It holds that $\|\mathrm{freq}_{k_j}(G_{i+1}) - \mathrm{freq}_{k_j}(G_j)\|_1 < 8\epsilon_j/(d\log(4/3))$ for all $j \le i+1$. ■

Now by the fact that $G_j \in \Pi_n$ is (ϵ_j, s_j)-hyperfinite, Claim 3.15 and Corollary 3.8, it follows that G_{i+1} is $(4\epsilon_j \log(4d/\epsilon_j), s_j)$-hyperfinite, for any $j \le i+1$.

In particular, let i' denote the index such that our algorithm outputs $G_{i'}$, i.e., $H^{(n)} = G_{i'}$. For any $\epsilon > 0$, let $j_\epsilon = \min\{i \mid 1 \le i \le i' \wedge 4\epsilon_i \log 4d/\epsilon_i \le \epsilon\}$. Then we define $\rho(\epsilon) := \max\{n_5(d), s_{j_\epsilon}\}$. By the above analysis, for any $n \in X$ with $n \ge n_5(d)$, we find a graph $H^{(n)} \in \Pi_n$ of size n that satisfies the following: for any $\epsilon > 0$, there exists j_ϵ such that by removing $(4\epsilon_{j_\epsilon} \log(4d/\epsilon_{j_\epsilon})) \cdot dn \le \epsilon dn$ edges, one can decompose $H^{(n)}$ into connected components each of which has size at most $s_{j_\epsilon} \le \rho(\epsilon)$. Here, it is important to note that j_ϵ is the same for any fixed ϵ and all $H^{(n)}$. Therefore, $\rho(\bullet)$ is the same function for all $H^{(n)}$), and it holds that $H^{(n)}$ is $(\epsilon, \rho(\epsilon))$-hyperfinite for any $\epsilon > 0$. □

Proof of Claim 3.15. Recall that $\epsilon_{i+1} = \epsilon_i/2$ and $\delta_{i+1} = \delta_i/2$ for all $i > 1$. We have

$$\|\mathrm{freq}_{k_j}(G_{i+1}) - \mathrm{freq}_{k_j}(G_j)\|_1 \le \sum_{\ell=j}^{i} \|\mathrm{freq}_{k_j}(G_{\ell+1}) - \mathrm{freq}_{k_j}(G_\ell)\|_1 \le \delta_j \sum_{\ell=j}^{i} \frac{1}{2^{\ell-j}} \le 2\delta_j,$$

where the first inequality follows from the triangle inequality and the second inequality follows from the convergence of the geometric series $\sum_{\ell=0}^{\infty} 2^{-\ell} = 2$. Since $\delta_j = \delta_1/2^{j-1}$, $\epsilon_j = \epsilon_1/2^{j-1}$ and $\delta_1 = 4\epsilon_1/d\log(4/3)$, it holds that $\delta_j = 4\epsilon_j/(d\log(4/3))$. □

3.4. Expander Subproperties

In the light of the previous result, a natural question is whether every testable infinite property that is not hyperfinite must contain an infinite subproperty that consists only of expander graphs or graphs that are close to an expander graph. Unfortunately, such a statement is not true as Theorem 3.3 shows.

3.3 Theorem. There exists an infinite graph property Π of bounded-degree graphs such that

- Π is testable (with two-sided error) with query complexity $\mathcal{O}(d/\epsilon^2)$,
- Π is not hyperfinite,
- every graph in Π differs in $\Omega(n)$ edges from every connected graph. ■

Proof of Theorem 3.3. We start by defining the graph property Π. It consists of all graphs $G = (V, E)$ with maximum degree d that have a single connected component with $\lceil |V|/2 \rceil$ vertices and the remaining $\lfloor |V|/2 \rfloor$ connected components are isolated vertices. We observe that Π is not hyperfinite as the big connected component may be an expander graph and so it requires to remove $\Omega(n)$ edges to partition it into small connected components. Furthermore, it is necessary to insert $\Omega(n)$ edges to make the graph connected, which is a necessary condition for having expansion greater than 0. Finally, we show that the property can be tested with query complexity $\mathcal{O}(d/\epsilon^2)$.

The algorithm consists of two stages. In the first stage, we sample $1000/\epsilon^2$ vertices uniformly at random and estimate the number of isolated vertices by averaging. We reject if this number differs from $\lfloor |V|/2 \rfloor$ by more than $\epsilon|V|/8$. In the second stage, we sample another $16/\epsilon$ vertices and perform, for every sampled vertex v, a BFS until we have explored the whole connected component of v or we have explored more than $12/\epsilon$ vertices. The tester rejects if it finds a connected component of size greater than 1 but smaller or equal to $12/\epsilon$. We may assume that the graph's size is larger than $24/\epsilon$ as otherwise, we can simply query the whole graph.

We now prove that the above algorithm is a tester for the property described above. Our analysis (in particular for the second stage) uses some ideas that were first introduced in an analysis of a connectivity tester in [GR02]. We first show that the tester accepts every $G \in \Pi$. By Hoeffding's inequality, the first stage of the tester approximates the number of isolated vertices in G with an additive error of $\epsilon|V|/8$ with probability at least $9/10$. If this approximation succeeds, the first stage of the tester does not reject. Furthermore, the second stage never rejects a graph $G \in \Pi$. Thus, the tester accepts with probability at least $9/10$. Next consider a graph that is ϵ-far from Π and begin with the following claim, which we prove after the proof of this theorem.

3.16 Claim. Let G be ϵ-far from Π. Then either the number of isolated vertices in G differs by more than $\epsilon|V|/4$ from $\lfloor |V|/2 \rfloor$ or there are more than $\epsilon|V|/12$ connected components of size at most $12/\epsilon$ that are not isolated vertices. ■

It remains to show that our tester rejects any G that is ϵ-far from Π. By Claim 3.16 we know that either the number of isolated vertices in G differs by more than $\epsilon|V|/4$ from $\lfloor |V|/2 \rfloor$ or G has at least $\epsilon|V|/12$ connected components of size at most $12/\epsilon$. In the first case, our algorithm rejects with probability at least $9/10$ as it approximates the number of isolated vertices with additive error $\epsilon|V|/8$ and rejects if the estimate differs by more than $\epsilon|V|/4$ from $\lfloor |V|/2 \rfloor$. In the second case we observe that for sufficiently large constant in asymptotic notation with probability at least $9/10$ we sample a connected component of size at most $12/\epsilon$. In this case our algorithm detects the component and rejects. Thus, with probability at least $9/10$ the algorithm rejects. The query complexity and running time of the algorithm are dominated by the second stage, which runs in $\mathcal{O}(d/\epsilon^2)$ time. □

Proof of Claim 3.16. Assume that the claim is not true and there is a graph G that is ϵ-far from Π, the number of isolated vertices in G differs by at most $\epsilon|V|/4$ from $\lfloor |V|/2 \rfloor$ and there are at most $\epsilon|V|/12$ connected components of size at most $12/\epsilon$ that are not isolated vertices. We will argue that in this case, we can modify at most ϵdn edges to turn G into a graph that has Π, which is a contradiction. We start with the connected components that are not isolated vertices. We can add a single edge to connect two such components. However, we must make sure that we are not violating the degree bound. If both connected components have a vertex of degree at most $d-1$, we can simply add an edge to connect them. If all vertices of a connected component have degree d then the component contains a cycle. We can remove an edge from the cycle without destroying connectivity. Thus, we need to modify at most 3 edges to connect two connected components. We observe that there are at most $\epsilon n/12$ connected components of size more than $12/\epsilon$ and so there are at most $\epsilon n/6$ connected components that are not isolated vertices. We can create a single connected component out of them by modifying $\epsilon n/2$ edges.

Our previous modifications did not change the number of isolated vertices in G, so it still differs by at most $\epsilon|V|/12$ from $\lfloor |V|/2 \rfloor$. If there are too many isolated vertices, we can connect each of them to the big connected component with at most 2 edge modifications resulting

in at most $\epsilon n/2$ modifications. If there are too few isolated vertices, we need to disconnect vertices from the big connected component. For this purpose consider a spanning tree T of the connected component. We will remove a leave of T. This can be done with d edge modifications and does not change connectivity. Thus we can create exactly $\lfloor |V|/2 \rfloor$ isolated vertices using at most $\epsilon dn/4$ modifications. Overall, the number of modifications is at most ϵdn, which proves that the graph was not ϵ-far from Π, which is a contradiction to our assumption. □

Since an expander graph is connected, it also follows that this property contains no graphs that are close to expander graphs.

Consider the disks of graphs from the property Π defined in the proof of Theorem 3.3. Recall that the graphs from the property consist of a connected graph on $\lceil |V|/2 \rceil$ vertices and $\lfloor |V|/2 \rfloor$ isolated vertices. We may view graphs in Π as the union of two graphs G_1 and G_2 of roughly the same size that satisfy two different properties: G_1 is connected and the G_2 has no edges. The disks of these graphs have two interesting properties: (i) no disk in G_1 occurs in G_2 and vice versa, and (ii) their centers cannot be adjacent in any graph. If G_1 and G_2 have the above properties then this means that the disks cannot *mix* in any connected component of another graph. Thus, we know whether they are supposed to come from G_1 or G_2, which is helpful to design a property tester. We remark that this phenomenon can also show up for other disks like, e. g., if G_1 is 4-regular and G_2 is 6-regular. We believe that understanding this phenomenon is important for a characterization of testable properties in bounded-degree graphs as we can use it to construct other constant-query testable properties in a similar way as above. This motivates the following definition.

3.17 Definition. We call two k-disk isomorphism types D_1, D_2 with roots u_1, u_2 *incompatible* if there exists no graph in which two adjacent vertices u_1 and u_2 have k-disk type D_1 and D_2, respectively. ■

3.5. Partitioning Theorem

The fact that there are constant-query testable properties that are composed of other properties with sets of incompatible disks (see Definition 3.17) leads to the question if we can always decompose the vertex set of a graph into sets such that the disk types behave *similarly* within each set. A simple partition would be to divide the vertex set according to disk isomorphism types. But such a partition is meaningless. In the light of previous work like [Alo+09], we consider the case that a partition has to have only a small fraction of the edges between the partition classes. We would like to obtain a partition into sets $S_1, \dots, S_r$ and a separator set T, such that no edges are between S_i and S_j for any $i \neq j$ and T is of small size. The next question is to specify what it means to behave *similarly*. One such specification is to ask that the disk distribution inside the partition is stable for every subset. Obviously, this cannot always be the case unless there is only one disk isomorphism type. Instead, we are only looking at sets that do not have too many outgoing edges. For these subsets we can show that they always have roughly the same disk distribution as their partition.

3.4 Theorem. Let $G = (V, E)$ be a bounded-degree graph. For every $k \geq 0$ and every $\delta \in (0, 1]$ there exists a function $f(\delta, d, k)$ such that V can be partitioned into $r \leq f(\delta, d, k)$ subsets $S_1, \dots, S_r$ and a set T such that

- for every $i \neq j$ there are no edges between S_i and S_j,
- $|T| \leq \delta d|V|$,

- and for every i and every subset X of S_i with $\text{cond}(X) \leq \delta^2$ it holds that

$$\|\text{freq}_k(X \mid G) - \text{freq}_k(S_i \mid G)\|_1 \leq 3\delta. \qquad \blacksquare$$

Proof. We will first construct a partition of V into sets $A_1, \ldots, A_t$ for some (sufficiently large) value of t and a set T so that (i) $|T| \leq \delta d|V|$ and (ii) there are no edges between any pair of A_i and A_j. Then we construct each set S_i as union of some of the sets A_j. Finally, we prove that the S_i satisfy the third property (the first two follow from the construction of the A_j).

We start with $T = \emptyset$ and $W = V$. Let A be a subset of vertices of W with $\text{cond}(A) \leq \delta$. We may assume that A contains no proper subset with this property (otherwise, we take this subset). We put the neighbors of vertices from A that are not in A into the set T and remove T from W. We store the set A as A_1 and remove it from W. We then repeat this process as long as possible, computing the sets $A_2, A_3, \ldots$. We observe that every vertex is removed at most once from W. Whenever we remove a set A_i, we move at most $\delta d|A_i|$ neighbors into T since $\text{cond}(A_i) \leq \delta$. Hence, $|T| \leq \delta d|V|$. Furthermore, we observe that by construction there are no edges between A_i and A_j for any $i \neq j$.

It remains to construct the sets S_i. For this purpose, we put a δ-net over the space of all disk frequency vectors, i. e., we compute a smallest set $N = \{v_1, \ldots, v_{|N|}\}$ of frequency vectors such that for every frequency vector there exists a vector in N within ℓ_1-distance at most δ). We observe that $|N|$ is a function of k, d and δ (similarly to the proof of Lemma 1.6). We then define S_i to be the union of all A_j that have v_i as the closest vector to their frequency vector. It remains to prove that the S_i satisfy the third property for δ^2. For this purpose consider an arbitrary subset $X \subseteq S_i$. We consider $X \cap A_j$ for the sets A_j whose union S_i is. If $X \cap A_j \neq A_j$ then we know that $\text{cond}(X \cap A_j) > \delta$. Recall that the edges that leave $X \cap A_j$ either go to $A_j \setminus X$ or to T, where $X \cap T = \emptyset$. If $\text{cond}(X) \leq \delta^2$, then it holds that at most a δ-fraction of the elements from X can be from a subset A_j with $\text{cond}(X \cap A_j) > \delta$. This is true as otherwise the number of edges crossing X and $V \setminus X$ is at least $\delta|X| \cdot \delta d$, which contradicts the assumption that $\text{cond}(X) \leq \delta^2$. Let J be the set of all indices j such that $A_j \cap X = A_j$. Hence we get

$$\begin{aligned}\text{freq}_k(X \mid G) &= \sum_j \sum_{x \in X \cap A_j} \frac{\text{freq}_k(x \mid G)}{|X|} \\ &= \frac{1}{|X|}\left(\sum_{j \in J} \sum_{x \in X \cap A_j} \text{freq}_k(x \mid G) + \sum_{j \notin J} \sum_{x \in X \cap A_j} \text{freq}_k(x \mid G)\right).\end{aligned}$$

Now let us define $X_1 = \{x \in X \mid x \in A_j, j \in J\}$ and $X_2 = X \setminus X_1$. We know that $|X_2| \leq \delta|X|$. We also observe that

$$\left\|\frac{1}{|X_1|}\sum_{x \in X_1} \text{freq}_k(x \mid G) - \text{freq}_k(S_i \mid G)\right\|_1 \leq \delta \quad \text{and} \quad \left\|\frac{1}{|X_2|}\sum_{x \in X_2} \text{freq}_k(x \mid G) - \text{freq}_k(S_i \mid G)\right\|_1 \leq 2$$

since all frequency vectors have ℓ_1-norm 1. It follows that

$$\begin{aligned}&\left\|\text{freq}_k(X \mid G) - \text{freq}_k(S_i \mid G)\right\|_1 \\ &= \left\|\frac{1}{|X|} \cdot \Big(\sum_{x \in X_1} \text{freq}_k(x \mid G) + \sum_{x \in X_2} \text{freq}_k(x \mid G)\Big) - \text{freq}_k(S_i \mid G)\right\|_1\end{aligned}$$

$$
\begin{aligned}
&= \Big\| \frac{1}{|X|} \cdot \Big(\sum_{x \in X_1} \operatorname{freq}_k(x \mid G) - |X_1| \cdot \operatorname{freq}_k(S_i \mid G) \\
&\qquad + \sum_{x \in X_2} \operatorname{freq}_k(x \mid G) - |X_2| \cdot \operatorname{freq}_k(S_i \mid G) \Big) \Big\|_1 \\
&\leq \frac{|X_1|}{|X|} \cdot \delta + \frac{|X_2|}{|X|} \cdot 2 \\
&\leq 3\delta.
\end{aligned}
$$

□

4. Testing Constant-Query Testable Properties in Random-Order Streams

Different types of sublinear algorithms share a lot of common ground. For example, techniques from property testing can often be applied in the design of local algorithms and vice versa. However, even models with different notions of complexity relate. In this chapter, we show that property testers for general graphs that have constant *query* complexity essentially subsample disks, and one-sided error testers imply property testers for random order streams that have logarithmic *space* complexity (measured in bits, i. e., a constant number of words). The results presented in this chapter are joint work with Artur Czumaj, Pan Peng and Christian Sohler [Czu+20].

4.1. Introduction

Streaming algorithms access their input as a stream. In particular, streaming algorithms for graphs can read an input graph as a stream of its edges only. Since storing the whole stream would reduce this setting to the classical setting where an algorithm has random access to its input, one objective of streaming algorithms is to minimize their space complexity. Streaming models have different flavors; e. g., streams can be *insertion-only*, *dynamic* (i. e., insertions and deletions) or *turnstile* (i. e., insertions and deletions do not need to be valid with respect to the current graph), and the order of edges can be *adversarial* or *random*. Streaming algorithms can be allowed only a *single pass* over the stream or *multiple passes*. In the following, we only consider the most popular variant, which is single pass algorithms and the variants mentioned above.

The setting of streaming is natural in a couple of scenarios. For example, imagine that we want to process a large data set stored on some external storage, but we only have limited computational resources available, say, a single machine for computing only. It would be very costly to provide random access to the external memory, and even if we would, the space on our local machine would prevent us from storing more than a tiny part of the input. However, if our algorithm would only read the input sequentially without storing it locally, we could simply *stream* the input to our local machine. In another setting, the input might be generated in real time and storing more than a few seconds might be infeasible because of the amount of data being generated.

4.1.1. Results in This Chapter

We relate property testers in two models: The first model is the *random-neighbor* query model, which is a special case of the general model. The set of input graphs is the set of all graphs, and a property tester obtains access to the input graph through random access to its adjacency list. In particular, a property tester can specify a vertex and gets an independently and uniformly

random neighbor in return. The second model is the random-order streaming model, where the stream is a uniformly random permutation of the edges. We show that one-sided error property testers with constant query complexity in the random-neighbor model give raise to one-sided error property testers for random-order streams.

4.1 Theorem. Every property that is constant-query testable with one-sided error in the random-neighbor model is also testable with one-sided error and space complexity $\mathcal{O}(\log n)$ in random-order streams. ■

In the *random-edge* model, the property tester can use an additional query type to sample edges independently and uniformly at random. We obtain a similar result for this model.

4.2 Theorem. Every property that is constant-query testable with one-sided error in the random-edge model is also testable with one-sided error and space complexity $\mathcal{O}(\log n)$ in random-order streams. ■

Note that our definition of property testing (see Definition 1.10) does not require testers to be uniform. This is crucial here because one can, e. g., reduce an EXPSPACE-complete problem to a constant-query testable graph property.[1]

In order to prove the results above, we first obtain another result that states that every (one-sided error or two-sided error) property tester in the random-neighbor model can be reduced to a canonical algorithm that subsamples disks.

4.3 Theorem (informal; cf. Theorem 4.15). If a property $\Pi = (\Pi_n)_{n\in\mathbb{N}}$ is testable with $q = q(\epsilon)$ queries in the random-neighbor model or the random-edge model, then it can also be tested by a canonical tester that subsamples $\hat{q}$ randomly chosen q-disks and accepts if and only if the explored subgraph does not contain any (forbidden) graph $F \in \mathcal{F}$, where $\hat{q}$ depends only on q, and $\mathcal{F}$ is a family of rooted graphs such that each graph $F \in \mathcal{F}_n$ is the union of $\hat{q}$ many $\hat{q}$-disks. ■

We prove Theorem 4.1 for the random-neighbor model in Section 4.4 and Theorem 4.3 in Section 4.3. The proofs of Theorem 4.1 for the random-edge model and of Theorem 4.2 are very similar, and the details of the necessary changes are described in Section 4.4.5.

4.1.2. Property Testing and Streaming

Algorithms for adversarial streams are very versatile, but it is known that solving many problems on graphs requires at least $\Omega(n)$ space [HRR99]. The probably most extensively studied graph problem in streaming is maximum matching. Goel, Kapralov, and Khanna [GKK12] and Kapralov [Kap13] showed that there is no $(1+\epsilon)$-approximation that requires only sublinear space in insertion-only streams, and Konrad [Kon15] showed that even a $\mathcal{O}(n^\epsilon)$-approximation requires $\Omega(n^{3/2-4\epsilon})$ space in turnstile streams. Assadi, Khanna, and Li [AKL17] proved that even an approximation of a maximum matching's *size* in insertion-only streams requires $\omega(n)$ space. For special classes like graphs with bounded arboricity, better bounds are known [BS15; Chi+16; MV16; Cor+17; Esf+18].

As a result, much work has been devoted to designing *semi-streaming* algorithms, which may use $\mathcal{O}(n \operatorname{poly}(\log n))$ space. These algorithms are not very interesting for sparse graphs,

[1] For example, consider the problem whether two regular expressions generate the same language, which is EXPSPACE-complete. Construct the following graph property: a graph G is in the property if and only if the two regular expressions represented by the bit-string encoding of n are equivalent. A tester for this property does not need to query the graph, but if it is uniform, it needs to solve the decision problem on n.

as superlinear space allows for storing the whole graph. This has lead to another branch in streaming algorithms that analyzes random order streams (see, e. g. [CCM08; KMM12; KKS14; Mon+17; PS18; Ass+19] and the survey by McGregor [McG14]), where the stream is a random permutation of the edges of an input graph. Following the thread of matching problems, Kapralov, Khanna, and Sudan [KKS14] proved that one can estimate the size of a maximum matching up to a polylogarithmic factor in random-order streams using $\text{poly}(\log n)$ space. On the other hand, Chakrabarti, Cormode, and McGregor [CCM08] proved that solving graph connectivity requires $\Omega(n)$ space even in random order streaming, which is close to the (tight) lower bound $\Omega(n \log n)$ for adversarial order insertion-only streams by Sun and Woodruff [SW15].

Turning to property testing, we need to point out the differences between the classical setting of testing in query models (like the bounded-degree model or the general model) and testing in streams. Most notably, the primary measure of complexity differs. In query models, we count queries to the oracle, which abstracts from counting the number of input bits read. Streaming algorithms read the whole input, so there is no point in analyzing this measure of complexity. However, streaming algorithms are not allowed to store the whole input, whereas property testing algorithms may store everything they ever learned about the input. Another difference is obviously the access to the input. In a query model, an algorithm may query arbitrary parts of the input in an adaptive fashion (although it has to start from zero knowledge), while a streaming algorithm has to accept an adversarial or random order of the input.

Neither regime is strictly more powerful than the other. For example, counting the number of edges is trivial in basically any streaming model, while it requires $\Omega(\sqrt{n})$ queries in the general model [GR06]. On the other hand, the query complexity of testing connectivity is constant in the bounded-degree model [GR97] as well as the general model [PR99], but testing connectivity in adversarial order insertion-only streams has space complexity between $\Omega(n^{1-8\epsilon})$ and $\mathcal{O}(n^{1-\epsilon \operatorname{poly}(\log n)})$ [HP16].

Huang and Peng [HP16] initiated the study of property testing in adversarial order dynamic streams by proving that connectivity, k-vertex connectivity, k-edge connectivity and cycle-freeness in general graphs as well as bipartiteness in planar graphs admit streaming property testers with space complexity $\mathcal{O}(n^{1-c\epsilon} \cdot \text{poly}(\log n))$ (for some universal $c > 0$). They also showed that the dependence on n is basically tight up to logarithmic factors for connectivity, cycle-freeness and bipartiteness in planar graphs. Turning the focus to random order streams, Monemizadeh et al. [Mon+17] proved that every property tester with constant query complexity in the bounded-degree model can be transformed into a streaming property tester with space complexity $\mathcal{O}(\log n)$. Constant-query property testers in bounded-degree graphs can be reduced to approximating the disk distribution of the graphs. In [Mon+17], the authors observe that one can estimate this distribution in random order streams. In particular, they show that one can simulate a BFS from some start vertex and collect adjacent edges when they appear in the stream. Since some edges of the BFS may arrive before any of their incident nodes are known, the collected graph may be incomplete. However, since the stream's order is uniformly random, the authors are able to infer the conditional probabilities of actually having a fixed disk type Δ when observing only some subgraph of it. Peng and Sohler [PS18] generalized their techniques and showed how to estimate the number of connected components and the weight of a minimum spanning tree in general graphs and the size of a maximum independent set in planar graphs with space complexity $\mathcal{O}(\log n)$ in random-order streams. Although the number of k-disk types and the size of k-disks may depend on n in general graphs, this can be

achieved by dividing the stream into two phases if one is only interested in k-disks of constant size (i. e., vertices of high degree can be ignored). Note that there is still a non-trivial difference to the bounded-degree model because large k-disks still *exist* in the graph. The first phase of their algorithm runs a more sophisticated version of BFS in the stream, and the second phase checks whether all relevant edges have been observed. If the length of the two phases are tuned accordingly, a witness for this event will be found during the second phase with large probability.

4.1.3. Open Problems and Future Work

We discuss all open problems related to Chapters 2 to 4 in this section. The main technical definition that supports all of the results in these chapters is the definition of disks. This is not a coincidence, as all constant-query testers can be canonized and turned into algorithms that (sub-)sample disks and use a dictionary to lookup their answers (see Theorems 3.11, 4.15, 4.24 and 4.33). The most begging open problem with respect to constant-query testers is probably to give a full characterization of constant-query testable properties in bounded-degree graphs. Although the technical definition of canonical testers is not a very insightful characterization on its own, we can obtain interesting results such as Theorem 3.2 by mixing in graph-theoretic results like Lemma 1.6. Similar results or results like Theorem 3.4 could be helpful for deriving a full characterization. In general, it is an interesting and probably rewarding challenge to obtain more results that have a close connection to disk frequency vectors.

4.4 Problem. Obtain a characterization of constant-query testable properties in the bounded-degree model that provides structural insight; study the connection between the combinatorial properties of graphs and their k-disk frequency vectors. ■

One might also think about *problematic* classes of graphs other than expander graphs. For example, consider an expander graph with girth $\Omega(\log n)$. Applying Theorem 2.3, we can modify the graph to obtain a graph that is far from being an expander. This graph seems somewhat similar in nature to the graph constructed in the proof of Theorem 3.3 (which states that there is a non-trivial property that is not constant-query testable and contains no expander graphs) in the sense that there is at least one expander subgraph but the whole graph is far from being an expander. However, graphs in this property still contain expanders as subgraphs. It would be interesting to know whether we can find a property that is not constant-query testable and lacks any connection to expander graphs in a stronger sense. One might also try to work towards this from the opposite direction by considering the role of sparse cuts. Hyperfinite graphs have quite strong guarantees regarding the existence of sparse cuts, but weaker assumptions might hold for all graphs of constant-query testable properties.

4.5 Problem. Find a non-trivial property that is not constant-query testable but does not contain *expander-like* subgraphs; study the role of small cuts with respect to constant-query testable properties. ■

Our knowledge becomes even more sparse when we turn our focus to general graphs. While it seems that not as many properties as in the bounded-degree model are constant-query testable, we are mostly clueless why they are testable or not on a higher level. For example, we do not know whether two planar graphs that have the same k-disk count vectors are (close to being) isomorphic, or whether hyperfiniteness or planarity can be tested with constant query

complexity[2]. Many of these questions seem to be harder to answer than for bounded-degree graphs because we do not know suitable techniques to exploit the fact that a constant-query algorithm can only see local bits of the potentially global view of a k-disk in a general graph.

4.6 Problem. Prove or disprove that if two planar graphs (without bound on the vertex degrees) have the same k-disk count vectors, then they are (close to) isomorphic. ■

Following the line of [Czu+11; Ede+11; Ito16], it would also be interesting to find or characterize properties that are constant-query testable for interesting subclasses such as (general) planar, bounded-arboricity, bounded-treewidth or power-law graphs. In these classes, the structure or the number of vertices whose k-disks have super-constant size is somehow restricted. Therefore, more constant-query testable properties might exist for these classes and finding such properties might be easier due to the additional structural guarantees (e. g., compare the results from [BKN16] with the results from [NS11]).

4.7 Problem. Study constant-query testable properties in general graphs that belong to some common class of graphs. ■

Another interesting direction is to take advantage of insights and techniques from property testers to design other algorithms. The results of [Mon+17; PS18] and Theorem 4.1 transform constant-query testers into $\mathcal{O}(\log n)$-space streaming testers. The area of local algorithms (see, e. g., [Rub+11; Alo+12; Man+12; MV13; LRR14; LRR16; Lev+17; LRY17; FPV18; LL18]) is also closely connected to property testing. Increasing the conceptual distance to property testing, Huang and Peng [HP16] and Henzinger and Peng [HP19] apply techniques from property testing to dynamic graph algorithms. Since data analysis is heavily parallelized nowadays, it seems appropriate to also reach into this direction. For example, Łącki et al. [Łąc+19] describe how to perform random walks, a crucial tools in property testing, in the MPC model. While property testing itself might not be a frequent application in data analysis, related learning tasks are, and sublinear techniques can hopefully provide a significant speedup.

4.8 Problem. Study how insights and tools form property testing can be applied in parallel computation settings. ■

4.1.4. Overview of the Analysis

The result about constant-space streaming algorithms for bounded-degree graphs by Monemizadeh et al. [Mon+17] is obtained by noting that any constant-query complexity tester basically estimates the distribution of local neighborhoods of the vertices (see Theorem 3.11) and emulating any such algorithm on a random-order graph stream using constant space. Unfortunately, this approach inherently relies on the assumption that the input graph has bounded degree. This limitation comes from two ends: on one hand, there has not been known any versatile description of testers for constant-query testable graph properties of general graphs, and on the other hand, the streaming approach from [Mon+17] relies on a breadth-first-search-like graph exploration that is possible (with constant space) only when the input graph has no high-degree vertices.

One important reason why the earlier approaches have been failing for the model of general graphs was our lack of understanding of constant-query testers in general graphs and the lack

[2]Note that not all planar graphs are hyperfinite in general graphs because planarity only guarantees the existence of $\mathcal{O}(\sqrt{n})$ separator *vertices*, but, e. g., there is no way to remove ϵm edges from a star to decompose it into constant-size connected components.

of techniques to appropriately emulate offline algorithms allowing many high-degree vertices.

A General and Simple Canonical Tester. To derive a canonical tester for constant-query testable properties in the random-neighbor model, we introduce the process *q-random BFS* (q-RBFS): It starts from any specified vertex v, and then performs a BFS-like exploration of depth q that is restricted to visiting at most q random neighbors at each step (see Definition 4.12 for the formal definition). We call the subgraph obtained by a q-RBFS a *q-bounded disk*. With the notion of q-RBFS and q-bounded disks, we are able to transform every constant-query tester for properties of general graphs into a *canonical tester* that works as follows: it samples q random vertices, performs a q-RBFS from each sampled vertex, and rejects if and only if the (non-induced) subgraph it has seen (which is a union of q-bounded disks) is isomorphic to some member of a family $\mathcal{F}$ of forbidden subgraphs (see Theorems 4.3 and 4.15). Furthermore, such a canonical tester preserves one-sided error, while the query complexity blows up exponentially. We believe that the exponential blow-up is necessary, even for bounded-degree graphs, as adaptivity is essential for property testing in sparse graphs [RS06; CG18]. In contrast, it is known that in the dense model a quadratic blow-up of the query complexity of canonical testers is sufficient [GT01].[3]

Canonical testers provide us a systematic view of the behavior of constant-query testers in the random-neighbor model. They further tell us that in order to test a constant-query testable property Π, it suffices to estimate the probability that some forbidden subgraph in $\mathcal{F}$ is found by a q-RBFS starting from a uniformly random vertex. Slightly more formal, we define the *reach probability* of a subgraph $F \in \mathcal{F}$ to be the probability that a q-RBFS starting from a uniformly randomly chosen vertex v sees a graph that is isomorphic to F. If we can estimate these reach probabilities in random order streams, then we can also test Π accordingly.

The problem with this approach is that it is hard to estimate the reach probabilities of subgraphs in $\mathcal{F}$. The main challenge here is that a forbidden subgraph $F \in \mathcal{F}_n$ may be the union of two or more subgraphs obtained from different q-RBFS, which may intersect with each other.

A Refined Canonical Tester. To cope with the challenge of estimating the reach probabilities of subgraphs in $\mathcal{F}$, we decompose each forbidden subgraph $F \in \mathcal{F}_n$ into all possible sets of intersecting q-bounded disks whose union is F and then try to recover F from these sets. In order to recover F from such a decomposition, we have to identify *vertices that are contained in more than one q-bounded disk of F*.

We refine the analysis of the canonical tester and separate the q-bounded disks explored by each q-RBFS and keep track of their intersections (see Theorem 4.24). We first observe that for every input graph G and every ϵ, there exists a *small fixed set* $V_\alpha \subseteq V$ of all vertices whose probability to be visited by a random q-RBFS from a random vertex exceeds some small threshold α (depending on q and ϵ, but independent of n). In other words, with constant probability, the subgraphs explored by multiple q-RBFS in the canonical tester will only overlap on vertices from V_α. Furthermore, we prove that the degree of all vertices in V_α is at least linear in n, and with constant probability, two random q-RBFS subgraphs will not share any edge. Since V_α has constant size, each q-bounded disk can be viewed as a *colored* q-bounded disk *type*

[3] In another line of research, trade-offs between adaptivity and non-adaptivity have been investigated by Canonne and Gur [CG18].

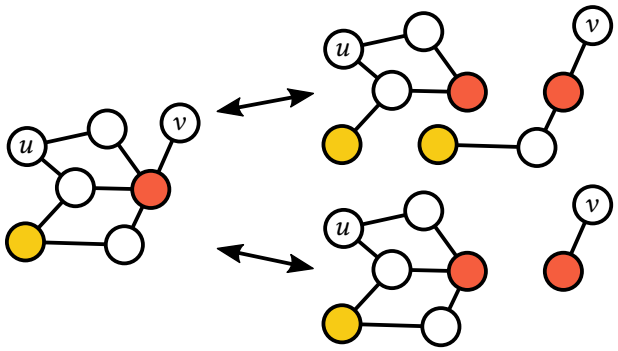


Figure 4.1.: Consider the graph on the left, which can be decomposed into colored 3-bounded disk types (which are rooted at u and v in this example) in more than one way. However, it is always possible to recover the original graph by identifying vertices of the same color. Furthermore, every mapping is bijective because every color is assigned at most once per disc. If the colored vertices correspond to the vertices in V_α, every forbidden graph $F \in \mathscr{F}_n$ from Theorem 4.15 corresponds to a decomposition into edge-disjoint colored q-bounded disks $F' \in \mathscr{F}'_n$ in Theorem 4.24, which intersect only at colored vertices.

such that each vertex in V_α is assigned a unique color from a constant-size palette. This way, it is possible to reversibly decompose each $F \in \mathscr{F}_n$ into a multiset of colored q-bounded disk types (actually, there may be many such multisets for each F): since the q-bounded disks that are explored by different q-RBFS intersect only at vertices in V_α, F is obtained by identifying vertices of the same color. See Fig. 4.1 for an example.

These properties are crucial to describe the forbidden subgraphs in terms of the graphs seen by the q many q-RBFS that the canonical tester performs plus a *constant-size* description of their interaction, i. e., how they overlap.

Simulation in the Streaming. In the streaming, in order to simulate q-RBFS, it is natural to consider the following procedure called *q-stream collect* (q-SC, see Algorithm 4.2) to explore the subgraph surrounding any specified vertex. It maintains a connected component C that initially contains only the start vertex. Whenever it reads an edge that connects to the current C and the augmented component may be observed by a run of q-RBFS, it adds the edge to C.

Note that one important feature of random-order streams is that we would see the right exploration (as in the query model) with constant probability, while it is challenging to verify if the subgraph we collected from the stream is indeed the right exploration (cf. [Mon+17; PS18] for an alternative discussion of this problem). In our setting, as we mentioned, another technical difficulty is to analyze whether subgraphs found by running the stream procedure multiple times *intersect* in exactly the same way as the q-bounded disks that are found by q-RBFS.

With the refined canonical tester, which specifies how different q-RBFS procedures intersect, we are able to simulate one-sided error constant-query testers in the random-neighbor model for general graphs in the random-order streaming model. Since the considered property Π is one-sided error testable in the random-neighbor model, it suffices to detect a forbidden subgraph F in the family $\mathscr{F}$ corresponding to Π with constant probability. That is, it suffices to

show that if the graph is far from having the property, then for any forbidden subgraph H that can be reached by the canonical tester with probability p, it can also be detected by *multiple* q-SC subroutines with probability at least cp for some suitable constant c.[4]

In order to do so, we first decompose the forbidden subgraphs that characterize the property into colored subgraphs, where each subgraph corresponds to a run of q-RBFS and vertices in V_α are colored with a unique color. Then, we prove that for a sufficiently large sample of vertices, the q-SC subroutines starting from these sampled vertices will collect, for each colored subgraph H, at least as many instances of H as the canonical property tester sees. Suppose that the input graph is far from the property. Since the subgraphs observed by the canonical tester intersect only at vertices in V_α (i. e., colored vertices), with constant probability it is possible to stitch a forbidden subgraph by identifying vertices of the same color in the analysis.

The analysis of this procedure is two-fold. First, we show that if a single run of q-RBFS from v sees a fixed colored q-bounded disk type with probability p (where the colored vertices are V_α), then a single run of q-SC from v sees this disc type with probability cp for some suitable constant c (see Corollary 4.28). The second step (which is the main technical part) is to show that if the probability that a q-RBFS from a random vertex sees a colored q-bounded disk type Δ is p, then with constant probability, for a sufficiently large sample set S, the calls to q-SC from vertices in S will also see a q-bounded disk type Δ, even though there are intersections from different q-SCs (see Lemma 4.29). Then we can show that if the input graph is far from the property, with constant probability, we can stitch the colored q-bounded disks to obtain a forbidden subgraph $F \in \mathscr{F}$ (see Theorem 4.1).

Finally, we remark that colors are only used in the analysis as the streaming algorithm can identify intersections of multiple q-SC by the vertex labels. However, the colors are crucial to the analysis: without colors, we cannot guarantee that the q-bounded disk types found by multiple q-SCs can be stitched in the same way as the q-bounded disk types found by q-RBFS. Here is an example: Consider some constant-query testable property Π such that the set of forbidden subgraphs $\mathscr{F}$ contains a graph F that is not a subgraph of any single q-bounded disk type (i.e, it is the union of at least two intersecting q-bounded disk types). For the sake of illustration, a concrete example is provided in Fig. 4.2. In order to reject, the canonical property tester needs to find at least two intersecting q-bounded disks such that their union contains F as a subgraph. However, even if we bound, for each *uncolored* q-bounded disk type Δ, the probability that q-SC finds Δ by some constant fraction of the probability that q-RBFS finds Δ, this is not sufficient to conclude that the probability that multiple q-SCs find a copy of F is bounded by a constant fraction of the probability that multiple q-RBFS find a copy of F. The reason is that q-SC might only find copies of Δ that are not intersecting, while q-RBFS might tend to find copies of Δ that intersect. Again, see Fig. 4.2 for an example. Therefore, we need to preserve, for each q-bounded disk type Δ, the information which of the corresponding vertices in the input graph are likely to be contained in more than one q-RBFS for the analysis.

[4]Note that this is not sufficient for simulating two-sided error testers. Let us take the property connectivity (which is two-sided error testable in the random-neighbor model) as an example. If the input graph is a path on n vertices, then a q-RBFS will detect a forbidden subgraph (i.e., a path of constant length that is not connected to the rest) corresponding to connectivity with zero probability, while a q-SC might see a forbidden subgraph with high constant probability. That is, in order to test connectivity, we need to be able to approximate the *frequencies* of the forbidden subgraphs, for which our current techniques fail.

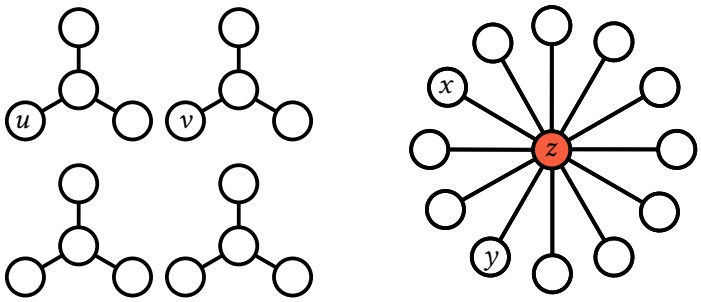


Figure 4.2.: The above graph, which is composed of 3-stars and a $\omega(1)$-star with root z and which should be thought of as a subgraph of some larger graph, illustrates the need for colors in our analysis of the streaming property tester. Although the 2-bounded disks of u, v x and y are all 3-stars (with constant probability over the randomness of the neighbor queries), exploring u and v by q-RBFS does not result in finding a 6-star, while it is likely to find a 6-star by exploring x and y. Even if we prove that the probability that a q-SC finds uncolored 3-stars is lower bounded by some constant fraction of the probability that q-RBFS finds uncolored 3-stars, we still cannot rule out that q-SC might tend to find leaves of the small stars (like u and v) while q-RBFS tends to find leaves of the big star (like x and y). Observe that here, z is the only vertex that is likely contained in two different q-RBFS due to its high degree.

4.2. Preliminaries

Streaming algorithms read their input sequentially. The objective is to minimize the maximum space used during the computation. In particular, streaming algorithms should use asymptotically less space than the size of the input.

4.9 Definition (Graph Streaming Algorithm). Given an input graph $G = (V, E)$, a (single-pass) graph streaming algorithm can access G by reading E as a sequence one time. The order of the sequence can be *adversarial*, i. e., fixed but unknown, or *random*, i. e., a uniformly random permutation of E. We denote the stream, i. e., the sequence of edges, by $\mathcal{S}(G)$. After reading an edge, the algorithm may perform arbitrary computations. The space complexity of the algorithm is the maximum number of bits stored by the algorithm at a single point in time. ■

In the random-neighbor model, a property tester can access the graph by querying random neighbors of vertices. In particular, given a vertex u, the oracle returns a neighbor v of u that is sampled independently and uniformly at random. We extend the definition of property testers from Section 1.3.3 to involve this randomness: in particular, a property tester does not get access to its input graph by query access to the representation function $f_G : D_n \to R_n$, but by access to a randomized algorithm that returns elements from R_n.

4.10 Definition (random-neighbor model). In the random-neighbor model, the universe $\mathcal{U}$ is the set of all graphs. The model provides access to the graph by randomized access to adjacency lists. Let $G = (V, E)$ be a graph. Without loss of generality, assume that $V = [n]$.

The representation domain is the set $D_n = \{(i, j) \mid i \in [n] \wedge j \in [n]\}$, and the representation codomain is the set $R_n = [n] \cup \{\star\}$. Then, f_G maps $(v, 1), \ldots, (v, d(v))$ bijectively to the neighbors of v and $f_G(v, d(v) + 1, \ldots, f_G(v, n) = \star$ for every $v \in V$.

Access to f_G is guarded by a randomized algorithm that returns, given a query vertex $v \in V$, the value of $f_G(v, i)$, where i is sampled independently and uniformly at random from $[d(v)]$ every time the algorithm is queried. The distance of two graphs G and H is defined as

$$\delta(G, H) := \frac{|E_G \oplus E_H|}{n + \max(|E_G|, |E_H|)}.$$ ■

In the random-edge model, a property tester can also access the graph by sampling edges independently and uniformly at random. This query has no conceptual arguments, it just returns an edge every time it is invoked.

4.11 Definition (random-edge model). The random-edge model is an extension of the random-neighbor model. Therefore, the universe $\mathcal{U}$ is the set of all graphs. Let $G = (V, E)$ be a graph. Without loss of generality, assume that $V = [n]$.

In addition to the random-neighbor model, this model provides access to G by a randomized algorithm that returns, given no input, the value $\{u, f_G(v, i)\}$, where u is sampled proportional to its degree and i is sampled independently and uniformly at random from $[d(v)]$ every time the algorithm is queried. In other words, the algorithm returns a random edge. ■

The random-edge model may seem very similar to the random-neighbor model, but they are not the same: Sampling edges uniformly at random is equivalent to sampling vertices proportional to their degree. So, e. g., while approximating the average degree of a graph has query complexity $\Omega(\sqrt{n})$ in the random-neighbor model [Fei06; GR06], it has constant query complexity in the random-edge model (since sampling edges uniformly at random is equivalent to sampling vertices proportional to their degree). The problem of sampling random edges in models that only allow access to vertices appears as an issue in [KKR04; BS15; Ede+15; ERS18], and the problem itself was solved essentially optimally by Eden and Rosenbaum [ER18].

Our canonical tester for general graphs and the transformation to random-order streams crucially depends on a randomized exploration of the graph that we call *q-random BFS* (q-RBFS). Informally, it subsamples from a BFS by choosing only a random subset of neighbors for each vertex that is visited.

q-RBFS

4.12 Definition (q-random BFS). Let $q > 0$ be an integer and $G = (V, E)$ be a simple graph. For any vertex $v \in V$, the q-random BFS (abbreviated as *q-RBFS*) explores a random subset of the q-disk of v in G iteratively as follows. First, it initializes a queue $Q = \{v\}$ and a graph $H = (\{v\}, \emptyset)$. Then, in every iteration, it pops a vertex u from Q and samples q random neighbors $s_{u,1}, \dots, s_{u,q}$ of u. For every edge $e = \{u, s_{u,i}\}$, it adds $s_{u,i}$ and the directed edge $(u, s_{u,i})$ to H. Furthermore, if $s_{u,i}$ has distance less than q from v in H and $s_{u,i}$ has not been added to Q before, $s_{u,j}$ is appended to Q. When Q is empty, all edges in H are made undirected (without creating parallel edges) and H is returned. See Algorithm 4.1. ■

Any output of q-RBFS can be described in a static form using the concept of bounded discs.

4.13 Definition (q-bounded disk). For a given $q \in \mathbb{N}$, graph $G = (V, E)$, and vertex $v \in V$, a q-bounded disk of v in G is any subgraph H of G that is rooted at v and can be returned by RandomBFS(G, v, q). In this case, vertex v is called a *root* of the q-bounded disk H and the maximum distance from v to any other vertex in H is called the *radius* of H. ■

All q-bounded disks that are root-preserving isomorphic form an equivalence class.

4.14 Definition (q-bounded disk type). Let H be a q-bounded disk. The equivalence class of H

Algorithm 4.1 q-random BFS

```
function RandomBFS(G, v, q)
    Q ← empty queue; enqueue(Q, v)
    ∀w ∈ V : ℓ[w] ← ∞
    ℓ[v] ← 0
    H ← ({v}, ∅) with v as root
    while Q not empty do
        u ← pop element from Q
        for 1 ≤ i ≤ q do
            s_{u,i} ← query oracle for random neighbor of u
            add vertex s_{u,i} and edge (u, s_{u,i}) to H
            if ℓ[u] < q − 1 ∧ ℓ[s_{u,i}] = ∞ then
                ℓ[s_{u,i}] ← ℓ[u] + 1
                enqueue(Q, s_{u,i})
            end if
        end for
    end while
    return undirected H without parallel edges
end function
```

with respect to $\cong$, i. e., the existence of a root-preserving isomorphism (see Definition 1.3), is called the q-bounded disk type of H. ■

4.3. Canonical Constant-Query Testers in General Graphs

In this section, we present the main result on canonical testers for constant-query testable properties in general graphs, i. e., Theorem 4.3. For simplicity of the presentation, we focus on the random-neighbor model. The results carry over to other models, and we discuss how the proof differs for the random-edge model in Section 4.4.5.

4.3.1. Canonical Testers I: A General Version

In the following, we show that any tester with query complexity $q = q(\epsilon, n)$ in the random-neighbor model can be simulated by a *canonical tester* that samples $\hat{q} = \mathcal{O}(q)$ vertices and rejects if and only if the union of the subgraphs induced by the $\hat{q}$-RBFS from the sampled vertices belongs to some family of forbidden graphs.

$\mathscr{F}, \mathscr{F}_\cdot$

4.15 Theorem (canonical tester). Let $\Pi = (\Pi_n)_{n\in\mathbb{N}}$ be a graph property that can be tested in the random-neighbor model with query complexity $q = q_\Pi(\epsilon, n)$ and error probability at most $1/3$. Then for every $\epsilon > 0$, there exists an infinite sequence $\mathscr{F} = (\mathscr{F}_n)_{n\in\mathbb{N}}$ such that for every $n \in \mathbb{N}$,

- $\mathscr{F}_n$ is a set of rooted graphs that is closed under isomorphism, and each graph $F \in \mathscr{F}_n$ is the union of $\hat{q}$ many $\hat{q}$-bounded disks;
- the property Π_n can be tested with error probability at most $1/3$ by the following canonical tester:

1. sample $\hat{q}$ vertices independently and uniformly at random and mark them roots,
2. for each sampled vertex v, perform a $\hat{q}$-RBFS starting at v,
3. reject if and only if the explored subgraph is root-preserving isomorphic to some $F \in \mathcal{F}_n$,

where $\hat{q} = cq$ for some constant $c > 1$. The query complexity of the canonical tester is $\hat{q}^{\hat{q}}$. Furthermore, if $\Pi = (\Pi_n)_{n\in\mathbb{N}}$ can be tested in the random-neighbor model with one-sided error, then the resulting canonical tester for Π has one-sided error too, i.e., the tester always accepts graphs satisfying Π. ■

Proof. Our proof follows the approach used earlier [GR09; CPS15], though our analysis requires some extensions to deal with general graphs (of possibly unbounded degree).

Let $\mathcal{T}$ be a tester for Π_n with error probability at most 1/6. Note that the query complexity of $\mathcal{T}$ is $\hat{q} = cq$ for some constant $c > 1$. We will first convert $\mathcal{T}$ into a tester $\mathcal{T}_1$ that samples a random subgraph H of the input graph and answers all of $\mathcal{T}$'s queries using this subgraph. In particular, it samples $\hat{q}$ vertices and then returns the output on the basis of the subgraphs explored by all $\hat{q}$-RBFS that start at these vertices. Then, we convert $\mathcal{T}_1$ into a tester $\mathcal{T}_2$ whose output depends only on the edges in the explored subgraph, the ordering of all explored vertices and its own coins. Next, we convert $\mathcal{T}_2$ into a tester $\mathcal{T}_3$ whose output is independent of the ordering of all explored vertices. Therefore, after sampling H, the probability that $\mathcal{T}_3$ accepts the input graph is equal for all query-answer sequences $((u_i) = v_i)_{i\in[\hat{q}]}$ that $\mathcal{T}$ may ask and observe. Finally, we convert $\mathcal{T}_3$ into a tester $\mathcal{T}_4$ that returns the output deterministically according to this unlabeled version of H where roots are marked identically.

$\mathcal{T}_1$ Let $\mathcal{T}_1$ be the tester that first samples a set S_0 of $\hat{q}$ vertices independently and uniformly at random and then explores a subgraph by starting a $\hat{q}$-RBFS at each of the vertices. We mark all vertices in S_0 as roots and denote the union of all subgraphs by $H = (V', E')$. We use $\mathcal{T}_1$ to simulate the execution of $\mathcal{T}$ in the following way. Given a random-neighbor query to a graph $G = (V, E)$, the tester $\mathcal{T}_1$ will on-the-fly select a random, uniformly distributed permutation $\pi : V \to V$ and provide oracle access to the permuted graph $\pi(G) = (V, \pi(E))$, where $\pi(E) := \{(\pi(u), \pi(v) \mid \{u, v\} \in E\}$. Initially, all vertices in S_0 are considered unused. In the simulation, when $\mathcal{T}$ makes a query for a random neighbor of vertex v and if v has not appeared in any prior query or answer, then the tester $\mathcal{T}_1$ allocates v to an unused vertex u in the sample set S_0, and we let $\pi(v) := u$ and u will be considered as used; otherwise $\mathcal{T}_1$ uses the allocation $\pi(v)$ determined in the previous steps of the tester $\mathcal{T}_1$. To answer the query for a random neighbor of v, $\mathcal{T}_1$ selects $w = s_{\pi(v),j+1}$ (see Algorithm 4.1) where j is the number of random neighbor queries of v that $\mathcal{T}$ has issued so far. If w has been selected as the image of some vertex in the permutation π before, then $\mathcal{T}_1$ returns $\pi^{-1}(w)$; otherwise $\mathcal{T}_1$ returns a random unused value (vertex label) x and we define $\pi(x) := w$. If $w \in S_0$, then w will be considered used. The returned values will then be fed into $\mathcal{T}$. The tester $\mathcal{T}_1$ makes the same decision as the final decision of $\mathcal{T}$ after receiving all the necessary query answers. Denoting the event that the algorithm chooses permutation π by π, it follows that

$$\Pr[\mathcal{T}_1 \text{ correctly answers } G] = \sum_{\pi} \Pr[\mathcal{T} \text{ correctly answers } \pi(G) \mid \pi] \cdot \Pr[\pi] \geq n! \frac{5}{6n!} = \frac{5}{6}.$$

$\mathscr{T}_2$ We make $\mathscr{T}_1$ label-oblivious by defining the new tester $\mathscr{T}_2$ to be the one that accepts G with the average probability that $\mathscr{T}_1$ accepts G over the choice of π (see the note on the complexity of this computation on page 46). We denote the event that the tester draws the random permutation π' by π'.

$$\Pr[\mathscr{T}_2 \text{ accepts } G \mid \pi] = \sum_{\pi'} \Pr[\mathscr{T}_1 \text{ accepts } G \mid \pi'] \cdot \Pr[\pi'] = \Pr[\mathscr{T}_1 \text{ accepts } G],$$

where we denote the event that the tester draws the random permutation π' by π'. Since it suffices to consider all labellings of H and random coins of $\mathscr{T}$, this does not require any additional queries.

$\mathscr{T}_3$ We make $\mathscr{T}_2$ oblivious of the order of S_0 and all $s_{v,i}$ by considering the uniform distribution $\mathscr{U}$ over all permutations of elements in S_0 and all permutations of $(s_{v,i})_{i\in[\hat{q}]}$ for all v. In particular, we let the resulting tester $\mathscr{T}_3$ accept with the average probability that $\mathscr{T}_2$ accepts G, where the probability is taken over the choice of $S \in \mathscr{U}$, i. e.,

$$\Pr[\mathscr{T}_3 \text{ accepts } G \mid S, \pi] = \sum_{S\in\mathscr{U}} \Pr[\mathscr{T}_2 \text{ accepts } G \mid S, \pi] \cdot \Pr[S \mid \pi] = \Pr[\mathscr{T}_2 \text{ accepts } G \mid \pi].$$

Again, this does not require any additional queries because we only need to consider all permutations of sampled vertices and random coins of $\mathscr{T}$.

$\mathscr{T}_4$ Let $\mathscr{T}_4$ be the tester obtained from $\mathscr{T}_3$ that accepts (with probability 1) the input graph if and only if the acceptance probability associated with the explored subgraph H is at least $1/2$. Since the acceptance probability of $\mathscr{T}_3$ does not change when vertices are relabeled or S is reordered, its depends only on H up to isomorphism and its internal randomness. Employing the proof of Lemma 4.4 in [GT03], we can prove that $\mathscr{T}_4$ is a tester for Π_n with error probability $1/3$. We replicate the argument for completeness: The acceptance probability of $\mathscr{T}_4$ on any input is as most twice as much as the acceptance probability of $\mathscr{T}_3$ because H is only accepted by $\mathscr{T}_4$ with probability 1 if $\mathscr{T}_3$ accepts H with probability at least $1/2$. If G is ϵ-far from Π, it is thus accepted with probability at most $2 \cdot 1/6 = 1/3$. Furthermore, if $G \in \Pi$, the expected rejection probability (where expectation is taken over S) is at least $5/6$, and by Markov's inequality, the probability that $\mathscr{T}_3$ rejects when sampling S is at most $1/3$. Therefore, the probability that $\mathscr{T}_4$ samples an S that is accepted by $\mathscr{T}_3$ with probability less than $1/2$ is at most $1/3$.

Note that the decision of $\mathscr{T}_4$ is deterministic after H is determined. We define $\mathscr{F}_n$ to be the set of graphs that is a union of $\hat{q}$ many $\hat{q}$-bounded disks on which the tester rejects.

Finally, let us observe that if the original tester for $\Pi = (\Pi_n)_{n\in\mathbb{N}}$ has one-sided error, then all steps of our simulations ensure that the resulting canonical tester has one-sided error too. In particular, we have that $\Pr[\mathscr{T}_1 \text{ correctly answers } G] = 1$ for all $G \in \Pi_n$, and all the remaining steps maintain one-sided error because they do not decrease the acceptance probability for any graph G that is accepted by $\mathscr{T}$ with probability at least $2/3 < 1$. □

4.3.2. Canonical Testers II: Identifying Vertices in the Intersecting Discs

Theorem 4.15 provides us a canonical way of testing constant-query testable properties (in the random-neighbor model) by relating the tester to a set of forbidden subgraphs $\mathscr{F}_n$ for every

$n \in \mathbb{N}$. However, as mentioned in Section 4.1.4, it is hard to directly use Theorem 4.15 to design and analyze our streaming testers due to the intersections of q-RBFS. In order to tackle this difficulty, we decompose each forbidden subgraph $F \in \mathscr{F}_n$ into all possible sets of intersecting q-bounded disks whose union is F. In order to recover F from such a decomposition, we have to identify and monitor vertices that are contained in more than one q-bounded disk of F.

Identifying Vertices With Large Reach Probability

In this section, we prove that with constant probability the q-bounded disks found by q-RBFS will only intersect on a small set of vertices V_α and the discs will not intersect on any edge.

We begin with a useful definition on the probability of reaching a vertex from a q-RBFS.

$q_{4.16}$, $r(\bullet)$, $V_\bullet$

4.16 Definition (reach probability). For each vertex v, the reach probability $r(v) \coloneqq r_q(v)$ of v is the probability that a q-RBFS starting at a uniformly randomly chosen vertex reaches v. For any α, $0 \leq \alpha \leq 1$, we let $V_\alpha \coloneqq \{v \in V : r(v) \geq \alpha\}$. For a fixed q, let $q_{4.16} \coloneqq \sum_{i=0}^{q} q^i = (q^{q+1}-1)/(q-1)$. ■

In the following lemma, we give an upper bound on the size of the set of vertices with constant reach probability, which also implies that with constant probability, the number of vertices visited by at least two q-RBFS performed by the canonical tester is small.

4.17 Lemma. For any $0 < \alpha \leq 1$, it holds that $|V_\alpha| \leq q_{4.16}/\alpha$. ■

Proof. Let $p_u(v)$ be the probability that a q-RBFS starting at vertex u discovers vertex v. Note that $r(v) = 1/n \sum_u p_u(v)$. Let X_u denote the number of vertices in the subgraph explored by a q-RBFS starting at vertex u, and let X denote the number of vertices in the subgraph explored by a q-RBFS starting at a vertex that is chosen independently and uniformly at random from V. Note that $\mathrm{E}[X_u] = \sum_v p_u(v)$, and thus,

$$\mathrm{E}[X] = \frac{1}{n}\sum_u \mathrm{E}[X_u] = \sum_u \frac{1}{n}\sum_v p_u(v) = \sum_v \frac{1}{n}\sum_u p_u(v) = \sum_v r(v)\,.$$

We observe that a q-RBFS starting at an arbitrary vertex u explores at most q^i vertices at distance i from u, which gives that $\mathrm{E}[X_u] \leq \sum_{i=0}^{q} q^i = q_{4.16}$, and hence also $\mathrm{E}[X] \leq q_{4.16}$.

Recall that $V_\alpha = \{v : r(v) \geq \alpha\}$. We have that

$$q_{4.16} \geq \mathrm{E}[X] = \sum_v r(v) \geq \sum_{v \in V_\alpha} r(v) \geq |V_\alpha| \cdot \alpha\,,$$

which concludes the lemma. □

We further show that with high probability, two q-RBFS starting from vertices chosen independently and uniformly at random will not share an edge (i.e., will not visit the same edge).

4.18 Lemma. Let $0 < \alpha \leq 1$, $n \geq q_{4.16} q/\alpha^2$ and let u, v be two randomly chosen vertices. Let H_u and H_v denote the subgraphs visited by two q-RBFS starting at u and v, respectively. Then with probability $1 - q_{4.16} q \cdot 2\alpha$, no edge will be contained in both H_u and H_v. ■

In order to prove the lemma, we first show that vertices with a large reach probability have large degree (i. e. linear in n).

4.19 Lemma. Let $0 < \alpha \leq 1$. It holds that for any $v \in V_\alpha$, $d(v) \geq n\alpha/q_{4.16}$. ■

Proof. Let $H = (S, E(H))$ be the subgraph explored by a q-RBFS starting at a vertex that is chosen uniformly at random from V. For each $0 \leq i \leq q$, we let $S_i \subseteq S$ denote the set of vertices at distance exactly i from the root of S. For any $v \in V$, let $p(v)$ be the probability that v is contained in S. We have the following claim (the proof is given subsequent to this one).

4.20 Claim. Let $0 \leq i \leq q$. For every non-isolated vertex $v \in V$, conditioned on the event that v is not contained in $\cup_{j \leq i-1} S_j$, the probability that v is contained in S_i is at most $q^i d(v)/n$. ■

By Claim 4.20, the probability that v is contained in S is at most

$$\sum_{i=0}^{q} \frac{q^i d(v)}{n} \leq \frac{q_{4.16} d(v)}{n}.$$

Recall that $V_\alpha = \{v : r(v) \geq \alpha\}$. By noting that $r(v)$ is exactly the probability that v is contained in S, we have that for any $v \in V_\alpha$,

$$\alpha \leq \frac{q_{4.16} d(v)}{n},$$

which gives that $d(v) \geq n\alpha/q_{4.16}$. This completes the proof of Lemma 4.19. □

Proof of Claim 4.20. We prove the claim by induction on i. If $i = 0$, then the probability that the q-RBFS visits v is $1/n$, and thus v is contained in S_0 with probability at most $1/n$.

Let us assume now that the statement of the claim holds for $i - 1$. That is, for any vertex u, the probability that u is contained in S_{i-1} is at most $q^{i-1} d(u)/n$.

Consider an arbitrary vertex v. By induction, for every neighbor u of v, u is contained in S_{i-1} with probability at most $q^{i-1} d(u)/n$. In the q-RBFS, each vertex w in S_{i-1} samples q neighbors of w independently and uniformly at random, which implies that the probability that v is contained in S_i is at most

$$\sum_{u \in \Gamma(v)} \frac{q^{i-1} d(u)}{n} \sum_{j=1}^{q} \frac{1}{d(u)} = \sum_{u \in \Gamma(v)} \frac{q^i}{n} = \frac{q^i d(v)}{n},$$

where $\Gamma(v)$ is the set of neighbors of v in G. This yields the proof of Claim 4.20. □

Now we are ready to prove Lemma 4.18, which upper bounds the probability that the two subgraphs explored by two q-RBFS starting at two random vertices share any edge.

Proof of Lemma 4.18. Let us consider an arbitrary edge $\{x, y\} \in E(H_u)$.

1. $x, y \in V_\alpha$: By Lemma 4.19, if $x, y \in V_\alpha$, then $d(x), d(y) \geq n\alpha/q_{4.16}$. Suppose that at least one of x, y, say x, is also discovered by the q-RBFS from v, i.e., $x \in V(H_v)$ (otherwise, $\{x, y\}$ will not be contained in H_v at all). Thus, the probability that $\{x, y\}$ will be contained in H_v (i.e., visited by q-RBFS from v) is at most

   $$\frac{q}{d(x)} \leq \frac{q_{4.16} q}{n\alpha} \leq \alpha.$$

   By the union bound, $\{x, y\}$ will be contained in H_v with probability at most $2q_{4.16} q/(n\alpha) \leq 2\alpha$.

2. $x, y \notin V_\alpha$: By definition of V_α, the probability that x is contained in $V(H_v)$ is at most α

(and similarly for y). By the union bound, $\{x, y\} \in E(H_v)$ with probability at most 2α.

3. $x \in V_\alpha, y \notin V_\alpha$: By the above analysis, if x is contained in $V(H_v)$, then $\{x, y\}$ will be contained in $E(H_v)$ with probability at most $q_{4.16}q/(n\alpha)$. Further note that y will be contained in $V(H_v)$ with probability at most α. Thus, the probability that $\{x, y\} \in E(H_v)$ is at most $q_{4.16}q/(n\alpha) + \alpha \leq 2\alpha$.

4. $x \notin V_\alpha, y \in V_\alpha$: By the analysis as for Item 3, the probability that $\{x, y\} \in E(H_v)$ with probability at most $q_{4.16}q/(n\alpha) + \alpha \leq 2\alpha$.

Furthermore, we note that $|E(H_u)| \leq q_{4.16}q$. This implies that with probability at least $1 - q_{4.16}q \cdot 2\alpha$, none of edges in $E(H_u)$ will be contained in H_v. □

Colored q-Bounded Disk Types

To identify vertices in V_α, we assign them unique colors for the analysis. We call a disc *r-colored* if in addition to uncolored vertices in the disc, some vertices in the disc may be colored with at most r colors, each color being used at most once. Two colored q-bounded disk types Δ_1 and Δ_2 (see Definition 4.14) are isomorphic to each other, denoted by $\Delta_1 \cong \Delta_2$, if there is a root-preserving isomorphism f from Δ_1 to Δ_2 (see Definition 1.3) that also preserves the colors, i. e., if and only if $u \in V(\Delta_1)$ is colored with color c, then $f(u) \in \Delta_2$ is colored with color c.

$\mathscr{H}_q$

4.21 Definition. Let $q > 0$ be an integer. We let $\mathscr{H}_q := \{\Delta_1, \ldots, \Delta_N\}$ denote the set of all possible r-colored q-bounded disk types, where N is the total number of such types. ■

For any given colored q-bounded disk type, we have the following definition on the probability of seeing such a disc type from a q-RBFS.

$\text{Reach}_{\cdot}(\bullet)$

4.22 Definition (reach probability of colored q-bounded disk types). Let $G = (V, E)$ be a graph of size n such that each vertex in V_α is assigned to a unique color. Let $\Delta \in \mathscr{H}_q$ be a colored q-bounded disk type. The *reach probability of* Δ *in* G is the probability that a q-RBFS from a random vertex in G reveals a graph that is (root- and color-preserving) isomorphic to Δ, that is

$$\text{Reach}_G(\Delta) := \Pr_{v,RBFS}[\textsc{RandomBFS}(G, v, q) \cong \Delta].$$

For a given vertex v, the reach probability of Δ from v in G is the probability that a q-RBFS from v in G induces a graph that is (root- and color-preserving) isomorphic to Δ, that is

$$\text{Reach}_G(v, \Delta) := \Pr_{RBFS}[\textsc{RandomBFS}(G, v, q) \cong \Delta].$$

■

Recall from Definition 4.13 that a q-bounded disk of v in G is any subgraph H of G that is rooted at v and can be returned by $\textsc{RandomBFS}(G, v, q)$. In order to estimate the reach probability of a colored q-bounded disk type, we consider, for each starting vertex v, the set of all possible colored q-bounded disks that one can see in a q-RBFS starting at v.

4.23 Definition (reach probability of colored q-bounded disks). Let $G = (V, E)$ be a graph in which all vertices in V_α are uniquely colored and let $v \in V$. A colored q-bounded disk of v is a q-bounded disk of v in G in which all vertices in V_α are colored. We let $\mathscr{C}_v$ denote the set of all possible colored q-bounded disks of v.[5]

[5]Note that the number $|\mathscr{C}_v|$ of colored q-bounded disks of v is not neccessarily a constant.

For any fixed colored q-bounded disk $C \in \mathscr{C}_v$, the reach probability of C from v is the probability that a q-RBFS from v sees exactly C, that is,

$$\mathrm{Reach}_G(v, C) \coloneqq \Pr_{BFS}[\textsc{RandomBFS}(G, v, q) = C] .$$

■

By our definition, the q-RBFS from a vertex v in the colored graph G (with vertices in V_α colored) will return exactly one colored q-bounded disk of v. For each colored q-bounded disk type Δ, we let $\mathscr{C}_v(\Delta)$ denote the subset of $\mathscr{C}_v$ which contains all colored q-bounded disks of v that are isomorphic to Δ. Therefore, we have the following observation.

$$\mathrm{Reach}_G(v, \Delta) = \sum_{D \in \mathscr{C}_v(\Delta)} \mathrm{Reach}_G(v, D) . \tag{4.1}$$

Canonical Testers With Distinguished Vertices in the Intersecting Discs

Now, we give a refined characterization of the family of forbidden subgraphs corresponding to any constant-query testable property in general graphs, which establishes the basis of our framework for transforming canonical constant-query testers in the random-neighbor model to the random order streaming model.

$\mathscr{F}', \mathscr{F}'_\bullet$

In our next theorem, we will consider partially vertex-colored graphs and q-bounded disks: we color each vertex in V_α with a unique color from a palette of size $|V_\alpha|$. Recall from Lemma 4.17 that $|V_\alpha| \leq q_{4.16}/\alpha$. We obtain canonical testers of constant-query testable properties by *forbidden colored q-bounded disks* instead of *forbidden subgraphs* (that can be composed of more than a single q-bounded disk). See Fig. 4.1 on page 61 for an example.

4.24 Theorem. Let $\Pi = (\Pi_n)_{n \in \mathbb{N}}$ be a graph property that is constant-query testable and let $\epsilon > 0$. Let $q = q(\epsilon)$ be the query complexity of a canonical tester $\mathscr{T}$ for Π with error probability at most $1/6$ (see Theorem 4.15) and let $\alpha \leq 1/(24q^q)$. There is an infinite sequence $\mathscr{F}' = (\mathscr{F}'_n)_{n \in \mathbb{N}}$ such that for any $n \geq q_{4.16} q/\alpha^2$, the following properties hold:

- $\mathscr{F}'_n$ is a set of graphs that is closed under isomorphism, and for each graph $F \in \mathscr{F}'_n$, there exists a family $\mathscr{D}_F$, and each $D \in \mathscr{D}_F$ is a multiset of q many $q_{4.16}/\alpha$-colored and rooted q-bounded disk types such that (i) the disc types are pairwise edge-disjoint, and (ii) the graph obtained by identifying all vertices of the same color in the bounded discs of D is isomorphic to F.
- For any graph $G = (V, E)$ of size n such that each vertex in V_α is colored uniquely, let S denote the set of q subgraphs obtained by performing q-RBFS starting at q vertices sampled independently and uniformly at random. Then,
  - if $G \in \Pi_n$, with probability at least $2/3$, there is no $F \in \mathscr{F}'_n$ such that F is isomorphic to a graph from S,
  - if G is ϵ-far from Π_n, with probability at least $2/3$, there exists $F \in \mathscr{F}'_n$ such that for some $D \in \mathscr{D}_F$, $D \cong S$ (preserving roots and colors),

  where the probability is taken over the randomness of S.

Furthermore, if Π can be tested with one-sided error, then for every $G \in \Pi_n$ there is no $F \in \mathscr{F}'_n$ such that $F \cong S$. ■

Proof. Let $\mathscr{F}_n$ be the family of forbidden graphs corresponding to the canonical tester $\mathscr{T}$ as defined in Theorem 4.15. We prove that if $\mathscr{T}$ finds $F \in \mathscr{F}_n$, then we can decompose F into $q_{4.16}/\alpha$-colored q-bounded disks with probability at least $5/6$. Since the tester errs with probability at most $1/6$, defining $\mathscr{F}'_n$ as the set of all F that can be decomposed in this way proves the claim.

Let F be the subgraph observed by $\mathscr{T}$ and let S be the corresponding multiset of q many q-bounded disks obtained by q-RBFS. By the definition of V_α and the union bound, there exist no $v \in F \setminus V_\alpha$ and $\Delta_1, \Delta_2 \in S$ such that $v \in V(\Delta_1) \cap V(\Delta_2)$ with probability at least $1 - q^q \cdot \alpha$. Condition on the event that all vertices that are contained in at least two disks from S are colored vertices. By Lemma 4.17, the number of colored vertices is at most $|V_\alpha| \leq q_{4.16}/\alpha$. Furthermore, by Lemma 4.18 and the union bound, with probability at least $1 - q^2 \cdot 2\alpha$, there is no pair $\Delta_1, \Delta_2 \in D$, edges $(u_1, v_1) \in E(\Delta_1), (u_2, v_2) \in E(\Delta_2)$ and a root- and color-preserving isomorphism f from Δ_1 to Δ_2 such that $(u_1, v_1) = (f(u_2), f(v_2))$. In summary, F can be decomposed as described with probability at least $1 - q^q\alpha - 2q^2\alpha > 5/6$. □

4.4. Transformation From The Query Model to Streaming

Given a canonical tester $\mathscr{T}$ for a property Π that is constant-query testable in the random-neighbor model, we transform it into a random-order streaming algorithm as follows. Recall from Theorem 4.15 that $\mathscr{T}$ explores the input graph by sampling vertices uniformly at random and running q-RBFS for each of these vertices. Only if the resulting subgraph contains an instance of a forbidden subgraph from a family $\mathscr{F}$, it rejects. It seems natural to define a procedure like q-RBFS for random order streams, namely a procedure StreamCollect($\mathcal{S}(G), v, q$) (q-*SC*), and let the streaming algorithm reject only if the union of all q-SC contains an instance of a graph from $\mathscr{F}$. However, this raises a couple of issues.

It seems hard to analyze the union of the subgraphs obtained by q-SC and relate it to the union of subgraphs observed by q-RBFS because the interference between two q-SC is quite different from the interference of two q-RBFS. Therefore, we use Theorem 4.24, which roughly says that we can decompose each forbidden subgraph into colored q-bounded disk types. This leads to the following idea: First, we prove that for any colored q-bounded disk type Δ, if q-RBFS finds an instance of Δ in the input graph with probability p (where colors correspond to intersections of multiple RBFS), then q-SC finds an instance of Δ with probability cp for some suitable constant c. Then, we prove that if S is a sufficiently large set of vertices sampled uniformly at random, for each colored q-bounded disk type Δ, the following holds: the fraction of q-bounded disks found by q-SCs started from S that are isomorphic to Δ is bounded from below by the probability that a q-RBFS from a random vertex sees a colored q-bounded disk that is isomorphic to Δ. Finally, in the next section, we conclude that if q-RBFS finds a forbidden subgraph $F \in \mathscr{F}$ with probability p, then the fraction of q-SC also finds this subgraph with probability cp (for some suitable constant c) because it will find the corresponding colored q-bounded disks that assemble F.

4.4.1. Collecting a q-Bounded Disk in a Graph Stream

In the streaming algorithm, we collect a q-bounded disk from a starting vertex v. We do this in a natural and greedy way, which we call *q-stream collect* (q-SC).

q-SC

4.25 Definition (q-stream collect). Let $H = (U, F)$ be a graph with $U = \{v\}$ and $F = \emptyset$. Given

access to a stream of edges, whenever we read an edge $\{u, w\}$ from the stream that is connected to our current graph H and adding $\{u, w\}$ to H does not violate the q-bounded radius of H, and the degree of u or the degree of w in H is still less than q^{2q}, we add it to F (and possibly add one of its endpoints to U); otherwise, we simply ignore the edge. Note that the algorithm does not assign colors to the subgraphs it explores. The procedure is formally defined in Algorithm 4.2. ■

Algorithm 4.2 Collecting a q-bounded disk from a vertex in stream

function StreamCollect($\mathcal{S}(G), v, q$)
  $U \leftarrow \{v\}$
  $\forall u \in V : d_u \leftarrow 0, \ell_u \leftarrow \infty$
  $\ell_v \leftarrow 0; F \leftarrow \emptyset$
  $H = (U, F)$ with v marked as root
  for $\{u, w\} \leftarrow$ next edge in the stream **do**
    if $(\{u, w\} \cap U \neq \emptyset)$ **then**
      if $(u \in U \Rightarrow (\ell_u < q \wedge d_u < q^{2q}) \vee (w \in U \Rightarrow (\ell_w < q \wedge d_w < q^{2q}))$ **then**
        $U \leftarrow U \cup \{u, w\}$
        $F \leftarrow F \cup \{\{u, w\}\}$
        $d_u \leftarrow d_u + 1; d_w \leftarrow d_w + 1$
        $\ell_u \leftarrow \min(\ell_u, \ell_w + 1); \ell_w \leftarrow \min(\ell_w, \ell_u + 1)$
      end if
    end if
  end for
  return H
end function

4.4.2. Relation of One q-SC and One q-RBFS

In the following, we show that for any vertex v, and any colored q-bounded disk C of v, the probability of collecting C from v by running q-SC on a random order edge stream is at least a constant factor of the probability of reaching C from v by running a q-RBFS on G. The statements in this section hold for a single run of q-SC.

We emphasize that the coloring does not need to be explicitly given. It is sufficient if it can be applied when random access to the graph is given. In particular, we may assign each vertex in V_α a unique color. This enables us to identify the vertices where multiple q-RBFS may intersect, which is crucial to apply Theorem 4.24 later.

4.26 Lemma. Let G be a vertex-colored graph. There exists a constant $c_{4.26}(q)$ depending on q, such that for any of its colored q-bounded disk C, it holds that $c_{4.26}(\bullet)$

$$\Pr_{\mathcal{S}(G)}[\text{StreamCollect}(\mathcal{S}(G), v, q) \text{ contains } C] \geq c_{4.26}(q) \cdot \text{Reach}_G(v, C)\,. \qquad ■$$

Proof. Note that both q-RBFS and q-SC can be viewed as random processes of revealing edges in G. Such a process starts from a fixed vertex $v_0 := v$. At each time step $t \geq 0$, new edge are revealed.

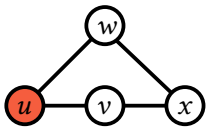


Figure 4.3.: Consider the disk rooted at u. The following edge ordering is good: $\{u,w\},\{u,v\},\{w,x\},\{v,x\}$. The ordering $\{u,v\},\{w,x\},\{u,w\},\{v,x\}$ is not good because there is no edge that connects $\{u,v\}$ to either w or x at the time when $\{w,x\}$ arrives, and it cannot be realized by q-RBFS or q-SC. The ordering $\{u,v\},\{v,x\},\{u,w\},\{w,x\}$ can be realized by q-SC, but not by q-RBFS, and therefore, it is not good either.

Recall that in every iteration, q-RBFS pops an element u from its queue Q, samples q random neighbors $s_{u,1},\dots,s_{u,q}$, adds $\{u,s_{u,1}\},\dots,\{u,s_{u,q}\}$ to its subgraph H and enqueues all $s_{u,i}$ that have not been visited yet. For any fixed v and any of its colored q-bounded disks C, we call an ordering σ of $E(C)$ *good* if it is a subsequence of the edges sampled by some run of q-RBFS on v. Note that q-RBFS may sample edges multiple times because neighbors are queried independently and uniformly at random. The ordering σ also defines an ordering of the vertices that corresponds to the order in which vertices are added to the queue Q. We have that

$$\mathrm{Reach}_G(v,C) = \sum_{\sigma\,:\,\mathrm{good}} \Pr_{RBFS}[\sigma]\,.$$

Let us consider an arbitrary good ordering σ of $E(C)$, and let $v_0 \coloneqq v, v_1, \dots, v_k$ be the corresponding vertex ordering of $V(C)$, where $k = |V(C)| \le q^{2q}$. We let $\mathscr{F}_t^\sigma$ denote the event that for any $0 \le i \le t$, when v_i is popped from the queue, the edges $\{v_i, w_{i,1}\},\dots,\{v_i,w_{i,j_i}\}$ corresponding to random neighbors of v_i are sampled in the order defined by σ, where $j_i \le q$. Note that if $j_i \neq q$, at least one neighbor of v_i was sampled more than once. We enumerate all possible combinations of $q - j_t$ positions in the sequence of the random neighbors $(s_{u,i})_{i\in[q]}$: Let $\mathscr{M}(j_i,q) = \{M \mid M \subseteq \{2,\dots,q\} \wedge |M| = q - j_i\}$. It holds that

$$\Pr_{RBFS}[\sigma] = \Pr_{RBFS}[\cap_{i\le k}\mathscr{F}_i^\sigma] = \Pr_{RBFS}[\mathscr{F}_0^\sigma]\cdot \Pr_{RBFS}[\mathscr{F}_1^\sigma \mid \mathscr{F}_0^\sigma]\cdots \Pr_{RBFS}[\mathscr{F}_k^\sigma \mid \cap_{i\le k-1}\mathscr{F}_i^\sigma]\,.$$

Consider an arbitrary $t \le k$ and observe that

$$\Pr_{RBFS}[\mathscr{F}_t^\sigma \mid \cap_{i\le t-1}\mathscr{F}_i^\sigma] = \left(\frac{1}{d(v_t)}\right)^{j_i} \sum_{M\in\mathscr{M}(j_i,q)} \prod_{i\in M} \frac{i-1-|M\cap[i-1]|}{d(v_t)} = \alpha(j_t,q)\left(\frac{1}{d(v_t)}\right)^q,$$

where $\alpha(j_t,q)$ is a constant depending on j_t and q. We compute the probability of seeing this edge ordering in random streaming order: It holds that

$$\Pr_{\mathscr{S}(G)}[\sigma] = \Pr_{\mathscr{S}(G)}[\cap_{i\le k}\mathscr{F}_i^\sigma] = \Pr_{\mathscr{S}(G)}[\mathscr{F}_0^\sigma]\cdot \Pr_{\mathscr{S}(G)}[\mathscr{F}_1^\sigma|\mathscr{F}_0^\sigma]\cdots \Pr_{\mathscr{S}(G)}[\mathscr{F}_k^\sigma|\cap_{i\le k-1}\mathscr{F}_i^\sigma]\,.$$

Recall that j_i denotes the number of edges that are collected from v_i in the ordering σ. Let $j_{\le i} \coloneqq \sum_{i'\le i} j_{i'}$ denote the number of edges after collecting edges from v_i. For any t, we let $\mathscr{G}_t$ denote the event that the first s_t in σ are streamed in the same order as in σ. By the definition of random-order streams, conditioned on $\cap_{i\le t-1}\mathscr{G}_i^\sigma$, the probability of seeing the next j_t edges from v_t is at least the probability that all the next j_t edges appear after the first $j_{\le t-1}$ edges,

times the probability that these j_t edges appear earlier than the remaining edges incident to v_t. That is, for any $t \leq k$,

$$\Pr_{\mathcal{S}(G)}[\mathcal{G}_t^\sigma | \cap_{i \leq t-1} \mathcal{G}_i^\sigma] \geq \left(\frac{1}{j_{\leq t-1}+1}\right)^{j_t} \cdot \min_{\lambda_t : 0 \leq \lambda_t \leq d(v_t)-j_t} \frac{(d(v_t)-\lambda_t-j_t)!}{(d(v_t)-\lambda_t)!} \geq \beta(j_t) \cdot \left(\frac{1}{d(v_t)}\right)^q,$$

where λ_t corresponds to the number of edges incident to v_t that appeared before the time we collect edges from v_t and $\beta(j_t)$ is a constant depending on j_t. We conclude that

$$\begin{aligned}
&\Pr_{\mathcal{S}(G)}[\textsc{StreamCollect}(\mathcal{S}(G), v, q) \text{ contains } C] \\
&\geq \sum_{\sigma : \text{good}} \Pr_{BFS}[\sigma] \\
&\geq \sum_{\sigma : \text{good}} \Pr_{\mathcal{S}(G)}[\mathcal{G}_0^\sigma] \cdot \Pr_{\mathcal{S}(G)}[\mathcal{G}_1^\sigma | \mathcal{G}_0^\sigma] \cdots \Pr_{\mathcal{S}(G)}[\mathcal{G}_k^\sigma | \cap_{i \leq k-1} \mathcal{G}_i^\sigma] \\
&\geq \sum_{\sigma : \text{good}} \prod_{t=0}^{k} \frac{\beta(j_t)}{\alpha(j_t)} \Pr_{RBFS}[\mathcal{G}_0] \cdot \Pr_{RBFS}[\mathcal{G}_1^\sigma | \mathcal{G}_0^\sigma] \cdots \Pr_{RBFS}[\mathcal{G}_k^\sigma | \cap_{i \leq k-1} \mathcal{G}_i^\sigma] \\
&= c_{4.26}(q) \sum_{\sigma : \text{good}} \Pr_{RBFS}[\sigma] \\
&= c_{4.26}(q) \cdot \text{Reach}_G(v, C),
\end{aligned}$$

where we define $c_{4.26}(q) \coloneqq \prod_{t=0}^{k} \beta(j_t)/\alpha(j_t)$. □

The following lemma performs the step from q-bounded disks to q-bounded disk *types*.

4.27 Lemma. Let Δ be a fixed colored q-bounded disk type. Let $X_{v,\Delta}$ denote the indicator variable for the event that q-SC starting at v collects a subgraph that contains a colored q-bounded disk of v that is isomorphic to Δ. Let Y_v denote the indicator variable that q-RBFS from v sees a colored q-bounded disk of v that is isomorphic to Δ. Then it holds that

$$\mathrm{E}_{\mathcal{S}(G)}[X_{v,\Delta}] \geq c_{4.26}(q) \cdot \mathrm{E}_{RBFS}[Y_v],$$

where $c_{4.26}(q)$ is the constant from Lemma 4.26. ■

Proof. Let $C \in \mathscr{C}_v(\Delta)$ denote any colored q-bounded disk that is isomorphic to Δ. Let $X_{v,C}$ be the indicator variable that q-SC from v collects a subgraph that contains C. Let $Y_{v,C}$ be the indicator variable that q-RBFS from v sees C. Then by linearity of expectation, we have

$$\mathrm{E}_{\mathcal{S}(G)}[X_{v,\Delta}] = \sum_{C \in \mathscr{C}_v(\Delta)} \mathrm{E}_{\mathcal{S}(G)}[X_{v,C}], \quad \mathrm{E}_{RBFS}[Y_v] = \sum_{C \in \mathscr{C}_v(\Delta)} \mathrm{E}_{RBFS}[Y_{v,C}].$$

Note that $\mathrm{E}_{\mathcal{S}(G)}[X_{v,C}] = \Pr_{\mathcal{S}(G)}[\textsc{StreamCollect}(\mathcal{S}(G), v, q) \text{ contains } C]$ and that $\mathrm{E}_{RBFS}[Y_{v,C}] = \text{Reach}_G(v, C)$. Now, the statement of the lemma follows from Lemma 4.26. □

Next, we consider the probability of seeing a colored q-disc type Δ. Note that $\mathrm{E}_{\mathcal{S}(G)}[X_{v,\Delta}] = \Pr_{\mathcal{S}(G)}[\textsc{StreamCollect}(\mathcal{S}(G), v, q) \text{ contains a subgraph } F \text{ with } F \cong \Delta]$. Furthermore, we have $\mathrm{E}_{RBFS}[Y_v] = \text{Reach}_G(v, \Delta)$. Thus, we have the following corollary.

4.28 Corollary. For any colored q-bounded disk type Δ, it holds that

$$\Pr_{\mathcal{S}(G)}[\textsc{StreamCollect}(\mathcal{S}(G), v, q) \text{ contains a subgraph } F \text{ with } F \cong \Delta] \geq c_{4.26}(q) \cdot \mathrm{Reach}_G(v, \Delta)\,.$$ ■

4.4.3. Relation of Multiple q-SC and q-RBFS

In the previous section, we related a single run of q-RBFS and a single run of q-SC. In particular, Corollary 4.28 states that if a q-RBFS starting from v finds some colored q-bounded disk type Δ with probability p, q-SC finds the same type Δ with probability $\Omega(p)$. However, the forbidden subgraphs that the property tester aims to find may be composed of more than one q-bounded disk. Therefore, we need to prove that if multiple runs of q-RBFS find q-bounded disk types $\Delta_1, \ldots, \Delta_k$ whose union contains an instance of a forbidden subgraph $F \in \mathcal{F}_n'$, then multiple runs of q-SC will find $\Delta_1, \ldots, \Delta_k$ with probability $\Omega(p)$.

We now show our main technical lemma on estimating the reach probability of q-bounded disk types in random order streams. Again, the coloring of vertices in G is implicit and only used for the analysis.

$q_\Delta, q_{4.29}$

4.29 Lemma. Let $G = (V, E)$ be a graph defined by a random order stream, let all vertices in V_α be colored, let $q > 0$ be an integer and let $q_{4.29} \coloneqq \sum_{i=0}^{q+1} q^{2qi}$; let $\delta > 0$, $\alpha = c_{4.26}(q)^4\delta^8/(10^9|\mathscr{H}_q|^2 \cdot q^{2q}q_{4.29})$. Let S denote a set of vertices that are chosen uniformly at random with

$$|S| = s = \max\left\{\frac{1}{20\sqrt{\alpha q^{2q} \cdot q_{4.29}}}, \frac{5000|\mathscr{H}_q|}{c_{4.26}(q)\delta^3}\right\}.$$

Let $\mathcal{J} \coloneqq \{H_v : H_v = \textsc{StreamCollect}(\mathcal{S}(G), v, q), v \in S\}$ denote the set of colored q-bounded disks collected by q-SC from vertices in S. For each type $\Delta \in \mathscr{H}_q$, let X_Δ denote the number of graphs H in $\mathcal{J}$ such that H contains a subgraph F with $F \cong \Delta$.

Then it holds that with probability at least $1 - 1/100$, for each type $\Delta \in \mathscr{H}_q$,

$$q_\Delta \coloneqq \frac{1}{c_{4.26}(q)} \cdot \frac{X_\Delta}{s} \geq \mathrm{Reach}_G(\Delta) - \delta\,,$$

where $c_{4.26}(q)$ is a constant from Corollary 4.28. ■

Proof. We first note that we only need to consider Δ with $\mathrm{Reach}_G(\Delta) \geq \delta$. As otherwise, the statement of the lemma trivially holds. Since we sampled a set S with $|S| \geq \Omega(\log(|\mathscr{H}_q|)/\delta^2)$, the following claim follows from the Chernoff bound.

4.30 Claim. For any $\Delta \in \mathscr{H}_q$, with probability (over the randomness of sampling S) at least $1 - 1/(400|\mathscr{H}_q|)$, it holds that

$$\left|\frac{\sum_{v\in S}\mathrm{Reach}_G(v, \Delta)}{|S|} - \mathrm{Reach}_G(\Delta)\right| \leq \frac{\delta}{2}\,. \tag{4.2}$$

■

Furthermore, similar to the proof of Lemma 4.18, we have the following claim (the proof appears on page 78).

4.31 Claim. Let α be $0 < \alpha \leq 1$, let $\alpha_0 \coloneqq \alpha q^{2q} \cdot q_{4.29}$ and let $n \geq q^{2q}q_{4.29}/\alpha^2$. Let H_u and H_v denote

the subgraphs collected by the q-SC starting at any two randomly chosen vertices u and v, respectively. Let

$$Y_{uv} \coloneqq \Pr_{\mathcal{S}(G)}[H_u \text{ and } H_v \text{ share some edge}].$$

Then with probability at least $1 - 2\sqrt{\alpha_0}$ (over the randomness of choosing u, v), it holds that

$$Y_{uv} \leq \sqrt{\alpha_0}. \tag{4.3}$$

■

From now on, we will condition on the following event $\mathscr{E}$: Eq. (4.2) holds as stated in Claim 4.30, and Eq. (4.3) holds for all $u, v \in S$ as stated in Claim 4.31. By Claim 4.30 and Claim 4.31, the event $\mathscr{E}$ holds with probability at least

$$1 - \frac{1}{400|\mathscr{H}_q|} \cdot |\mathscr{H}_q| - \alpha q^{2q} \cdot q_{4.29} \cdot s^2 \geq 1 - \frac{1}{200}$$

by our choice of α and s.

Let us now consider a fixed type $\Delta \in \mathscr{H}_q$. By Corollary 4.28, for any fixed v, we know that $\mathrm{E}_{\mathcal{S}}[X_{v,\Delta}] \geq c_{4.26}(q) \cdot \mathrm{Reach}_G(v, \Delta)$. Therefore, our estimate q_Δ for Δ satisfies that

$$\mathrm{E}_{\mathcal{S}}[q_\Delta] \geq \frac{1}{c_{4.26}(q)} \cdot c_{4.26}(q) \cdot \frac{\sum_{v \in S} \mathrm{Reach}_G(v, \Delta)}{|S|} \geq \mathrm{Reach}_G(\Delta) - \frac{\delta}{2},$$

where the last inequality follows from Eq. (4.2). The variance of our estimator is bounded as follows (the proof appears on page 79).

4.32 Claim. Let $\Delta \in \mathscr{H}_q$ be a type such that $\mathrm{Reach}_G(\Delta) > \delta$. Then

$$\mathrm{Var}_{\mathcal{S}}[q_\Delta] \leq \frac{\mathrm{E}_{\mathcal{S}}[q_\Delta]}{s \cdot c_{4.26}(q)} + \frac{\sqrt{\alpha q^{2q} \cdot q_{4.29}}}{c_{4.26}(q)^2}.$$ ■

Observe that $\mathrm{E}_{\mathcal{S}}[q_\Delta] \geq \mathrm{Reach}_G(\Delta) - \delta/2 \geq \delta/2$, as we have assumed that $\mathrm{Reach}_G(\Delta) \geq \delta$. Let $\eta = \delta/2$, and we apply Chebyshev's inequality to obtain that

$$\begin{aligned}
\Pr_{\mathcal{S}}[|q_\Delta - \mathrm{E}_{\mathcal{S}}[q_\Delta]| \geq \eta \,\mathrm{E}_{\mathcal{S}}[q_\Delta]] &\leq \frac{\mathrm{Var}_{\mathcal{S}}[q_\Delta]}{\eta^2 \,\mathrm{E}_{\mathcal{S}}[q_\Delta]^2} \\
&\leq \frac{1}{\eta^2} \cdot \frac{\mathrm{Var}_{\mathcal{S}}[q_\Delta]}{\mathrm{E}_{\mathcal{S}}[q_\Delta]^2} \\
&\leq \frac{1}{\eta^2} \left(\frac{1}{s \cdot c_{4.26}(q) \,\mathrm{E}_{\mathcal{S}}[q_\Delta]} + \frac{\sqrt{\alpha q^{2q} \cdot q_{4.29}}}{c_{4.26}(q)^2 \,\mathrm{E}_{\mathcal{S}}[q_\Delta]^2} \right) \\
&\leq \frac{8}{s \cdot c_{4.26}(q) \delta^3} + \frac{16\sqrt{\alpha q^{2q} \cdot q_{4.29}}}{c_{4.26}(q)^2 \delta^4} \\
&\leq \frac{1}{200|\mathscr{H}_q|},
\end{aligned}$$

Reminder: Lemma 4.29

$$|S| = s = \max\left\{\frac{1}{20\sqrt{\alpha q^{2q} \cdot q_{4.29}}}, \frac{5000|\mathscr{H}_q|}{c_{4.26}(q)\delta^3}\right\}, \quad \alpha = \frac{c_{4.26}(q)^4\delta^8}{10^9|\mathscr{H}_q|^2 \cdot q^{2q} q_{4.29}}$$

where the last inequality follows from our setting that of α and s. Therefore, with probability at least $1 - 1/200$, for all $\Delta \in \mathscr{H}_q$ with $\mathrm{Reach}_G(\Delta) \geq \delta$, we have that

$$q_\Delta \geq (1-\eta)\,\mathrm{E}_{\mathcal{S}}[q_\Delta] \geq \left(1 - \frac{\delta}{2}\right)\left(\mathrm{Reach}_G(\Delta) - \frac{\delta}{2}\right) \geq \mathrm{Reach}_G(\Delta) - \delta\,.$$

Finally, with probability at least $1 - 1/200 - 1/200 > 99/100$, we have that $q_\Delta \geq \mathrm{Reach}_G(\Delta) - \delta$ for all $\Delta \in \mathscr{H}_q$. This finishes the proof of the lemma. □

Proofs of Claim 4.31 and Claim 4.32

Proof of Claim 4.31. We note that it suffices to prove that the average of $Y_{u,v}$ is at most $2\alpha_0$, i. e.,

$$\frac{1}{n^2} \cdot \sum_{u,v} Y_{uv} \leq 2\alpha_0\,. \tag{4.4}$$

Assume that Eq. (4.4) holds. Let X denote the number of pairs u, v satisfying Eq. (4.3). Then it holds that

$$\frac{1}{n^2} \cdot \left((n^2 - X) \cdot \sqrt{\alpha_0}\right) \leq 2\alpha_0\,,$$

which gives that $X/n^2 \geq 1 - 2\sqrt{\alpha_0}$. This will complete the proof of Claim 4.31.

In the following, we prove Eq. (4.4). In order to do so, we first need to modify the corresponding parts in the proof of Lemma 4.18. That is, for the given $\alpha > 0$, we can define a set V'_α that contains all vertices that can be collected with probability at least α by invoking StreamCollect$(\mathcal{S}(G), v, q)$ from a uniformly randomly chosen vertex v. We can show that $|V'_\alpha| \leq q_{4.29}/\alpha$ in the same way as in the proof of Lemma 4.17, with the only difference that q-SC might see at most $q_{4.29}$ vertices (instead of $q_{4.16}$ vertices). Now, we show that for any $v \in V'_\alpha$, it holds that $d(v) \geq n\alpha/q_{4.29}$, which is similar to what we proved in Lemma 4.19 for the query model.

Let $S =$ StreamCollect$(\mathcal{S}(G), v, q)$ for a randomly chosen $v \in V$, and let $S_i \subseteq S$ denote the set of vertices of distance exactly i from the root of S. Similar to Claim 4.20, we first prove by induction the following statement:

($\star$) For $0 \leq i \leq q+1$, and for every non-isolated vertex $u \in V$, conditioned on the event $\mathscr{E}_{i-1}$ that u is not contained in $\cup_{j \leq i-1} S_j$, the probability that u is contained in S_i is at most $q^{2qi}d(u)/n$.

However, there are some subtle differences in the analysis of the induction from the proof of Claim 4.20. Details follow.

First, the statement ($\star$) is true for $i = 0$ as we sample u with probability $1/n$. Consider the case $i = 1$. If we see u conditioned on the event that we did not see it in S_0, this implies that

one of the neighbors of u, say $w \in \Gamma(u)$, is in S_0, which happens with probability $1/n \leq d(w)/n$. Then, q-SC will add the edge $\{w, u\}$ if it is among the first q^{2q} edges of all $d(w)$ edges that are incident to w. By the union bound, the probability that $u \in S_1$ conditioned on $\mathscr{E}_0$ is therefore at most

$$\sum_{w \in \Gamma(u)} \frac{d(w)}{n} \cdot \frac{q^{2q}}{d(w)} = \frac{q^{2q} d(u)}{n} .$$

Assume the statement holds for $i-1$ and let $i \geq 2$. We note that conditioned on $\mathscr{E}_{i-1}$, for any $w \in \Gamma(u)$, either $\{u, w\}$ has already appeared before the timestamp τ_w when we explore w, or $\{u, w\}$ appeared after we explore w. In the former case, we will not see u from w. In the latter case, suppose further that there are exactly j edges incident to w that appear after τ_w. Note that since $\{u, w\}$ appears after τ_w and w is not in S_0, it holds that $j \geq 1$. Then the probability that we will collect u from w is at most

$$\frac{j}{d(w) - j + 1} \cdot \frac{q^{2q}}{j} = \frac{q^{2q}}{d(w)}$$

because with probability $j/d(w)$ the edge $\{u, w\}$ appears after τ_w (i.e., after the $d(w) - j$ edges that are incident to w and that appear before τ_w), and conditioned on the event that $\{u, w\}$ appears after τ_w, with probability at most q^{2q}/j, the edge $\{u, w\}$ appears among the first q^{2q} edges of all the j edges incident to w that appear after τ_w. Since $q^{2q}/d(w)$ is independent of j, the probability that we collect u from w is at most

$$\begin{aligned} &\sum_{j=1}^{d(w)} \Big(\Pr[u \text{ will be collected from } w \mid j \text{ neighbors appear after we explore } w] \\ &\qquad \cdot \Pr[j \text{ neighbors appear after we explore } w] \Big) \\ &\leq \frac{q^{2q}}{d(w)} . \end{aligned}$$

Therefore, by induction on $i-1$, the probability that u is contained in S_i is at most

$$\sum_{w \in \Gamma(u)} \frac{q^{2q(i-1)} d(w)}{n} \frac{q^{2q}}{d(w)} \leq \sum_{w \in \Gamma(u)} \frac{q^{2qi}}{n} \leq \frac{q^{2qi} d(u)}{n} .$$

This finishes the proof of the statement $(\star)$. It follows that the probability that u is contained in S is at most

$$\sum_{i=0}^{q+1} \frac{q^{2qi} d(u)}{n} = \frac{q_{4.29} d(u)}{n} .$$

Recall that V'_α is the set of vertices that are contained in S with probability at least α. Therefore, as in the proof of Lemma 4.19, we have $d(v) \geq n\alpha / q_{4.29}$ for any $v \in V'_\alpha$.

Finally, we use the same argument as in the proof of Lemma 4.18 (with the only difference that we use $q_{4.29}$ instead of $q_{4.16}$) to show that with probability $1 - 2\alpha \cdot q^{2q} \cdot q_{4.29}$, no edge will be contained in both H_u and H_v. This finishes the proof of Eq. (4.4) and Claim 4.31. □

Proof of Claim 4.32. Note that $\mathrm{Var}_{\mathcal{S}}[q_\Delta] = 1/(s^2 c_{4.26}(q)^2) \cdot \mathrm{Var}_{\mathcal{S}}[\sum_{v \in S} X_{v,\Delta}]$. We further have

that

$$\begin{aligned}\mathrm{Var}_{\mathcal{S}}\left[\sum_{v\in S} X_{v,\Delta}\right] &= \mathrm{E}_{\mathcal{S}}\left[\left(\sum_{v\in S} X_{v,\Delta}\right)^2\right] - \left(\sum_{v\in S}\mathrm{E}_{\mathcal{S}}[X_{v,\Delta}]\right)^2 \\ &= \sum_{v\in S}\mathrm{E}_{\mathcal{S}}[X_{v,\Delta}^2] + \sum_{\substack{u,v\in S\\ u\neq v}}\mathrm{E}_{\mathcal{S}}[X_{u,\Delta}X_{v,\Delta}] - \sum_{v\in S}(\mathrm{E}_{\mathcal{S}}[X_{v,\Delta}])^2 - \sum_{\substack{u,v\in S\\ u\neq v}}\mathrm{E}_{\mathcal{S}}[X_{u,\Delta}]\,\mathrm{E}_{\mathcal{S}}[X_{v,\Delta}] \\ &= \sum_{v\in S}\mathrm{E}_{\mathcal{S}}[X_{v,\Delta}] - \sum_{v\in S}(\mathrm{E}_{\mathcal{S}}[X_{v,\Delta}])^2 + \sum_{\substack{u,v\in S\\ u\neq v}}\mathrm{E}_{\mathcal{S}}[X_{u,\Delta}X_{v,\Delta}] - \sum_{\substack{u,v\in S\\ u\neq v}}\mathrm{E}_{\mathcal{S}}[X_{u,\Delta}]\,\mathrm{E}_{\mathcal{S}}[X_{v,\Delta}]\,.\end{aligned}$$

Let $u, v \in S$, and let $\mathscr{C}_u, \mathscr{C}_v$ be the sets of colored q-bounded disks rooted at u and v that are isomorphic to Δ, respectively. Consider the execution of q-SC on $\mathcal{S}(G)$ for u and v, respectively. To simplify notation, we define the random variables $H_u = (U_u, F_u) \coloneqq \textsc{StreamCollect}(\mathcal{S}(G), u, q)$ and $H_v = (U_v, F_v) \coloneqq \textsc{StreamCollect}(\mathcal{S}(G), v, q)$. By the definition of expectation,

$$\begin{aligned}\mathrm{E}_{\mathcal{S}}[X_{u,\Delta}X_{v,\Delta}] &= \Pr_{\mathcal{S}}[X_{u,\Delta} = 1 \cap X_{v,\Delta} = 1] \\ &= \sum_{H\in\mathscr{C}_u}\sum_{H'\in\mathscr{C}_v}\Pr_{\mathcal{S}}[H_u = H \cap H_v = H'] \\ &= \sum_{\substack{H\in\mathscr{C}_u, H'\in\mathscr{C}_v,\\ E(H)\cap E(H')=\emptyset}}\Pr_{\mathcal{S}}[H_u = H \cap H_v = H'] + \sum_{\substack{H\in\mathscr{C}_u, H'\in\mathscr{C}_v,\\ E(H)\cap E(H')\neq\emptyset}}\Pr_{\mathcal{S}}[H_u = H \cap H_v = H']\,.\end{aligned}$$

We further note that for any two H and H', if $E(H) \cap E(H') = \emptyset$, then the events $H_u = H$ and $H_v = H'$ are independent. Therefore, we have that

$$\sum_{\substack{H\in\mathscr{C}_u, H'\in\mathscr{C}_v\\ E(H)\cap E(H')=\emptyset}}\Pr_{\mathcal{S}}[H_u = H\cap H_v = H'] = \sum_{\substack{H\in\mathscr{C}_u, H'\in\mathscr{C}_v\\ E(H)\cap E(H')=\emptyset}}\Pr_{\mathcal{S}}[H_u = H]\Pr_{\mathcal{S}}[H_v = H'] \leq \mathrm{E}_{\mathcal{S}}[X_{u,\Delta}]\cdot\mathrm{E}_{\mathcal{S}}[X_{v,\Delta}]\,.$$

Furthermore, by Claim 4.31 it holds that

$$\sum_{\substack{H\in\mathscr{C}_u, H'\in\mathscr{C}_v,\\ E(H)\cap E(H')\neq\emptyset}}\Pr_{\mathcal{S}}[H_u = H \cap H_v = H'] = \Pr_{\mathcal{S}}[H_u \text{ and } H_v \text{ visited the same edge}] \leq \sqrt{\alpha q^{2q}\cdot q_{4.29}}\,,$$

where the last inequality follows from our condition that event $\mathscr{E}$ holds. Thus, we have that for any two $u, v \in S$,

$$\mathrm{E}_{\mathcal{S}}[X_{u,\Delta}X_{v,\Delta}] \leq \mathrm{E}_{\mathcal{S}}[X_{u,\Delta}]\,\mathrm{E}_{\mathcal{S}}[X_{v,\Delta}] + \sqrt{\alpha q^{2q}\cdot q_{4.29}}\,.$$

This implies that

$$\begin{aligned}\mathrm{Var}[\sum_{v\in S} X_{v,\Delta}] &= \sum_{v\in S}\mathrm{E}_{\mathcal{S}}[X_{v,\Delta}] - \sum_{v\in S}(\mathrm{E}_{\mathcal{S}}[X_{v,\Delta}])^2 + \sum_{\substack{u,v\in S\\ u\neq v}}\mathrm{E}_{\mathcal{S}}[X_{u,\Delta}X_{v,\Delta}] - \sum_{\substack{u,v\in S\\ u\neq v}}\mathrm{E}_{\mathcal{S}}[X_{u,\Delta}]\,\mathrm{E}_{\mathcal{S}}[X_{v,\Delta}]\\ &\leq \sum_{v\in S}\mathrm{E}_{\mathcal{S}}[X_{v,\Delta}] - \sum_{v\in S}(\mathrm{E}_{\mathcal{S}}[X_{v,\Delta}])^2 + s^2\sqrt{\alpha q^{2q}\cdot q_{4.29}}\\ &\leq \sum_{v\in S}\mathrm{E}_{\mathcal{S}}[X_{v,\Delta}] + s^2\sqrt{\alpha q^{2q}\cdot q_{4.29}}\,.\end{aligned}$$

Plugging this into $\mathrm{Var}_{\mathcal{S}}[q_\Delta] = 1/(s^2 c_{4.26}(q)^2)\cdot \mathrm{Var}_{\mathcal{S}}[\sum_{v\in S} X_{v,\Delta}]$ finishes the proof:

$$\mathrm{Var}[q_\Delta] \leq \frac{\sum_{v\in S}\mathrm{E}_{\mathcal{S}}[X_{v,\Delta}] + s^2\sqrt{\alpha q^{2q}\cdot q_{4.29}}}{s^2 c_{4.26}(q)^2} \leq \frac{\mathrm{E}_{\mathcal{S}}[q_\Delta]}{s\cdot c_{4.26}(q)} + \frac{\sqrt{\alpha q^{2q}\cdot q_{4.29}}}{c_{4.26}(q)^2}\,. \qquad \square$$

4.4.4. The Streaming Property Tester

We transform constant-query property testers (with one-sided error) into constant-space streaming property testers to prove Theorem 4.1. The main idea is to explore the streamed graph by q-SC and look for the forbidden subgraphs in $\mathscr{F}_n$ that characterize Π (see Theorem 4.15). However, in the underlying analysis, we use the (reversible) decomposition of the forbidden subgraphs in $\mathscr{F}_n$ into $\mathscr{F}_n'$ (see Theorem 4.24) to prove the following: if $\mathscr{T}$ finds the colored q-bounded disks $\Delta_1,\dots,\Delta_k$ that compose a forbidden subgraph $F\in\mathscr{F}_n'$ with probability p, then the streaming tester will find at least as many copies of $\Delta_1,\dots,\Delta_k$ as $\mathscr{T}$ (see Lemma 4.29) and it can stitch F with these copies. With these tools at hand, we can incorporate our analysis from previous sections to complete the proof of Theorem 4.1.

Proof of Theorem 4.1. We let $q_0 = q_0(\epsilon)$ denote the query complexity of Π. We present our testing algorithm. Let $q = c\cdot q_0$ for some constant c from Theorem 4.15. Let

$$\alpha = \frac{c_{4.26}(q)^4\delta^8}{10^9|\mathscr{H}_q|^2\cdot q^{2q}q_{4.29}}, \text{ where } \delta = \frac{1}{200|\mathscr{H}_q|}\,.$$

If $n\leq n_0 := q_{4.16}q/\alpha^2$, then we simply store the whole graph. If $n>n_0$, we proceed as follows. Let $\mathscr{F}_n$ be the set of forbidden subgraphs that characterize Π as stated in Theorem 4.15. We sample

$$s = \max\left\{\frac{1}{20\sqrt{\alpha q^{2q}\cdot q_{4.29}}}, \frac{5000|\mathscr{H}_q|}{c_{4.26}(q)\delta^3}\right\}$$

vertices $S\subseteq V$ independently and uniformly at random and we run StreamCollect$(\mathcal{S}(G), v, q)$ for each $v\in S$ to obtain a subgraph $H_v = (V_v, E_v)$ of G. If $H = \cup_{v\in S}H_v$ contains a forbidden subgraph $F\in\mathscr{F}_n$, the tester rejects, otherwise it accepts. See Algorithm 4.3.

Algorithm 4.3 Testing graph property Π in random order stream

function StreamTest($\mathcal{S}(G), n, \epsilon, \mathcal{F}_n$)
  $S \leftarrow$ sample s vertices independently and uniformly at random from V
  for all $v \in S$ **do**
    $H_v \leftarrow (V_v, E_v) =$ StreamCollect($\mathcal{S}(G), v, q$)
  end for
  $H \leftarrow (\cup_v V_v, \cup_v E_v)$
  if there exists $F \in \mathcal{F}_n$ such that H contains a subgraph F **then**
    reject
  else
    accept
  end if
end function

The space complexity of the algorithm is $\mathcal{O}_q(s \cdot q^q) = \mathcal{O}_{q_0}(1)$ words, i. e. $\mathcal{O}_{q_0}(\log n)$ bits. For the correctness of the algorithm, we note that for any property Π that is constant-query testable with one-sided error, with probability 1, we will not see any $F \in \mathcal{F}_n'$ if the graph G satisfies Π.

On the other hand, if G is ϵ-far from satisfying Π, then by Theorem 4.24, with probability at least $2/3$, the subgraph S spanned by the union of q-bounded disks rooted at q uniformly sampled vertices from G will span a subgraph that is isomorphic to some $F \in \mathcal{F}_n'$. Note that, in contrast to the algorithm above, the analysis uses the decomposition of forbidden subgraphs in $\mathcal{F}_n$ into colored q-discs given by Theorem 4.24. The key idea is to use the q-bounded disks found by q-SC and the implicit colors (which are not observed by q-SC, but can be used in the analysis to identify vertices in V_α) to stitch forbidden subgraphs from $\mathcal{F}_n'$ that are discovered by q-RBFS. We prove that with sufficient probability, for each colored q-bounded disk Δ, q-SC finds at least as many copies of Δ as q-RBFS, and therefore, it can reproduce the same types of forbidden subgraphs from $\mathcal{F}_n'$.

By Markov's inequality and the union bound, the probability that at least one q-RBFS in the canonical tester for Π will return a colored q-bounded disk that is isomorphic to a disc Δ' such that $\mathrm{Reach}_G(\Delta') < 2\delta = 1/(100|\mathcal{H}_q|)$ is at most $1/100$. Let $\mathcal{D}$ be the set of all colored q-bounded disks Δ such that $\mathrm{Reach}_G(\Delta) \geq 2\delta$.

By Lemma 4.29, with probability at least $1 - 1/100$, for every $\Delta \in \mathcal{D}$, the number of graphs H_v obtained by q-SC that contain a subgraph isomorphic to Δ is at least $100|\mathcal{H}_q| \cdot \mathrm{Reach}_G(\Delta) \geq 1$. By (implicitly) coloring all vertices in V_α, it follows from Theorem 4.24 that H contains a forbidden subgraph from $\mathcal{F}_n'$ with probability $1 - 1/100 - 1/100 > 2/3$. □

4.4.5. Property Testing in the Random-Edge Model

For the random-edge model, one can obtain the following extension of Theorem 4.15.

4.33 Theorem (Canonical Tester). Let $\Pi = (\Pi_n)_{n\in\mathbb{N}}$ be a graph property that can be tested in the random-edge model with query complexity $q = q(\epsilon, n)$ and error probability at most $1/3$. Then for every ϵ, there exists an infinite sequence $\mathcal{F} = (\mathcal{F}_n)_{n\in\mathbb{N}}$ such that for every $n \in \mathbb{N}$,

- $\mathcal{F}_n$ is a set of rooted graphs such that each graph $F \in \mathcal{F}_n$ is the union of $\hat{q}$ many $\hat{q}$-bounded disks;

- the property Π_n can be tested with error probability at most $1/3$ by the following canonical tester:
  1. sample $\hat{q}$ vertices independently and uniformly at random and mark them red roots;
  2. for each sampled vertex v, perform a $\hat{q}$-RBFS starting at v;
  3. sample $\hat{q}$ vertices i.wp.r. and mark them blue roots;
  4. for each sampled edges $e = \{u, v\}$, perform $\hat{q}$-RBFS starting at u and v;
  5. reject if and only if the explored subgraph is root-preserving isomorphic to some $F \in \mathscr{F}_n$,

where $\hat{q} = cq$ for some constant $c > 1$. The query complexity of the canonical tester is $q^{\mathcal{O}(q)}$. Furthermore, if $\Pi = (\Pi_n)_{n\in\mathbb{N}}$ can be tested in the random-edge model with one-sided error, then the resulting canonical tester for Π has one-sided error too, i.e., the tester always accepts graphs satisfying Π. ■

To prove this theorem, it is sufficient to observe that sampling an edge uniformly at random is equivalent to sampling a vertex proportional to its degree and choosing a uniformly random neighbor. In particular, let X be a vertex that is sampled proportionally to its degree, i. e., $\Pr[X = v] = d(v)/m$. Let Y be a uniformly random neighbor of X. Then, for every $\{u, v\} \in E$, $\Pr[(X, Y) = \{u, v\}] = 1/m$. Now, the proof is analogous to the proof of Theorem 4.15.

We can also have a refined canonical tester for the random-edge model that uses colors to distinguish intersecting vertices, which corresponds to the one from Theorem 4.24 for the random-neighbor model. Again, such a canonization can be proved in a similar way as before, with the only difference that we also need to define the reach probability of a vertex corresponding to a q-RBFS from a uniformly sampled edge. This difference can be easily handled by viewing the process of sampling an edge as the (equivalent) process of sampling a vertex with probability proportional to its degree.

Finally, noting that a set of $\hat{q}$ random edges can be easily obtained by taking the first $\Theta(\hat{q})$ edges from the random-order stream, we can use the above refined canonical tester and the previous emulation proof in Section 4.4 to transform a tester in the random-edge query model to the random-order streaming model.

5. Testing Outerplanarity and Other Forbidden Minors

Graph minors are a structural concept of graphs that is among the best studied research objects in graph theory. Consequently, the minor-freeness tester with two-sided error and constant query complexity for bounded-degree graphs by Benjamini, Schramm, and Shapira [BSS08] received a significant amount of attention and has lead to a tail of follow-up work. Until answered affirmingly by Kumar, Seshadhri, and Stolman [KSS18], one of the open questions in graph property testing was whether minor-freeness can be tested with one-sided error and query complexity $\mathcal{O}(\sqrt{n})$, as conjectured by Benjamini, Schramm, and Shapira. In this chapter, we show that, among other minor-closed properties, outerplanarity can be tested with roughly $\mathcal{O}(n^{2/3})$ queries. The results presented in this chapter are joint work with Reut Levi, Maximilian Wötzel and Yadu Vasudev [Fic+18].

5.1. Introduction

Subgraphs are a fundamental concept of *structure* in graphs. Given a graph H of constant size, H-subgraph freeness can be tested with one-sided error and with constant query complexity [GR02]: roughly, the tester samples $\Theta(1/\epsilon)$ vertices and checks their $|H|$-disks for occurrences of H. However, subgraphs capture only local structure. Even some fundamental properties like being cycle-free (i. e., a tree) cannot be characterized by a finite family of subgraphs because they depend on structures of super-constant size. On the other hand, cycles do share common structure except for their length. This gives rise to a concept more general than subgraphs: *minors*. We say that a graph G is H-minor free if one cannot obtain H from G by a sequence of (i) vertex or edge removals and (ii) edge contractions, i. e., identifications of two adjacent vertices. With this definition at hand, a graph is cycle-free (i. e., a tree) if and only if it is K_3-minor free. It is also well known that a graph is planar if and only if it is K_5-minor free and $K_{3,3}$-minor free [Kur30].

In this chapter, we study one-sided error testing of minor-freeness for *separable* minors (see Definition 5.6) in bounded-degree graphs with query complexity $\mathcal{O}(n^{2/3}/\epsilon^5)$. This class of minors includes $K_{2,k}$, the k-circus graph and the $(k \times 2)$-grid, which gives rise to one-sided error testers for outerplanarity and being a cactus graph.

5.1.1. Results in This Chapter

The following main theorem is derived from the more general Theorem 5.28, which may be of independent interest.

5.1 Theorem. Given a bounded-degree graph G and a parameter $\epsilon > 0$, for every constant k, there is a one-sided error ϵ-tester with query complexity and running time $\tilde{\mathcal{O}}(n^{2/3}/\epsilon^5)$ for $\mathcal{F}$-minor

freeness, where $\mathcal{F}$ is a family of forbidden minors such that there exists $H \in \mathcal{F}$ and H is a minor of $K_{2,k}$, the k-circus graph or the $(k \times 2)$-grid. ■

When $\mathcal{F} = \{K_{2,3}, K_4\}$, it follows that outerplanarity can be tested with one-sided error. Similarly, when $\mathcal{F}$ consists of the diamond graph (K_4 with one missing edge), being a cactus graph can be tested with one-sided error.

5.2 Corollary. Testing outerplanarity with one-sided error has query complexity and running time $\tilde{\mathcal{O}}(n^{2/3}/\epsilon^5)$. Testing the property of being a cactus graph with one-sided error has query complexity and running time $\tilde{\mathcal{O}}(n^{2/3}/\epsilon^5)$. ■

5.1.2. Property Testing of Minor-Freeness

For two-sided error testers, the problem of testing minor freeness in bounded-degree graphs is essentially settled. Benjamini, Schramm, and Shapira [BSS08] proved that H-minor freeness and, more generally, every minor-closed property can be tested with constant query complexity. Their result was improved by Hassidim et al. [Has+09], who developed local graph partitioning oracles that give query access to a (black-box) partition of a hyperfinite input graph (see page 41). Since minor-free graphs are hyperfinite, the tester can basically decompose the graph into parts of constant size and check for forbidden minors (of constant size) after it has confirmed that the graph is hyperfinite (see Section 3.1.2). The partitioning oracle was improved for minor-free graphs by Levi and Ron [LR15], resulting in a query complexity that is quasi-polynomial in $1/\epsilon$. For constant-query testable properties, research in property testing distinguishes between polynomial and super-polynomial (often exponential) dependence of the query complexity on ϵ. A purely polynomial dependence was proved for outerplanarity by Yoshida and Ito [YI10] and for bounded-treewidth graphs by Edelman et al. [Ede+11]. Eventually, Kumar, Seshadhri, and Stolman [KSS19] proved that the query complexity is polynomial in d and ϵ for every fixed set of minors. This work draws on the random-walk based approach for one-sided error testing of minor-freeness [KSS18], which is discussed below.

Czumaj et al. [Czu+14] developed one-sided error property testers to test H-minor freeness if H is a tree or a cycle. If H is a tree of size k, the query complexity of their tester is at most $\text{poly}(d^{k(16d/\epsilon)^{4k+2}})$. The essence of this tester's analysis is a technical result that roughly states the following: if, for some function f, the $f(\epsilon, d, k)$-disk of a vertex contains no subset S such that $|E(S, V \setminus S)| \leq \epsilon d|S|$, then this disk contains a tree-minor of any tree of size at most k. Thus, if the graph is ϵ-far from being H-free, one can either decompose G into small pieces by removing an $\epsilon/2$-fraction of edges, which reduces the problem to finding minors of constant size, or many disks contain minors of every small tree. Phrasing this differently, it is sufficient to test for small minors of H, which reduces to the simple task of testing subgraph freeness, which again has constant query complexity (see [GR02]). This is reminiscent of the two-sided error tester by Hassidim et al. [Has+09], which employs a partitioning oracle to decompose the graph into pieces of constant size. In contrast, the oracle in [Has+09] may fail to return a partition with small cut, although it may exist. This is fine because the tester in [Has+09] has two-sided error. The main difference between these two decompositions is that for arbitrary minor-free graphs, it may be necessary to charge the cut size of a part to another part in the partition, while the result of [Czu+14] for trees states that every part can pay for its own cut (or the disk contains the forbidden minor).

If, on the other hand, we want to test H-minor freeness and H is a cycle of length k, the tester

of Czumaj et al. [Czu+14] has query complexity $\tilde{\mathcal{O}}(\text{poly}(d^k/\epsilon) \cdot \sqrt{n})$. The dependency on n is not a deficiency of their tester: Goldreich and Ron [GR02, Proposition 4.3] proved a lower bound of $\Omega(\sqrt{n})$ for testing cycle freeness with one-sided error, and Czumaj et al. [Czu+14] extend this result to testing H-minor freeness for any graph H that contains a cycle. Their tester for cycle freeness is also nearly optimal in terms of the length of the cycle it finds. A result by Bollobás and Thomason [BT97, Theorem 5] implies that any graph that is ϵ-far from being cycle-free contains a cycle of length at most $\log(2\epsilon n)/(2\epsilon)$. On the other hand, there exist graphs that are ϵ-far from being cycle-free but have girth $2 \log n/(\log d)$ (e. g., Ramanujan graphs, see Lemma 6.22). The tester by Czumaj et al. finds a cycle of length at most $\text{poly}(d^k \log(n)/\epsilon)$.

While the tester for tree minors is based on BFS, the tester for cycle freeness uses random walks. In particular, the tester employs a local graph transformation that reduces testing cycle freeness to testing bipartiteness. For the latter problem, Goldreich and Ron [GR99] developed a one-sided error tester with query complexity $\mathcal{O}(\sqrt{n})$. The bipartiteness tester by Goldreich and Ron uses random walks to find odd cycles. The random walks are run from a set of sampled vertices. If the input graph is far from being bipartite, there exist two walks that start from the same vertex, collide and form an odd cycle with constant probability. It is not hard to show that this is true for expander graphs, which have the property that random walks mix rapidly, but the analysis of the non-expander case is quite involved.

After the result that is presented in this chapter was published, Kumar, Seshadhri, and Stolman [KSS18] showed that one can test H-minor freeness for every graph H with query complexity $\mathcal{O}(n^{1/2+o(1)})$. This resolves an open problem by Benjamini, Schramm, and Shapira [BSS08] up to a factor of $n^{o(1)}$: Benjamini, Schramm, and Shapira [BSS08] conjectured that testing minor freeness with one-sided error has query complexity $\mathcal{O}(\sqrt{n})$. Like the work by Czumaj et al. [Czu+14], the tester uses random walks. However, their analysis is not based on a reduction to the bipartiteness tester by Goldreich and Ron [GR99], but inspired by their algorithm: Suppose that we want to find a K_k-minor. When using random walks, a natural approach is to find k vertices X and run $\mathcal{O}(\sqrt{n})$ random walks from these vertices. We would hope that for each pair of vertices $u, v \in X$, a random walk from u and a random walk from v collide because of a birthday paradox. We might be able to deal with this problem if the input graph had reasonably small mixing times. However, even if this was the case, we would still need to guarantee that these paths do not intersect each other so that we can contract them *separately* to construct the minor's edges. Kumar, Seshadhri, and Stolman [KSS18] show that there are two cases: The first case is that random walks spread to many vertices for a constant fraction of start vertices, but do not necessarily mix rapidly beyond that. In this case, there is a linear number of vertices S and a large set of vertices R such that random walks starting from S (i) visit vertices from R often and (ii) visit each vertex in R with roughly the same probability. Therefore, they obtain the sample $X \subseteq R$ by running random walks from S. It is then shown that the guarantees on the behavior (with respect to R) of the random walks' starting from X suffice to apply a birthday paradox argument and prove that the paths obtained will not intersect after spreading out from X. The second case is that for many vertices, the random walks are trapped within small sets of vertices. In this case, it is shown that one can identify these small sets by applying partitioning techniques by Spielman and Teng [ST13] and run a $\mathcal{O}(n^2)$ time minor-freeness verifier by Kawarabayashi, Kobayashi, and Reed [KKR12]. However, the analysis is also quite sophisticated and combines several known techniques [LS90; KPS08] with the analysis from the first case.

5.1.3. Open Problems and Future Work

As mentioned in Section 5.1.2, the most related problem of obtaining a tight upper bound for one-sided error property testing of minor freeness has been essentially resolved by Kumar, Seshadhri, and Stolman [KSS18] recently. On top of that, the same authors proved that testing minor freeness with two-sided error has query complexity $\text{poly}(\epsilon, d)$. However, the best result regarding partition oracles for minor-free graphs is still the pseudo-polynomial bound by Levi and Ron [LR15].

5.3 Problem. Prove or disprove the existence of a partition oracle with polynomial query complexity for minor-free graphs. ■

See also Problem 4.6 for another problem related to planar graphs.

5.1.4. Overview of the Analysis

One of the first approaches that may come to one's mind is to exploit that minor-free graphs are hyperfinite. For example, we might hope that we can use a partitioning oracle to recover the parts of an $(\epsilon/2, s)$-hyperfinite partition for some $s \in \mathcal{O}(1)$ and remove the $\epsilon dn/2$ edges in the cut between the parts (given a vertex v, a partitioning oracle returns the part of v in some partition; see Section 2.1.4). We would like to argue that the remaining graph is still $\epsilon/2$-far from being minor-free if it was ϵ-far originally, so that it is sufficient to sample $\mathcal{O}(1)$ parts and check these for forbidden minors. However, there is a caveat: if the input graph is not hyperfinite, there is no guarantee that there exists a partition with a cut of size $\mathcal{O}(\epsilon dn)$.

We use a different partitioning instead, which is not related to hyperfinite partitions. In a nutshell, we show that one can locally partition every graph into parts of size roughly $\mathcal{O}(n^{1/3})$ so that it is sufficient to either (i) explore a few random parts or (ii) find a large cut between two parts to prove the existence of a forbidden minor. The first case is reminiscent of the case that the input graph is far from being minor-free but hyperfinite (technically, say, $(\epsilon/2, \mathcal{O}(n^{1/3}))$-hyperfinite). The second case deals with graphs that are far from being minor-free, but our partition contains parts with a large cut in between. We show that this is already a sufficient condition for the existence of the minors mentioned in Theorem 5.1, so we only have to certify the cut size in this case.

The partitioning we use is adapted from Lenzen and Levi [LL18] and provides us with a few useful guarantees. First, all parts are connected so that we can construct a spanning tree of each part. Second, all parts have bounded diameter, namely, $\mathcal{O}(\log n)$. The first case, checking a part for forbidden minors given access to a partitioning oracle, is straightforward. Consider the second case, i. e., the cut of the partition is greater than $\epsilon dn/2$ but the input graph is ϵ-far from being minor-free. Tailoring to the two guarantees from the partitioning, we define *separability* of a graph (see Definition 5.6) so that for every partition we may construct, the cut between two parts is small. We show that if a graph does not contain one of the minors from Theorem 5.1, then it is separable.

As mentioned in Section 5.1.2, one of the main issues with finding minors is to construct disjoint paths that constitute the minor. If a graph is not separable, then there exist at least two parts with a large cut. We show that one can leverage the structure of the parts and the cut in between to construct disjoint paths that form a forbidden minor. If the graph is far from being minor-free and the cut between the parts of the partition is large, then it is likely that a random edge is part of a minor *and* lies in the cut between two parts. Therefore, we can

identify the two parts and compute the size of the cut between them to certify the existence of a forbidden minor.

It remains to construct a suitable partition. The partitioning of [LL18] can be constructed locally, i. e., given a vertex v, one can explore the graph starting from v to construct its part. As mentioned above, we are interested in connected parts of size roughly $\mathcal{O}(n^{1/3})$, and we need to compute the cut size between two parts. We will call these parts *core clusters*. Technically, we obtain core clusters from a more coarse partition, called *Voronoi cells*, which is built by sampling *center* vertices and assigning the remaining vertices to the nearest center. Unfortunately, even the distance to the nearest center might be rather large for some vertices. For further analysis, we need to single these vertices out and process them as *remote clusters*. However, the cut size between remote clusters is $\mathcal{O}(\epsilon dn)$ and therefore the cut can be removed. Furthermore, we show that one can explore a superset of a remote cluster, which is sufficient to find forbidden minors in a remote cluster.

We refine Voronoi cells into core clusters by a simple cutting procedure on the spanning trees of the Voronoi cells. Core clusters are small enough so that we can always explore the whole core cluster of a given vertex. In light of the concept of separability introduced above, we would like to compute the pairwise cuts of an edge's core clusters. However, the potential size of all pairwise cuts between core clusters can be quite large although separability does not certify the existence of a forbidden minor (i. e., each cut is small, but the number of pairs of core clusters is too large to yield a sufficiently small bound on the total cut size). To finish the argument, we merge some core clusters into *super clusters*. Super clusters can be too large to be fully explored, but they support membership queries. Since Voronoi cells and core clusters are connected, this allows us to compute the cut size between a core cluster and a super cluster and to apply separability. Finally, this partition satisfies our requirements: roughly speaking, if all pairs of core clusters and super clusters are separable, the total cut size is $\mathcal{O}(\epsilon dn)$ and we can remove the cut. Otherwise, we obtain a certificate of a forbidden minor due to a large cut between two clusters.

5.2. Preliminaries

In this chapter, all graphs are bounded-degree graphs unless stated otherwise. The total order on the vertices (see Section 1.3.1) induces a total order r on the edges of the graph in the following straightforward manner: $r((u,v)) < r((u',v'))$ if and only if $\min\{u,v\} < \min\{u',v'\}$ or $\min\{u,v\} = \min\{u',v'\}$ and $\max\{u,v\} < \max\{u',v'\}$. The total order on the vertices also induces an order on those vertices visited by a BFS starting from any given vertex v, and whenever we refer to a BFS in this chapter, we mean that it is performed according to this order. Whenever referring to one of the above orders, we may refer to the *rank* of an element in the respective order. This is simply the index of the respective element when listing all the elements according to the order starting with the smallest element.

The $(k \times 2)$-grid and the k-circus graph referenced in Theorem 5.1 are defined as follows.

5.4 Definition ($k \times 2$**-grid).** The $(k \times 2)$-grid is the graph with vertex set $\{x_i, y_i\}_{i=1}^{k}$ and edge set $\{\{x_i, x_{i+1}\}, \{y_i, y_{i+1}\}\}_{i=1}^{k-1} \cup \{\{x_i, y_i\}\}_{i=1}^{k}$. ■

5.5 Definition (k**-circus graph [Bod93]).** The k-circus graph is the graph with vertex set $\{x\} \cup \{y_i, z_i\}_{i=1}^{k}$ and edge set $\{\{y_i, z_i\}, \{z_i, z_{i+1}\}, \{x, y_i\}\}_{i=1}^{k}$. ■

We say that two subsets of vertices A and B are *adjacent* if $E(A,B) \neq \emptyset$. A graph is *separable*

if every pair of disjoint sets of connected vertices such that one set has small diameter has a small cut.

5.6 Definition (separability). A graph $G = (V, E)$ is (f, g)-*separable* if for every pair of disjoint sets of vertices $A, B \subseteq V$ such that $G[A]$ and $G[B]$ are connected and the diameter of $G[A]$ is at most g it holds that $|E(A, B)| \leq f$. ■

We use the following two results by Erdös and Szekeres [ES35] and Hall [Hal35], respectively, to prove the separability of several minors.

5.7 Lemma [ES35]. Given a sequence of natural numbers $S = (s_i)_{i \in [n]}$ of length n, there exists a subsequence of length $\sqrt{n}$ that is either monotonically increasing or monotonically decreasing. ■

5.8 Lemma (Hall's marriage theorem; [Hal35]). Given a bipartite graph $G = (V_1 \cup V_2, E)$, there exists a subset $E' \subseteq E$ that matches every vertex in V_1 to vertex in V_2 if and only if $|\Gamma(V_1)| \geq |V_1|$ for every $S \subset V_1$. ■

The following definition and result are used to construct a partition of the input graph.

5.9 Definition (exponential distribution). Given a parameter $\lambda > 0$, the probability density function f_λ and the cumulative distribution function F_λ of the exponential distribution are defined as

$$f_\lambda(x) = \begin{cases} \lambda e^{-\lambda x} & x \geq 0 \\ 0 & x < 0 \end{cases} \qquad F_\lambda(x) = \begin{cases} 1 - e^{-\lambda x} & x \geq 0 \\ 0 & x < 0\,. \end{cases}$$

Let X be a random variable that follows the exponential distribution. The expected value of X is $\mathrm{E}[X] = \lambda^{-1}$. The distribution is memoryless, i. e., $\Pr[X > x + y \mid X > x] = \Pr[X > y]$ for all $x, y \geq 0$. ■

The following tail inequality for the exponential distribution is implicit in [MPX13]. We recite a proof from [EN17] for completeness.

5.10 Lemma [MPX13]. Let $x_1 \leq \ldots \leq x_n$ be arbitrary real values and let $X_1, \ldots, X_n$ independent random variables that obey the exponential distribution. Let $M = \max_i\{X_i - x_i\}$ and $I = \{i \mid X_i - x_i \geq M - 1\}$. Then, for any $t \in [n]$,

$$\Pr[|I| \geq t] = (1 - e^{-\lambda})^{t-1}\,.$$ ■

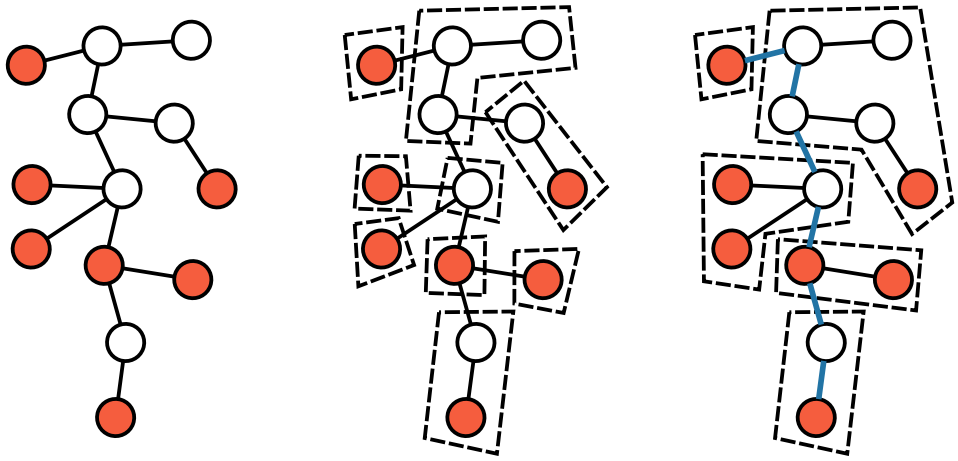
Proof [EN17, Lemma 2.1]. Let Y_t be the t-th largest random variable among $\{X_i - x_i\}_i$. For any $y \in \mathbb{R}$, if we condition on $Y_t = y$, then the event $|I| \geq t$ is the event that $Y_1, \ldots, Y_{t-1} \in [y, y + 1]$. Since the exponential distribution is memoryless, $\Pr[Y_i > Y_t + 1 \mid Y_i > Y_t] = \Pr[Y_i > 1]$ for every $i < t$, and therefore $\Pr[Y_i \leq Y_t + 1 \mid Y_i > Y_t] = \Pr[Y_i \leq 1]$. By the independence of the $\{X_i\}_i$, it follows that

$$\Pr[|I| \geq t \mid Y_t = y] = (1 - e^{-\lambda})^{t-1}\,.$$

Since this bound does not depend on t, the law of total probability implies the claim. □

5.3. Separability and Evidence for Minors

In this section we prove combinatorial lemmas that give sufficient conditions for the existence of a $(k \times 2)$-grid minor, a k-circus minor and a $K_{2,k}$-minor in a graph. To this end, we will



(a) Input graph T

(b) Contraction of vertices with degree 2

(c) Contraction of subtrees into L

Figure 5.1.: Construction of a path minor L (colored blue) on relevant vertices Q (colored in red) in a given tree T. Initially, all vertices form a partition on their own (see Fig. 5.1a; partitions not shown). Then, all edges that are incident to a non-relevant vertex of degree at most two are contracted (see Fig. 5.1b; contracted parts are surrounded by dashed polygons). Finally, we choose a path of maximum length L and contract the remaining subtrees into it (see Fig. 5.1c; L is shown in blue).

consider the following auxiliary graph that is defined with respect to a partition of another graph's vertex set.

5.11 Definition. Let $G = (V, E)$ be a graph and $\mathscr{P}$ a partition of its vertex set. The graph $G[\mathscr{P}]$ is defined as the graph with vertex set $\mathscr{P}$, and $\{P, P'\} \subseteq \mathscr{P}$ is an edge if and only if there are vertices $v \in P, u \in P'$ such that $\{u, v\} \in E$. ■

In other words, $G[\mathscr{P}]$ results from contracting all edges $\{u, v\} \in E \cap (P \times P)$ for every $P \in \mathscr{P}$. Notice that if $G[P]$ is connected for every $P \in \mathscr{P}$, then $G[\mathscr{P}]$ is isomorphic to the minor of G obtained by contracting every edge of $G[P]$ for all $P \in \mathscr{P}$, so we also refer to this minor by $G[\mathscr{P}]$. In order to show that graphs that do not contain one of the minors mentioned above are separable, we first show that we can construct special path and star minors from trees. In particular, we show that one can mark arbitrary vertices as *relevant* so that not too many relevant vertices are contracted into each other.

5.12 Lemma. Given a tree $T = (V, E)$ and a subset of *relevant* vertices $Q \subseteq V$, there exists a partitioning $\mathscr{P}$ of V such that $T[\mathscr{P}]$ is a path minor of T of length $\log_d |Q|$ and for every $P \in \mathscr{P}$, it holds that $P \cap Q \neq \emptyset$. ■

Proof. We will construct the claimed partition of T (see Fig. 5.1). Initially, let $\mathscr{P} = \{\{v\} \mid v \in V\}$. While there exists an edge (P, P') in $T[\mathscr{P}]$ such that P' does not contain a relevant vertex and the degree of P' is at most two, update $\mathscr{P} = \mathscr{P} \setminus P'$ and $P = P \cup P'$. Repeat this process until no such edges remain (see Fig. 5.1b). Note that the resulting $T[\mathscr{P}]$ is still a tree of maximum degree d and that every part contains at most one relevant vertex, that is, $|P \cap Q| \leq 1$ for all $P \in \mathscr{P}$.

Since $T[\mathscr{P}]$ is a tree with at least $|Q|$ vertices, it has diameter $\ell \geq \log_d |Q|$. Let $L =$

$(P_1, P_2, \dots, P_\ell)$ be a simple path in $T[\mathscr{P}]$ between a pair of leaves, of maximum length. For every part $P_i \in L$, let T_i be the subtree rooted at P_i in $T[\mathscr{P}]$ that is obtained by (virtually) removing the (at most two) edges between P_i and its neighbors in L. Every part $P_i \in L$ that contains no relevant vertex has at least one neighbor that is not in L, otherwise it is of degree at most two and would have been merged previously. For every such part, the tree T_i must contain at least one part that has non-empty intersection with Q, otherwise it would have been contracted. We update the partition by setting $P_i = \bigcup_{P \in T_i} P$ for every part $i \in [\ell]$ and setting $\mathscr{P} = \{P_i\}_{i=1}^{\ell}$ (see Fig. 5.1c).

Now, we have that for every $i \in [\ell]$, $P_i \cap Q \neq \emptyset$, and we have also not shortened the path. Therefore, $T[\mathscr{P}]$ is the desired path minor of T. □

5.13 Lemma. Given a rooted tree $T = (V, E)$ with height h (for some $h \in \mathbb{N}$) and a subset of *relevant* vertices $Q \subseteq V$, there exists a partition $\mathscr{P}$ of V such that $T[\mathscr{P}]$ is a star minor of T with $|Q|/(2h)$ leaves and for every $P \in \mathscr{P}$, it holds that $P \cap Q \neq \emptyset$. ■

Proof. Without loss of generality, assume that the root is not in Q. Let S be the set of maximal paths in T that start at the root and end at some vertex in Q. The length of each path in S is bounded by h, and therefore there exist at least $|Q|/h$ such paths. Since all paths in S start at the same vertex and they are maximal, their end vertices are pairwise distinct. Removing all vertices that are not contained in any path of S and contracting all but the end edges of the paths in S, gives a star minor with at least $|Q|/h$ relevant leaves. Finally, we can contract an arbitrary leaf into the root to make the part of the root relevant. □

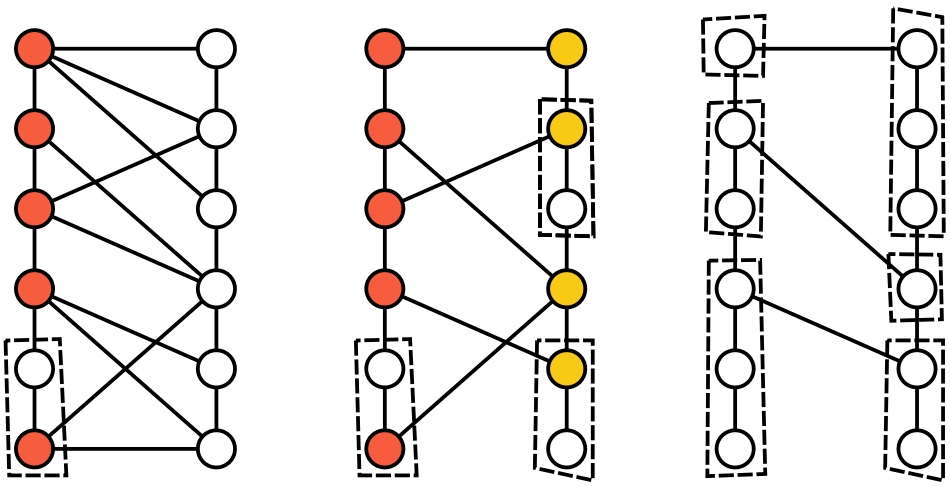
The common idea of the following proofs is to consider a partition of a graph into two parts such that there is a large cut, and to apply Lemma 5.12 and / or Lemma 5.13 to these parts in order to construct the desired minors.

5.14 Lemma. Let $G = (V, E)$ be a graph that does not contain the $(k \times 2)$-grid as a minor. Then, G is $(d^{1+d^{k^2+1}}, n)$-separable. ■

Proof. We prove the contrapositive of the statement of the lemma (see Fig. 5.2). Let V_1, V_2 be a partition of V such that $E(V_1, V_2) > d^{1+d^{k^2+1}}$. Let T_1 (resp. T_2) be a spanning tree of $G[V_1]$ (resp. $G[V_2]$). Let Q_1 be the set of vertices in V_1 that have a neighbor in V_2. Since $|E(V_1, V_2)| \geq d^{1+d^{k^2+1}}$, the size of the set Q_1 is at least $d^{d^{k^2+1}}$. By Lemma 5.12, there exists a partition $\mathscr{P}_1$ of V_1 such that $T_1[\mathscr{P}_1]$ is a path minor of T_1 of length $r \geq \log_d |Q_1| \geq d^{k^2+1}$ (see Fig. 5.2a). Let $(u_1, u_2, \dots, u_r\}$ be this path minor. For each vertex u_j in the path minor $T_1[\mathscr{P}_1]$, it holds that $|\Gamma(u_j) \cap V_2| \geq 1$. Now, for each vertex u_j in the path minor, remove all the edges to V_2 except the one of lowest rank, so that $|\Gamma(u_j) \cap V_2| = 1$.

Since the degree of G is at most d, the number of vertices in V_2 adjacent to Q_1 that remain after these edge deletions is at least d^{k^2}. Denote this set of vertices by Q_2. By Lemma 5.12, there exists a partition $\mathscr{P}_2$ of V_2 such that $T_2[\mathscr{P}_2]$ is a path minor of length $t \geq \log_d |Q_2| \geq k^2$ (see Fig. 5.2b). Let $(v_1, v_2, \dots, v_t)$ be this path. Since $|\Gamma(u_j) \cap T_2[\mathscr{P}_2]| = 1$ for all $u_j \in T_1[\mathscr{P}_1]$, we have $t \leq r$. Furthermore, each vertex $v_j \in T_2[\mathscr{P}_2]$ has at least one neighbor in $T_1[\mathscr{P}_1]$. Therefore, by Lemma 5.8 theorem, there is a matching of size at least k^2 between $T_1[\mathscr{P}_1]$ and $T_2[\mathscr{P}_2]$.

By Lemma 5.7, there is a set of k vertices, say $u_{i_1}, u_{i_2}, \dots, u_{i_k}$ and $v_{j_1}, v_{j_2}, \dots, v_{j_k}$, such that $i_1 \leq i_2 \leq \cdots \leq i_k$ and $j_1 \leq j_2 \leq \cdots \leq j_k$, and $\{u_{i_l}, v_{j_l}\}$ is an edge in the matching. Contract the



(a) Construction of $T[\mathscr{P}_1]$ on V_1 (left side)

(b) Construction of $T[\mathscr{P}_2]$ on V_2 (right side)

(c) A (3×2)-grid minor

Figure 5.2.: Construction of a grid minor. The six left vertices belong to V_1, the six right vertices belong to V_2. In Fig. 5.2a, a path minor $T[\mathscr{P}_1]$ is constructed on V_1 so that not too many vertices that connect to V_2 are contracted into each other (Q_1, colored in red). Then, in Fig. 5.1b, a path minor $T[\mathscr{P}_2]$ is constructed on V_2 so that not too many vertices that connect to $T[\mathscr{P}_1]$ are contracted into each other (Q_2, colored in yellow). Finally, in Fig. 5.1c, Lemma 5.8 is invoked to construct a grid minor on $T[\mathscr{P}_1]$, $T[\mathscr{P}_2]$ and the edges in between.

edges in the path to the vertices $u_{i_1}, u_{i_2}, \dots, u_{i_k}$ and $v_{j_1}, v_{j_2}, \dots, v_{j_k}$ to obtain the $(k \times 2)$-grid minor (see Fig. 5.2c). □

5.15 Lemma. Let $G = (V, E)$ be a graph that does not contain the k-circus as a minor. Then, G is $(2hd^{2+k^2}, h)$-separable for every $h \in \mathbb{N}$. ■

Proof. We prove the contrapositive of the statement of the lemma. This proof is very similar to the proof of Lemma 5.14. The only difference is that (i) we consider a partition V_1, V_2 of V such that $E(V_1, V_2) > 2hd^{2+k^2}$ and the diameter of $G[V_1]$ is at most h and (ii) we apply Lemma 5.13 instead of Lemma 5.12 to $G[V_1]$. In particular, let T_1 (resp. T_2) be a spanning tree of $G[V_1]$ (resp. $G[V_2]$). Let Q_1 be the set of vertices in V_1 that have a neighbor in V_2. Since $|E(V_1, V_2)| \geq 2hd^{2+k^2}$, the size of the set Q_1 is at least $2hd^{1+k^2}$. By Lemma 5.13, there exists a partition $\mathscr{P}_1$ of V_1 such that $T_1[\mathscr{P}_1]$ is a star minor of T_1 with $r \geq d^{1+k^2}$ leaves. Let $\{u_1, u_2, \dots, u_r\}$ be the leaves of this star minor. For each leaf u_j of the star minor $T_1[\mathscr{P}_1]$, it holds that $|\Gamma(u_j) \cap V_2| \geq 1$. Now, for each leaf u_j in the star minor, remove all the edges to V_2 except the one of lowest rank, so that $|\Gamma(u_j) \cap V_2| = 1$. The remaining proof is analogous to the proof of Lemma 5.14. □

5.16 Lemma. Let $G = (V, E)$ be a graph that does not contain the complete bipartite graph $K_{2,k}$ as a minor. Then, G is $(2dkh, h)$-separable for every $h \in \mathbb{N}$. ■

Proof. Observe that k-circus contains $K_{2,k}$ as a minor, and therefore it follows from Lemma 5.15 that G is $(2hd^{2+k^2}, h)$-separable. However, a better bound can be achieved directly as follows: instead of applying Lemma 5.12 to $G[V_2]$ in the proof of Lemma 5.15 (that is, carrying out the shared part of the proofs of Lemma 5.15), one can simply contract V_2 to a single vertex to obtain $K_{2,k}$. □

5.4. Underlying Partitions

In this section we describe a method to partition the graph into small connected parts with certain properties that enable us to apply separability Lemmas 5.14 to 5.16. The partition technique is very similar to the one that appears in [LL18], which is used for the local construction of sparse spanning subgraphs. We make minor adaptation to suit our needs.

Three different partitions are described next. One is a refinement of the other two. As described in more detail in the next sections, the properties of these partitions are as follows. The refined partition into *core clusters* can be locally recovered. The edge cut of this partition is not necessarily small, even if the input graph excludes the forbidden minor. The second partition into *Voronoi cells* will be useful for checking the edge cut of the third partition into *super clusters*, which in turn is guaranteed to have a small edge cut in case the graph excludes the forbidden minor. See Fig. 5.3 for an illustration of the following definitions.

$\epsilon_1, \epsilon_2, \ell, t$

Parameters. The input parameters are ϵ_1 and ϵ_2. We sample ℓ uniformly at random from $[\log n/\log(1+\epsilon_2), \log n/\log(1+\epsilon_2) + d/\epsilon_2]$, and let $t := n^{1/3} \ln n \cdot \ell(d+1)/\epsilon_1$. The parameter ℓ affects the diameter of the parts of the partition. It is picked randomly so as to ensure that only a small fraction of the edges are in the edge cut of the partition.

5.4.1. Voronoi Cells

Centers Pick a set $V_C \subset |V|$ of $\Theta(\epsilon_1 n^{2/3}/\ln n)$ vertices at random. We shall refer to the vertices in V_C as *centers*. For each vertex $v \in V$, its *center*, denoted by $\text{cent}(v)$, is the center which is closest to v among all centers (break ties between centers according to the rank). $V_C, \text{cent}(\bullet)$

Remote vertices Define $R := \{v \mid \Gamma_\ell(v) \cap V_C = \emptyset\}$ where V_C is the set of centers. We call the vertices in R *remote* and abbreviate $\bar{R} := V \setminus R$. $R, \bar{R}$

Voronoi cells The *Voronoi cell* of a vertex $v \in \bar{R}$ is $\text{Vor}(v) := \{u \in \bar{R} \mid \text{cent}(u) = \text{cent}(v)\}$. $\text{Vor}(\bullet)$

We deal with the partitioning of remote vertices later. Given a vertex $v \in \bar{R}$, one can determine its center by exploring its ℓ-hop neighborhood. However, it is much more costly to find all vertices that belong to its Voronoi cell, which may have size $\Omega(n)$. We now describe how to further refine the partition given by the Voronoi cells so that the number of vertices in each cluster is $\tilde{\mathcal{O}}(n^{1/3}d/\epsilon_1)$.

5.4.2. Core Clusters

For each Voronoi cell, consider a rooted BFS tree spanning it. For every $v \in \bar{R}$, let $p(v)$ denote the *parent* of v in this BFS tree. If v is a center then $p(v) = v$. For every $v \in \bar{R} \setminus V_C$, let $T(v)$ denote the subtree of v in the above-mentioned BFS tree when we remove the edge $\{v, p(v)\}$. Now consider a Voronoi cell. We define the *core cluster* of a vertex v as follows (where t is defined as above):

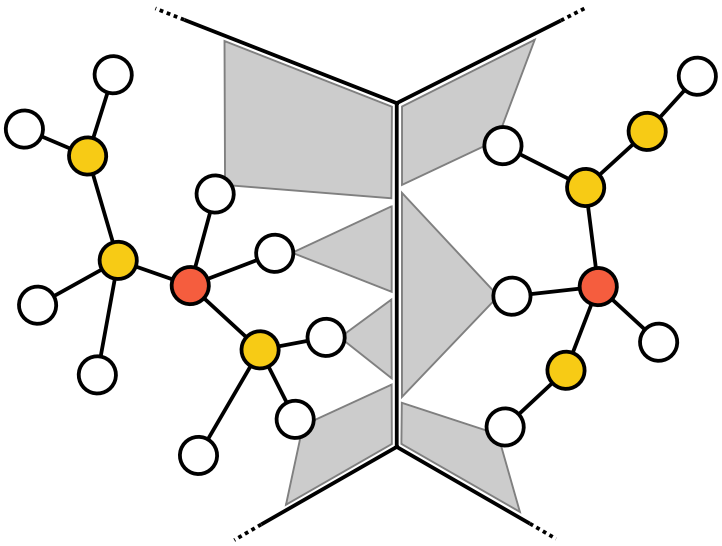
1. If $|\text{Vor}(v)| \le t$ then the core cluster of v is $\text{Vor}(v)$.
2. If $|T(v)| \ge t$, where $|T(v)|$ denotes the number of vertices in $T(v)$, then the core cluster of v is the singleton $\{v\}$.
3. Otherwise, v has a unique ancestor u for which $|T(u)| < t$ and $|T(p(u))| \ge t$. The core cluster of v is the set of vertices in $T(u)$.

For a vertex $v \in \bar{R}$, let $\text{clus}(v)$ denote the cluster of v. For a cluster C, let $\text{cent}(C)$ denote the center of the vertices in C (all the vertices in the cluster have the same center). Let $\text{Vor}(C)$ denote the Voronoi cell of the vertices in C. $\text{clus}(\bullet), \text{cent}(\bullet)$

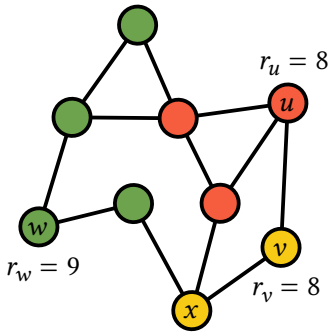
This describes a partition of V into R and $\bar{R}$, a refinement of $\bar{R}$ into Voronoi cells, and a refinement of this partition into core clusters (see Fig. 5.3a). It was shown in [LL18] that the number of core clusters is not much higher than the number of Voronoi cells.

5.17 Lemma ([LL18, Lemma 1]). The number of core clusters, denoted by γ, is at most $|V_C| + n\ell(d+1)/t$. ■ γ

Proof. There are three types of core clusters as described above. The number of core clusters of type 1 is at most the number of Voronoi cells, $|V_C|$. Let P be the set of all maximal paths that start at some center vertex and end at some vertex in a core cluster of type 2. Since all vertices whose core clusters are of type 2 have at least t children in the Voronoi cells' BFS trees, $|P|$ is at most n/t. The number of core clusters of type 2 is bounded by the total length of all paths in P, $\ell \cdot n/t$. Consider the children of end vertices of all paths in P. By the maximality of the paths, these children are in core clusters of type 3. The number of children is bounded by $d \cdot n\ell/t$. □



(a) *Voronoi cells and core clusters.* Centers (red) constitute the Voronoi cells. Vertices with more than t children in their BFS subtrees are singletons (yellow). Every other vertex and the vertices in its subtree form a core cluster (white, subtrees simplified / omitted).



(b) *Remote clusters.* Assume that $r_y \leq 5$ for $y \notin \{u, v, w\}$ and that the ranks are $u < v < w$. Then, $m_v(x) = m_w(x) = 7, m_u(x) = 6$ and x belongs to the remote cluster of v (yellow) because it has lower rank than w. Furthermore, $M(x) = \{u, v, w\}$.

Figure 5.3.: Partitioning a graph into Voronoi cells, core clusters and remote clusters.

Note that core clusters are, like Voronoi cells, connected.

5.18 Lemma. For every vertex $v \in \bar{R}$, $\mathrm{clus}(v)$ is connected. ■

Proof. This follows from the construction since every core cluster is either a Voronoi cell, a singleton vertex, or the subtree of the BFS tree of a Voronoi cell. □

Even more, Voronoi cells are still connected if one removes a cluster that is not a singleton.

5.19 Lemma. Let $v \in \bar{R}$ be a vertex such that $\mathrm{clus}(v)$ is not a singleton. Then $G[\mathrm{Vor}(v) \setminus \mathrm{clus}(v)]$ is connected. ■

Proof. Observe that if $\mathrm{clus}(v)$ is not a singleton, then it is the subtree of the BFS tree of $G[\mathrm{Vor}(v)]$ rooted at its center. Therefore, removing $\mathrm{clus}(v)$ does not disconnect $G[\mathrm{Vor}(v)]$. □

In contrast to Voronoi cells, core clusters are guaranteed to be sufficiently small by construction, which allows us to fully explore them in an efficient manner. An explicit procedure for this is given in Section 5.5.1. However, it might still be possible for the overall edge cut to be large, even if there are only few edges in individual cuts between two core clusters. To this end, we group core clusters and consider the cut between pairs of a core cluster and such a resulting *super cluster* instead.

$\partial\bullet$ **5.20 Definition.** For a core cluster A, define its *adjacent vertices* $\partial A := \{v \mid u \in A \wedge v \in \bar{R} \setminus A \wedge \{u, v\} \in E\}$, i.e., the set of vertices that are adjacent to a vertex in A, excluding A. ■

5.21 Definition. Define the *adjacent centers* of a set of vertices A to be $\{\mathrm{cent}(v) \mid v \in A\}$. ■

p **Marked clusters** Each center is *marked* independently with probability $p := 1/n^{1/3}$. If a center is marked, then we say that its Voronoi cell is marked and all the clusters in this cell are

marked as well.

Super clusters Let A be a cluster which is not marked but is adjacent to at least one marked cluster. Let $\{u, v\}$ be the edge with minimum rank such that $u \in A$ and $v \in B$, where B is a marked cluster. We say that the cluster A *joins* the cluster B. The *super cluster* of B consists of B and all the clusters which join B.

After considering pairs of clusters and super clusters (and bounding the number of edges between them), only few pairs of core clusters (A, B) are left such that neither A nor B are member of a super cluster with high probability.

5.22 Lemma. With probability at least $1 - o(1)$, it holds that $|\text{clus}(\partial A)| \leq n^{1/3} \log n$ for every core cluster A that is not adjacent to a marked cluster. ■

Proof. The probability that cluster A is not adjacent to a marked cell is $(1-p)^{|\text{clus}(\partial A)|} \leq e^{-p|\text{clus}(\partial A)|}$. Therefore, if $|\text{clus}(\partial A)| > p^{-1} \ln n$, then the probability that A is not adjacent to a marked cluster is at most $1/n$. Since there are at most $\tilde{\mathcal{O}}(n^{2/3})$ many core clusters, the lemma follows from a union bound. □

This settles the three partitions of vertices from $\bar{R}$ into Voronoi cells, core clusters and super clusters. We now describe a way to partition the remote vertices into remote clusters (see Fig. 5.3b) so that (with high probability) the total number of edges that go out from each remote cluster is at most $\mathcal{O}(\epsilon_2 nd)$ even if the graph is far from being H-minor free. Basically, this implies that one can test a remote cluster isolated from the remaining graph because all outgoing edges can be removed such that G, which was $(\epsilon_1 + \epsilon_2)$-far from being H-minor free, is still ϵ_1-far from the property. The partitioning uses ideas of Elkin and Neiman [EN17].

5.4.3. Remote Clusters

We will first describe the algorithm of Elkin and Neiman [EN17]. Given an integer h and a parameter $0 < \delta \leq 1$, each vertex v draws r_v according to the exponential distribution with parameter $\beta = \ln(n/\delta)/h$. Each vertex v receives r_u from every vertex u within distance at most h, and stores the values $m_u(v) = r_u - \text{dt}_{G[R]}(u, v)$. We use this technique to obtain a partition of R as follows. Every vertex $v \in R$ is *assigned* to the vertex $u \in R$ such that $m_u(v) = \max_{w \in R}\{m_w(v)\}$, if there is more than one such vertex, pick the one with minimum rank (see Fig. 5.3b). We say that u is the *leader* of v denoted by $L(v)$ and that $\{w \in R \mid L(w) = L(v)\}$ is the *remote cluster* of v. We note that we run this algorithm on $G[R]$ (namely, we calculate $m_u(v) = r_u - \text{dt}_{G[R]}(u, v)$), therefore a vertex u cannot be assigned to a vertex on a different connected component in $G[R]$.

Like core clusters, remote clusters are also connected.

5.23 Lemma. For every $v \in R$, the subgraph induced on the remote cluster of v is connected. ■

Proof. Consider any shortest path between v and $v' := L(v)$ in $G[R]$. Let w be the neighbor of v on this shortest path. We claim that $L(w) = L(v)$. From this the lemma follows by induction on the distance to $L(v)$. Assume to the contrary that $w' \neq v'$ where $w' := L(w)$. Then either $m_{w'}(w) > m_{v'}(w)$ or, $m_{w'}(w) = m_{v'}(w)$ and the rank of w' is higher than the rank of v'. Since $m_{v'}(v) = m_{v'}(w) - 1$ and $m_{w'}(v) \geq m_{w'}(w) - 1$ we obtain that in both cases w' is the leader of v, contrary to our assumption. □

5.24 Lemma [EN17, Claim 2.3]. With probability at least $1 - \delta$, it holds that $r_v < h$ for all $v \in V$. ■

Proof. For any $v \in R$, $\Pr[r_u \geq h] = e^{-\beta k} = \delta/n$. The claim follows by applying the union bound. □

Similarly as in [EN17], we define $M(v) = \{u \mid m_u(v) \geq \max_{w \in R}\{m_w(v) - 1\}\}$, for every $v \in R$. We will use an observation about the size of this set to argue that the number of cut-edges between remote clusters is small.

5.25 Lemma [EN17, Lemma 2.2]. For every $v \in R$, $\mathrm{E}[M(v)] \leq (n/\delta)^{1/h}$. ■

Proof. Let $v \in R$. Note that $|M(v)| = t$ happens if and only if there are t vertices u such that $m_u(v) \geq \max_{w \in R}\{m_w(v) - 1\}$. By Lemma 5.10,

$$\mathrm{E}[|M(v)|] = \sum_{t=1}^{n} t \cdot \Pr[|M(v)| = t] = \sum_{t=1}^{n} \Pr[|M(v)| \geq t] \leq \sum_{t=0}^{\infty} (1 - e^{-\lambda})^t = e^{-\lambda} = \left(\frac{n}{\delta}\right)^{1/h}.$$

□

K For $\delta = 1/n^{b-1}$ and $h = \ell$, we obtain that $\mathrm{E}[M(v)] \leq (1 + \epsilon_2)$. Define the *edge-cut of R* to be $K = \{\{u, v\} \in E \mid v \in R \wedge u \in R \wedge (L(u) \neq L(v))\}$.

5.26 Lemma. With probability at least $99/100$, $|K| \leq 100\epsilon_2 d|R| \leq 100\epsilon_2 dn$. ■

Proof. Define $B = \{v \in R \mid M(v) > 1\}$. Observe that $e \cap B \neq \emptyset$ for any edge $e \in K$. To see this, consider an edge $\{u, v\} \in K$. Let u' and v' denote the leaders of u and v, respectively, and assume w.l.o.g. that u' has higher rank than v'. By the triangle inequality $\mathrm{dt}_{v'}(u, \leq) \, \mathrm{dt}(v', v) + 1$, thus we have that $m_{v'}(u) \geq m_{v'}(v) - 1$. Since u' has higher rank than v' it holds that $m_{u'}(u) \leq m_{v'}(v)$, and so it follows that $m_{v'}(u) \geq m_{u'}(u) - 1$. Thus $\{v', u'\} \subseteq M(u)$, which implies that $u \in B$, as required.

Hence $|K| \leq d|B|$. By Lemma 5.25 and the linearity of expectation, we obtain that with probability at least $99/100$, $|B| \leq 100\epsilon_2|R|$ and so we get the desired bound on $|K|$. □

It remains to bound the number of edges between remote clusters and core clusters.

5.27 Lemma [LL18, Lemma 6]. $\mathrm{E}[|E(R, \bar{R})|] \leq \epsilon_2 n$. ■

Proof. Observe that for every edge $\{u, v\} \in E$, there is at most one integer r such that $\Gamma_r(u) \cap V_C = \emptyset$ and $\Gamma_r(v) \cap V_C \neq \emptyset$. If no such r exists or $\ell \neq r$, then $\{u, v\}$ is not in $E(R, \bar{R})$. Over the random choice of ℓ, the probability that $\{u, v\}$ is a cut edge of R and $\bar{R}$ is at most $\Pr[\ell = r] \leq \epsilon/d$. The claim follows by the linearity of expectation. □

5.5. The Algorithm

We prove the following result. Theorem 5.1 follows by plugging in Lemmas 5.14 to 5.16.

f, g **5.28 Theorem.** Let $\mathcal{F}$ be a finite family of graphs such that there exists $H \in \mathcal{F}$ and $f = f(n, \mathcal{F}, d)$, $g = g(n, \mathcal{F}, d)$, $g \geq \ell$ (where ℓ is defined in Section 5.4) such that every n-vertex graph that is H-minor free is (f, g)-separable. For every $\epsilon > 0$, there is a one-sided ϵ-tester that given a query access to an n-vertex graph, G tests whether G is $\mathcal{F}$-minor free (i.e., G is R-minor free, for every $R \in \mathcal{F}$). The query complexity of the tester is $\tilde{\mathcal{O}}(n^{2/3} f^3/\epsilon^5)$. If $\mathcal{F}$ includes a planar graph, then the running time is $\tilde{\mathcal{O}}(n^{2/3} f^3/\epsilon^5)$ as well. ■

We first analyze a global version of the tester (see Algorithm 5.1) and show how to turn it into a local algorithm in Section 5.5.1. Our tester draws $\Theta(f/\epsilon)$ edges at random from the input graph G. It follows that at least one of these edges is part of a forbidden minor with

constant probability if G is ϵ-far from being $\mathcal{F}$-minor free. For the sake of this exposition, think of the graph being partitioned into core clusters and remote clusters. To reveal a forbidden minor from $\mathcal{F}$, the algorithm employs this partitioning.

We conduct the following case analysis. Either, an edge that is one out of many that connect a cluster A and an adjacent (disjoint) super cluster B will be sampled. Since non-remote clusters have diameter at most $\ell \leq g$, by the (f, g)-separability of $\mathcal{F}$-minor free graphs, a large cut between A and B implies the existence of H as a minor (see steps 2c and 2d). Otherwise, one can show that the total number of edges between clusters is at most $\epsilon dn/2$. This implies that the edges between clusters can be removed such that the graph is still $\epsilon/2$-far from being $\mathcal{F}$-minor free. Then, it suffices to look for a minor-instance of a graph from $\mathcal{F}$ inside the clusters of each edge (see step 2b).

Therefore, it suffices to bound the number of edges between clusters under the promise that the edge-cut between every cluster and an adjacent (disjoint) super cluster is small. Since a naive bound over all pairs of clusters is quite costly, we classify edges and analyze each class independently. First, we observe that the total number of edges between remote clusters and clusters is $\mathcal{O}(\epsilon dn)$ with constant probability. It remains to bound the number of edges between core clusters. To this end, we analyze the total number of edges between core clusters within the same Voronoi cell, the total number of edges between two unmarked core clusters and the total number of edges between core clusters and super clusters separately. This covers all relevant edges between clusters at least once and gives an upper bound on their total number.

Theorem 5.28 follows from the the efficient implementation (Section 5.5.1) and the correctness and the soundness of the tester (Section 5.5.2).

Note on the Running Time. The running time of our $\mathcal{F}$-minor testing algorithm is $\tilde{\mathcal{O}}(n^{2/3}f^3/\epsilon^5)$ if $\mathcal{F}$ contains a planar graph H'. In steps 2c, 2(d)i and 2(d)ii, Algorithm 5.1 rejects based on the number of edges crossing a cut. The time complexity for these steps is the same as the query complexity to compute the cut size. In step 2b, Algorithm 5.1 tests if any graph in $\mathcal{F}$ exists as a minor in the cluster C of size at most $t = \tilde{\mathcal{O}}(n^{1/3}f/\epsilon^2)$. To check this we proceed as follows. Let $m = |2V(H')| + 4|E(H')|$. Now check whether $G[C]$ has treewidth at most 20^{2m^5}. This can be done in time $\mathcal{O}(t)$ (see [Bod06]). By a theorem of Robertson, Seymour, and Thomas [RST94], if the treewidth is more than 20^{2m^5}, then $G[C]$ contains H' as a minor and the algorithm can reject right away. Otherwise, we know that $G[C]$ has treewidth at most 20^{2m^5}. Now, there is an $\mathcal{O}(t)$-time algorithm by Courcelle to test if $G[C]$ contains H as a minor [Cou90].

5.5.1. Efficient Implementation

In this subsection we describe how Algorithm 5.1 can be implemented in query and time complexity $\tilde{\mathcal{O}}(n^{2/3}f^3/\epsilon^5) \cdot \mathrm{poly}(d)$. For a vertex $v \in V$, define $i_v := \arg\min_i\{\Gamma_i(v) \geq y\}$ where $y := 10n^{1/3}\log^2 n/\epsilon_1$. Let E denote the event that $\Gamma_{i_v}(v) \cap V_C \neq \emptyset$ for all $v \in V$. Since E occurs with probability $1 - n \cdot (1 - |V_C|/n)^y \geq 99/100$, we condition on this event henceforth. The following subroutines are sufficient in order to implement Algorithm 5.1:

1. Given a vertex $v \in \bar{R}$, finding $\mathrm{clus}(v)$. The query complexity of finding $\mathrm{cent}(v)$ is bounded by $y \cdot d$. That is, $\mathrm{cent}(v)$ is found after performing a BFS from v for at most i_v levels. Furthermore, the path connecting v and $\mathrm{cent}(v)$ in $T(\mathrm{cent}(v))$ is also found (this is the shortest path between v and $\mathrm{cent}(v)$ of smallest lexicographical order). Therefore it is

Algorithm 5.1 Test $\mathscr{F}$-minor freeness

1. Partition V according to the partition described in Section 5.4 with parameters $\epsilon_1 = \epsilon/(100f)$ and $\epsilon_2 = \epsilon/2000$.
2. Sample $1000f/\epsilon$ random edges from G. For each sampled edge $\{u, v\}$:
   a) Find the cluster for each endpoint, denoted by C_v and C_u, respectively.
   b) If both u and v belong to the same cluster C, check that $G[C]$ is $\mathscr{F}$-minor free, if it is not, then **reject**.
   c) If either for $w = u$ or for $w = v$ it holds that: $w \in \bar{R}$, clus(w) is not a singleton, and $|E(\text{Vor}(w) \setminus \text{clus}(w), \text{clus}(w))| > f$, then **reject**.
   d) If both u and v are in $\bar{R}$ and and both C_u and C_v are not singletons then:
      i. If $|E(C_v, C_u)| > f$, then **reject**.
      ii. If C_v (and symmetrically for C_u) joins a cluster $C \neq C_u$, then let $A := \bigcup_{v \in \partial C \setminus C_u} \text{Vor}(v)$. If $|E((A \cup C) \setminus C_u, C_u)| > f$, then **reject**.
3. **accept**

Reminder: Parameters

$$\ell \in \left[\frac{\log n}{\log(1+\epsilon_2)}, \frac{\log n}{\log(1+\epsilon_2)} + \frac{d}{\epsilon_2}\right], \quad t = \frac{n^{1/3} \ln n \cdot \ell(d+1)}{\epsilon_1}$$

possible to explore $T(\text{cent}(v))$ with query complexity yd per step. In order to determine clus(v), it is sufficient to explore $T(v)$ and $T(a)$ for any ancestor a of v up to t vertices. Therefore, the total query complexity is at most $\ell \cdot t \cdot yd = \tilde{\mathcal{O}}(n^{2/3} f^2 d^4/\epsilon^4)$ per iteration.

2. Given a subset of vertices A, finding clus(∂A) can be done in query complexity yd times the total number of edges which are incident to vertices in A. If A is a cluster then the latter is bounded by td. Therefore we obtain a bound of $\tilde{\mathcal{O}}(n^{2/3} f^2 d^4/\epsilon^3)$ queries per iteration of step 2c and step 2(d)ii.

3. Finally, instead of finding the remote-cluster of a vertex $v \in R$ it suffices to find a connected induced subgraph that contains the remote-cluster of v. This is achievable by first finding the corresponding leader and then exploring the ℓ-hop neighborhood of the leader. To find the leader, it is sufficient to explore the ℓ-hop neighborhood of v and then for every vertex in it, to determine whether it is in R or not (determining whether a vertex is in R requires yd queries). With this at hand, it is possible to simulate the result of the leader-decomposition algorithm for v. Observe that both v and the leader of v are in R, therefore (under the assumption that E occurred) it is possible to explore their ℓ-hop neighborhood in yd time. The query complexity for each iteration of this step is bounded by $\tilde{\mathcal{O}}(n^{2/3} f^2/\epsilon^2) \cdot \text{poly}(d)$.

5.5.2. Correctness

5.29 Lemma. Algorithm 5.1 accepts every graph G that is $\mathscr{F}$-minor free. ■

Proof. The completeness of the test is based on the separability of H-minor free graphs. We show that if the algorithm rejects, then G contains a graph from $\mathscr{F}$ as minor.

Step 2b rejects only if G contains a graph from $\mathscr{F}$ as a minor. To apply (f, g)-separability to step 2c, it suffices to note that for any $w \in \bar{R}$ such that $\mathrm{clus}(w)$ is not a singleton, $G[\mathrm{Vor}(w) \setminus \mathrm{clus}(w)]$ is connected by Lemma 5.19.

To apply the separability to step 2(d)i, it suffices to note that C_v and C_u are disjoint and that $G[C_v]$ and $G[C_u]$ are both connected by Lemma 5.18. To apply the separability to step 2(d)ii, we need to show that $G[(A \cup C) \setminus C_u]$ is connected, since it is clearly disjoint from C_u, the correctness then follows. There are two cases. If $(A \cup C) \cap C_u = \emptyset$, then $G[(A \cup C) \setminus C_u] = G[A \cup C]$ is connected. Otherwise, since C_u is not a singleton and since ∂C contains a vertex v which is in $\mathrm{Vor}(C_u) \setminus C_u$, the claim follows from Lemma 5.19. □

5.30 Lemma. Algorithm 5.1 rejects every graph G that is ϵ-far from being H-minor free with probability $2/3$. ■

Proof. Assume that G is ϵ-far from being H-minor free. Let $\mathscr{P}$ denote the partition obtained by the algorithm (namely, the partition of the entire graph as described in Section 5.4 with parameters $\epsilon_1 = \epsilon/(100f)$ and $\epsilon_2 = \epsilon/2000$). We say that an edge $e = \{u, v\}$ violates the separability property with respect to $\mathscr{P}$ if either:

1. There exist a core cluster $A \in \bar{R}$ and a cluster or a super cluster, $B \in \bar{R}$ such that $e \in E(A, B)$ and $|E(A, B)| > f$,
2. or, if either for $w = u$ or for $w = v$ it holds that: $w \in \bar{R}$, $\mathrm{clus}(w)$ is not a singleton, and $|E(\mathrm{Vor}(w) \setminus \mathrm{clus}(w), \mathrm{clus}(w))| > f$.

Let $\mathscr{E}$ denote the set of edges which violate the (f, g)-separability property with respect to $\mathscr{P}$. If $|\mathscr{E}| > \epsilon_1 nd$, then with probability at least $99/100$, the algorithm finds a violation in one of the steps: step 2c, step 2(d)i or step 2(d)ii. Note that we do not need to check any edges between a core cluster and a remote cluster nor any edges between two remote clusters. By Markov's inequality (Lemma 1.17) and Lemma 5.27, with probability at least $99/100$, $|E(R, \bar{R})| \leq 100\epsilon_2 n$. By Lemma 5.26, with probability at least $99/100$, $|K| \leq 100\epsilon_2 nd$. Thus, after removing these $|E(R, \bar{R}) \cup K| \leq \epsilon dn/3$ edges, the graph is still $\epsilon/3$-far from being $\mathscr{F}$-minor free.

Assume that $|\mathscr{E}| \leq \epsilon_1 nd$. We will show that with probability at least $96/100$, we can separate G into clusters by removing at most $\epsilon_1 nd \cdot 500f = \epsilon dn/2$ edges. Therefore, the resulting graph is $\epsilon/2$-far from being $\mathscr{F}$-minor free, and with probability at least $2/3$, the algorithm rejects in step 2b.

Separating G Into Clusters.

1. As argued above, $|E(R, \bar{R}) \cup K| \leq 200\epsilon_2 nd$ with probability at least $98/100$. Therefore, we can separate G into $G[\bar{R}]$ and $G[R_1], \ldots, G[R_j]$, where $R_1, \ldots, R_j$ is the partition of R into remote clusters.
2. Next, we remove all the edges in $\mathscr{E}$ (at most $\epsilon_1 nd$). In order to separate each Voronoi cell into its core clusters we simply remove all the edges between different clusters in the

same Voronoi cell. The number of edges which are incident to singleton clusters are at most $d\gamma$. Since we removed the edges in $\mathscr{E}$, for a cluster A which is not a singleton, we have that $E(A, \mathrm{Vor}(A) \setminus A) \leq f$. Therefore by removing at most $\gamma(d+f)$ edges we separate all the Voronoi cells into clusters.

3. By Lemma 5.22, with probability at least $1 - o(1)$, we can separate all the clusters, A, for which $\mathrm{clus}(\partial A)$ does not contain a marked center by removing at most $3\gamma f p^{-1} \ln n$ edges.

4. The expected number of marked clusters is γp, therefore with probability at least $99/100$ the number of marked clusters is at most $100\gamma p$. Thus, with probability at least $99/100$ the number of pairs $A, B \in \bar{R}$ such that A is a cluster and B is a super cluster is at most $\gamma \cdot 100\gamma p$. Since we removed all edges in $\mathscr{F}$, we have that $E(A,B) < f$ for each such pair. Therefore, the number of edges between clusters and super clusters is at most $100 f \gamma^2 p$.

Recalling from Lemma 5.17 that $s = \epsilon_1 n^{2/3} / \ln n$, we can choose $\epsilon_1 = \epsilon/(100f)$ and $\epsilon_2 = \epsilon/2000$ such that

$$\begin{aligned} & 200\epsilon_2 nd + \Big[\epsilon_1 nd + \gamma(d+f)\Big] + \frac{3\gamma \ln n}{p} + 100 f^2 \gamma p \\ &= \frac{\epsilon dn}{10} + \Big[\frac{\epsilon dn}{100 f} + \frac{\epsilon n^{2/3}}{100 f \ln n}(d+f)\Big] + \frac{\epsilon n}{100} + \frac{\epsilon^2 n}{100 f \ln^2 n} \\ &\leq \epsilon dn/2 \,. \end{aligned}$$

Hence, with probability at least $96/100$, we can separate G into clusters and remote clusters by removing at most $\epsilon dn/2$ edges. □

6. Distributed Testing of Conductance

The traditional setting of property testing is sequential: An algorithm run on a single processor may ask an oracle questions about the input and has to decide a problem. However, parallel and distributed computing has become highly relevant in practice within the last decades. In this chapter, we study property testing in a distributed setting, where the input also defines a distributed model of computation, namely, the CONGEST model. The results presented in this chapter are joint work with Yadu Vasudev [FV18].

6.1. Introduction

Sequential computation is the oldest and still a very dominant model in computer science. Although parallel computing has its share in theoretical computer science and even more so in practice, one reason for the popularity of sequential computation models is that it is often much simpler to analyze one processor instead of multiple processors *and* their interactions. However, in light of large data, a single point of computation is often undesirable or even impractical. In this chapter, we consider property testing in the distributed LOCAL and CONGEST models, where processors and communication are tightly related to the input data. In the LOCAL model, the input is a graph $G = (V, E)$, and each vertex is a processor that knows only its vertex id (usually a random number from $\text{poly}(n)$). Computation takes place in rounds, and in each round, a processor may send a message to each of its neighbors in the graph. In the CONGEST model, the message length is bounded by $\mathcal{O}(\log n)$ bits, which essentially allows to send a constant number of vertex ids. After the last round, the output of the algorithm is given by function of the output of all vertices.

This model might seem a bit odd at first: The number of processors scales linear with the input size, but every processor has very limited capabilities due to its restricted input and possibly small number of neighbors. In particular, the whole power of the model *arises* from communication between processors, as compared to parallel computation models, where one often aims to avoid too much communication between processors. On top of that, even the topology of the communication network depends heavily on the input and is basically unknown to the processors. Why should we consider such a model?

One reason is that it models locality in a very strong sense. In the LOCAL model, every local view is part of the computation simultaneously. More precisely, it is possible to let every vertex v know its r-disk in round r (in the CONGEST model, this is a bit more restricted). However, there is no way of merging these views to obtain the global structure of the graph efficiently. This makes the LOCAL and the CONGEST models good candidates to study the power of locality. A more practical scenario is that the input graph is actually a network, and every vertex corresponds to an individual that is interested in solving some problem on this network or a superior authority that is interested in solving a problem with the help of the individuals. In this chapter, we consider testing conductance in the CONGEST model, which, as a property, implies fast rumor spreading in the graph. If the input graph is, e. g., a mobile network or a

peer-to-peer network, conductance is an interesting property for the carrier or maintainer of the network to measure the efficiency of message passing.

6.1.1. Results in This Chapter

The main result in this chapter is a distributed property tester in the CONGEST model for conductance (of general graphs). We note that the definition of testing is slightly weakened as there is a gap between the conductance bounds that hold in the far-case.

6.1 Theorem. Testing whether a graph $G = (V, E)$ has conductance at least Φ or is ϵ-far from having conductance at least $\Phi^2/1000$ with two-sided error has complexity $\mathcal{O}(\log(n+m)/(\epsilon\Phi^2))$ in the CONGEST model. ■

This tester can be implemented such that *all vertices accept or all vertices reject*, which is a stronger decision rule than *all vertices accept or at least one vertex rejects* considered in previous work. If the input graph is connected, it is not necessary to pass the size of the graph as input to the vertices. On the other hand, because communication between vertices of different connected components is not possible, an algorithm in the LOCAL model cannot distinguish a graph G with high conductance from a graph G' that is composed of two isolated copies of G, which implies conductance 0, without knowledge of n. We show that our main result is tight up to the dependence on ϵ and Φ.

6.2 Theorem. Let $\epsilon, d > 0$ be constants, and let $\Phi, c > 0$ be constants that depend on d. Testing whether a d-regular graph $G = (V, E)$ has conductance at least Φ or is ϵ-far from having conductance at least $c\Phi^2$ requires $\Omega_{\Phi,\epsilon}(\log(n+m))$ rounds of communication in the LOCAL and CONGEST models. ■

6.1.2. Expansion and Conductance in Sequential Property Testing

In the classic, sequential setting of property testing, the problem of testing conductance in bounded-degree graphs, or expansion respectively, was first studied by Goldreich and Ron [GR00] and Czumaj and Sohler [CS07]. Goldreich and Ron [GR00] proposed a random-walk based tester with two-sided error and query complexity roughly $\mathcal{O}(\sqrt{n} \cdot \text{poly}(\epsilon))$, which they could not fully analyze for graphs that are ϵ-far from the property. Czumaj and Sohler [CS07] were the first to close this gap by a combinatorial argument in their analysis. Their property tester accepts graphs with expansion Φ and it rejects graphs that are ϵ-far from graphs with expansion $\Theta(\Phi^2/(d^2 \log(n/\epsilon)))$. It has query complexity $\tilde{\mathcal{O}}(d^2\sqrt{n}/(\Phi^2\epsilon^3))$. Independently of each other, Kale and Seshadhri [KS08] and Nachmias and Shapira [NS10] removed the logarithmic dependence of the far-case on n. Their testers accept graphs that have expansion at least Φ and reject graphs that are ϵ-far from having expansion at least $\Omega(\Phi^2)$. The testers have comparable running time; the algorithm given in Kale and Seshadhri [KS08] has running time $\tilde{\mathcal{O}}(n^{1/2+\delta}/(\epsilon\Phi^2))$ for any $\delta > 0$.

An algorithm for testing the cluster structure of graphs was developed by Czumaj, Peng, and Sohler [CPS15]. Testing conductance in general graphs in the stronger rotation map model with query complexity roughly $\tilde{\mathcal{O}}(\sqrt{m}/\Phi^2)$ was studied by Li, Pan, and Peng [LPP11]. They developed a reduction to the bounded-degree case. Testing conductance properties restricted to small sets was studied by Li and Peng [LP15]. However, the optimal query complexity for testing conductance in general graphs of unbounded degree is still open.

6.1.3. The Field of Distributed Property Testing

One of the first adaptions of property testing to distributed computing was performed by Brakerski and Patt-Shamir [BP09]. Their algorithm finds *near-cliques* of linear size, i. e., subgraphs that are ϵ-close to being a clique of linear size. The round complexity is $2^{\text{poly}(\epsilon)}$ in the CONGEST model. Censor-Hillel et al. [Cen+16] initiated a rigorous study of one-sided error property testing in the LOCAL and in the CONGEST models. They show that every constant-query testable property in the dense graph model such that the existence of a connected witness is guaranteed for graphs that are not in the property, can be tested with $\mathcal{O}(q^2)$ queries, where q is the query complexity of the property tester in the dense graph model. This result is obtained by using the fact that every constant-query tester in the dense graph model can be made non-adaptive (see Section 3.1.2). They also prove that triangle-freeness has round complexity $\mathcal{O}(1/\epsilon^2)$ and that bipartiteness and cycle-freeness have round complexity $\text{poly}(1/\epsilon \log(n/\epsilon))$ and $\mathcal{O}(\log n/\epsilon)$, respectively; they show lower bounds of $\Omega(\log n)$ for the latter two. Levi, Medina, and Ron [LMR18] propose a one-sided error tester for planarity in the CONGEST model that has query complexity $\mathcal{O}(\log(n)/\text{poly}(\epsilon))$, which is shown to be tight up to the dependence on ϵ. Their algorithm is based on a partition with parts of diameter $\text{poly}(1/\epsilon)$ and a sparse cut. Then, the algorithms proceeds by constructing an underlying spanning tree and a planar embedding; if the graph is ϵ-far from being planar, it either fails to construct the embedding or many of the edges that do not belong to the spanning tree satisfy some hierarchical condition. The approach of constructing an underlying structure is slightly related to work discussed and presented in Chapter 5.

Most attention in distributed property testing has been focused on subgraph-freeness testing. All of the following results refer to one-sided error testers. Fraigniaud et al. [Fra+16] studied these problems for subgraphs of size 4 and showed that it has constant round complexity in the CONGEST model. Fraigniaud and Olivetti [FO17] proved that this upper bound on the complexity also holds for cycles of any length k, and Even et al. [Eve+17] extended this class of subgraphs to graphs that contain a *special* set of cycles. In [Eve+17], it is also shown that one can detect a fixed tree deterministically in constant time, and that K_k-freeness can be tested in $\mathcal{O}(m^{1/2-1/(s-2)\cdot\text{poly}(\epsilon)})$ rounds. Furthermore, they describe how to transform a distributed property tester whose round complexity is bounded by a function $f(\ell)$ of the diameter ℓ of the input graph into a distributed property tester with round complexity $\mathcal{O}(\log(n)/\epsilon + f(\log(n)/\epsilon))$.

Distributed random walks have been studied by Das Sarma et al. [Das+13] and Molla and Pandurangan [MP17]. In particular, Molla and Pandurangan [MP17] show that one can approximate the mixing time τ_v of a vertex v in $\mathcal{O}(\tau_v \log n)$ rounds by running $\text{poly}(n)$ random walks v and comparing their endpoint distribution to the stationary distribution. The graph's mixing time $\tau = \max_v \tau_v$ relates to the conductance by $c_1\Phi^2/\log n \leq 1/\tau \leq c_2\Phi$, where c_1, c_2 are some universal constants. A straightforward approach based on [MP17] leads to an $\mathcal{O}(n \log^2(n)/\Phi^2)$ round algorithm for *approximating* Φ with a multiplicative gap of $\Theta(\Phi/\log n)$. In comparison, our *tester's* gap does not depend on n and its complexity is only logarithmic in n. One reason is that if the graph is *far from* having conductance $\Omega(\Phi^2)$, there exist many vertices with large mixing times compared to the case that the graph has conductance Φ (see the proof of Theorem 6.1 for details). This is not necessarily the case if the graph is not ϵ-far from having conductance $\Omega(\Phi^2)$.

6.1.4. Open Problems and Future Work

From the perspective of property testing in query models, there is a substantial body of work that ports the two main exploration techniques – BFS and random walks – to the CONGEST model. The transformation by Even et al. [Eve+17] provides the first more general result. Similar to the query model (see Section 4.1.3), it is desirable to better understand which properties can be tested with constant (or logarithmic) complexity in the LOCAL and CONGEST models. Comparing the two regimes, it would also be interesting to understand the different powers and relate them to classes of properties that have different complexities in the LOCAL or CONGEST model and the corresponding query models. One might even think about parameterizing the complexity by the number of processors. For example, this might give results for the CONGEST model, the MPC model and a (centralized) query model at once and provide useful insights into the role of distributed computing and parallel processing in graph property testing.

6.3 Problem. Obtain more general results for property testing in the LOCAL or CONGEST models; obtain results that cover multiple models with varying level of distribution / parallelism. ■

Structural results in distributed property testing of *bounded-degree* graphs might also lead to new insights regarding constant-query testable properties in the bounded-degree query model. For example, a LOCAL tester for bounded-degree graphs has access to every k-disk, but it cannot assemble even an approximation of the whole k-disk frequency vector. This might lead to results that explain some of the structure of constant-query testable properties, similar to the results presented in Chapter 3. Maybe one can also relate the problem of learning the 2-disk of high-degree nodes in the CONGEST model (without bound on the degrees), i. e., the problem of congestion, with the problem of exploring all neighbors of a high-degree node in the general model.

6.4 Problem. Study distributed property testing in bounded-degree graphs with respect to structural insights and the relation to constant-query testable properties; extend these studies to the CONGEST model with unbounded degrees and the general model. ■

6.1.5. Overview of the Analysis

The proof of the tester is sketched along the lines of the algorithm in Section 6.3.1. For the lower bound, we construct two graphs and prove that they cannot be distinguished by a tester with round complexity $o_{\Phi,\epsilon}(\log(n+m))$. The first graph is an expander graph with girth $\Theta(\log n)$, and the second graph is a graph that results from rewiring this expander so that there exist two disjoint connected components (see Proposition 2.11). The latter graph is a bad expander; however, its $\Theta(\log n)$-disk distribution is not changed due to rewiring. Therefore, any tester needs to query beyond the boundary of these disks to distinguish the two graphs. Intuitively, this is sufficient. However, it remains to prove that a tester cannot use the vertex labels of the graphs to distinguish between them. We do this by defining an auxiliary model of distributed computing on unlabeled graphs, and prove that this model is as powerful as the CONGEST model. In particular, we show that vertices in the auxiliary model can assign themselves labels so that no two vertices have the same label with high probability.

6.2. Preliminaries

Distributed Computing. In the distributed computational model, a computation network $G = (V, E)$ with a processor associated to each vertex $v \in V$ is given. Each processor v has access to numbered communication channels to its neighbors in G. Additionally, it may have some specific input $I(v)$. The computation operates in synchronized rounds that are divided into three phases. In each round, each processor may do some local computation first, then it may send a message to each of its neighbors, and finally it receives the messages sent from its neighbors.

DCM(G, p_G, I)

6.5 Definition (distributed computational model, DCM). Let $G = (V, E)$ be a graph and $p_G = (p_v)_{v \in V}$ with $p_v : [d(v)] \to \Gamma(v)$ be a bijective function, i. e., an adjacency list representation of G. Let $I : V \to \{0,1\}^*$ be a mapping from the set of vertices to bit strings. An instance of the distributed computational model on G, p_G and I, called DCM(G, p_G, I), is defined as follows. Each vertex $v \in V$ is a processor that has communication access to its neighbors $p_v(1), \dots, p_v(d(v))$ by ports numbered $1, \dots, d(v)$. The model operates in synchronized rounds, where each round r consists of three phases: (i) Each vertex performs local computation, (ii) each vertex v sends a message to its neighbor $p_v(i)$, denoted $s_r(v, i)$, for all $i \in d(v)$, (iii) each vertex u receives a message from its neighbor $p_u(j)$, for all $j \in d(u)$. The distributed computational model DCM is the set of all instances DCM(G, p_G, I). ■

The LOCAL model is the subset of the DCM such that for each vertex $v \in V$, the input $I(v)$ is only n and a numerical vertex identifier from $[n^c]$ for some universal constant c. The CONGEST model is the subset of the LOCAL model such that the size of each message $s_r(v, i)$ is restricted to $c \log n$ bits.

A distributed network decision algorithm DNDA($\mathscr{A}, O$) is an algorithm $\mathscr{A}$ that is deployed to the vertices of a DCM to decide a property of an instance of the model. In particular, the output of $\mathscr{A}$ is a single bit, and the final decision is obtained by applying a function $O(\cdot)$ to (the sequence of) all vertices' answers.

DNDA($\mathscr{A}, O$)

6.6 Definition (distributed network decision algorithm). Let $\mathscr{A}$ be an algorithm that takes a bit string as input and outputs a single bit, and let $O : \{0,1\}^* \to \{0,1\}$ be a function. When the distributed network decision algorithm DNDA($\mathscr{A}, O$) is run on an instance DCM(G, p_G, I), a copy of $\mathscr{A}$ is deployed to every vertex v with input $I(v)$ and run in parallel as described in Definition 6.5. We refer to the copy of $\mathscr{A}$ deployed to v by $\mathscr{A}_v$. When every vertex v_i has terminated its computation with output bit b_{v_i}, the decision of DNDA($\mathscr{A}, O$) is $O(b_{v_1} b_{v_2} \cdots b_{v_n})$. ■

Distributed Property Testing. All graphs in this chapter have unbounded degree if not stated otherwise. A distributed property testing algorithm is a distributed algorithm as defined in Definition 6.6 that accepts graphs that have a property, and rejects graphs that are ϵ-far from the property. We adapt the definitions for the query model as described in Section 1.3 accordingly.

6.7 Definition (distributed tester). A two-sided error distributed ϵ-tester for a property Π is a DNDA($\mathscr{A}, O$), where $O(b_{v_1} b_{v_2} \cdots b_{v_n}) = 1$ if and only if $b_{v_i} = 1$ for all $v_i \in V$ such that the following conditions hold: (i) if G has the property Π, then, with probability at least $2/3$, $b_{v_i} = 1$ for all $v_i \in V$, (ii) if G is ϵ-far from Π, then, with probability at least $2/3$, there exists a $v_i \in V$ such that $b_{v_i} = 0$. ■

The guarantees given by our tester are actually a bit stronger in the sense that the tester can be modified so that either $b_v = 0$ or $b_v = 1$ for all $v \in V$ simultaneously.

W, N, D, L

Before we describe the algorithm, we give a few useful definitions and lemmas. A *lazy random walk* on a graph $G = (V, E)$ on n vertices is a random walk on the graph where at each vertex v the walk chooses to stay at v with probability $1/2$ and chooses a neighbor u with probability $1/(2d(v))$. The walk matrix $W = [w_{uv}]_{u,v\in[n]}$ is defined by $w_{uv} := 1/2$ if $u = v$, $w_{uv} := 1/(2d(v))$ if $u \neq v, \{u, v\} \in E$ and $w_{uv} := 0$ otherwise. Notice that for irregular graphs, W is not symmetric. To analyze these random walks, one can draw on the *normalized walk matrix*, which is a symmetric matrix similar to W. The normalized walk matrix N of G is $D^{-1/2}WD^{1/2}$, where D is the diagonal matrix with $D(u, u) := d(u)$. The *(normalized) Laplacian* becomes useful when studying cuts in a graph. It is defined by $L := I - D^{-1/2}AD^{-1/2}$.

$\mu_\cdot, f_\cdot$

Since N is a real symmetric matrix, it has real eigenvalues. Let $1 = \mu_1, \ldots, \mu_n \geq 0$ be its eigenvalues, and let $\{f_i\}_{i\in[n]}$ be its orthonormal eigenbasis. We have $f_1 = \sqrt{\pi}$, where π is the random walk's *stationary distribution*. In particular, it is well known that $\pi_v = d(v)/(2m)$. For more details on spectral graph theory, refer to the book by Chung [Chu97].

It is well known that graphs with high conductance have small diameter.

6.8 Lemma [Chu89, cf. Theorem 2]. Let $G = (V, E)$ be a graph with conductance Φ. The diameter of G is at most $(3/\Phi)\ln(m)$. ■

Sinclair [Sin93] proved that there is a tight connection between the conductance and the mixing time of random walks. In particular, the L_2 distance of any starting distribution π' to π after $\Phi^{-2}\log n$ steps is $\mathcal{O}(()1/n)$.

6.9 Lemma [Sin93, cf. Theorem 2.5]. Let $G = (V, E)$ be a graph with conductance Φ. For any starting distribution π', it holds that $\|W^\ell\pi' - \pi\|_2 \leq \left(1 - \Phi^2/2\right)^\ell$. ■

The following decomposition of the walk matrix turns out to be useful in the context of graph cuts.

6.10 Lemma [LPW09, Lemma 12.2]. [1] Let G be a graph. The walk matrix W can be decomposed as

$$\frac{W^\ell(u, v)}{\pi(v)} = 1 + \sum_{i=2}^{n} \mu_i^\ell \frac{f_i(u)f_i(v)}{\sqrt{d(u)d(v)}}.$$

■

6.3. Testing Using Random Walks

In this section will present the distributed algorithm for testing whether a graph has conductance at least Φ or is ϵ-far from having conductance at least $\Phi^2/1000$. The core idea of the algorithm is to perform random walks from a small set of vertices and test whether these walks converge to the stationary distribution rapidly, which is the case for graphs with high conductance. It is based on the ideas of Kale and Seshadhri [KS08] and Goldreich and Ron [GR00].

[1] In [LPW09], the result is stated for a right stochastic walk matrix and its *right* eigenbasis that is orthonormal with respect to the non-standard inner product $\langle f, g\rangle_\pi = \sum f(x)g(x)\pi(x)$. The statement here is adapted to our notation.

6.3.1. Testing Algorithm

We discuss the algorithm from a global point of view instead of describing an algorithm $\mathscr{A}$ that is executed by every vertex to provide a better explanation of the interactions between vertices.

Lemma 6.8 implies that if the graph has high conductance, then it has diameter $\mathcal{O}(\log(n)/\Phi)$, which we want to use as an assumption in the algorithm later. To test the diameter, we perform a BFS of depth $\mathcal{O}(\log(n)/\Phi)$ of the graph starting from an arbitrary vertex. Initially, every vertex chooses itself as root of the BFS and announces itself as root to all its neighbors. To break the symmetry between the vertices, a vertex accepts every vertex with a lower identifier than its current root as new root and forwards its messages. If the diameter is $\mathcal{O}(\log(n)/\Phi)$, a unique root has been chosen after $\mathcal{O}(\log(n)/\Phi)$ rounds and every vertex knows its parent and its children in the BFS tree. Otherwise, at least one of the remaining candidates will reject. Algorithm 6.3 gives a formal description of the BFS.

From now on, assume that the diameter is $\mathcal{O}(\log(n)/\Phi)$. Using the previously computed BFS tree, we can compute the number of edges in the graph by summing up vertex degrees from the leaves to the root and transmitting this number to all vertices afterwards. Algorithm 6.4 describes the procedure in detail.

The key technical lemma from [KS08] for bounded-degree graphs states that if a graph is ϵ-far from having conductance $\Omega(\Phi^2)$, then there exists an $\Omega(\epsilon)$-fraction of *weak* vertices such that random walks starting from these vertices converge only slowly to the stationary distribution. Therefore, a sample $S \subset V$ of size $\mathcal{O}(1)$ will likely contain a weak vertex (technically, we sample each vertex v independently into S with probability $\Theta(d(v)/\epsilon m)$. By Markov's inequality, we may reject if S is much larger than its expected size.). We extend this lemma to unbounded degree graphs. Then, we perform $N = n^{100}$ random walks of length $\ell = 40/\Phi^2 \cdot \log n = \mathcal{O}(\log(n)/\Phi^2)$ starting from each of the vertices in S to approximate the rate of convergence.

The crucial point here is that in each round of the algorithm, we do not send the full trace of every random walk. Instead, for every origin $v \in S$, every vertex $u \in V$ only transmits the total number of random walks that are leaving it through an edge $\{u, w\}$ to its neighbor $w \in \Gamma(u)$. Since the size of S is constant, we require $\mathcal{O}(\log n)$ bits per edge to communicate this. On the other hand, this information is sufficient because we are only interested in the distribution of endpoints of the lazy random walks for every $v \in S$. Algorithm 6.2 gives a formal description of this procedure. Finally, the estimated distribution of endpoints is used to approximate the distance to the stationary distribution for each $v \in S$. The whole algorithm is summarized in Algorithm 6.1.

6.3.2. Spectral Graph Theory Results

First, we show that either the estimates $\widehat{W}^{\ell}_{v,u}$ of Algorithm 6.2 are good or the algorithm rejects in line 19 because G has low conductance.

6.11 Lemma. Consider Algorithm 6.2. For every $v, u \in V$, it holds with probability at least $1 - m^{-10}$ that (i) $|\widehat{W}^{\ell}_{v,u} - W^{\ell}(v,u)| \leq m^{-20}$ and, conditioned on the previous, (ii) if $\widehat{W}^{\ell}_{v,u} < m^{-2}$ then G has conductance less than Φ. ∎

Algorithm 6.1 Conductance tester

```
1:  function TestConductance(G = (V, E), n, Φ)
2:      BFS(G, 6/Φ ln n)                          ▷ construct BFS of depth 6/Φ log n, Algorithm 6.3
3:      if BFS visited less than n vertices then reject
4:      end if
5:      m ← AggegrateSum(G, 12/Φ ln n, f)          ▷ f(v) := d(v)/2, Algorithm 6.4
6:      let every vertex v ∈ V do
7:          with probability min{1, 10^4 d(v)/2εm}, mark v
8:      end let
9:      S ← marked v, r ← root of BFS tree
10:     if |S| > 10^5/ε then reject
11:     end if
12:     RandomWalk(G, S, 40/Φ^2 · log n, n^100)    ▷ compute local s_{v,u}, Algorithm 6.2
13:     for all v ∈ S do s_v ← AggegrateSum(G, 12/Φ ln n, f)   ▷ f(u) := s_{v,u}, Alg. 6.4
14:     end for
15:     let every vertex v ∈ V do
16:         if s_v ≤ m^{-15} for all v ∈ S then accept
17:         else reject
18:         end if
19:     end let
20: end function
```

Proof. We have $E[\widehat{W}^\ell_{v,u}] = W^\ell(v,u)$. By Hoeffding's inequality, it holds that

$$\Pr[|\widehat{W}^\ell_{v,u} - E[\widehat{W}^\ell_{v,u}]| \geq m^{-10}] \leq 2\exp\left(-\frac{N}{3m^{40}}\right) \leq m^{-10}\,. \tag{6.1}$$

Condition on $|\widehat{W}^\ell_{v,u} - E[\widehat{W}^\ell_{v,u}]| < m^{-20}$, which happens with probability at least $1 - 1/m^{10}$. If $\widehat{W}^\ell_{v,u} < m^{-2}$, then

$$W^\ell(v,u) = E[\widehat{W}^\ell_{v,u}] < \widehat{W}^\ell_{v,u} + m^{-20} = m^{-2} + m^{-10} < 2m^{-2}\,.$$

Let $\pi' = \mathbb{1}_v$. We bound $\|W^\ell\pi' - \pi\|_2$ from below.

$$\|W^\ell\pi' - \pi\|_2 \geq |W^\ell(v,u) - d(u)/2m| \geq -(2m^{-2} - 1/(2m)) \geq 1/(4m)\,.$$

By the contrapositive of Lemma 6.9, G has conductance less than Φ. □

Furthermore, Lemma 6.11 implies that the estimates s_v in Algorithm 6.1 (see line 13) are also good if Algorithm 6.2 has not rejected before.

6.12 Lemma. Consider Algorithm 6.1. With probability at least $1 - m^{-8}$ it holds for every $v \in S$ in line 13 that

$$\left|\|W^\ell(v,\cdot) - \pi\|_2^2 - s_v\right| \leq 3m^{-19}$$ ■

Proof. Let $v \in S$. We have the following equality for the discrepancy of the distribution of the

Algorithm 6.2 Perform random walks

function RandomWalk(G, S, ℓ, N)
  let *every vertex* $v \in S$ **do**
    sample $u_1, \dots, u_N$ independently according to We_v
    for all $w \in \Gamma(v)$ **do** send $(v, v, |\{i \mid w = u_i\}|)$ to w
    end for
  end let
  for ℓ rounds, **let** *every vertex* x **do**
    receive $(u_1, x'_1, k_1), (u_2, x'_2, k_2), \dots$
    for all $v \in S$ **do**
      sample $u_1, \dots, u_{n_v}$ independently according to We_x, where $n_v = \sum_{u_i = v} k_i$
      for all $w \in \Gamma(v)$ **do** send $(v, x, |\{i \mid w = u_i\}|)$ to w
      end for
    end for
  end for
  let *every vertex* $u \in V$ **do**
    receive $(u_1, x'_1, k_1), (u_2, x'_2, k_2), \dots$
    for all $v \in S$ **do**
      $\widehat{W}^{\ell}_{v,u} \leftarrow \sum_{v_i = v} n_i / N$
      if $\widehat{W}^{\ell}_{v,u} \le 2m^{-2}$ **then reject**
      end if
      $s_{v,u} \leftarrow (\widehat{W}^{\ell}_{v,u} - \frac{d(v)}{2m})^2$
    end for
  end let
end function

Algorithm 6.3 Construct BFS tree

function BFS(G, D)
  let *every vertex* v **do**
    $T_v \leftarrow (v, \cdot)$ ▷ set root to itself, parent to empty
    $minid \leftarrow v$
    send $\{v, v\}$ to every neighbor $u \in \Gamma(v)$
  end let
  for D rounds, **let** *every vertex* w **do**
    $R_v \leftarrow \{\{v', u'\}$ received $\mid u' \in \Gamma(w)\}$
    $\{v, u\} \leftarrow \arg\min_{\{v', u'\} \in R_v} v'$
    if $T_w = (\cdot)$ or $v' < minid$ **then**
      $T_w \leftarrow \{v, u\}$ ▷ set root to v, parent to u
      send $\{v, w\}$ to all neighbors $\neq u$
    end if
  end for
end function

Algorithm 6.4 Aggregate sum of vertex values and propagate it to all vertices

Require: $\forall v : v$ has local information $f(v)$
Ensure: $\forall v : v$ has information $\sum_{u \in V} f(u)$
function AggregateSum(G, D, $f : V \to \mathbb{R}$)
  for D rounds, **let** *every vertex* v **do**
    if v received partial sums s_u from all its children u in BFS tree **then**
      $s_v \leftarrow f(v) + \sum_u s_u$
      send s_v to parent in BFS tree
    end if
  end for
  let *vertex root r of BFS tree* **do**
    send total sum $s = \sum_v f(v)$ to all children
  end let
  for D rounds, **let** *every vertex* v **do**
    if v received total sum s from its parent **then**
      send s_v to all children in BFS tree
    end if
  end for
  return s_v ▷ consider s_v to be the output of the algorithm
end function

endpoints of random walks that start at v and the stationary distribution:

$$\|W^\ell(v,\cdot) - \pi\|_2^2 = \sum_{u \in V} \left(W^\ell(v,u) - \frac{d(u)}{2m} \right)^2 . \tag{6.2}$$

By Lemma 6.11, we know that for every $u \in V$ we have $|\widehat{W}^\ell_{u,v} - W^\ell(v,u)| \leq m^{-20}$ with probability $1 - 1/m^9$. Then, by the triangle inequality,

$$\left| \left(W^\ell(v,u) - \frac{d(u)}{2m} \right)^2 - s_{v,u} \right| \leq 3m^{-20} .$$

Combining this with Eq. (6.2), a union bound over all $u \in V$ implies that with probability at least $1 - n \cdot m^{-10} \geq 1 - m^{-9}$, we have that

$$\left| \|W^\ell(v,\cdot) - \pi\|_2^2 - \sum_{u \in V} s_{v,u} \right| \leq 3m^{-19} .$$

A union bound over all $v \in S$ gives that with probability at least $1 - |S|/m^{-9} \geq 1 - m^{-8}$, for every $v \in S$,

$$\left| \|W^\ell(v,\cdot) - \pi\|_2^2 - \sum_{u \in V} s_{v,u} \right| \leq 3m^{-19} . \qquad \square$$

The proof of completeness is a straightforward application of the results from the previous section.

6.13 Lemma (Completeness). Let $G = (V, E)$ be a graph with conductance at least Φ. Then, with probability at least $2/3$, each vertex in G accepts when it runs Algorithm 6.1. ■

Proof. The probability that the algorithm rejects in Line 10 of Algorithm 6.1 is at most $1/10$, and we assume, for the remainder of the proof, that this event did not occur. If G has conductance at least Φ, then from Lemma 6.9 we know that for every vertex v

$$\|W^\ell(\cdot, v) - \pi\|_2^2 \leq \left(1 - \Phi^2/2\right)^{2\ell} \leq \exp(-\Phi^2\ell/2) \leq m^{-20}\,.$$

Lemma 6.12 implies that with probability at least $9/10$, it holds that $\left|\|W^\ell(\cdot, v) - \pi\|_2^2 - s_v\right| \leq 3m^{-19}$. Conditioning on this event, every vertex accepts in line 16 of Algorithm 6.1. □

To complete the analysis of the tester, we show that whenever the graph is ϵ-far from having conductance $\Phi^2/1000$, the tester rejects with probability at least $2/3$. To this end, we actually show that if the volume of weak vertices is small, then the graph can be converted to another graph G' by modifying at most ϵm edges such that the conductance is at least $\Phi^2/1000$. The idea of the analysis is due to Kale and Seshadhri [KS08], who analyzed a classic property tester for testing expansion in graphs with vertex degrees bounded by a constant. We deviate from their analysis where it becomes necessary to take care of arbitrary vertex degrees.

Let a vertex $v \in V$ be called *weak* if $\|W^\ell(v, \cdot) - \pi\|_2 > 6m^{-15}$. The following lemma states that if there exists a set of vertices S with small conductance, then there exists a set of weak vertices T whose volume is at least a constant fraction of the volume of S.

6.14 Lemma. Let $S \subset V$ be such that $\mathrm{vol}(S) \leq \mathrm{vol}(\bar{S})$ and $\mathrm{cond}(S) \leq \delta$. Then, for any $\ell \in \mathbb{N}$ and any $0 < \theta \leq 1/10$, there exists a set $T \subseteq S$ such that $\mathrm{vol}(T) \geq \theta\,\mathrm{vol}(S)$ and for every $v \in T$, it holds that

$$\|W^\ell(v, \cdot) - \pi\|_2^2 > \frac{1}{80m^7}(1 - 4\delta)^{2\ell}\,.$$

■

The proof can be found in Section 6.3.3. We can use Lemma 6.14 to separate weak vertices from the remaining graph.

6.15 Lemma. Let $G = (V, E)$ be a graph. If the volume of weak vertices in G is at most $(1/100)\epsilon m$, then there is a partition of V into $P \cup \bar{P}$ such that $\mathrm{vol}(P) \leq \epsilon m/10$ and $\Phi(G[\bar{P}]) \geq \Phi^2/256$. ■

Proof. We partition the graph recursively into two sets $(P, \bar{P})$. At the beginning, $P_0 = \emptyset$ and $\bar{P}_0 = V$. As long as there is a cut $(C_i, \bar{C}_i)$ in $\bar{P}_{i-1}$ in step i with $\mathrm{vol}(C_i) \leq \mathrm{vol}(\bar{C}_i)$ and $|E(C_i, \bar{C}_i)|/\mathrm{vol}(C_i) \leq \Phi^2/256$, we set $P_i = P_{i-1} \cup C_i$ and $\bar{P}_i = V - P_i$. We continue this until we don't find such a cut or the condition $\mathrm{vol}(P_{i+1}) \leq \mathrm{vol}(\bar{P}_{i+1})$ would be violated. The number of edges going across the cut $(P, \bar{P})$ is at most $\sum_i |E(C_i, \bar{C}_i)|$. Therefore, $|E(P, \bar{P})| \leq (\Phi^2/256)\sum_i \mathrm{vol}(C_i) \leq (\Phi^2/256)\,\mathrm{vol}(P)$.

Now, assume that $\mathrm{vol}(P) > \epsilon m/10$. Lemma 6.14 implies that there exists $P' \subseteq P$ such that $\mathrm{vol}(P') \geq \mathrm{vol}(P)/10 > \epsilon m/100$ (where $\theta = 1/10$) and for all $v \in P'$ we have

$$\|W^\ell(v, \cdot) - \pi\|_2 > \frac{1}{80m^7}(1 - 4\Phi^2/256)^{2\ell} > \frac{1}{80m^{10}}\,.$$

This means that P' contains only weak vertices and has volume at least $\epsilon m/100$, which contradicts our assumption that the volume of weak vertices in G is at most $\epsilon m/100$. Therefore, $\mathrm{vol}(P) \leq \epsilon m/10$ when the partitioning terminates. Hence $\Phi(G[\bar{P}]) \geq \Phi^2/256$. □

Finally, the following lemma states that few edge modifications in a graph with separated weak vertices are sufficient to make it a graph with high conductance.

6.16 Lemma [LP15, Lemma 9]. Let $G = (V, E)$ be a graph. If there exists a set $P \subseteq V$ such that $\mathrm{vol}(P) \leq \epsilon m/10$ and the subgraph $G[V - P]$ is a Φ'-expander, then there exists an algorithm that modifies at most ϵm edges to get a $\Phi'/3$-expander $G' = (V, E')$. ■

Combining the results on the separation of weak vertices and patching the graph (Lemmas 6.14 to 6.16) and approximating the endpoint distribution (Lemmas 6.11 and 6.12), we prove the soundness of the algorithm.

6.17 Lemma (Soundness). Let $G = (V, E)$ be a graph. If G is ϵ-far from having conductance at least $\Phi^2/768$, then, with probability at least $2/3$, each vertex in G rejects when it runs Algorithm 6.1. ■

Proof. First we note that if the volume of weak vertices is less than $\epsilon m/100$, then by Lemmas 6.15 and 6.16, the graph is ϵ-close to having conductance at least $\Phi^2/768$. Therefore, the volume of weak vertices is at least $\epsilon m/100$. Each vertex v is contained in S with probability $\Theta(d(v)/\epsilon m)$. Hence, the expected number of weak vertices that are present in the sample S is at least 100. Therefore, with probability at least $9/10$, at least one weak vertex is sampled in S.

If $W^\ell(v, u) < m^{-2}$ for some $v \in S, u \in V$, then with probability at least $9/10$, $\widehat{W}^\ell_{v,u} < 2m^{-2}$ by Lemma 6.11. In this case, the algorithm will reject in line 19 of Algorithm 6.2. If $W^\ell(v, u) \geq m^{-2}$ for all $v \in S, u \in V$, then with probability at least $9/10$, it holds that $\left|\|W^\ell(v, \cdot) - \pi\|_2 - s_v\right| \leq 3m^{-19}$ for every $v \in S$ by Lemma 6.12. Since at least one vertex $v \in S$ is weak, i. e., $\|W^\ell(v, \cdot) - \pi\|_2 > 6m^{-15}$, the algorithm rejects in line 17 of Algorithm 6.1. □

6.3.3. Existence of Weak Vertices

In this section, we prove Lemma 6.14. We would like to show that if the conductance of some set S of vertices is small, then a constant fraction of the volume of S belongs to some (basically) weak vertices. The following statement is a preliminary version of the result that we aim for. It proves the existence of a single vertex with some (unknown) volume only.

6.18 Lemma. Let $S \subset V$ be such that $\mathrm{vol}(S) \leq \mathrm{vol}(\bar{S})$ and $\mathrm{cond}(S) \leq \delta$. Then, for any $\ell \in \mathbb{N}$, there exists a vertex $v \in S$ such that

$$\|W^\ell(v, \cdot) - \pi\|_2^2 > \frac{1}{16m^7}(1 - 4\delta)^{2\ell}\,.$$ ■

Proof. We will argue that $(1/s)\sum_{x \in S}\|W^\ell(x, \cdot) - \pi\|_2^2 > (1 - 4\delta)^{2\ell}/(16m^7)$ and apply an averaging argument to conclude.

Assume for the moment that

$$\sum_{\substack{i \geq 2 \\ \mu_i > \tau}}^{n} \left(\sum_{x \in S} \sqrt{d(x)} f_i(x)\right)^2 \geq \frac{\mathrm{vol}(S)}{4}, \text{ where } \tau = (1 - 4\delta)\,. \tag{6.3}$$

Then, the following calculation concludes the proof:

$$\begin{aligned}
&\frac{1}{s}\sum_{x\in S}\|W^\ell(x,\cdot)-\pi\|_2^2\\
=&\frac{1}{s}\sum_{x\in S}\left\|\frac{D}{2m}\left(2mD^{-1}W^\ell(x,\cdot)-1\right)\right\|_2^2\\
\overset{(a)}{=}&\frac{1}{4m^2 s}\sum_{x\in S}\left\|D\sum_{i\geq 2}^{n}\mu_i^\ell\frac{f_i(x)}{\sqrt{d(x)}}D^{-1/2}f_i\right\|_2^2\\
\overset{(b)}{\geq}&\frac{1}{4m^2 n}\left\|\sum_{x\in S}\frac{D^{1/2}}{s\cdot d(x)}\sum_{i\geq 2}^{n}\mu_i^\ell\sqrt{d(x)}f_i(x)f_i\right\|_2^2\\
\geq&\frac{1}{4m^2n^3s^2}\tau^{2\ell}\left\|\sum_{\substack{i\geq 2\\ \mu_i>\tau}}^{n}\sum_{x\in S}\sqrt{d(x)}f_i(x)f_i\right\|_2^2\\
\overset{(c)}{\geq}&\frac{1}{4m^2n^3s^2}\tau^{2\ell}\sum_{\substack{i\geq 2\\ \mu_i>\tau}}^{n}\left(\sum_{x\in S}\sqrt{d(x)}f_i(x)\right)^2\|f_i\|_2^2\\
\overset{(d)}{\geq}&\frac{1}{4m^2n^3s^2}\tau^{2\ell}\frac{\mathrm{vol}(S)}{4}\\
\overset{(e)}{\geq}&\frac{1}{16m^7}(1-4\delta)^{2\ell},
\end{aligned}$$

where (a) follows from Lemma 6.10, (b) follows from Jensen's inequality, (c) follows because $\{f_i\}_{i\in[n]}$ are orthogonal, (d) follows by Eq. (6.3) and for (e), we assume without loss of generality $\mathrm{vol}(S)\geq 1$.

The remaining calculation is similar to the proof of [KS08, Lemma 3.5]. We prove Eq. (6.3). Let $u = D^{1/2}1_S$. Denote $\alpha_i \coloneqq \left\langle D^{1/2}1_S, f_i\right\rangle$. Representing u in the orthonormal eigenbasis $\{f_i\}_{i\in[n]}$ of N, we have

$$u=\sum_{i=1}^{n}\alpha_i f_i\,.$$

By the definition of the normalized Laplacian,

$$u^TLu = u^TIu - u^TNu\,. \tag{6.4}$$

Observe that

$$u^TIu = \|u\|_2^2 = \sum_i^n \alpha_i^2 \tag{6.5}$$

$$\|u\|_2^2 = \sum_{i\in S}\sqrt{d(i)}^2 = \mathrm{vol}(S)\,. \tag{6.6}$$

The second term of the right-hand side of Eq. (6.4) is equal to

$$\begin{aligned} u^T N u = \left(\sum_{i=1}^{n} \alpha_i f_i^T\right) N \left(\sum_{i=1}^{n} \alpha_i f_i\right) &= \left(\sum_{i=1}^{n} \alpha_i f_i^T\right)\left(\sum_{i=1}^{n} \alpha_i \mu_i f_i\right) \\ &= \left(\sum_{\substack{i,j=1 \\ i=j}}^{n} \alpha_i^2 \mu_i f_i^T f_j\right) + \left(\sum_{\substack{i,j=1 \\ i\neq j}}^{n} \alpha_i^2 \mu_i f_i^T f_j\right) \\ &= \sum_{i=1}^{n} \alpha_i^2 \mu_i + 0\,. \end{aligned} \tag{6.7}$$

Combining Eqs. (6.4), (6.5) and (6.7), we get that

$$u^T L u = u^T I u - u^T N u = \|u\|_2^2 - \sum_{i=1}^{n} \alpha_i^2 \mu_i\,. \tag{6.8}$$

On the other hand,

$$u^T L u = \sum_{\{i,j\}\in E} \left(\frac{u_i}{\sqrt{d(i)}} - \frac{u_j}{\sqrt{d(j)}}\right)^2 = \sum_{\{i,j\}\in E(S,\bar{S})} (1-0) \leq \delta \operatorname{vol}(S)\,. \tag{6.9}$$

Equations (6.6), (6.8) and (6.9) imply

$$\sum_{i\in[n]} \alpha_i^2 \mu_i \geq (1-\delta)\operatorname{vol}(S)\,. \tag{6.10}$$

Let H be the eigenvalues $\mu_i > 1 - 4\delta$, and define $x \coloneqq \sum_{\mu\in H} \alpha_i^2$. Rewriting Eq. (6.10), we get that

$$\begin{aligned} & x + \left(\sum_{i\in[n]} \alpha_i^2 - x\right)(1-4\delta) \geq (1-\delta)\operatorname{vol}(S) \\ \Leftrightarrow \quad & 4\delta x + \operatorname{vol}(S)(1-4\delta) \qquad \geq (1-\delta)\operatorname{vol}(S), \text{ by Eq. (6.5)} \\ \Leftrightarrow \quad & x \qquad\qquad\qquad\qquad\qquad \geq \frac{3\operatorname{vol}(S)}{4} \end{aligned} \tag{6.11}$$

Note that

$$\alpha_1 = \left\langle D^{1/2} 1_S, \sqrt{2m}^{-1} D^{1/2} 1 \right\rangle = \frac{1}{\sqrt{2m}} \sum_{i\in S} d(i) \leq \frac{\operatorname{vol}(S)}{\sqrt{2\operatorname{vol}(S)}} = \sqrt{\frac{\operatorname{vol}(S)}{2}}\,.$$

Therefore, we have that

$$\sum_{\substack{i\geq 2 \\ \mu_i > \tau}}^{n} \left(\sum_{x\in S} \sqrt{d(x)} f_i(x)\right)^2 = x - \alpha_1^2 \geq \frac{\operatorname{vol}(S)}{4}\,. \tag{6.12}$$

□

Actually, our goal is to get a result that is a bit stronger than Lemma 6.18. However, it follows from Lemma 6.18 in the same way as [KS08, Lemma 3.6] follows from [KS08, Lemma 3.5]. It states that even if we exclude some vertices that account for a small fraction of the total volume of S, there exists a basically weak vertex.

6.19 Lemma. Let $T \subseteq S \subset V$ be such that $\mathrm{vol}(S) \leq \mathrm{vol}(\bar{S})$, $\mathrm{cond}(S) \leq \delta$ and $\mathrm{vol}(T) = (1-\theta)\,\mathrm{vol}(S)$ for some $0 < \theta \leq 1/10$. Then, for any $\ell \in \mathbb{N}$, there exists a vertex $v \in T$ such that

$$\|W^\ell(v,\cdot) - \pi\|_2^2 > \frac{1}{80m^7}(1-4\delta)^{2\ell}\,. \qquad \blacksquare$$

Proof. Let $u_S := D^{1/2}1_S$ and $u_T := D^{1/2}1_T$. Let $\alpha_i := \langle u_S, f_i\rangle$ and $\beta_i := \langle u_T, f_i\rangle$. Equation (6.11) holds, where $H = \{\mu_i \mid \mu_i > (1-4\delta)\}$:

$$\sum_{i\in H}\alpha_i^2 > \frac{3\,\mathrm{vol}(S)}{4}\,.$$

It holds that

$$\sum_{i\in H}(\alpha_i-\beta_i)^2 \leq \sum_i(\alpha_i-\beta_i)^2 = \|u_S-u_T\|_2^2 = \sum_{x\in S}d(x) - \sum_{x\in T}d(x) = \mathrm{vol}(S) - \mathrm{vol}(T) = \theta\,\mathrm{vol}(S)\,.$$

Using the triangle inequality $\|a-b\|_2 \geq \|a\|_2 - \|b\|_2$ on the subspace spanned by the basis H, we have

$$\begin{aligned}\sum_{i\in H}\beta_i^2 \geq \left[\sqrt{\sum_{i\in H}\alpha_i^2} - \sqrt{\sum_{i\in H}(\alpha_i-\beta_i)^2}\right]^2 &> \left[\sqrt{\frac{3\,\mathrm{vol}(S)}{4}} - \sqrt{\theta\,\mathrm{vol}(S)}\right]^2 \\ &= \frac{3\,\mathrm{vol}(S)}{4} - \sqrt{\frac{3\theta}{4}}\,\mathrm{vol}(S) + \theta\,\mathrm{vol}(S) \\ &> \frac{11}{20}\,\mathrm{vol}(S)\,.\end{aligned}$$

Observe that $\beta_1^2 \geq \alpha_1^2 \geq \frac{\mathrm{vol}(S)}{2}$. Similar to Eq. (6.10), we have

$$\sum_{\substack{i=2\\ \mu_i>\tau}}^{n}\left(\sum_{x\in T}\sqrt{d(x)}f_i(x)\right)^2 = \sum_{i\in H}\beta_i^2 - \beta_1^2 \geq \frac{\mathrm{vol}(S)}{20}\,.$$

Therefore, we obtain that

$$\frac{1}{s}\sum_{x\in T}\|W^\ell(x,\cdot)-\pi\|_2^2 \geq \frac{\mathrm{vol}(S)}{80m^2n^3s^2}(1-4\delta)^{2\ell}\,. \qquad \square$$

6.14 Lemma. Let $S \subset V$ be such that $\mathrm{vol}(S) \leq \mathrm{vol}(\bar{S})$ and $\mathrm{cond}(S) \leq \delta$. Then, for any $\ell \in \mathbb{N}$ and any $0 < \theta \leq 1/10$, there exists a set $T \subseteq S$ such that $\mathrm{vol}(T) \geq \theta\,\mathrm{vol}(S)$ and for every $v \in T$, it holds that

$$\|W^\ell(v,\cdot) - \pi\|_2^2 > \frac{1}{80m^7}(1-4\delta)^{2\ell}\,.$$

■

Proof. Apply Lemma 6.19, mark the weak vertex that has been found in T and exchange it for some vertex in $S - T$ that has not been marked yet. By Lemma 6.19, we can repeat this process until $\text{vol}(T) \geq \theta \text{vol}(S)$. □

6.3.4. Extension to Unknown Graph Size

We describe how to get rid of the assumption that the size n of the graph G is known to the tester if G is connected. Note that without any prior knowledge of G, no distributed tester can distinguish between a graph with conductance Φ and two distinct copies of it (the latter is ϵ-far from being a graph with conductance Φ^c for $\epsilon < \Phi^c/2$, $c \geq 1$).

First, we describe a slightly simpler version of the final algorithm. In the setting of the simpler algorithm, we mark a single vertex v that will initiate the test and will also give the final answer of the tester. We call this vertex the maintainer (of the graph). The algorithm can be easily adapted to the CONGEST model.

Let $v \in V$ be a fixed vertex. The algorithm either makes n available at all vertices and runs Algorithm 6.1 afterwards or v rejects because G does not have conductance Φ. If G has conductance Φ, the algorithm never rejects.

We start with an initial set $S = \{v\}$ that is grown in two phases. In the first phase, we extend S to $S \cup \Gamma(S)$ as long as $\text{cond}(S) \geq \Phi$. In particular, v starts a BFS and in every round, the vertices in the last level report their degree and the number of neighbors outside of S to their parents. Similar to Algorithm 6.1, these are aggregated and sent to v along the edges of the BFS tree. If $\text{cond}(S) < \Phi$ for the first time, the algorithm proceeds to the second phase. It continues the BFS for $-\log(\text{vol}(S))/\log(1-\Phi)$ rounds and stops. If any vertex in the graph notices a neighbor that is not in S after these rounds, then $S \neq V$ and the algorithm rejects. Otherwise, we have obtained the value of $n = |S|$ that can be sent to all vertices, and we continue by executing Algorithm 6.1.

6.20 Lemma. Let $G = (V, E)$ be a graph and $\Phi \in [0, 1]$. There is an algorithm that computes n if G has conductance at least Φ. Otherwise, it either computes n or rejects. The round complexity is $\mathcal{O}(\log m/\log(1-\Phi))$. ■

Proof. It is easy to see that if the algorithm explores the whole graph, it computes n correctly, and else it rejects. Without loss of generality, let G have conductance Φ. Let S_i be the set S after i rounds and let $\bar{S}_i = V - S_i$. We denote the last round of the first (second) phase by k (ℓ).

In the first phase, we have that $\text{vol}(S_i) \geq (1+\Phi) \cdot \text{vol}(S_{i-1})$ for every round i and by induction,

$$k \leq \frac{\log \text{vol}(S_k)}{\log(1+\Phi)} \leq \frac{\log m}{\log(1+\Phi)}.$$

We also have that $\text{vol}(S_k) \geq m/2 \geq \text{vol}(\bar{S}_k)$ because G has conductance Φ. In the second phase, we have that $\text{vol}(\bar{S}_i) \leq (1-\Phi) \cdot \text{vol}(\bar{S}_{i-1})$ for every round i. By induction,

$$\ell - k \geq -\frac{\log m}{\log(1-\Phi)} \geq \frac{\log \text{vol}(\bar{S}_k)^{-1}}{\log(1-\Phi)}$$

implies that that $\text{vol}(\bar{S}_\ell) = 0$. Therefore, the algorithm has explored the whole graph. Clearly, $\ell \in \mathcal{O}(\log m/\log(1-\Phi))$. □

To transform the algorithm into a tester in the CONGEST model, we start with each vertex being a maintainer initially. In every round every vertex chooses the vertex with the smallest id it has ever received a message from to be the maintainer and it forwards only this vertex' messages (the latter maintains the congestion bound). At the end of the algorithm, if G has conductance Φ, then there is only one maintainer (the vertex with the smallest id) and the algorithm continues by executing Algorithm 6.1. Otherwise, there might be multiple vertices that are still maintainers. However, none of these vertices has explored the whole (connected) graph, so all of them send a broadcast message to reject.

6.4. Lower Bound for the LOCAL Model

In this section, we prove a lower bound of $\Omega_{\Phi,\epsilon}(\log(n+m))$ on the round complexity for testing the conductance of a graph in the LOCAL model regardless of how the final decision of the tester is derived from the single votes of the vertices.

6.4.1. Setup of the Lower Bound

We extend the definition of k-disks by an order on the neighbors of every vertex that relates to the port numbers in the LOCAL model (see Definition 1.3).

6.21 Definition (ordered k-disk). For any $v \in V$, the *ordered k-disk* of v, denoted by $\text{odisk}_k(G, v)$, is the k-disk of v, $\text{disk}_k(G, v)$, that stores, for every vertex $u \in \text{disk}_k(G, v)$, a total order on its neighbors $w \in \Gamma(u)$. Two k-disks are isomorphic if and only if there exists a *root-and-order-preserving graph isomorphism* between them, i. e., a graph isomorphism that identifies the roots and preserves the orders of neighbors. ■

We need the following two lemmas to obtain the distribution over graphs to prove the lower bound.

6.22 Lemma ([LPS88]; cf. [12, Section 16.8.3]). For every $n' \in \mathbb{N}$ and every $d' \in \mathbb{N}$ there exists a d-regular graph G of size n such that G has conductance $\Phi(G) = 1/\sqrt{2d}$ and girth $2\log n/\log d$, and $n \geq n'$, $d \geq d'$. ■

The second lemma states that we can sparsify an arbitrary cut $E(V_1, V_2)$ in a d-regular graph with girth $3k$ without changing $\text{odisk}_k(G, v)$ for any $v \in V$. In particular, it states that we can remove two edges in the cut and add them somewhere else, or the cut has size $\text{poly}(d^k)$ only. It is obtained as a special case of Proposition 2.11 by observing that we can assume $D_k = 1$ and $\lambda = 0$.

6.23 Lemma (cf. Proposition 2.11). Let $G = (V, E)$ be a d-regular graph with $\text{girth}(G) \geq 3k$ for $k \geq 2$ and let $V_1 \uplus V_2 = V$ be a partitioning of V. Then either there exists a graph $H = (V, F)$ such that

$$\begin{gathered}\text{girth}(H) \geq 3k \\ |F \cap (V_1 \times V_2)| \leq |E \cap (V_1 \times V_2)| - 2 \\ \text{odisk}_k(H, w) = \text{odisk}_k(G, w) \forall w \in V\end{gathered}$$

or $E(V_1, V_2) \leq 6d^{3k}$. ■

Proof. We note that there is only one k-disk isomorphism type in a d-regular graph with girth$(G) \geq 3k$. As this isomorphism type is a full d-ary unlabeled tree, every isomorphism between two k-disks of this type is also an isomorphism between two ordered k-disks of this type. Therefore, we may assume that $D_k = 1$ in the statement of Proposition 2.11, which implies $\lambda = 0$. The claim follows. □

To prove the lower bound, we use an auxiliary model we call the ISO-LOCAL model. In this model, the input $I(\cdot)$ is empty but an additional oracle provides every vertex v with the ability to construct the isomorphism type of $\text{odisk}_r(G, v)$ in round r if it knows the type of $\text{odisk}_{r-1}(G, u_i)$ of its neighbors $u_1, \dots, u_{d(v)}$. Note that the ISO-LOCAL model is not a DCM due to the additional oracle.

6.24 Definition (ISO-LOCAL model). Let $\text{DCM}(G, p_G, I)$ be a DCM instance such that $I(\cdot)$ maps the whole domain to n. In addition to sending and receiving messages, in every round r every vertex v is provided access to a function $e_{r,v} : (\mathbb{N} \cup \{\star\})^r \times (\mathbb{N} \cup \{\star\})^r \to \{0, 1\}$ during the local computation phase. The value of $e_{r,v}((i_1, \dots, i_{r'}), (j_1, \dots, j_{r''}))$ is 1 if and only if $p'_v(i_1, \dots, i_{r'}) = p'_v(j_1, \dots, j_{r''})$, where

$$p'_v(i_1, \dots, i_r) := \begin{cases} v & \text{if } i_r = \star \\ p_{p'_v(\star, i_1, \dots, i_{r-1})}(i_r) & \text{otherwise}. \end{cases}$$

The instance $\text{DCM}(G, p_G, I)$ equipped with such an oracle is called ISO-LOCAL. ■

In other words, $p'_v(\cdot)$ takes a path of length at most r that starts at v and that is defined by a sequence of port numbers as input. Then, it maps the path to its endpoint in V. Finally, $e_{r,v}(\cdot)$ tells whether two such paths end at the same vertex.

The ISO-LOCAL model is a graph where the nodes are not labeled by any strings. To argue the lower bound, we need to prove the existence of graphs that have same local neighborhoods such that one is a good expander and the other is far from having good conductance. Now it is possible that the algorithm can glean information about the different graphs based on the vertex labels even if the local neighborhoods are identical. Without ISO-LOCAL, we would need to argue that a randomized algorithm cannot deduce information from the vertex labels in the LOCAL model directly. This would be easy if for every good expander G we use, there is a bad expander H with exactly the same set of (labeled) k-disks. However, this is not the case as it would imply that G and H are isomorphic. The ISO-LOCAL model formalizes the intuition that, still, isomorphic k-disks should be sufficient to establish the lower bound even for randomized algorithms.

It is a basic observation that a distributed algorithm can only depend on information that has reached it until the moment it performs the computation in question.

6.25 Lemma (folklore; cf. [Lin92, Section 2]). Let $\text{DNDA}(\mathscr{A}, O)$ be a DNDA. After r rounds, the state of $\mathscr{A}_v$ may depend only on $d(v)$, $I(v)$, the state of $\mathscr{A}_u$ at time $r - \text{dt}(v, u)$ for vertices u with $\text{dt}(v, u) < r$ and the random coins of $\mathscr{A}$. ■

6.4.2. Proof of the Lower Bound

Let $G = (V, E)$ be an expander graph obtained by applying Lemma 6.22 and let $k = \Theta(\log n)$. Observe that if a graph is d-regular and it has girth $3k$, then all its k-disks are pairwise isomorphic. In particular, all k-disks are full d-ary trees of depth k.

We will prove that a distributed algorithm DNDA($\mathscr{A}, O$) with round complexity r in the ISO-LOCAL model decides based on the set of views – ordered r-disk isomorphism types – that the different instances of $\mathscr{A}$ have (see Lemma 6.26). Using Proposition 2.11, it will be easy to come up with a graph H that is a bad expander but whose ordered k-disks are isomorphic to the ones of G. This implies a lower bound of $k = \Theta(\log n)$ for testing conductance in the ISO-LOCAL model (see Proposition 6.27). Finally, we prove that a lower bound on the round complexity of a tester in the ISO-LOCAL model implies the same bound in the LOCAL model. Actually, we prove the contrapositive: A tester in the LOCAL model implies a tester in the ISO-LOCAL model (see Proposition 6.28).

6.26 Lemma. Let DNDA($\mathscr{A}, O$) be a deterministic DNDA in the ISO-LOCAL model. The output of $\mathscr{A}_v$ depends only on the isomorphism type of $\text{odisk}_r(G, v)$. ■

Proof. Instead of analyzing DNDA($\mathscr{A}, O$), we will analyze a canonical algorithm DNDA($\mathscr{B}, O$) that simulates DNDA($\mathscr{A}, O$) depending only on the isomorphism type of $\text{odisk}_r(G, v)$. Employing $\mathscr{B}$, we prove the following statement by induction: After the local computation phase of round r, the state of $\mathscr{A}_v$ depends only on the type of $\text{odisk}_r(G, v)$.

The first local computation phase of $\mathscr{A}_v$ can only depend on the port numbering and $I(v)$ (the empty string). Therefore, $\mathscr{B}_v$ can simulate the execution of the first round of $\mathscr{A}_v$.

Let the current round be $r > 1$. Algorithm $\mathscr{B}_v$ maintains a rooted graph H_v that resembles $\text{odisk}_r(G, v)$. The adjacency lists of H_v are ordered according to $(p_v)_{v \in V}$. Let $H_v(r)$ be the value of H_v after the computation phase of round r. In the send phase, vertex v sends $H_v(r)$ to each of its neighbors. In the receive phase, vertex v receives graphs $H_{u_1}(r), \dots, H_{u_{d(v)}}(r)$ from its neighbors $u_1, \dots, u_{d(v)}$. In the subsequent computation phase of round $r + 1$, vertex v extends $H_v(r) = \text{odisk}_r(G, v)$ to $\text{odisk}_{r+1}(G, v) = H_v(r + 1)$ by querying $e_{r,v}$ on all pairs of vertices of $V(H_v(r)) \cup V(H_{u_1}(r)) \cup \dots \cup V(H_{u_{d(v)}}(r))$ to identity vertices and patching the different views together.

Note that $H_v(r + 1)$ also provides the isomorphism type of $\text{odisk}_{r-\text{dt}_G(v,u)}(G, u)$ for every vertex u at distance at most r from v. Since the adjacency lists of H_v are ordered according to the port numbering, it is also possible to reconstruct $e_{r-\text{dt}_G(v,u),u}(\cdot)$. By the induction hypothesis, $\mathscr{B}_v$ can now simulate round $r - \text{dt}_G(v, u)$ of $\mathscr{A}_u$ for every such u. By Lemma 6.25, this is enough to simulate the local computation phase of round r of $\mathscr{A}_u$. □

We show that there is no tester for conductance in the ISO-LOCAL model with query complexity $\omega(\log n)$.

6.27 Proposition. Let $G = (V, E)$ be d-regular graph on n vertices, and let $\Phi = 1/\sqrt{2d}$ be a constant. Any algorithm for testing if G has conductance at least Φ or is ϵ-far from having conductance at least $c\Phi^2$ (for constants c and ϵ) in the ISO-LOCAL model that succeeds with probability $2/3$ requires $\Omega(\log n)$ rounds of communication. ■

Proof. Let $G = (V, E)$ be a d-regular graph provided by Lemma 6.22 and set $k = (1/3) \log_d((c\Phi^2 - \epsilon)dn/6)$. Without loss of generality assume that n is even, and let $S \subset V$ be a set of size $n/2$. Apply Proposition 2.11 (with $V_1 = S$ and $V_2 = V - S$) repeatedly to G until $|E(S, V - S)| \leq 6d^{3k}$ holds. Let $H = (V, E')$ be the resulting graph. We have that $|E'(S, V - S)| \leq (c\Phi^2 - \epsilon)dn$, and $\text{vol}(S) = nd/2$. Therefore, H is ϵ-far from having conductance $c\Phi^2$. Let $\mathscr{D}_G$ ($\mathscr{D}_H$) be the uniform distribution over all ISO-LOCAL models DCM(G, p_G, I) (DCM(H, p_H, I)) such that p_G (p_H) ranges over all possible mappings, i. e., port numberings.

We use Yao's principle to prove the lower bound. Let DNDA($\mathscr{A}, O$) be a tester for conductance

that has round complexity smaller than k in the ISO-LOCAL model. Since G is d-regular and girth$(G) \geq 3k$, odisk$_k(G, v)$ is a full d-ary tree of depth k for every $v \in V$. For any pair $u, v \in V$, we have that odisk$_k(G, u)$ is equal to odisk$_k(H, v)$ by Proposition 2.11. By Lemma 6.26, DNDA$(\mathscr{A}, O)$ cannot distinguish between G and H. □

To complete the proof of the lower bound, we show that each vertex in the graph in the ISO-LOCAL model can choose an id randomly.

6.28 Proposition. Let DNDA$(\mathscr{A}, O)$ be a randomized tester in the LOCAL model that succeeds with probability p. Then, there is a randomized tester DNDA$(\mathscr{B}, O)$ in the ISO-LOCAL model that succeeds with probability at least $p - o(1)$, and has the same round complexity. ■

Proof. We make a simple modification to $\mathscr{A}$ to obtain $\mathscr{B}$: In the first local computation phase, $\mathscr{B}_v$ draws a random number id_v uniformly from $\{1, \dots, n^3\}$ and feeds it into $\mathscr{A}_v$ as $I(v)$. Then, $\mathscr{A}_v$ is executed as normal. For $u, v \in V$, the probability that id_u and id_v are equal is $1/n^3$. Applying a union bound, with probability $1 - o(1)$, it holds that $id_u \neq id_v$ for every $u, v \in V$. We then run algorithm $\mathscr{A}$ on this new instance and output the result. □

A. List of Coauthored Sources

The main body of this work is based on previous works [FPS15; Fic+18; FV18; FPS19; Czu+20] that were coauthored by the author of this thesis. This section is provided to comply with good scientific practice [TU 17a; TU 17b]. All related work that was not coauthored by the author is listed in the respective chapters.

Chapter 2 is based on joint work with Pan Peng and Christian Sohler [FPS15]. All authors contributed equally to the work of [FPS15] presented in this thesis, which is listed in the following. The original texts of Section 2.1 were written for this thesis. In regards of the results, Theorem 2.3 matches Theorem 1 and Theorem 2.4 matches Theorem 2 from [FPS15]. Section 2.2 was rewritten and some of parts of it were merged with other preliminaries in Section 1.3. Sections 2.3 and 2.4 were slightly modified to fix minor errors and improve readability.

Chapter 3 is based on joint work with Pan Peng and Christian Sohler [FPS19]. All authors contributed equally to the work of [FPS19] presented in this thesis, which is listed in the following. The original texts of Section 3.1 were written for this thesis. In regards of the results, Theorem 3.2 matches Theorem 1.1, Theorem 3.3 matches Theorem 1.2 and Theorem 3.4 matches Theorem 5.1 in [FPS19]. Section 3.2 was rewritten and some of parts of it were merged with other preliminaries in Section 1.3. Section 3.3 was modified by fixing minor errors, extending some proofs and improving readability. Most notably, this affects the proof of Theorem 3.11 and the structure of Section 3.3.3. Sections 3.4 and 3.5 were slightly modified to fix minor errors and improve readability.

Chapter 4 is based on joint work with Artur Czumaj, Pan Peng and Christian Sohler [Czu+20]. All authors contributed equally to the work of [Czu+20] presented in this thesis, which is listed in the following. The original texts of Section 4.1 to Section 4.1.3 were written for this thesis. Section 4.1.4 was slightly modified to improve readability. In regard of the results, Theorem 4.1 matches Theorem 1.4, Theorem 4.2 matches Theorem 6.3 and Theorem 4.15 matches Theorem 1.2 and Theorem 6.2 from [Czu+20]. Section 4.2 was rewritten and some parts of it were merged with other preliminaries in Section 1.3. Sections 4.3 and 4.4 were slightly modified to improve readability.

Chapter 5 is based on joint work with Reut Levi, Maximilian Wötzel and Yadu Vasudev [Fic+18]. All authors contributed equally to the work of [Fic+18] presented in this thesis, which is listed in the following. The original texts of Section 5.1 were written for this thesis. In regard of the results, Theorem 5.1 matches Theorem 1 from [Fic+18]. Section 5.2 was rewritten and some parts of it were merged with other preliminaries in Section 1.3. Sections 5.3 to 5.5 were modified to fix minor errors, add omitted proofs and improve readability.

Chapter 6 is based on joint work with Yadu Vasudev [FV18]. All authors contributed equally to the work of [FV18] presented in this thesis, which is listed in the following. The original texts of Section 6.1 were written for this thesis. In regard of the results, Theorem 6.1 matches Theorem 1 and Theorem 6.2 matches Theorem 2 from [FV18]. Section 6.2 was rewritten and some parts of it were merged with other preliminaries in Section 1.3. Sections 6.3 and 6.4 were slightly modified to improve readability.

B. Bibliography

[AH74] A. V. Aho and J. E. Hopcroft. *The Design and Analysis of Computer Algorithms*. Addison-Wesley Longman Publishing Co., Inc., 1974. ISBN: 978-0-201-00029-0 (cit. on pp. 33, 35).

[AL07] D. Aldous and R. Lyons. "Processes on Unimodular Random Networks". In: *Electronic Journal of Probability* 12 (2007). DOI: 10.1214/EJP.v12-463 (cit. on p. 24).

[Alo+09] N. Alon, E. Fischer, I. Newman, and A. Shapira. "A Combinatorial Characterization of the Testable Graph Properties: It's All About Regularity". In: *SIAM Journal on Computing* 39.1 (2009). DOI: 10.1137/060667177 (cit. on pp. 37–40, 52).

[AK02] N. Alon and M. Krivelevich. "Testing K-Colorability". In: *SIAM Journal on Discrete Mathematics* 15.2 (2002). DOI: 10.1137/S0895480199358655 (cit. on p. 17).

[Alo+99] N. Alon, M. Krivelevich, E. Fischer, and M. Szegedy. "Efficient Testing of Large Graphs". In: *Proceedings of the 40th Annual Symposium on Foundations of Computer Science (FOCS)*. 1999. DOI: 10.1109/SFFCS.1999.814642 (cit. on p. 39). Journal version: N. Alon, E. Fischer, M. Krivelevich, and M. Szegedy. "Efficient Testing of Large Graphs". In: *Combinatorica* 20.4 (2000). DOI: 10.1007/s004930070001.

[AST90] N. Alon, P. Seymour, and R. Thomas. "A Separator Theorem for Graphs with an Excluded Minor and Its Applications". In: *Proceedings of the 22nd Annual ACM Symposium on Theory of Computing (STOC)*. Association for Computing Machinery, 1990. DOI: 10.1145/100216.100254 (cit. on p. 14).

[Alo11] N. Alon. "Proof of a Conjecture by László Lovász". Bertinoro Workshop on Sublinear Algorithms. 2011 (cit. on pp. 15, 22).

[Alo+12] N. Alon, R. Rubinfeld, S. Vardi, and N. Xie. "Space-Efficient Local Computation Algorithms". In: *Proceedings of the 23rd Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. SIAM, 2012. DOI: 10.1137/1.9781611973099.89 (cit. on p. 59).

[Ass+19] S. Assadi, M. Bateni, A. Bernstein, V. Mirrokni, and C. Stein. "Coresets Meet EDCS: Algorithms for Matching and Vertex Cover on Massive Graphs". In: *Proceedings of the 30th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. Society for Industrial and Applied Mathematics, 2019. DOI: 10.1137/1.9781611975482.98 (cit. on p. 57).

[AKL17] S. Assadi, S. Khanna, and Y. Li. "On Estimating Maximum Matching Size in Graph Streams". In: *Proceedings of the 28th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. Society for Industrial and Applied Mathematics, 2017. DOI: 10.1137/1.9781611974782.113 (cit. on p. 56).

[BKN16] J. Babu, A. Khoury, and I. Newman. "Every Property of Outerplanar Graphs Is Testable". In: *Proceedings of the 19th / 20th International Workshops on Approximation Algorithms for Combinatorial Optimization Problems / Randomization and Computation (APPROX / RANDOM)*. Vol. 60. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2016. DOI: 10.4230/LIPIcs.APPROX-RANDOM.2016.21 (cit. on p. 59).

[Bat96] J. Bather. "A Conversation with Herman Chernoff". In: *Statistical Science* 11.4 (1996). DOI: 10.1214/ss/1032280306 (cit. on p. 19).

[BS01] I. Benjamini and O. Schramm. "Recurrence of Distributional Limits of Finite Planar Graphs". In: *Electronic Journal of Probability* 6 (2001). DOI: 10.1214/EJP.v6-96 (cit. on p. 24).

[BSS08] I. Benjamini, O. Schramm, and A. Shapira. "Every Minor-Closed Property of Sparse Graphs Is Testable". In: *Proceedings of the 40th Annual ACM Symposium on Theory of Computing (STOC)*. Association for Computing Machinery, 2008. DOI: 10.1145/1374376.1374433 (cit. on pp. 40, 41, 43, 85–87). Journal version: "Every Minor-Closed Property of Sparse Graphs Is Testable". In: *Advances in Mathematics* 223.6 (2010). DOI: 10.1016/j.aim.2009.10.018.

[BLR90] M. Blum, M. Luby, and R. Rubinfeld. "Self-Testing/Correcting with Applications to Numerical Problems". In: *Proceedings of the 22nd Annual ACM Symposium on Theory of Computing (STOC)*. Association for Computing Machinery, 1990. DOI: 10.1145/100216.100225 (cit. on p. 11). Journal version: M. Blum, M. Luby, and R. Rubinfeld. "Self-Testing/Correcting with Applications to Numerical Problems". In: *Journal of Computer and System Sciences* 47.3 (1993). DOI: 10.1016/0022-0000(93)90044-W.

[BLR93] M. Blum, M. Luby, and R. Rubinfeld. "Self-Testing/Correcting with Applications to Numerical Problems". In: *Journal of Computer and System Sciences* 47.3 (1993). DOI: 10.1016/0022-0000(93)90044-W (cit. on p. 11).

[Bod93] H. L. Bodlaender. "On Linear Time Minor Tests with Depth-First Search". In: *Journal of Algorithms* 14.1 (1993). DOI: 10.1006/jagm.1993.1001 (cit. on p. 89).

[Bod06] H. L. Bodlaender. "A Linear-Time Algorithm for Finding Tree-Decompositions of Small Treewidth". In: *SIAM Journal on Computing* (2006). DOI: 10.1137/S0097539793251219 (cit. on p. 99).

[BT97] B. Bollobás and A. Thomason. "On the Girth of Hamiltonian Weakly Pancyclic Graphs". In: *Journal of Graph Theory* 26.3 (1997). DOI: 10.1002/(SICI)1097-0118(199711)26:3<165::AID-JGT7>3.0.CO;2-P (cit. on p. 87).

[BP09] Z. Brakerski and B. Patt-Shamir. "Distributed Discovery of Large Near-Cliques". In: *Distributed Computing*. Springer, Berlin, Heidelberg, 2009. DOI: 10.1007/978-3-642-04355-0_22 (cit. on p. 105). Journal version: "Distributed Discovery of Large Near-Cliques". In: *Distributed Computing* 24.2 (2011). DOI: 10.1007/s00446-011-0132-x.

[BS15] M. Bury and C. Schwiegelshohn. “Sublinear Estimation of Weighted Matchings in Dynamic Data Streams”. In: *Proceedings of the 23rd Annual European Symposium on Algorithms (ESA)*. Springer Berlin Heidelberg, 2015. DOI: 10.1007/978-3-662-48350-3_23 (cit. on pp. 56, 64).

[CG18] C. L. Canonne and T. Gur. “An Adaptivity Hierarchy Theorem for Property Testing”. In: *computational complexity* 27.4 (2018). DOI: 10.1007/s00037-018-0168-4 (cit. on p. 60).

[Car07] C. Carathéodory. “Über den Variabilitätsbereich der Koeffizienten von Potenzreihen, die gegebene Werte nicht annehmen”. In: *Mathematische Annalen* 64.1 (1907). DOI: 10.1007/BF01449883 (cit. on p. 24).

[Cay89] Cayle, Arthur. “A Theorem on Trees”. In: *The quarterly journal of pure and applied mathematics* 23 (1889) (cit. on p. 33).

[Cen+16] K. Censor-Hillel, E. Fischer, G. Schwartzman, and Y. Vasudev. “Fast Distributed Algorithms for Testing Graph Properties”. In: *Proceedings of the 30th International Symposium on Distributed Computing (DISC)*. Springer Berlin Heidelberg, 2016 (cit. on p. 105). Journal version: “Fast Distributed Algorithms for Testing Graph Properties”. In: *Distributed Computing* 32.1 (2019). DOI: 10.1007/s00446-018-0324-8.

[CCM08] A. Chakrabarti, G. Cormode, and A. McGregor. “Robust Lower Bounds for Communication and Stream Computation”. In: *Proceedings of the 40th Annual ACM Symposium on Theory of Computing (STOC)*. Association for Computing Machinery, 2008. DOI: 10.1145/1374376.1374470 (cit. on p. 57).

[CH67] G. Chartrand and F. Harary. “Planar Permutation Graphs”. In: *Annales de l’I.H.P. Probabilités et statistiques* 3.4 (1967) (cit. on p. 14).

[Che52] H. Chernoff. “A Measure of Asymptotic Efficiency for Tests of a Hypothesis Based on the Sum of Observations”. In: *The Annals of Mathematical Statistics* 23.4 (1952). DOI: 10.1214/aoms/1177729330 (cit. on p. 19).

[Chi+16] R. Chitnis, G. Cormode, H. Esfandiari, M. Hajiaghayi, A. McGregor, M. Monemizadeh, and S. Vorotnikova. “Kernelization via Sampling with Applications to Finding Matchings and Related Problems in Dynamic Graph Streams”. In: *Proceedings of the 27th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. Society for Industrial and Applied Mathematics, 2016. DOI: 10.1137/1.9781611974331.ch92 (cit. on p. 56).

[Chu89] F. R. K. Chung. “Diameters and Eigenvalues”. In: *Journal of the American Mathematical Society* 2.2 (1989). DOI: 10.1090/S0894-0347-1989-0965008-X (cit. on p. 108).

[Chu97] F. R. K. Chung. *Spectral Graph Theory*. American Mathematical Soc., 1997. ISBN: 978-0-8218-8936-7 (cit. on p. 108).

[12] *Combinatorial Scientific Computing*. Chapman & Hall/CRC Computational Science 12. CRC Press, 2012. ISBN: 978-1-4398-2735-2 (cit. on p. 119).

[CF12] D. Conlon and J. Fox. "Bounds for Graph Regularity and Removal Lemmas". In: *Geometric and Functional Analysis* 22.5 (2012). DOI: 10.1007/s00039-012-0171-x (cit. on p. 39).

[Cor+17] G. Cormode, H. Jowhari, M. Monemizadeh, and S. Muthukrishnan. "The Sparse Awakens: Streaming Algorithms for Matching Size Estimation in Sparse Graphs". In: *Proceedings of the 25th Annual European Symposium on Algorithms (ESA)*. Vol. 87. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2017. DOI: 10.4230/LIPIcs.ESA.2017.29 (cit. on p. 56).

[Cou90] B. Courcelle. "The Monadic Second-Order Logic of Graphs. I. Recognizable Sets of Finite Graphs". In: *Information and Computation* 85.1 (1990). DOI: 10.1016/0890-5401(90)90043-H (cit. on p. 99).

[Czu+11] A. Czumaj, M. Monemizadeh, K. Onak, and C. Sohler. "Planar Graphs: Random Walks and Bipartiteness Testing". In: *Proceedings of the 52nd Annual IEEE Symposium on Foundations of Computer Science (FOCS)*. 2011. DOI: 10.1109/FOCS.2011.69 (cit. on pp. 40, 59).

[CSS09] A. Czumaj, A. Shapira, and C. Sohler. "Testing Hereditary Properties of Nonexpanding Bounded-Degree Graphs". In: *SIAM Journal on Computing* 38.6 (2009). DOI: 10.1137/070681831 (cit. on p. 40).

[CS07] A. Czumaj and C. Sohler. "Testing Expansion in Bounded-Degree Graphs". In: *Proceedings of the 48th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*. 2007. DOI: 10.1109/FOCS.2007.33 (cit. on p. 104). Journal version: A. Czumaj and C. Sohler. "Testing Expansion in Bounded-Degree Graphs". In: *Combinatorics, Probability and Computing* 19 (2010). DOI: 10.1017/S096354831000012X.

[Czu+20] A. Czumaj, H. Fichtenberger, P. Peng, and C. Sohler. "Testable Properties in General Graphs and Random Order Streaming". In: *To appear in the proceedings of the 23rd / 24th International Workshops on Approximation Algorithms for Combinatorial Optimization Problems / Randomization and Computation (APPROX / RANDOM)*. 2020 (cit. on pp. 55, 123).

[Czu+14] A. Czumaj, O. Goldreich, D. Ron, C. Seshadhri, A. Shapira, and C. Sohler. "Finding Cycles and Trees in Sublinear Time". In: *Random Structures & Algorithms* 45.2 (2014). DOI: 10.1002/rsa.20462 (cit. on pp. 86, 87).

[CPS16] A. Czumaj, P. Peng, and C. Sohler. "Relating Two Property Testing Models for Bounded Degree Directed Graphs". In: *Proceedings of the 48th Annual ACM Symposium on Theory of Computing (STOC)*. Association for Computing Machinery, 2016. DOI: 10.1145/2897518.2897575 (cit. on p. 44).

[CPS15] A. Czumaj, P. Peng, and C. Sohler. "Testing Cluster Structure of Graphs". In: *Proceedings of the 47th Annual ACM Symposium on Theory of Computing (STOC)*. Association for Computing Machinery, 2015. DOI: 10.1145/2746539.2746618 (cit. on pp. 66, 104).

[Das+13] A. Das Sarma, D. Nanongkai, G. Pandurangan, and P. Tetali. "Distributed Random Walks". In: *Journal of the American Mathematical Society* 60.1 (2013). DOI: 10.1145/2432622.2432624 (cit. on p. 105).

[Die17] R. Diestel. *Graph Theory*. Springer-Verlag, 2017. ISBN: 978-3-662-53621-6 (cit. on p. 13).

[Ede+11] A. Edelman, A. Hassidim, H. N. Nguyen, and K. Onak. "An Efficient Partitioning Oracle for Bounded-Treewidth Graphs". In: *Proceedings of the 14th / 15th International Workshops on Approximation Algorithms for Combinatorial Optimization Problems / Randomization and Computation (APPROX / RANDOM)*. Springer Berlin Heidelberg, 2011. DOI: 10.1007/978-3-642-22935-0_45 (cit. on pp. 59, 86).

[Ede+15] T. Eden, A. Levi, D. Ron, and C. Seshadhri. "Approximately Counting Triangles in Sublinear Time". In: *Proceedings of the 56th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*. 2015. DOI: 10.1109/FOCS.2015.44 (cit. on p. 64). Journal version: T. Eden, D. Ron, and C. Seshadhri. "On Approximating the Number of K-Cliques in Sublinear Time". In: *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing (STOC)*. Association for Computing Machinery, 2018. DOI: 10.1145/3188745.3188810.

[ERS18] T. Eden, D. Ron, and C. Seshadhri. "On Approximating the Number of K-Cliques in Sublinear Time". In: *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing (STOC)*. Association for Computing Machinery, 2018. DOI: 10.1145/3188745.3188810 (cit. on p. 64).

[ER18] T. Eden and W. Rosenbaum. "On Sampling Edges Almost Uniformly". In: *Proceedings of the 1st Symposium on Simplicity in Algorithms (SOSA)*. Vol. 61. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2018. DOI: 10.4230/OASIcs.SOSA.2018.7 (cit. on p. 64).

[Ele07a] G. Elek. "Note on Limits of Finite Graphs". In: *Combinatorica* 27.4 (2007). DOI: 10.1007/s00493-007-2214-8 (cit. on p. 24).

[Ele10] G. Elek. "On the Limit of Large Girth Graph Sequences". In: *Combinatorica* 30.5 (2010). DOI: 10.1007/s00493-010-2559-2 (cit. on p. 24).

[Ele07b] G. Elek. "The Combinatorial Cost". In: *L'Enseignement Mathématique* 53.3-4 (2007). DOI: 10.5169/seals-109545 (cit. on p. 14).

[EN17] M. Elkin and O. Neiman. "Efficient Algorithms for Constructing Very Sparse Spanners and Emulators". In: *Proceedings of the 28th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. Society for Industrial and Applied Mathematics, 2017. DOI: 10.1145/3274651 (cit. on pp. 90, 97, 98).

[ES35] P. Erdös and G. Szekeres. "A Combinatorial Problem in Geometry". In: *Compositio Mathematica* 2 (1935) (cit. on p. 90).

[Esf+18] H. Esfandiari, M. Hajiaghayi, V. Liaghat, M. Monemizadeh, and K. Onak. "Streaming Algorithms for Estimating the Matching Size in Planar Graphs and Beyond". In: *ACM Trans. Algorithms* 14.4 (2018). DOI: 10.1145/3230819 (cit. on p. 56).

[Eve+17] G. Even, O. Fischer, P. Fraigniaud, T. Gonen, R. Levi, M. Medina, P. Montealegre, D. Olivetti, R. Oshman, I. Rapaport, and I. Todinca. "Three Notes on Distributed Property Testing". In: *Proceedings of the 31st International Symposium on Distributed Computing (DISC)*. Vol. 91. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2017. DOI: 10.4230/LIPIcs.DISC.2017.15 (cit. on pp. 105, 106).

[Fei06] U. Feige. "On Sums of Independent Random Variables with Unbounded Variance and Estimating the Average Degree in a Graph". In: *SIAM Journal on Computing* 35.4 (2006). DOI: 10.1137/S0097539704447304 (cit. on p. 64).

[FPV18] U. Feige, B. Patt-Shamir, and S. Vardi. "On the Probe Complexity of Local Computation Algorithms". In: *Proceedings of the 45th International Colloquium on Automata, Languages, and Programming (ICALP)*. Vol. 107. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2018. DOI: 10.4230/LIPIcs.ICALP.2018.50 (cit. on p. 59).

[FPS19] H. Fichtenberger, P. Peng, and C. Sohler. "Every Testable (Infinite) Property of Bounded-Degree Graphs Contains an Infinite Hyperfinite Subproperty". In: *Proceedings of the 30th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. Society for Industrial and Applied Mathematics, 2019. DOI: 10.1137/1.9781611975482.45 (cit. on pp. 37, 123).

[Fic14] H. Fichtenberger. *Explicit Upper Bounds on the Minimum Size of Planar Graphs That Satisfy a Given Distribution of K-Disks*. 2014 (cit. on pp. 24, 43).

[Fic+18] H. Fichtenberger, R. Levi, Y. Vasudev, and M. Wötzel. "A Sublinear Tester for Outerplanarity (and Other Forbidden Minors) With One-Sided Error". In: *Proceedings of the 45th International Colloquium on Automata, Languages, and Programming (ICALP)*. Vol. 107. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2018. DOI: 10.4230/LIPIcs.ICALP.2018.52 (cit. on pp. 85, 123).

[FPS15] H. Fichtenberger, P. Peng, and C. Sohler. "On Constant-Size Graphs That Preserve the Local Structure of High-Girth Graphs". In: *Proceedings of the 18th / 19th International Workshops on Approximation Algorithms for Combinatorial Optimization Problems / Randomization and Computation (APPROX / RANDOM)*. Vol. 40. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2015. DOI: 10.4230/LIPIcs.APPROX-RANDOM.2015.786 (cit. on pp. 21, 123).

[FV18] H. Fichtenberger and Y. Vasudev. "A Two-Sided Error Distributed Property Tester For Conductance". In: *Proceedings of the 43rd International Symposium on Mathematical Foundations of Computer Science (MFCS)*. Vol. 117. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2018. DOI: 10.4230/LIPIcs.MFCS.2018.19 (cit. on pp. 103, 123).

[FN07] E. Fischer and I. Newman. "Testing versus Estimation of Graph Properties". In: *SIAM Journal on Computing* 37.2 (2007). DOI: 10.1137/060652324 (cit. on pp. 39, 40).

[FO17] P. Fraigniaud and D. Olivetti. "Distributed Detection of Cycles". In: *Proceedings of the 29th ACM Symposium on Parallelism in Algorithms and Architectures (SPAA)*. Association for Computing Machinery, 2017. DOI: 10.1145/3087556.3087571 (cit. on p. 105).

[Fra+16] P. Fraigniaud, I. Rapaport, V. Salo, and I. Todinca. "Distributed Testing of Excluded Subgraphs". In: *Proceedings of the 30th International Symposium on Distributed Computing (DISC)*. Springer Berlin Heidelberg, 2016. DOI: 10.1007/978-3-662-53426-7_25 (cit. on p. 105).

[GKK12] A. Goel, M. Kapralov, and S. Khanna. "On the Communication and Streaming Complexity of Maximum Bipartite Matching". In: *Proceedings of the 23rd Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. Society for Industrial and Applied Mathematics, 2012. DOI: 10.1137/1.9781611973099.41 (cit. on p. 56).

[GGR96] O. Goldreich, S. Goldwasser, and D. Ron. "Property Testing and Its Connection to Learning and Approximation". In: *Proceedings of 37th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*. 1996. DOI: 10.1109/SFCS.1996.548493 (cit. on pp. 17, 39). Journal version: O. Goldreich, S. Goldwasser, and D. Ron. "Property Testing and Its Connection to Learning and Approximation". In: *Journal of the American Mathematical Society* 45.4 (1998). DOI: 10.1145/285055.285060.

[GT01] O. Goldreich and L. Trevisan. "Three Theorems Regarding Testing Graph Properties". In: *Proceedings of the IEEE International Conference on Cluster Computing*. 2001. DOI: 10.1109/SFCS.2001.959922 (cit. on p. 60). Journal version: O. Goldreich and L. Trevisan. "Three Theorems Regarding Testing Graph Properties". In: *Random Structures & Algorithms* 23.1 (2003). DOI: 10.1002/rsa.10078.

[Gol17] O. Goldreich. *Introduction to Property Testing*. Cambridge University Press, 2017. ISBN: 978-1-107-19405-2 (cit. on pp. 11, 13, 16).

[GR99] O. Goldreich and D. Ron. "A Sublinear Bipartiteness Tester for Bounded Degree Graphs". In: *Combinatorica* 19.3 (1999). DOI: 10.1007/s004930050060 (cit. on p. 87).

[GR06] O. Goldreich and D. Ron. "Approximating Average Parameters of Graphs". In: *Proceedings of the 11th / 12th International Workshops on Approximation Algorithms for Combinatorial Optimization Problems / Randomization and Computation (APPROX / RANDOM)*. Springer Berlin Heidelberg, 2006. DOI: 10.1007/11830924_34 (cit. on pp. 57, 64). Journal version: "Approximating Average Parameters of Graphs". In: *Proceedings of the 11th / 12th International Workshops on Approximation Algorithms for Combinatorial Optimization Problems / Randomization and Computation (APPROX / RANDOM)*. Springer Berlin Heidelberg, 2006. DOI: 10.1007/11830924_34.

[GR09] O. Goldreich and D. Ron. "On Proximity Oblivious Testing". In: *Proceedings of the 41st Annual ACM Symposium on Theory of Computing (STOC)*. Association for Computing Machinery, 2009. DOI: 10.1145/1536414.1536436 (cit. on pp. 37, 40, 44, 66). Journal version: O. Goldreich and D. Ron. "On Proximity-Oblivious Testing". In: *SIAM Journal on Computing* 40.2 (2011). DOI: 10.1137/100789646.

[GR00] O. Goldreich and D. Ron. "On Testing Expansion in Bounded-Degree Graphs". In: (2000) (cit. on pp. 104, 108). Journal version: "On Testing Expansion in Bounded-Degree Graphs". In: *Studies in Complexity and Cryptography. Miscellanea on the Interplay between Randomness and Computation*. Springer Berlin Heidelberg, 2011. DOI: 10.1007/978-3-642-22670-0_9.

[GR97] O. Goldreich and D. Ron. "Property Testing in Bounded Degree Graphs". In: *Proceedings of the 29th Annual ACM Symposium on Theory of Computing (STOC)*. Association for Computing Machinery, 1997. DOI: 10.1145/258533.258627 (cit. on pp. 18, 37, 40, 57). Journal version: "Property Testing in Bounded Degree Graphs". In: *Algorithmica* 32.2 (2002). DOI: 10.1007/s00453-001-0078-7.

[GR02] O. Goldreich and D. Ron. "Property Testing in Bounded Degree Graphs". In: *Algorithmica* 32.2 (2002). DOI: 10.1007/s00453-001-0078-7 (cit. on pp. 51, 85–87).

[GS12] O. Goldreich and I. Shinkar. "Two-Sided Error Proximity Oblivious Testing". In: *Proceedings of the 15th / 16th International Workshops on Approximation Algorithms for Combinatorial Optimization Problems / Randomization and Computation (APPROX / RANDOM)*. Springer Berlin Heidelberg, 2012. DOI: 10.1007/978-3-642-32512-0_48 (cit. on p. 40). Journal version: "Two-Sided Error Proximity Oblivious Testing". In: *Proceedings of the 15th / 16th International Workshops on Approximation Algorithms for Combinatorial Optimization Problems / Randomization and Computation (APPROX / RANDOM)*. Springer Berlin Heidelberg, 2012. DOI: 10.1007/978-3-642-32512-0_48.

[GT03] O. Goldreich and L. Trevisan. "Three Theorems Regarding Testing Graph Properties". In: *Random Structures & Algorithms* 23.1 (2003). DOI: 10.1002/rsa.10078 (cit. on pp. 39, 40, 67).

[Hal35] P. Hall. "On Representatives of Subsets". In: *Journal of the London Mathematical Society* s1-10.1 (1935). DOI: 10.1112/jlms/s1-10.37.26 (cit. on p. 90).

[Has+09] A. Hassidim, J. Kelner, H. Nguyen, and K. Onak. "Local Graph Partitions for Approximation and Testing". In: *Proceedings of the 50th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*. 2009. DOI: 10.1109/FOCS.2009.77 (cit. on pp. 40, 41, 86).

[HRR99] M. R. Henzinger, P. Raghavan, and S. Rajagopalan. "Computing on Data Streams". In: American Mathematical Society, 1999 (cit. on p. 56).

[HP19] M. Henzinger and P. Peng. "Constant-Time Dynamic $(\Delta+1)$-Coloring and Weight Approximation for Minimum Spanning Forest: Dynamic Algorithms Meet Property Testing". In: *Computing Research Repository* (2019). arXiv: 1907.04745 (cit. on p. 59).

[Hoe63] W. Hoeffding. "Probability Inequalities for Sums of Bounded Random Variables". In: *Journal of the American Statistical Association* 58.301 (1963). DOI: 10.1080/01621459.1963.10500830 (cit. on p. 19).

[HM16] L. Hosseini and P. O. de Mendez. "Treeable Graphings Are Local Limits of Finite Graphs". In: *Computing Research Repository* (2016). arXiv: 1601.05580 (cit. on p. 24).

[HP16] Z. Huang and P. Peng. "Dynamic Graph Stream Algorithms in o(n) Space". In: *Proceedings of the 43rd International Colloquium on Automata, Languages, and Programming (ICALP)*. Vol. 55. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2016. DOI: 10.4230/LIPIcs.ICALP.2016.18 (cit. on pp. 57, 59). Journal version: "Dynamic Graph Stream Algorithms in o(n) Space". In: *Algorithmica* 81.5 (2019). DOI: 10.1007/s00453-018-0520-8.

[Ito16] H. Ito. "Every Property Is Testable on a Natural Class of Scale-Free Multigraphs". In: *Proceedings of the 24th Annual European Symposium on Algorithms (ESA)*. Vol. 57. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2016. DOI: 10.4230/LIPIcs.ESA.2016.51 (cit. on p. 59).

[ITY12] H. Ito, S.-I. Tanigawa, and Y. Yoshida. "Constant-Time Algorithms for Sparsity Matroids". In: *Proceedings of the 43rd International Colloquium on Automata, Languages, and Programming (ICALP)*. Springer Berlin Heidelberg, 2012. DOI: 10.1007/978-3-642-31594-7_42 (cit. on p. 40).

[IY17] K. Iwama and Y. Yoshida. "Parameterized Testability". In: *ACM Trans. Comput. Theory* 9.4 (2017). DOI: 10.1145/3155294 (cit. on p. 40).

[Jen06] J. L. W. V. Jensen. "Sur les fonctions convexes et les inégalités entre les valeurs moyennes". In: *Acta Mathematica* 30 (1906). DOI: 10.1007/BF02418571 (cit. on p. 19).

[KPS08] S. Kale, Y. Peres, and C. Seshadhri. "Noise Tolerance of Expanders and Sublinear Expander Reconstruction". In: *Proceedings of the 49th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*. 2008. DOI: 10.1109/FOCS.2008.65 (cit. on p. 87). Journal version: "Noise Tolerance of Expanders and Sublinear Expansion Reconstruction". In: *SIAM Journal on Computing* 42.1 (2013). DOI: 10.1137/110837863.

[KS08] S. Kale and C. Seshadhri. "An Expansion Tester for Bounded Degree Graphs". In: *Proceedings of the 35th International Colloquium on Automata, Languages, and Programming (ICALP)*. Springer Berlin Heidelberg, 2008. DOI: 10.1007/978-3-540-70575-8_43 (cit. on pp. 104, 108, 109, 113, 115, 117). Journal version: S. Kale and C. Seshadhri. "An Expansion Tester for Bounded Degree Graphs". In: *SIAM Journal on Computing* 40.3 (2011). DOI: 10.1137/100802980.

[Kap13] M. Kapralov. "Better Bounds for Matchings in the Streaming Model". In: *Proceedings of the 24th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. Society for Industrial and Applied Mathematics, 2013. DOI: 10.1137/1.9781611973105.121 (cit. on p. 56).

[KKS14] M. Kapralov, S. Khanna, and M. Sudan. "Approximating Matching Size from Random Streams". In: *Proceedings of the 25th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. Society for Industrial and Applied Mathematics, 2014. DOI: 10.1137/1.9781611973402.55 (cit. on p. 57).

[KKR04] T. Kaufman, M. Krivelevich, and D. Ron. "Tight Bounds for Testing Bipartiteness in General Graphs". In: *SIAM Journal on Computing* 33.6 (2004). DOI: 10.1137/S0097539703436424 (cit. on p. 64).

[KKR12] K.-i. Kawarabayashi, Y. Kobayashi, and B. Reed. "The Disjoint Paths Problem in Quadratic Time". In: *Journal of Combinatorial Theory, Series B* 102.2 (2012). DOI: 10.1016/j.jctb.2011.07.004 (cit. on p. 87).

[KY13] K.-i. Kawarabayashi and Y. Yoshida. "Testing Subdivision-Freeness: Property Testing Meets Structural Graph Theory". In: *Proceedings of the 45th Annual ACM Symposium on Theory of Computing (STOC)*. Association for Computing Machinery, 2013. DOI: 10.1145/2488608.2488663 (cit. on p. 40).

[KS96] J. Komlós and M. Simonovits. *Szemerédi's Regularity Lemma and Its Applications in Graph Theory*. 1996 (cit. on p. 39).

[Kon15] C. Konrad. "Maximum Matching in Turnstile Streams". In: *Proceedings of the 23rd Annual European Symposium on Algorithms (ESA)*. Springer Berlin Heidelberg, 2015. DOI: 10.1007/978-3-662-48350-3_70 (cit. on p. 56).

[KMM12] C. Konrad, F. Magniez, and C. Mathieu. "Maximum Matching in Semi-Streaming with Few Passes". In: *Proceedings of the 15th / 16th International Workshops on Approximation Algorithms for Combinatorial Optimization Problems / Randomization and Computation (APPROX / RANDOM)*. Springer Berlin Heidelberg, 2012 (cit. on p. 57).

[KSS18] A. Kumar, C. Seshadhri, and A. Stolman. "Finding Forbidden Minors in Sublinear Time: A N1̂/2+o(1)-Query One-Sided Tester for Minor Closed Properties on Bounded Degree Graphs". In: *Proceedings of the 59th Annual Symposium on Foundations of Computer Science (FOCS)*. 2018. DOI: 10.1109/FOCS.2018.00055 (cit. on pp. 41, 85–88).

[KSS19] A. Kumar, C. Seshadhri, and A. Stolman. "Random Walks and Forbidden Minors II: A Poly(D ϵ-1)-Query Tester for Minor-Closed Properties of Bounded Degree Graphs". In: *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing (STOC)*. ACM, 2019. DOI: 10.1145/3313276.3316330 (cit. on pp. 41, 86).

[Kur30] C. Kuratowski. "Sur le problème des courbes gauches en Topologie". In: *Fundamenta Mathematicae* 15.1 (1930) (cit. on pp. 14, 85).

[Łąc+19] J. Łącki, S. Mitrović, K. Onak, and P. Sankowski. "Walking Randomly, Massively, and Efficiently". In: *Computing Research Repository* (2019). arXiv: 1907.05391 (cit. on p. 59).

[LL18] C. Lenzen and R. Levi. "A Centralized Local Algorithm for the Sparse Spanning Graph Problem". In: *Proceedings of the 45th International Colloquium on Automata, Languages, and Programming (ICALP)*. Vol. 107. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2018. DOI: 10.4230/LIPIcs.ICALP.2018.87 (cit. on pp. 59, 88, 89, 94, 95, 98).

[LMR18] R. Levi, M. Medina, and D. Ron. "Property Testing of Planarity in the CONGEST Model". In: *Proceedings of the 37th ACM Symposium on Principles of Distributed Computing (PODC)*. ACM, 2018. DOI: 10.1145/3212734.3212748 (cit. on p. 105).

[Lev+17] R. Levi, G. Moshkovitz, D. Ron, R. Rubinfeld, and A. Shapira. "Constructing near Spanning Trees with Few Local Inspections". In: *Random Structures & Algorithms* 50.2 (2017). DOI: 10.1002/rsa.20652 (cit. on p. 59).

[LR15] R. Levi and D. Ron. "A Quasi-Polynomial Time Partition Oracle for Graphs with an Excluded Minor". In: *ACM Trans. Algorithms* 11.3 (2015). DOI: 10.1145/2629508 (cit. on pp. 86, 88).

[LRR16] R. Levi, D. Ron, and R. Rubinfeld. "A Local Algorithm for Constructing Spanners in Minor-Free Graphs". In: *Proceedings of the 19th / 20th International Workshops on Approximation Algorithms for Combinatorial Optimization Problems / Randomization and Computation (APPROX / RANDOM)*. Vol. 60. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2016. DOI: 10.4230/LIPIcs.APPROX-RANDOM.2016.38 (cit. on p. 59).

[LRR14] R. Levi, D. Ron, and R. Rubinfeld. “Local Algorithms for Sparse Spanning Graphs”. In: *Proceedings of the 17th / 18th International Workshops on Approximation Algorithms for Combinatorial Optimization Problems / Randomization and Computation (APPROX / RANDOM)*. Vol. 28. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2014. DOI: 10.4230/LIPIcs.APPROX-RANDOM.2014.826 (cit. on p. 59).

[LRY17] R. Levi, R. Rubinfeld, and A. Yodpinyanee. “Local Computation Algorithms for Graphs of Non-Constant Degrees”. In: *Algorithmica* 77.4 (2017). DOI: 10.1007/s00453-016-0126-y (cit. on p. 59).

[LPW09] D. A. Levin, Y. Peres, and E. L. Wilmer. *Markov Chains and Mixing Times*. American Mathematical Society, 2009. ISBN: 978-0-8218-4739-8 (cit. on p. 108).

[LPP11] A. Li, Y. Pan, and P. Peng. “Testing Conductance in General Graphs”. In: *Electronic Colloquium on Computational Complexity (ECCC)* (2011) (cit. on p. 104).

[LP15] A. Li and P. Peng. “Testing Small Set Expansion in General Graphs”. In: *Proceedings of the 32nd International Symposium on Theoretical Aspects of Computer Science (STACS)*. Vol. 30. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2015. DOI: http://dx.doi.org/10.4230/LIPIcs.STACS.2015.622 (cit. on pp. 104, 114).

[Lin92] N. Linial. “Locality in Distributed Graph Algorithms”. In: *SIAM Journal on Computing* 21.1 (1992). DOI: 10.1137/0221015 (cit. on p. 120).

[LT79] R. Lipton and R. Tarjan. “A Separator Theorem for Planar Graphs”. In: *SIAM Journal on Applied Mathematics* 36.2 (1979). DOI: 10.1137/0136016 (cit. on p. 14).

[LS90] L. Lovasz and M. Simonovits. “The Mixing Rate of Markov Chains, an Isoperimetric Inequality, and Computing the Volume”. In: *Proceedings of the 31st Annual Symposium on Foundations of Computer Science (STOC)*. 1990. DOI: 10.1109/FSCS.1990.89553 (cit. on p. 87).

[Lov12] L. Lovász. *Large Networks and Graph Limits*. American Mathematical Society, 2012. ISBN: 978-0-8218-9085-1 (cit. on pp. 22, 24, 39).

[LPS88] A. Lubotzky, R. Phillips, and P. Sarnak. “Ramanujan Graphs”. In: *Combinatorica* 8.3 (1988). DOI: 10.1007/BF02126799 (cit. on p. 119).

[Man+12] Y. Mansour, A. Rubinstein, S. Vardi, and N. Xie. “Converting Online Algorithms to Local Computation Algorithms”. In: *Proceedings of the 39th International Colloquium on Automata, Languages, and Programming (ICALP)*. Springer Berlin Heidelberg, 2012. DOI: 10.1007/978-3-642-31594-7_55 (cit. on p. 59).

[MV13] Y. Mansour and S. Vardi. “A Local Computation Approximation Scheme to Maximum Matching”. In: *Proceedings of the 16th / 17th International Workshops on Approximation Algorithms for Combinatorial Optimization Problems / Randomization and Computation (APPROX / RANDOM)*. Springer Berlin Heidelberg, 2013. DOI: 10.1007/978-3-642-40328-6_19 (cit. on p. 59).

[McG14] A. McGregor. “Graph Stream Algorithms: A Survey”. In: *SIGMOD Rec.* 43.1 (2014). DOI: 10.1145/2627692.2627694 (cit. on p. 57).

[MV16] A. McGregor and S. Vorotnikova. "Planar Matching in Streams Revisited". In: *Proceedings of the 19th / 20th International Workshops on Approximation Algorithms for Combinatorial Optimization Problems / Randomization and Computation (APPROX / RANDOM)*. Vol. 60. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2016. DOI: 10.4230/LIPIcs.APPROX-RANDOM.2016.17 (cit. on p. 56).

[MPX13] G. L. Miller, R. Peng, and S. C. Xu. "Parallel Graph Decompositions Using Random Shifts". In: *Proceedings of the 25th Annual ACM Symposium on Parallelism in Algorithms and Architectures (SPAA)*. Association for Computing Machinery, 2013. DOI: 10.1145/2486159.2486180 (cit. on p. 90).

[MU17] M. Mitzenmacher and E. Upfal. *Probability and Computing: Randomization and Probabilistic Techniques in Algorithms and Data Analysis*. Cambridge University Press, 2017. ISBN: 978-1-107-15488-9 (cit. on pp. 13, 19).

[MP17] A. R. Molla and G. Pandurangan. "Distributed Computation of Mixing Time". In: *Proceedings of the 18th International Conference on Distributed Computing and Networking (ICDCN)*. Association for Computing Machinery, 2017. DOI: 10.1145/3007748.3007784 (cit. on p. 105).

[Mon+17] M. Monemizadeh, S. Muthukrishnan, P. Peng, and C. Sohler. "Testable Bounded Degree Graph Properties Are Random Order Streamable". In: *Proceedings of the 44th International Colloquium on Automata, Languages, and Programming (ICALP)*. Vol. 80. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2017. DOI: 10.4230/LIPIcs.ICALP.2017.131 (cit. on pp. 57, 59, 61).

[MR95] R. Motwani and P. Raghavan. *Randomized Algorithms*. Cambridge University Press, 1995. ISBN: 978-0-521-47465-8 (cit. on pp. 13, 17).

[NS10] A. Nachmias and A. Shapira. "Testing the Expansion of a Graph". In: *Information and Computation* 208.4 (2010). DOI: 10.1016/j.ic.2009.09.002 (cit. on p. 104).

[NS11] I. Newman and C. Sohler. "Every Property of Hyperfinite Graphs Is Testable". In: *Proceedings of the 43rd Annual ACM Symposium on Theory of Computing (STOC)*. Association for Computing Machinery, 2011. DOI: 10.1145/1993636.1993726 (cit. on pp. 37, 40, 41, 59). Journal version: I. Newman and C. Sohler. "Every Property of Hyperfinite Graphs Is Testable". In: *SIAM Journal on Computing* 42.3 (2013). DOI: 10.1137/120890946.

[PR99] M. Parnas and D. Ron. "Testing the Diameter of Graphs". In: *Proceedings of the 2nd / 3rd International Workshops on Approximation Algorithms for Combinatorial Optimization Problems / Randomization and Computation (APPROX / RANDOM)*. Springer Berlin Heidelberg, 1999. DOI: 10.1007/978-3-540-48413-4_9 (cit. on pp. 18, 57). Journal version: "Testing the Diameter of Graphs". In: *Random Structures & Algorithms* 20.2 (2002). DOI: 10.1002/rsa.10013.

[PS18] P. Peng and C. Sohler. "Estimating Graph Parameters from Random Order Streams". In: *Proceedings of the 29th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. Society for Industrial and Applied Mathematics, 2018. DOI: 10.1137/1.9781611975031.157 (cit. on pp. 57, 59, 61).

[RS06] S. Raskhodnikova and A. Smith. “A Note on Adaptivity in Testing Properties of Bounded Degree Graphs”. In: *Electronic Colloquium on Computational Complexity (ECCC)* (2006) (cit. on p. 60).

[RST94] N. Robertson, P. Seymour, and R. Thomas. “Quickly Excluding a Planar Graph”. In: *Journal of Combinatorial Theory, Series B* 62.2 (1994). DOI: 10.1006/jctb.1994.1073 (cit. on p. 99).

[RS04] N. Robertson and P. D. Seymour. “Graph Minors. XX. Wagner’s Conjecture”. In: *Journal of Combinatorial Theory, Series B* 92.2 (2004). DOI: 10.1016/j.jctb.2004.08.001 (cit. on p. 41).

[RS96] R. Rubinfeld and M. Sudan. “Robust Characterizations of Polynomials with Applications to Program Testing”. In: *SIAM Journal on Computing* 25.2 (1996). DOI: 10.1137/S0097539793255151 (cit. on p. 11).

[RS92] R. Rubinfeld and M. Sudan. “Self-Testing Polynomial Functions Efficiently and over Rational Domains”. In: *Proceedings of the 3rd Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. Society for Industrial and Applied Mathematics, 1992 (cit. on p. 11).

[Rub+11] R. Rubinfeld, G. Tamir, S. Vardi, and N. Xie. “Fast Local Computation Algorithms”. In: *Computing Research Repository* (2011). arXiv: 1104.1377 (cit. on p. 59).

[Sch11] O. Schramm. “Hyperfinite Graph Limits”. In: *Selected Works of Oded Schramm*. Springer New York, 2011. DOI: 10.1007/978-1-4419-9675-6_19 (cit. on p. 41).

[Sin93] A. Sinclair. *Algorithms for Random Generation and Counting: A Markov Chain Approach*. Birkhauser Verlag, 1993. ISBN: 978-0-8176-3658-6 (cit. on p. 108).

[ST13] D. Spielman and S. Teng. “A Local Clustering Algorithm for Massive Graphs and Its Application to Nearly Linear Time Graph Partitioning”. In: *SIAM Journal on Computing* 42.1 (2013). DOI: 10.1137/080744888 (cit. on p. 87).

[Ste09] E. Steinitz. “Bedingt Konvergente Reihen Und Konvexe Systeme”. In: *Journal für die reine und angewandte Mathematik (Crelle’s Journal)* 1913.143 (2009). DOI: 10.1515/crll.1913.143.128 (cit. on p. 24).

[SW15] X. Sun and D. P. Woodruff. “Tight Bounds for Graph Problems in Insertion Streams”. In: *Proceedings of the 18th / 19th International Workshops on Approximation Algorithms for Combinatorial Optimization Problems / Randomization and Computation (APPROX / RANDOM)*. Vol. 40. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2015. DOI: 10.4230/LIPIcs.APPROX-RANDOM.2015.435 (cit. on p. 57).

[TY15] S.-I. Tanigawa and Y. Yoshida. “Testing the Supermodular-Cut Condition”. In: *Algorithmica* 71.4 (2015). DOI: 10.1007/s00453-013-9842-8 (cit. on p. 40).

[TU 17a] TU Dortmund University, ed. *Regeln Guter Wissenschaftlicher Praxis an Der TU Dortmund*. 2017 (cit. on p. 123).

[TU 17b] TU Dortmund University, ed. *Rules of Good Scientific Practice at TU Dortmund University*. 2017 (cit. on p. 123).

[Wag37] K. Wagner. “Über eine Eigenschaft der ebenen Komplexe”. In: *Mathematische Annalen* 114.1 (1937). DOI: 10.1007/BF01594196 (cit. on p. 14).

[YI08] Y. Yoshida and H. Ito. "Property Testing on K-Vertex-Connectivity of Graphs". In: *Proceedings of the 35th International Colloquium on Automata, Languages, and Programming (ICALP)*. Springer Berlin Heidelberg, 2008 (cit. on p. 40). Journal version: "Property Testing on K-Vertex-Connectivity of Graphs". In: *Algorithmica* 62.3 (2012). DOI: 10.1007/s00453-010-9477-y.

[YI10] Y. Yoshida and H. Ito. "Testing Outerplanarity of Bounded Degree Graphs". In: *Proceedings of the 13th / 14th International Workshops on Approximation Algorithms for Combinatorial Optimization Problems / Randomization and Computation (APPROX / RANDOM)*. Springer Berlin Heidelberg, 2010 (cit. on p. 86). Journal version: "Testing Outerplanarity of Bounded Degree Graphs". In: *Algorithmica* 73.1 (2015). DOI: 10.1007/s00453-014-9897-1.

[YK12] Y. Yoshida and Y. Kobayashi. "Testing the (s,t)-Disconnectivity of Graphs and Digraphs". In: *Theoretical Computer Science* 434 (2012). DOI: 10.1016/j.tcs.2012.01.045 (cit. on p. 40).

[YYI09] Y. Yoshida, M. Yamamoto, and H. Ito. "An Improved Constant-Time Approximation Algorithm for Maximum Matchings". In: *Proceedings of the 41st Annual ACM Symposium on Theory of Computing (STOC)*. Association for Computing Machinery, 2009. DOI: 10.1145/1536414.1536447 (cit. on p. 40).