Testing C_k-freeness in Bounded Admissibility Graphs

Christine Awofeso
Birkbeck, University of London, UK

Patrick Greaves
Birkbeck, University of London, UK

Oded Lachish
Birkbeck, University of London, UK

Amit Levi
University of Haifa, Haifa, Israel

Felix Reidl
Birkbeck, University of London, UK

Abstract

We study C_k-freeness in sparse graphs from a property testing perspective, specifically for graph classes with bounded r-admissibility. Our work is motivated by the large gap between upper and lower bounds in this area: C_k-freeness is known to be testable in planar graphs [4], but not in graphs with bounded arboricity for $k > 3$ [7]. There are a large number of interesting graph classes that include planar graphs and have bounded arboricity (e.g. classes excluding a minor), calling for a more fine-grained approach to the question of testing C_k-freeness in sparse graph classes.

One such approach, inspired by the work of Nesetril and Ossona de Mendez [11], is to consider the graph measure of r-admissibility, which naturally forms a hierarchy of graph families $\mathcal{A}_1 \supset \mathcal{A}_2 \supset \ldots \supset \mathcal{A}_\infty$ where $\mathcal{A}_r$ contains all graph classes whose r-admissibility is bounded by some constant. The family $\mathcal{A}_1$ contains classes with bounded arboricity, the class $\mathcal{A}_\infty$ contains classes like planar graphs, graphs of bounded degree, and minor-free graphs. Awofeso *et al.* [3] recently made progress in this direction. They showed that C_4- and C_5-freeness is testable in $\mathcal{A}_2$. They further showed that C_k-freeness is *not* testable in $\mathcal{A}_{\lfloor k/2 \rfloor - 1}$ and conjectured that C_k-freeness is testable in $\mathcal{A}_{\lfloor k/2 \rfloor}$. In this work, we prove this conjecture: C_k-freeness is indeed testable in graphs of bounded $\lfloor k/2 \rfloor$-admissibility.

2012 ACM Subject Classification Author: Please fill in 1 or more \ccsdesc macro

Keywords and phrases Property Testing, Sparse Graphs, Cycle, Admissibility

Digital Object Identifier 10.4230/LIPIcs.ICALP.2025.104

1 Introduction

Finding a k-cycle (C_k) as a subgraph is a fundamental problem in graph theory with applications in network analysis, bioinformatics, and theoretical computer science. Given a graph $G = (V, E)$, the goal is to detect cycles of exactly k vertices. Various algorithmic approaches have been proposed, including combinatorial search techniques [9], matrix-based methods using spectral graph theory [2], and more recent advancements leveraging graph sparsification and algebraic techniques [12, 5].

In this paper, the focus is on a variation of this problem, where access to the input algorithm is through an oracle that answers queries of the sort: "What is the degree of vertex v?", "What is the i-th neighbour of the vertex v?", and "are the vertices u and v neighbours?". The goal in this variation is to determine whether a graph is C_k-free (it does not have a subgraph isomorphic to C_k), given the size of the input graph and oracle access to the graph, using the minimal possible number of queries.

A deterministic algorithm of this type, for determining whether a graph is C_k-free, may require a number of queries that is at least linear in the size of the graph. Therefore, to reduce the number of queries, an alternative approach is to require the algorithm only to distinguish between graphs that are C_k-free and those that are *far* from being C_k-free, which means that they require the removal of many edges to become C_k-free. The algorithm is also allowed to be probabilistic and is only required to produce the correct answer with high probability. Observe that if the algorithm is required to be error-free when the input graph is C_k-free (as is the case with the algorithm presented here), it must explicitly identify a C_k subgraph to conclude that the graph is *not* C_k-free, essentially solving the C_k detection problem for inputs that are far from being C_k-free.

This relaxation of requirements is captured by the framework of *property testing*. In this framework, the problem described above is referred to as *testing C_k-freeness*. A central goal is to show that the number of queries required for testing C_k-freeness, when the input is restricted to graphs from a specific family, is independent of the size of the input graph, yet may depend on the parameters of the class and the distance parameter (which is a fixed parameter provided with the input) that governs the 'farness' of the input from C_k-freeness. If such an algorithm exists for some family of graphs, then we say that C_k-freeness is *testable* for this family.

The results in this work focus on testing C_k-freeness for sparse graph families, specifically those with a bounded average degree. This line of research was initiated in the seminal work of Goldriech and Ron [8] who showed that the query complexity of testing C_k-freeness in graphs with maximum degree d is $O(d^k)$. Although that result was promising, Alon *et al.* [1] later showed that the query complexity testing triangle-freeness (C_3-freeness) in graphs with constant average degree is $\Theta(\sqrt{n})$, where n is the number of vertices in the input graph.

This raised the question of whether C_k-freeness has a better query complexity in families of graphs that are strict subsets of graphs with a bounded average degree. Czumaj et al. [4] showed that the more general property of H-freeness (i.e. graphs that don't have a subgraph isomorphic to H) is testable in planar graphs, which implies that C_k-freeness is also testable in this setting. Subsequently, Levi [10] showed that the special case of triangle-freeness is testable for graphs with bounded *arboricity*, a superset of planar graphs. However, Eden et al. [7] recently showed that this does not extend to larger cycles: testing C_4-freeness and C_5-freeness has query complexity $\Omega(n^{1/4})$ in graphs of bounded arboricity, and testing C_k-freeness for $k \geq 6$ has query complexity $\Omega(n^{1/3})$. This left open the question of which (strict) subsets of classes with bounded arboricity still allow testing of C_k-freeness.

Awofeso et al. [3] recently presented results in this direction. They considered a hierarchy of sparse graph classes defined by the r-admissibility measure, which was first introduced by Dvořák [6] and plays an important role in the study of sparse classes [11, 14]. We postpone the rather technical definition to the next section; for now let us note that graphs of bounded 1-admissibility are equivalent to graphs of bounded arboricity, and graphs whose r-admissibility is bounded for *any* integer r include many well-known sparse classes such as planar graphs, graphs with bounded genus, graphs of bounded degree and graphs excluding a (topological) minor. Consequently, if we define $\mathcal{A}_r$ as the family of all graph classes whose r-admissibility is bounded by some constant, then we have an infinite hierarchy of classes $\mathcal{A}_1 \supset \mathcal{A}_2 \supset \ldots \supset \mathcal{A}_\infty$ between graphs of bounded arboricity ($\mathcal{A}_1$) and classes whose r-admissibility is bounded for any value of r ($\mathcal{A}_\infty$, also known as *bounded expansion* classes [11]). Note that all these inclusions are proper, for example, the class $\mathcal{K}^{(r)}$ consisting of all cliques with every edge is replaced with a length r path, is contained in $\mathcal{A}_r$ but not in $\mathcal{A}_{r+1}$.

In this hierarchy, Awofeso *et al.* [3] showed that testing C_4- and C_5-freeness is possible in $\mathcal{A}_2$, the graphs with bounded 2-admissibility. They supplement this with a lower bound, which shows that for all $r \geq 4$ the number of queries required for C_{2r+1}-freeness testing and C_{2r}-freeness testing of classes in $\mathcal{A}_{r-1}$ is polynomial in the size of the input graph. The two results lead them to conjecture that C_{2r}- and C_{2r+1}-freeness should be testable for classes in $\mathcal{A}_r$, i.e. classes with bounded r-admissibility.

In this work, we complete the picture and prove the following.

▶ **Theorem** (Informal). *For any integer $r > 1$, there exists a testing algorithm for C_{2r}-freeness and C_{2r+1}-freeness of bounded r-admissible graphs with query complexity depending only on r,p and the proximity parameter ϵ, where p is the bound on the r-admissibility bound.*

We prove the theorem for C_{2r+1}-freeness, but a minor (and slightly simpler) version of our proof implies the same result C_{2r}-freeness. We also provide the lower bounds that do not appear in [3]. Specifically, that for every $r \geq 2$, C_{2r}-freeness and C_{2r+1}-freeness are not testable in bounded $(r-1)$-admissible graphs.

Techniques and their relation to existing work

The results in this paper are derived from the classical property tester for C_k-freeness in bounded-degree graphs. The tester selects a small initial random subset of the vertices of a graph that is far from being C_k-free; then, with sufficiently high probability, one of these vertices participates in a C_k subgraph of the input graph (a subgraph of the input graph that is isomorphic to C_k). Since the input graph has a bounded degree, starting a "bounded range" breadth-first search (BFS) from each one of the vertices in the initial set of vertices will result in the discovery of a C_k subgraph with sufficient probability. If this event occurs, then the graph is rejected and, otherwise, it is accepted. Clearly, this algorithm will always accept a C_k-free graph.

Suppose that we tried to use a variation of this tester on a graph with a bounded average degree, where the latter bound is provided as part of the input. We call this variation *pseudo-BFS* (PBFS). To guarantee that the PBFS's query complexity remains low, the algorithm first computes some γ that depends only on the given average degree and then behaves like the bounded-degree tester until it encounters a vertex with a degree greater than γ (a *heavy vertex*). When this occurs, the search only includes a size γ randomly selected subset of the vertex's neighbours. If the PBFS initiates a search on a vertex that is part of a C_k *and* this C_k contains at most one heavy vertex, then the PBFS will uncover all vertices of said C_k. However, if a C_k contains at least two non-neighbouring heavy vertices, then the PBFS might not find all the vertices of this C_k copy with sufficiently high probability since it only sees a small random subset of a heavy vertex's neighbours. The lower bound result is based on this scenario.

This raised the question of whether the PBFS approach still works if we impose more restrictions than just the bounded average degree. In [3] it was shown that C_4-freeness is testable in $\mathcal{A}_2$ using PBFS as the testing algorithm. The technique involves showing that if the input graph has bounded 2-admissibility and is far from being C_4-free, then with sufficiently high probability the PBFS will start its search in a vertex that belongs to a C_4 with at most two heavy vertices and the crucial insight is that these two heavy vertices necessarily have a large joint neighbourhood of non-heavy vertices—meaning that if the PBFS locates one of these joint neighbours, it will immediately find both heavy vertices in the next step and two steps later a C_4 will be detected with high probability. This technique is sufficient to establish testability of C_7-freeness in 3-admissible graphs, but not for C_8-freeness in 4-admissible graphs. The reason is that a C_8 might contain two heavy vertices of distance exactly four, which means that these paths include vertices that are not neighbours of either of the heavy endpoints, a case that is not covered by this technique.

We model our approach on [3] by using a process called *trimming*. That is, given an input graph G that is far from being C_k-free, we carve out a subgraph $\hat{G}$ of G by removing just enough edges from G to ensure that $\hat{G}$ has some required structural properties and is still far from being C_k-free (with some changes to the "farness" parameter). The graph $\hat{G}$ is only used for the analysis of our algorithm. Specifically, it is shown that if $\hat{G}$ has a C_k subgraph and one of its vertices is detected when our algorithm runs on G, then with high probability, a C_k-subgraph will be detected.

The main tools used here are various structural properties of graphs with bounded admissibility. For example, a graph G with bounded r-admissibility admits a total ordering $<$ of its vertex set $V(G)$ such that every heavy vertex $v \in G$ can only reach a bounded number of heavy vertices $u < v$ via paths of length at most r. This allows us to enforce that in the trimmed graph $\hat{G}$, if such a path of length at most r between u and v exists, then there must be a large number of such paths in $\hat{G}$ and therefore also in G (since otherwise they would have been removed from $\hat{G}$).

2 Preliminaries

$\mathbb{N}$,$\mathbb{R}$,$[k]$

We use $\mathbb{N}$ for the set of integer numbers, $\mathbb{R}$ for the set of real numbers, $\mathbb{R}^+$ for the set of positive reals. For an integer k, we use $[k]$ as a shorthand for the set $\{1, 2, \dots, k\}$. For integers k_1 and k_2 such that $k_1 < k_2$ we use $[k_1, k_2]$ as a shorthand for the set $\{k_1, k_1 + 1 \dots, k\}$ and $(k_1, k_2]$ as a shorthand for the set $\{k_1 + 1, k_1 + 2, \dots, k\}$. For a graph G, we use $V(G)$ and $E(G)$ to refer to its vertex and edge set, respectively. All graphs considered in this work are simple undirected graphs. The degree of a vertex $v \in V(G)$, denoted $\deg_G(v)$, is the number of edges incident on v. $N(v)$ is the set of neighbours of the vertex v. The distance between two vertices $u, v \in V(G)$ is denoted $\mathrm{dist}_G(u, v)$. The distance between a vertex $u \in V(G)$ and a subset $V' \subseteq V(G)$, is the minimum of $\mathrm{dist}_G(u, v)$ over every $v \in V'$.

xPy

For sequences of vertices $x_1x_2 \dots x_\ell$ (and in particular, paths), we use the shorthand x_1Px_ℓ to represent the sequence, and use $\mathrm{length}(x_1Px_\ell)$ to denote the length of the corresponding path. In addition, we use Px_ℓ (or x_1P) to refer to a subsequence of x_1Px_ℓ. All paths considered in this work are simple paths. Though paths here are undirected, we often treat them as directed by specifying a start and end vertex.

Property testing

graph property, far, close

A *graph property* (or simply *property*) $\mathcal{P}$ is a class of graphs closed under isomorphism. We say that a graph G *has the property* $\mathcal{P}$ if $G \in \mathcal{P}$.

We say that a graph G with n vertices and **at most** m edges is *ϵ-far* from $\mathcal{P}$ if at least ϵm edge modifications are required to make the graph have the property. Otherwise, we say that it is *ϵ-close* to $\mathcal{P}$.

Property tester

▶ **Definition 1.** *A* property tester *for a property $\mathcal{P}$ of graphs is a randomized algorithm $\mathcal{T}$ that receives as input parameters $n, m \in \mathbb{N}$, $\epsilon > 0$ and oracle access to a graph G with n vertices and* **at most** *m edges. If the graph G is ϵ-far from $\mathcal{P}$, then $\mathcal{T}$ rejects with probability at least $2/3$. If the graph $G \in \mathcal{P}$, then $\mathcal{T}$ accepts with probability 1 (if the tester has* one-sided *error) or with probability at least $2/3$ (if the tester has* two-sided *error).*

In this work, we consider the setting of sparse graphs where the oracle access to G is defined as follows.

▶ **Definition 2** (Sparse graph oracle). *Given a graph G, a* sparse graph oracle *can answer the following queries for vertices $u, v \in V(G)$:*

- *The degree $\deg_G(v)$ of a vertex v (degree query).*
- *The ith neighbour of v in G (neighbour query).*
- *Whether $\{u, v\}$ is an edge in G (adjacency query).*

By combining these queries, the whole neighbourhood of a vertex v can be revealed by using $1+\deg_G(v)$ queries. The oracle returns the special symbol '$\perp$' when asked a query without sensible answers, e.g. when asked to return the i-th neighbour of a vertex v with $\deg_G(v) < i$.

In this paper, we consider a particular set of graphs (which will be defined later on) for the property of C_k-freeness. That is, a graph that does not have a subgraph isomorphic to C_k. We refer to the problem as *C_k-freeness.* We call a graph C_k-free if it is contained in the C_k-freeness property.

C_k-freeness

In further sections, we use the term *knowledge-graph* to refer to the graph that includes all the vertices and edges revealed by the algorithm queries.

Knowledge-graph

3 Graph admissibility

We start this section by presenting the definitions required for defining when a graph is (p, r)-admissible. Afterwards, we provide the actual definition and a structural lemma for (p, r)-admissible graphs. Then, we explain why this lemma is critical and provide an extra definition and two more structural claims.

An *ordered graph* is a pair $\mathbb{G} = (G, \leq)$ where G is a graph and $\leq$ is a total order relation on $V(G)$. We write $\leq_{\mathbb{G}}$ to denote the ordering of $\mathbb{G}$ and extend this notation to derive the relations $<_{\mathbb{G}}$, $>_{\mathbb{G}}$, $\geq_{\mathbb{G}}$.

$\mathbb{G}$, ordered graph

To define *r-admissibility* we need the following ideas and notations.

▶ **Definition 3.** *Let $\mathbb{G} = (G, \leq)$ and $v \in V(G)$. A path vPx is r-*admissible *in $\mathbb{G}$ if length$(vPx) \leq r$, $x <_{\mathbb{G}} v$ and $\min_{w \in P} w >_{\mathbb{G}} v$ (where the minimum is with respect to the ordering in $\mathbb{G}$). That is, the path goes from v to x using only vertices w such that $w >_{\mathbb{G}} v$ and x satisfy $v >_{\mathbb{G}} x$.*

r-admissible path

▶ **Definition 4.** *For every integer $i > 0$ we let $\mathrm{Target}^i_{\mathbb{G}}(v)$ be the set of all vertices in $u \in V(G)$ such that $u <_{\mathbb{G}} v$ and u is reachable from v via an r-*admissible *path vPu of length at most i. We omit the subscript $\mathbb{G}$ when it is clear from context.*

Target

▶ **Definition 5.** *An r-*admissible path packing *is a collection of paths $\{vP_ix_i\}_i$ with joint* root *v and the additional properties that every path vP_ix_i is r-admissible and the subpaths P_ix_i are all pairwise vertex-disjoint. In particular, all endpoints $\{x_i\}_i$ are distinct. We write $\mathrm{pp}^r_{\mathbb{G}}(v)$ to denote the maximum size of any r-admissible path packing rooted at v.*

$(r, \mathbb{G})$-admissible path packing
$\mathrm{pp}^r_{\mathbb{G}}$

Examples of 2- and 3-admissible path packings are depicted in Figure 1.

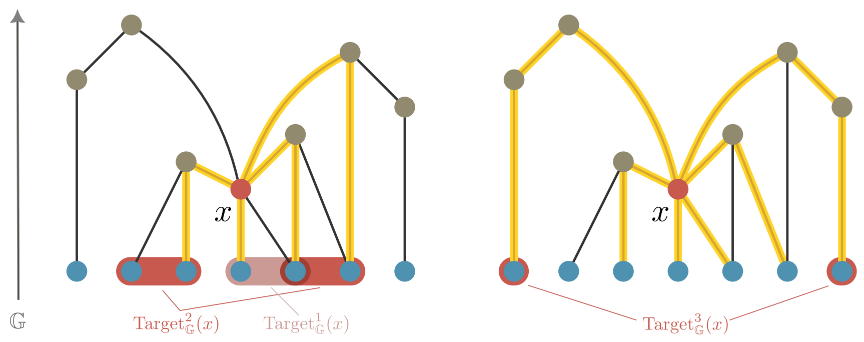
▶ **Definition 6** (Admissibility). *The r-*admissibility *of an ordered graph $\mathbb{G}$, denoted $\mathrm{adm}_r(\mathbb{G})$ and the admissibility of an unordered graph G, denoted $\mathrm{adm}_r(G)$ are*[1]

$\mathrm{adm}_r(\mathbb{G})$, $\mathrm{adm}_r(G)$

$$\mathrm{adm}_r(\mathbb{G}) := \max_{v \in \mathbb{G}} \mathrm{pp}^r_{\mathbb{G}}(v) \quad \textit{and} \quad \mathrm{adm}_r(G) := \min_{\mathbb{G} \in \pi(G)} \mathrm{adm}_r(\mathbb{G}),$$

where $\pi(G)$ is the set of all possible orderings of G.

[1] Note that some authors choose to define the admissibility as $1 + \max_{v \in \mathbb{G}} \mathrm{pp}^r_{\mathbb{G}}(v)$ as this matches with some other, related measures.



x
x
$\mathbb{G}$
$\mathrm{Target}^2_{\mathbb{G}}(x)$
$\mathrm{Target}^1_{\mathbb{G}}(x)$
$\mathrm{Target}^3_{\mathbb{G}}(x)$

Figure 1 An example of a maximum 2-admissible (left) and 3-admissible path packing for a vertex x in an ordered graph $\mathbb{G}$. The order is depicted by the height of the vertices, with the exception of the blue vertices who appear before x in $\mathbb{G}$ and whose relative ordering is not important here.

Admissibility ordering If $\mathbb{G}$ is an ordering of G such that $\mathrm{adm}_r(\mathbb{G}) = \mathrm{adm}_r(G)$, then we call $\mathbb{G}$ an *admissibility ordering* of G. The 1-admissibility of a graph coincides with its degeneracy, and therefore such orderings are easily computable in linear time. For $r \geq 2$, an optimal ordering can also be computed in linear time in n if the class has *bounded expansion*, i.e. if the graph class has bounded admissibility for *every* r (see [6]).

Now we are ready to define the set of graphs we consider for testing C_{2r+1}-freeness.

(p,r)-admissible, adm_r-bounded ▶ **Definition 7.** *We say that a graph G is* (p,r)-admissible *if* $\mathrm{adm}_r(G) = p$. *Note that by definition, if a graph G is (p,r)-admissible, it is also (p,r')-admissible for all $r' \leq r$. We call a graph class* adm_r-bounded *if all of its members are (p,r)-admissible for some finite value p.*

We state the following fact regarding (p,r)-admissible graphs.

▶ **Fact 8.** *If G is (p,r)-admissible, then $|E(G)| \leq p \cdot |V(G)|$.*

The next lemma is a well-known structural result for admissible graphs. This lemma is the critical component of the result of this paper.

▶ **Lemma 9.** *Let $\mathbb{G} = (G, \leq)$ be such that $\mathrm{adm}_r(G) = p$, then for every $v \in V(G)$, and $h \in [r]$ we have $|\mathrm{Target}^h_{\mathbb{G}}(v)| \leq p(p-1)^{h-1}$.*

We refer the reader to [3] (Proposition 3) for a proof.

Lemma 9, is the structural property of bounded admissibility graphs that enables the result in this paper. To understand the importance of the above lemma, fix $\epsilon > 0$ and suppose that we have a $(2,2)$ admissible graph G with ordering $\mathbb{G} = (G, \leq)$. Suppose also that G is ϵ-far from being C_4-free and all the C_4-subgraphs in it have exactly two vertices of degree 2 and 2 vertices of degree $16n^{1/2}/\epsilon$. Let Q be the set of degree $16n^{1/2}/\epsilon$ vertices and let W be the set of degree 2 vertices. Consider any ordering $\leq$ placing the vertices in Q before the vertices in W. For an example of such a graph see Section 7. Note that that graph is not $(2,2)$-admissible, however, a $(2,2)$-admissible graph with such attribute exists. Suppose that we create a new graph $\hat{G}$ from G by doing the following: (i) for every $u \in Q$ and $v \in N(u)$ we remove uv if $u \geq_{\mathbb{G}} v$, and (ii) for every $u \in Q$ and v, such that $u \geq_{\mathbb{G}} v$, if there exists a maximum set of less than $n^{1/2}/(16\epsilon)$ edge disjoint paths uwv in $\hat{G}$, then we remove all the edges in these paths from $\hat{G}$.

It is easy to see that in (i) at most two edges are removed for every vertex in Q. So, after the removal of all these edges $\hat{G}$ is still far from C_4-freeness. Regarding (ii) it can be shown, using Lemma 9, that after (i), for every $u \in Q$ there are at most 4 vertices in $v \in V(G)$ such that $u \geq_{\mathbb{G}} v$ and u and v are connected by a path of length 2. This, along with some accounting, implies that after (ii), far fewer than $\epsilon|E(G)|$ edges were removed from G to obtain $\hat{G}$. Thus, $\hat{G}$ is $\epsilon/2$-far from being C_4-free.

Now, since $\hat{G}$ is $\epsilon/2$-far from being C_4-free, because of the type of C_4s it has, a constant portion of the vertices of W are in such C_4. So, an algorithm that randomly selects a few vertices from the graph, with high probability, will pick one of these vertices. Suppose that this was the algorithm described in the introduction. Once a vertex from W in a C_4-subgraph of $\hat{G}$ is selected, its neighbours $x, y \in Q$ are discovered. Now, since in $\hat{G}$, x and y are connected by a path of length 2, x and y have many common vertices (at least $n^{1/2}/(16\epsilon)$) and hence the algorithm will discover one of them with high probability. This will lead to the discovery of a C_4-subgraph in G.

This technique cannot be generalized beyond $k = 7$, since for $k > 7$ the C_k-subgraphs may include more complicated subpaths. To deal with this issue, we need the following definition.

▶ **Definition 10.** *Let $\mathbb{G} = (G, \leq)$. A path uLv is a* chain *if for every $w \in v(uL)$, we have $w >_{\mathbb{G}} v$.* *chain*

▶ **Proposition 11.** *Let $r, p \in \mathbb{N}$, $\mathbb{G} = (G, \leq)$ be such that $\mathrm{adm}_r(G) = p$, uLv be an r-admissible path in G, $x \in L$ and $y \in \{u, v\}$. The subpath $xL'y$ of uLv is a chain.*

Proof. By the definition of an r-admissible path for every $z \in xL'$, we have $z >_{\mathbb{G}} \max_{\mathbb{G}}\{u, v\}$ and, in particular, $x >_{\mathbb{G}} \max_{\mathbb{G}}\{u, v\}$. Thus, by definition, $xL'y$ is a chain. ◀

Lemma 9 describes how sets of vertices reachable through r-admissible paths from a fixed start vertex are bounded by a function of $\mathrm{adm}_r(G)$ and r. We can obtain a comparable bound for the more general cases where the vertices are now reachable via chains:

▶ **Lemma 12.** *Let $p, r \in \mathbb{N}$, $\ell \in [r]$ and $\mathbb{G}$ be an ordered (p, r)-admissible graph. Let $P_1, \ldots, P_t$ be a collection of chains of length at most ℓ that all start at the same vertex $x \in \mathbb{G}$ and each end with a vertex smaller than x under $\leq_{\mathbb{G}}$.*

Then the set $W := \{\min_{\mathbb{G}} P_i\}_{i \in [t]}$ which contains the respective minima, under $\leq_{\mathbb{G}}$, of each of the above paths has size at most p^{r^2}.

Proof. Let $V_{<x} := \{y \in V(G) \mid y <_{\mathbb{G}} x\}$, and $s = \max_{i \in [t]} |V(P_i) \cap V_{<x}|$. That is, s is the maximum number of vertices of $V_{<x}$ in a path P_i. Next, we show by induction that the set $\{\min_{\mathbb{G}} P_i\}_{i \in [t]}$ has size at most p^{rs}, since $s < \ell$ this implies the lemma.

For $s = 0$ the statement becomes vacuous, so consider $s = 1$. This is the case that for every $i \in [t]$, the only vertex of P_i smaller than x is its other end point. Since all paths P_i are r-chains the set of these endpoints is a subset of $\mathrm{Target}^r_{\mathbb{G}}(x)$. Therefore, by Lemma 9, $|\{\min_{\mathbb{G}} P_i\}_{i=1}^t| \leq p(p-1)^{r-1} < p^r$ which provides us with the inductive base.

Assume that the statement holds for all $s' < s$, where $s \leq \ell$, and all collections $Q_1, \ldots, Q_h$ of r-chains of length $\leq \ell$, such that for every $i \in [h]$, Q_i starts at the vertex $z \in \mathbb{G}$ and ends at some vertex smaller than z under $\leq_{\mathbb{G}}$, and $|V(Q_i) \cap V_{<z}| \leq s'$.

Let $\mathcal{P} = \{P_i\}_{i=1}^t$ and for every $i \in [t]$, let $y_i \in V(P_i)$ be such that $y_i <_{\mathbb{G}} x$ and $P_i = xL_iy_iR_i$, where $|V(xL_iy_i) \cap V_{<y_i}| = 1$. According to the base of the induction $|\{y_i\}_{i=1}^t| \leq p^r$.

Pick a vertex $y \in \{y_i\}_{i=1}^t$ and let $\mathcal{P}_y$ be the collection of all the paths $y_i R_i$ such that $y_i = y$. Since $y <_G x$, every path in $\mathcal{P}_y$ is a subpath of a path in $\mathcal{P}$, we can conclude that for every $yR \in \mathcal{P}_y$, we have $|V(yR) \cap V_{<y}| < s$. Thus, by the induction assumption, $|\{\min_{\mathbb{G}} P\}_{P \in \mathcal{P}_y}| \leq p^{r(s-1)}$. Finally, by construction, we have $|\{\min_{\mathbb{G}} P\}_{P \in \mathcal{P}}| \leq p^r \cdot |\{\min_{\mathbb{G}} P\}_{P \in \mathcal{P}_y}| \leq p^{rs}$. ◀

4 Nadirs and Braids

braid, full-braid

▶ **Definition 13.** *For every pair of vertices $u, v \in V(G)$, a braid is a set of **edge-disjoint** r-admissible paths each having v as the start vertex and u as the end vertex, and all having the same length. A braid is a* full-braid *if the number of paths it has is maximum.*

length, ends

We use a tilde on an upper case letter to denote a braid, for example $\tilde{B}$. In addition, we use the following notations for paths L and braids $\tilde{B}$:

- $|\tilde{B}|$ is the number of paths in $\tilde{B}$.
- $\text{length}(\tilde{B})$ is the length of the paths in $\tilde{B}$.
- $\text{ends}(\tilde{B})$, is an order pair (v, u) where v is the start vertex of every path in $\tilde{B}$, and u is the end vertices of every path in $\tilde{B}$ and otherwise it is undefined.

Nadir

▶ **Definition 14.** *The* nadir *of a path uLv, denoted* $\text{nadir}(uLv)$, *is the vertex $x \in V(L)$ such that for every $y \in V(L)$ we have $y \geq_{\mathbb{G}} x$. The* nadir *of a braid $\tilde{B}$, denoted* $\text{nadir}(\tilde{B})$ *is the set* $\{\text{nadir}(uLv)\}_{uLv \in \tilde{B}}$.

δ-weak, δ-strong

▶ **Definition 15.** *Let $\tilde{B}$ be a braid and $\delta > 0$, a vertex $v \in \text{nadir}(\tilde{B})$ is* δ-weak *for $\tilde{B}$, if the maximum size of a braid $\tilde{B}' \subseteq \tilde{B}$ such that $\text{nadir}(\tilde{B}') = \{v\}$ is at most $\deg_G(v)/\delta$ and otherwise it is* δ-strong *for $\tilde{B}$.*

δ-braids

▶ **Definition 16.** *For every $\delta \in \mathbb{N}$, a braid $\tilde{D}$ is a* δ-braid *if every vertex in $\text{nadir}(\tilde{D})$ is δ-weak for $\tilde{D}$.*

▶ **Proposition 17.** *Let $r, p \in \mathbb{N}$, $\mathbb{G} = (G, \leq)$ be such that $\text{adm}_r(G) = p$, uLv an r-admissible path in G, $x \in L$ and $y = \text{nadir}(uLv)$. Then, $x \geq_{\mathbb{G}} y$ and the subpath $xL'y$ of uLv is a chain.*

Proof. By the definition of an r-admissible path for every $z \in xL'$, we have $z >_{\mathbb{G}} \text{nadir}(uLv)$. Thus, by definition, $x \geq_{\mathbb{G}} y$ and $xL'y$ is a chain. ◀

▶ **Lemma 18.** *Let $p, r \in \mathbb{N}$, $\ell \in [r]$ and G be an ordered (p, r)-admissible graph and $u, v \in V(G)$ such that $u >_{\mathbb{G}} v$. If there exists a braid $\tilde{B}$ such that $\text{ends}(\tilde{B}) = (u, v)$, $\text{length}(\tilde{B}) \leq r$ and $|\text{nadir}(\tilde{B})| \geq 2r^2 \cdot p^{r^2}$, then there exists a braid $\tilde{B}' \subset \tilde{B}$ such that $|\tilde{B}'| \geq 2r$ and the only vertices that are shared between the paths of $\tilde{B}'$ are u and v.*

Proof. Let $\tilde{B}^* \subseteq \tilde{B}$ where for every $x \in \text{nadir}(\tilde{B}^*)$, there exists exactly one path uLv such that $x = \text{nadir}(uLv)$. By assumption, $|\tilde{B}^*| \geq 2r \cdot p^{r^2}$. Let W be the set of all vertices in the paths of $\tilde{B}^*$ except u and v.

We first argue that every vertex $w \in W$ can be contained in at most p^{r^2} members of $\tilde{B}^*$. To that end, assume that w is contained in members $uL_1v, \ldots, uL_tv \in \tilde{B}^*$. Let $x_1, \ldots, x_t$ be the respective nadirs of these subgraphs under $\mathbb{G}$, so $x_i := \text{nadir}(uL_iv)$. For every $i \in [t]$, let $wL_i'x_i$ be a subpath of uL_iv. Since each of the paths $uL_1v, \ldots, uL_tv$ is r-admissible, by Proposition 17, all the paths $wL_1'x_1, \ldots, wL_t'x_t$ are chains. We can therefore invoke Lemma 12 on the paths $wL_1'x_1, \ldots, wL_t'x_t$ to conclude $t \leq p^{r^2}$. Thus, w can be contained in at most p^{r^2} members of $\tilde{B}^*$.

It follows immediately that every path in $\tilde{B}^*$ shares vertices that are not u or v with at most $r \cdot p^{r^2}$ other paths of $\tilde{B}^*$. Consequently, a simple greedy selection process can construct the claimed braid $\tilde{B}'$ such that $|\tilde{B}'| \geq 2r$. ◀

5 Trimming

Trimming

In this section, we present the procedure that we refer to as *trimming*. It is important to note that the trimming procedure is used only for the analysis of the algorithm we present. This is the reason that trimming can read the entire input graph.

Trimming receives as input a graph G that is (p,r)-admissible, an ordering $\mathbb{G}$ and the parameters $p,r \in \mathbb{N}$ and $\gamma,\delta \in \mathbb{R}^+$. From now on, in this section, assume that these parameters are fixed. Trimming creates a subgraph $\hat{G}$ of G that has a number of properties that are essential for the analysis of our algorithm. For example, with the right choice of parameters, if G is ϵ-far from C_{2r+1}-freeness, then $\hat{G}$ is $\epsilon/2$-far from C_{2r+1}-freeness; for every $v,u \in V(G)$, if there exists an full-braid $\tilde{B}$ in $\hat{G}$ such that $\text{ends}(\tilde{B}) = (v,u)$ and $\text{length}(\tilde{B}) \le r$, then $|\tilde{B}| \ge \deg_G(v)/\gamma$. Note that the choice of $\deg_{\mathbf{G}}(v)$ and not $\deg_{\hat{\mathbf{G}}}(v)$, in the last inequality, is crucial for our proof to work and that $\hat{G}$ is (p,r)-admissible since it is a subgraph of G.

The trimming procedure

Initialize $\hat{G}$ to be equal to G and repeat the following steps until each one does not result in any edge removal:

1. For every distinct $v \in V(G)$, $u \in \text{Target}^r_{\mathbb{G}}(v)$, $\ell \in [r]$, and full-braid $\tilde{B}$ in $\hat{G}$ such that $\text{length}(\tilde{B}) = \ell$ and $\text{ends}(\tilde{B}) = (v,u)$, if $|\tilde{B}| < \deg_G(v)/\gamma$, then all the edges participating in the paths of $\tilde{B}$ are removed from $E(\hat{G})$.
2. For every distinct $v \in V(G)$, $u \in \text{Target}^r_{\mathbb{G}}(v)$, $\ell \in [r]$, and δ-braid $\tilde{D}$ in $\hat{G}$ such that $\text{length}(\tilde{D}) = \ell$ and $\text{ends}(\tilde{D}) = (v,u)$, if $|\tilde{D}| < \deg_G(v)/\gamma$, then all the edges participating in the paths of $\tilde{D}$ are removed from $E(\hat{G})$.

Next, we show that the graph $\hat{G}$ created by using the trimming process is far from being C_{2r+1}-free.

▶ **Lemma 19.** *Let $p,r \in \mathbb{N}$, $\epsilon > 0$, $\gamma \in \mathbb{R}^+$, and G be a (p,r)-admissible graph that is ϵ-far from C_{2r+1}-freeness. If $\gamma > 8rp^r/\epsilon$, then the graph $\hat{G}$ created by trimming G with parameters p, r and γ, is $\epsilon/2$-far from C_{2r+1}-freeness and (p,r)-admissible.*

Proof. Note first that $\hat{G}$ is a subgraph of G since it was created from G by taking a copy of G and removing specific edges from it. So, to prove the claim, we only need to show that at most $\epsilon|E(G)|/2$ edges have been removed from G to obtain $\hat{G}$. Since G is ϵ-far from being C_{2r+1}-free, this implies the claim.

Trimming step (1). For every $v \in V(G)$ and $u \in \text{Target}^r_{\mathbb{G}}(v)$, the maximum number of edges removed in this step is at most the number of edges in the paths of a full-braid $\tilde{B}$ such that $\text{length}(\tilde{B}) = \ell$ and $|\tilde{B}| \le \deg_G(v)/\gamma$. Thus, for every such v and u at most $r \deg_G(v)/\gamma$ edges are removed from $\hat{G}$. By Lemma 9, $|\text{Target}^r_{\mathbb{G}}(v)| \le p^r$, and therefore the total number of edges removed in this step is at most $\sum_{v \in V(G)} rp^r \deg_G(v)/\gamma \le (\epsilon/8) \sum_{v\in V(G)} \deg_G(v) = \epsilon|E(G)|/4$, where the inequality follows since $\gamma > 8rp^r/\epsilon$.

Trimming step (2). The proof is the same as the previous step. ◀

▶ **Proposition 20.** *Let $v,u \in V(G)$ be such that $u >_{\mathbb{G}} v$ and assume that $\hat{G}$ was created by trimming with parameters $r,p \in \mathbb{N}$ and $\gamma,\delta \in \mathbb{R}^+$. Then*

1. *if $uv \in E(\hat{G})$ then $\deg_G(u) \le \gamma$,*
2. *if $\hat{G}$ has an r-admissible path uLv, then there exists a full-braid $\tilde{B}$ in $\hat{G}$ such that ends$(\tilde{B}) = (u,v)$, length$(\tilde{B})$ = length(uLv) and $|\tilde{B}| \ge \deg_G(u)/\gamma$, and*

3. *if $\hat{G}$ has an r-admissible path uLv, and there does not exist a $\tilde{B}$ in $\hat{G}$ such that* $\mathrm{nadir}(\tilde{B}) = \mathrm{nadir}(uLv)$*, ends*$(\tilde{B}) = (u, v)$*, length*$(\tilde{B}) =$ *length*(uLv), $|\tilde{B}| \geq \deg_G(v)/\delta$*, then there exists a δ-braid $\tilde{D}$ in $\hat{G}$ such that ends*$(\tilde{D}) = (u, v)$*, length*$(\tilde{D}) =$ *length*(uLv) *and* $|\tilde{D}| \geq \deg_G(v)/\gamma$.

Proof. (1) By definition, the braid $\tilde{B}$ that contains only the pass vu is a full-braid, such that $\mathrm{ends}(\tilde{B}) = vu$ and $\mathrm{length}(\tilde{B}) = 1$. Since $|\tilde{B}| = 1 \leq \deg_G(v)/\gamma$, the edge vu is removed from $\hat{G}$ during step 1 of Trimming.
(2) The proof follows directly from the definition of a full-braid and Step (1) of trimming.
(3) The proof follows directly from Step (2) of trimming. ◀

6 The main algorithm

Algorithm 1

```
Input: $r, p \in \mathbb{N}$, $\epsilon \in \mathbb{R}^+$, oracle access to a graph $G$, and $|V(G)|$

1  Compute the values of $\xi_1$, $\xi_2$ and $\xi_3$
2  Set $S = \emptyset$
3  Repeat $\xi_1$ times                                          // Selection Loop
4      add to $S$ a u.a.r selected vertex of $V(G)$
5  for $i = 1, 2, \ldots, \xi_2$ do                              // Outer Loop
6      Set $S' = \emptyset$
7      for $v \in S$ do                                          // Middle Loop
8          Query the degree of $v$
9          if $\deg_G(v) \leq \xi_3$ then
10             Query the identity of all neighbours of $v$ and add them to $S'$
11         else
12             Repeat $\xi_3$ times                              // Inner Loop
13                 select independently and u.a.r $k \in [\deg_G(v)]$
14                 query the identity of the $k$'th neighbour of $v$ and add it to $S'$
15     Set $S = S \cup S'$
16     if the knowledge-graph has a $C_{2r+1}$-subgraph then
17         return REJECT
18 return ACCEPT
```

We refer to the loop on line 3 of Algorithm 1, as *Selection Loop*, to the loop on line 5 of Algorithm 1, as *Outer Loop*, to the loop on line 7 as *middle loop* and to the loop on line 12 as *Inner Loop*. In all of this section, G is a (p, r)-admissible graph G with n vertices, $\mathbb{G}$ is an ordering of G, $\hat{G}$ is a subgraph of G obtained by trimming with the parameters $r, p \in \mathbb{N}$, $\gamma, \delta \in \mathbb{R}^+$ and when we use the parameter ξ_3 we assume that it is a multiple of $2r$. Note that by Fact 8, Algorithm 1 receives a bound on the number of edges implicitly, since the value of p is part of its input.

Assumptions ξ_3

The following lemma follows directly from Algorithm 1.

▶ **Lemma 21.** *The query complexity of Algorithm 1 is $O(\xi_1 \cdot \xi_3^{\xi_2})$.*

In this section, we use the term *knowledge-graph* to refer to the subgraph of G that is discovered by queries of Algorithm 1. *knowledge-graph*

6.1 Proof overview of the main theorem

Bounding the query complexity of Algorithm 1 and proving that it returns ACCEPT given a C_{2r+1}-free input graph is rather simple. The challenge is to prove that Algorithm 1 returns REJECT with probability at least 2/3 on an input graph that is (p, r)-admissible and ϵ-far from being C_{2r+1}-free. This is the focus of this subsection; suppose that Algorithm 1 is given as input $p, r \in \mathbb{N}$, $\epsilon > 0$, oracle access to a (p, r)-admissible graph G that is ϵ-far from being C_{2r+1}-free.

The proof is divided into three phases: (i) showing that, with high probability, Algorithm 1 discovers a set of vertices D in some C_{2r+1}-subgraph H of $\hat{G}$, such that every path in H that does not include a vertex from D has a length of at most r. (ii) showing that if (i) occurred, then with high probability, all the vertices of a C_{2r+1}-subgraph H' of $\hat{G}$ are discovered; and (iii) showing that if (ii) occurred, then with high probability, the knowledge-graph includes a C_{2r+1}-subgraph. Note that since $\hat{G}$ is a subgraph of G, this implies that, with high probability, a C_{2r+1}-subgraph H' of $\hat{G}$ is discovered. We next further explain each phase starting with (iii), proceeding to (i), and ending with (ii).

(iii) The knowledge-graph includes all the vertices of a C_{2r+1}-subgraph H' of $\hat{G}$. By (1) of Proposition 20, every edge uv in $E(H')$ is incident to a vertex with degree at most γ in G. Thus, line 9 of Algorithm 1 ensures that the edge uv is discovered.

(i) By using (1) of Proposition 20, it is shown that a significant portion of the vertices x in G are such that: $\deg_G(x) \leq \gamma$, x is in some C_{2r+1}-subgraph H of $\hat{G}$, $\text{dist}_H(x, y) = r$, where y is the smallest vertex in H according to $\mathbb{G}$. The value of ξ_1 is set so that with high probability a vertex such as x is discovered. As in (iii), both of x's neighbours in H will also be discovered. Now, notice that x and y are selected so that the shorter path in H between x and y is a chain of length r. Since H is in $\hat{G}$, so is the chain above. Thus, by Lemma 23 (appears later), with high probability, y will also be discovered. Thus, a set of vertices as D above is discovered with high probability. The formal proof appears later in Lemma 24.

(ii) Suppose that a set of vertices like D above was discovered by Algorithm 1 and H is the C_{2r+1}-subgraph of $\hat{G}$ such that $D \subseteq V(H)$. Note that if we can ensure that, with high probability, we find a vertex w such that $D \cup \{w\}$ satisfies the same attributes as D (though not necessarily with the initial subgraph H), then we are guaranteed that, with high probability, all the vertices of some C_{2r+1} subgraph of $\hat{G}$ will eventually be discovered.

Let y be the smallest vertex in $V(H) \setminus D$ according to $\mathbb{G}$. Assume that y is on a subpath uLv of H such that $V(uLv) \cap D = \{u, v\}$. If $y <_{\mathbb{G}} \max_{\mathbb{G}}\{u, v\}$, then, as in (i), there exists a chain of length at most r in $\hat{G}$ between one of the vertices u and v and the vertex y. By the same reasoning as in (i), with high probability this vertex is discovered. Next, we describe what happens if $y >_{\mathbb{G}} \max_{\mathbb{G}}\{u, v\}$.

Suppose that $u >_{\mathbb{G}} v$. If there exists a braid $\tilde{B}$ such that $|\tilde{B}| \geq \deg_G(u)/\delta$ and $y = \text{nadir}(\tilde{B})$, then by future choice of the value of ξ_3, with high probability, a vertex $v' \in N(v)$ and one of the paths in $\tilde{B}$ is selected. The fact that $y = \text{nadir}(\tilde{B})$ ensures that there exists a chain of length at most r in $\hat{G}$ between v' and v, containing the vertex y. As in (i), this ensures that y is discovered with high probability. This is proved later on formally in Lemma 25. Hence, it remains to deal with the case that a braid like $\tilde{B}$ does not exist.

The nonexistence of a braid like $\tilde{B}$ implies that there exists a δ-braid $\tilde{D}$ such that

$|\tilde{D}| \geq \deg_G(u)/\gamma$. Notice that according to the definition of a δ-braid, this implies that $|\operatorname{nadir}(\tilde{D})| \geq \delta/\gamma$. The value of δ is selected to be significantly higher than γ, thus ensuring that $|\operatorname{nadir}(\tilde{D})|$ is sufficiently large. Using a similar proof to the previous case with some significant extra work and the help of a sufficiently large value of ξ_3, it is shown later in Lemma 26, that eventually a sufficiently large portion of the vertices in $\operatorname{nadir}(\tilde{D})$ are discovered. Once this happens, we are guaranteed by Lemma 18, that there are paths $uL_1v, \dots, uL_{2r}v$ in $\hat{G}$ such that the L_i are vertex-disjoint. This implies that one of these paths together with H includes a C_{2r+1}-subgraph H' of $\hat{G}$ such that $D \cup \{y'\} \subseteq V(H')$, where y' is the nadir of the path, and $D \cup \{y'\}$ satisfies the same attributes as D. The formal proof for this case appears later in Lemma 27.

6.2 Proof of the main theorem

▶ **Lemma 22.** *Let $\tau < \xi_2$ and S be the set of all vertices of an C_{2r+1}-subgraph of $\hat{G}$. If $\xi_3 \geq \gamma$ and all the vertices of S are in the knowledge-graph at the end of iteration τ of Outer Loop, then with probability 1, at the end of iteration τ of Outer Loop, the knowledge-graph is not C_{2r+1}-free.*

Proof. Let H be a C_{2r+1} subgraph of $\hat{G}$ and let $uv \in E(H)$, be such that $u >_{\mathbb{G}} v$ and $uv \in E(\hat{G})$. By Proposition 20, $\deg_G(u) \leq \gamma$. Thus, according to Inner Loop, if both u and v are in the knowledge-graph at the end of iteration τ of Outer Loop, then because $\xi_3 \geq \gamma$ with probability 1, at the end of iteration τ of Outer Loop, the edge uv is also in the knowledge-graph. The same applies to all other edges. ◀

▶ **Lemma 23.** *Let $\ell \in [r]$, $p_{23}(\ell) = \ell(1-\gamma^{-1})^{\xi_3}$, $\tau \leq \xi_2 - \ell$ and $u, v \in V(G)$ be such that $u >_{\mathbb{G}} v$. Assume that there exists a chain uLv in $\hat{G}$ such that length$(uLv) = \ell$. If u is in the knowledge-graph at the end of iteration τ of Outer Loop, then with probability at least $1 - p_{23}(\ell)$, at the end of iteration $\tau + \ell$ of Outer Loop, v is also in the knowledge-graph.*

Proof. We proceed by induction on the value of ℓ. Suppose that $\ell = 1$. Thus, u and v are neighbours in $\hat{G}$ and therefore, by Proposition 20, $\deg_G(u) \leq \gamma$. Thus, according to the contents of Outer Loop, with probability 1, at the end of iteration $\tau + 1$ of Outer Loop, v is also in the knowledge-graph.

Assume by induction that the lemma holds for every chain L in $\hat{G}$, where $\operatorname{length}(L) < \ell$. Let x be the closest vertex to u in uLv such that $x <_{\mathbb{G}} u$ and let $uL'x$ be a subpath of uLv. By construction, for every $h \in uL'$, we have $h >_{\mathbb{G}} x$ and therefore $uL'x$ is an r-admissible path in $\hat{G}$.

Suppose that $x \neq v$. As $1 < \operatorname{length}(uL'x) < \ell$, according to the induction assumption, given that u is in the knowledge-graph at the end of iteration τ of Outer Loop, with probability at least $1 - \operatorname{length}(uL'x)(1-\gamma^{-1})^{\xi_2}$, at the end of iteration $\tau + \operatorname{length}(uL'x)$ of Outer Loop, x is also in the knowledge-graph.

Let xL^*v be a subpath of uLv. Since xL^*v is a subpath of uLV, we conclude that $\operatorname{length}(xL^*v) < \ell$. Thus, by the induction assumption, given that x is in the knowledge-graph at the end of iteration $\tau + \operatorname{length}(uL'x)$ of Outer Loop, with probability at least $1 - \operatorname{length}(xL^*v)(1-\gamma^{-1})^{\xi_3}$, at the end of iteration $\tau + \ell$ of Outer Loop, the vertex v is also in the knowledge-graph. Consequently, together with the previous, given that u is in the knowledge-graph at the end of iteration τ of Outer Loop, with probability at least $1 - (\operatorname{length}(uL'x) + \operatorname{length}(xL^*v))(1-\gamma^{-1})^{\xi_3} = 1 - \ell(1-\gamma^{-1})^{\xi_3}$, at the end of iteration $\tau + \ell$ of Outer Loop, the vertex v is also in the knowledge-graph.

Suppose that $x = v$. By definition, uLv is an r-admissible path in $\hat{G}$. Therefore, by Proposition 20, $\hat{G}$ has a full-braid $\tilde{B}$, such that $\text{ends}(\tilde{B}) = (u, v)$ and $\text{length}(\tilde{B}) = \ell$.

Since there are at least $\deg_G(u)/\gamma$ edge-disjoint paths in $\tilde{B}$, at least $\deg_G(u)/\gamma$ of the neighbours of the vertex u are in one of the paths of $\tilde{B}$. Thus, according to the contents of Inner Loop, given that u is in the knowledge-graph at the end of iteration τ of Outer Loop, with probability at least $1 - (1 - \gamma^{-1})^{\xi_3}$, at the end of iteration $\tau + \ell$ of Outer Loop, a vertex y adjacent to u in a path of $\tilde{B}$ is also in the knowledge-graph.

Let $uyL"v$, be the specific r-admissible path of $\tilde{B}$ that includes y. By Proposition 11, $yL"v$ is a chain. Since $\text{length}(yL"v) = \ell - 1$, by the induction assumption, given that y is in the knowledge-graph at the end of iteration $\tau + 1$ of Outer Loop, with probability at least $1 - (\ell - 1)(1 - \gamma^{-1})^{\xi_3}$, at the end of iteration $\tau + \ell$ of Outer Loop, the vertex v is also in the knowledge-graph. Consequently, in this case, given that u is in the knowledge-graph at the end of iteration τ of Outer Loop, with probability at least $1 - (1+\ell-1)(1-\gamma^{-1})^{\xi_3} = 1 - \ell(1-\gamma^{-1})^{\xi_3}$, at the end of iteration $\tau + \ell$ of Outer Loop, the vertex v is also in the knowledge-graph. ◀

▶ **Lemma 24.** *Let $p_{24}(r) = p_{23}(r) + (1 - \epsilon p/(2\gamma))^{\xi_1}$. If $\hat{G}$ is ϵ-far from C_{2r+1}-freeness, then with probability at least $1 - p_{24}(r)$, at the end of iteration r of Outer Loop the knowledge-graph includes a set of vertices D, such that for some C_{2r+1}-subgraph H of $\hat{G}$ we have $D \subseteq V(H)$ and for every path uLv in H, where $D \cap V(uLv) = \{u, v\}$, we have $\text{length}(uLv) \leq r$.*

Proof. Let T be the set of all vertices $x \in V(G)$ such that $\deg_G(x) \leq \gamma$. For every C_{2r+1}-subgraph H of $\hat{G}$ let $\text{anchor}(H)$ be the vertex in $V(H)$ such that for every $y \in V(H)$ we have $y \geq_{\mathbb{G}} \text{anchor}(H)$, and let $\text{stern}(H)$ be an arbitrary vertex in $V(H) \cap T$ such that $\text{dist}_H(\text{anchor}(H), \text{stern}(H)) = r$. Let K be the set of all vertices x such that $x = \text{stern}(H)$ for some C_{2r+1}-subgraph H of $\hat{G}$.

Let H be an arbitrary C_{2r+1}-subgraph of $\hat{G}$. $\text{stern}(H)$ exists since there are exactly two vertices $x \in V(H)$ such that $\text{dist}_H(\text{anchor}(H), x) = r$, they are neighbours in $\hat{G}$ and therefore, by Proposition 20, at least one of them is in T.

We first show that, with probability at least $1 - (1 - \epsilon p/(2\gamma))^{\xi_1}$, after Selection Loop and prior to the first iteration of Outer Loop, the set S of Algorithm 1 includes a vertex from K. Afterwards, we show that if the above occurred, then with probability at least $1 - p_{23}(r)$, at the end of iteration r of Outer Loop, the knowledge-graph includes a set D as required.

Suppose that we removed from $\hat{G}$ every edge adjacent to a vertex in K, then by construction, the graph we get is C_{2r+1}-free. Since, by Lemma 19, $\hat{G}$ is $\epsilon/2$-far from being C_{2r+1}-free it holds that $|K|\gamma \geq \epsilon np/2$. Thus, according to Selection Loop, with probability at least $1 - (1 - \epsilon p/(2\gamma))^{\xi_1}$, after Selection Loop and prior to the first iteration of Outer Loop, the set S of Algorithm 1 includes a vertex from K.

Suppose that H is a C_{2r+1}-subgraph of $\hat{G}$ and that $v = \text{stern}(H)$ is set S of Algorithm 1 after Selection Loop and prior to the first iteration of Outer Loop. Let D be the set of vertices that includes v, v's neighbours in H and $\text{anchor}(H)$. We note that the set D satisfies the requirements of the lemma.

According to the Inner Loop, with probability 1, at the end of the first iteration of Outer Loop, the neighbours of v in H will also be in the knowledge-graph. By construction $v >_{\mathbb{G}} \text{anchor}(H)$ and $\hat{G}$ has a chain of length r whose end-vertices are v and $\text{anchor}(H)$. By Lemma 23, with probability at least $1 - p_{23}(r)$, at the end of iteration r of Outer Loop, the knowledge-graph includes the vertex $\text{anchor}(H)$. Applying a union bound finishes the proof. ◀

▶ **Lemma 25.** *Let $\ell \in [r]$, $\tau \in \xi_2 - \ell$, $p_{25}(\ell) = (1 - \delta^{-1})^{\xi_3} - \ell(1 - \gamma^{-1})^{\xi_3}$, and $u, v, w \in V(G)$ be such that $u >_{\mathbb{G}} v$. Suppose that there exists a braid $\tilde{B}$ in $\hat{G}$ such that $\text{nadir}(\tilde{B}) = w$,* *ends*$(\tilde{B}) = (u,v)$*, length*$(\tilde{B}) = \ell$ *and* $|\tilde{B}| \geq \deg_G(v)/\delta$*. If* u *is in the knowledge-graph at the end of iteration* τ *of Outer Loop, then with probability at least* $1 - p_{25}(\ell)$*, at the end of iteration* $\tau + \ell$ *of Outer Loop,* w *is also in the knowledge-graph.*

Proof. Since $|\tilde{B}| \geq \deg_G(v)/\delta$, by the same reasoning as in Lemma 23, if u is in the knowledge-graph at the end of iteration τ of Outer Loop, then with probability at least $1 - (1 - \delta^{-1})^{\xi_3}$, at the end of iteration $\tau + 1$ of Outer Loop, the knowledge-graph includes a vertex y that is adjacent to u in a path of $\tilde{B}$. Assume that $y \neq \text{nadir}(\tilde{B}) = w$, since otherwise we are done. Let $vyL'u$, be the specific r-admissible path of $\tilde{B}$ that includes y. Let yL^*w we a subpath of $vyL'u$. By Proposition 17 $y >_{\mathbb{G}} w$ and yL^*w is a chain.

Using the same reasoning as in Lemma 23, if u is in the knowledge-graph at the end of iteration $\tau + 1$ of Outer Loop, then with probability at least $1 - (\ell - 1)(1 - \gamma^{-1})^{\xi_3}$, at the end of iteration $\tau + 1$ of Outer Loop, the knowledge-graph includes w. An application of a union bound concludes the proof. ◀

▶ **Lemma 26.** *Let* $\ell \in [r]$*,* u, v *in*$V(G)$ *and* $p_{26}(\ell) = \ell(\delta/(8\gamma^2))(1 - \gamma^{-1})^{\xi_3} + e^{-\frac{\delta}{16\gamma^3}}$*. Assume that* $\hat{G}$ *has a* δ*-braid* $\tilde{D}$ *such that ends*$(\tilde{D}) = (v,u)$*, length*$(\tilde{D}) = \ell$*, and* $|\tilde{D}| \geq \deg_G(v)/\gamma$*. Assume also that* v *is in the knowledge-graph at the end of iteration* $\tau \leq \xi_2 - \ell$ *of Outer Loop. With probability at least* $1 - p_{26}(\ell)$*, at the end of iteration* $\tau + \ell$ *of Outer Loop, the knowledge-graph also includes a size* $\gamma/(8\delta^2)$ *subset of* $\text{nadir}(\tilde{D})$*.*

Proof. Let $S' \subseteq N_G(v)$ be the set of all vertices adjacent to v in a path of $\tilde{D}$. The following analysis is for iteration $\tau + 1$ of Outer Loop.

For every $i \in [\xi_3]$, let S_i be the set of all vertices of S' that are in the knowledge-graph at the end of iteration i of the inner loop, and let $W_i = (S_i, \text{nadir}(\tilde{D}), F)$ be a bipartite graph such that $xy \in F$ if and only if $x \in S_i$, $y \in \text{nadir}(\tilde{D})$ and there exists a path L^* in $\tilde{D}$ such that $x \in V(L^*)$ and $y = \text{nadir}(L^*)$. Also, let Y be the size of a maximum matching in W_{ξ_3}.

We show first that if $Y \geq \delta/(8\gamma^2)$, then with probability at least $1 - \ell(\delta/(8\gamma^2))(1 - \gamma^{-1})^{\xi_3}$, at the end of iteration $\tau + \ell$ of Outer Loop, the knowledge-graph includes at least $\delta/(8\gamma^2)$ vertices from $\text{nadir}(\tilde{D})$. Afterwards, we show that with probability at least $1 - e^{-\frac{\delta}{16\gamma^3}}$, we have $Y \geq \delta/(8\gamma^2)$, which together with the previous implies the lemma.

Suppose that $Y \geq \delta/(8\gamma^2)$, and let $M \subseteq F$ be a size $\delta/(8\gamma^2)$ matching in W_{ξ_3}. For every edge $xy \in M$, where $x \in S$ and $y \in \text{nadir}(\tilde{D})$, the braid $\tilde{D}$ has an r-admissible path uL^*v in $\tilde{D}$ such that $x \in V(L^*)$ and $y = \text{nadir}(L^*)$. By Proposition 17 the subpath $xL''y$ of uL^*v is a chain in $\hat{G}$. Thus, by Lemma 23, for every edge $xy \in M$, if x is in the knowledge-graph at the end of iteration $\tau + 1$ of Outer Loop, then with probability at least $1 - \ell(1 - \gamma^{-1})^{\xi_3}$, at the end of iteration $\tau + \ell$ of Outer Loop, the vertex y is also in the knowledge-graph. Thus, by the union bound, given that $Y \geq \delta/(8\gamma^2)$, with probability at least $1 - \ell(\delta/(8\gamma^2))(1 - \gamma^{-1})^{\xi_3}$, the knowledge-graph includes at least $\delta/(8\gamma^2)$ vertices of $\text{nadir}(\tilde{D})$.

Now, we show that, with probability at least $1 - e^{-\frac{\delta}{16\gamma^3}}$, at the end of iteration $\tau + \ell$ of Outer Loop, $Y \geq \delta/(8\gamma^2)$. We use martingales for this purpose.

For every $i \in [\xi_3]$, let X_i be the identity of the neighbour of v selected in iteration i of Inner Loop, and let $W_i = (S_i, \text{nadir}(\tilde{D}), F_i)$ be a bipartite graph such that $S_i = \{X_j\}_{j=1}^{i}$ and $xy \in F_i$ if and only if $x \in S_i$, $y \in \text{nadir}(\tilde{D})$, $x \geq_{\mathbb{G}} y$ and $\hat{G}$ has a chain xLy such that $\text{length}(xLy) \leq \ell$.

For every $i \in [\xi_3]$, let Y_i be the size of a maximum matching in W_i minus the size of a maximum matching in W_{i-1} and $Y' = \sum_{i=1}^{\delta/(2\gamma)} \mathbb{E}[Y_i]$. Since $\xi_3 \geq \delta/(2\gamma)$, by using the linearity of expectation, we have $\mathbb{E}[Y] \geq \mathbb{E}[Y']$. Next, we show that for every $i \in [\delta/(2\gamma)]$, we have $\mathbb{E}[Y_i] \geq 1/(2\gamma)$. Together, with the above, this implies that $\mathbb{E}[Y'] \geq \delta/(4\gamma^2)$.

Fix $j \in [2, \xi_3]$, and M_{j-1} to be a maximum matching in W_{j-1}. Note that if X_j, is a vertex in S that is not in W_{j-1} and there exists a vertex $y \in \text{nadir}(\tilde{D})$, such that y is not in M_{j-1} and $\hat{G}$ has a chain xLy such that $\text{length}(xLy) \leq \ell$, then W_j has an edge that does not share a vertex with any edge in M_{j-1} and therefore $Y_j = 1$. We next lower bound the probability that this happens.

By the definition of a δ-braid, for every $y \in \text{nadir}(\tilde{D})$, there are at most $\deg_G(u)/\delta$ paths in $\tilde{D}$. So, if we remove from $\tilde{D}$ all the the paths in which the vertices of $V(M_{j-1}) \cap \text{nadir}(\tilde{D})$ participate, then we removed from $\tilde{D}$ at most $j \cdot \deg_G(u)/\delta \leq (\delta/2\gamma) \cdot \deg_G(u)/\delta = \deg_G(u)/(2\gamma)$ edges. Thus, after the removal of these edges the resulting braid $\tilde{D}'$ includes at least $\deg_G(u)/(2\gamma)$ paths. So, the probability that a vertex selected in Inner Loop will be in one of the paths of $\tilde{D}'$ is at least $1/(2\gamma)$. Note that if this event occurs for some $i \in [2, \delta/(2\gamma)]$, then $Y_i - Y_{i-1} = 1$. Thus, we conclude that, indeed, for every $i \in [\delta/(2\gamma)]$, we have $\mathbb{E}[Y_i] \geq 1/(2\gamma)$.

We use the vertex exposure martingale to show that with sufficiently high probability the maximum matching in the random graph $W := W_{\xi_3}$ is sufficiently large. We reveal the whole graph W, iterating over $i = 1, \ldots, \delta/(2\gamma)$, and for every $i \in [\delta/(2\gamma)]$ we reveal the identity of X_i and the identities of its neighbours in W and the edges of W incident on X_i. For every $t \in [\delta/(2\gamma)]$, we let $Z_t = \mathbb{E}[Y' \mid X_t, \ldots, X_1]$ and let $Z_0 = \mathbb{E}[Y']$.

We note that for every $t \in [\delta/(2\gamma)]$, we have $|Z_{t+1} - Z_t| \leq 1$ since Z_{t+1} has only one fixed vertex more than Z_t. Thus, by the Azuma Hoeffding tail bound, with probability at least $1 - e^{-\frac{2(\delta/(8\gamma^2))^2}{\delta/(2\gamma)}} = 1 - e^{-\frac{\delta}{16\gamma^3}}$, we have $|Z_{\delta/(2\gamma)} - Z_0| \leq \delta/(8\gamma^2)$. This implies that with probability at least $1 - e^{-\frac{\delta}{16\gamma^3}}$, we have $Y \geq \delta/(8\gamma^2)$. ◀

▶ **Lemma 27.** *Let $\tau \in [\xi_2 - r^2]$, H a C_{2r+1}-subgraph of $\hat{G}$ and $D \subset V(H)$ such that every subpath of H that does not include vertices from D has length at most r. If $\delta/(8\gamma^2) \geq 2r^2 \cdot p^{r^2}$, and all vertices of D were in the knowledge-graph at the end of iteration τ of Outer Loop, then with probability at least $1 - 2r \cdot \max\{p_{23}(\ell), p_{25}(\ell), p_{26}(\ell)\}$, at the end of iteration $\tau + r^2$ of Outer Loop, all the vertices of some C_{2r+1}-subgraph of $\hat{G}$ are in the knowledge-graph.*

Proof. We shall show that if all vertices of D are in the knowledge-graph at the end of iteration τ of Outer Loop, then with probability at least $1 - \max\{p_{23}(\ell), p_{25}(\ell), p_{26}(\ell)\}$, at the end of iteration $\tau + \ell$ of Outer Loop, the knowledge-graph also includes a vertex w' such that $D \cup \{w'\}$ are all vertices of some C_{2r+1}-subgraph of $\hat{G}$ (not necessarily H). By the union bound, this implies the lemma.

Let $w \in V(H) \setminus D$ be such that for every $y \in V(H) \setminus D$ we have $y \geq_{\mathbb{G}} v$. Let $u, v \in D$ be such that $u >_{\mathbb{G}} v$ and the path uLv in H satisfies $V(L) \cap D = \emptyset$.

Suppose first that $w <_{\mathbb{G}} v$, by construction the path $vL'w$ of H is a chain of length at most r in $\hat{G}$. Thus, by Lemma 23, with probability at least $1 - p_{23}(\ell)$, at the end of iteration $\tau + \ell$ of Outer Loop, the knowledge-graph also includes w. Note that $D \cup \{w\} \subseteq V(H)$.

Assume now that $w >_{\mathbb{G}} u$, then by the definition of a nadir, $w = \text{nadir}(uLv)$. Suppose also that there exists a braid $\tilde{B}$ in $\hat{G}$ such that $\text{nadir}(\tilde{B}) = w$, $\text{ends}(\tilde{B}) = (u, v)$, $\text{length}(\tilde{B}) \leq r$ and $|\tilde{B}| \geq \deg_G(v)/\delta$. By Lemma 25, with probability at least $1 - p_{25}(\ell)$, at the end of iteration $\tau + \ell$ of Outer Loop, w is also in the knowledge-graph. Note that $D \cup \{w\} \subseteq V(H)$.

Finally, suppose that there does not exist a braid $\tilde{B}$ in $\hat{G}$ such that $\text{nadir}(\tilde{B}) = w$, $\text{ends}(\tilde{B}) = (u, v)$, $\text{length}(\tilde{B}) \leq r$ and $|\tilde{B}| \geq \deg_G(v)/\delta$. Then, by (3) of Proposition 20, there exists a δ-braid $\tilde{D}$ such that $\text{ends}(\tilde{D}) = (u, v)$, $\text{length}(\tilde{D}) \leq \text{length}(uLv)$ and $|\tilde{D}| \geq \deg_G(v)/\gamma$. By Lemma 26, with probability at least $1 - p_{26}(\ell)$, at the end of iteration $\tau + \ell$ of Outer Loop, the knowledge-graph also includes a size $\delta/(8\gamma^2)$ subset W of $\text{nadir}(\tilde{D})$. Since we assumed that $\delta/(8\gamma^2) \geq 2r^2 \cdot p^{r^2}$, by Lemma 18, the graph $\hat{G}$ includes a braid $\tilde{D}' \subset \tilde{D}$

such that $\mathrm{nadir}(\tilde{D}') \subset W$, $|\tilde{D}'| \geq 2r$ and the only vertices that are shared between the paths of $\tilde{D}'$ are u and v. Thus, $\tilde{D}'$ has a path uL^*v such that $\mathrm{length}(uL^*v) = \mathrm{length}(\tilde{D})$, $\mathrm{nadir}(uL^*v) \in W$, $V(L^*) \cap V(H) \subseteq \{\mathrm{nadir}(uL^*v)\}$. Consequently, H together with uL^*v includes a C_{2r+1}-subgraph H' of $\hat{G}$, such that $D \cup \{\mathrm{nadir}(uL^*v)\} \subseteq V(H')$. ◀

▶ **Theorem 28.** *Given oracle access to a C_{2r+1}-free graph, Algorithm 1 returns "ACCEPT". Let $p, r \in \mathbb{N}$ and $\epsilon \in \mathbb{R}^+$. There exist values for ξ_1, ξ_2 and ξ_3, that depend only on p,r and ϵ for which the following holds: on input p,r and ϵ, oracle access to a graph G and $|V(G)|$, Algorithm 1 computes the values of ξ_1, ξ_2 and ξ_3, and if G is (p, r) admissible and ϵ-far from being C_{2r+1}-free, with probability at least $2/3$, Algorithm 1 returns "REJECT". The query complexity of Algorithm 1 depends only on p,r and ϵ.*

Proof. The first part of the claim follows from the fact that Algorithm 1 returns "REJECT" only if its knowledge-graph is not C_{2r+1}-free. Since the knowledge-graph of Algorithm 1 is a subgraph of the input graph, then it will not return "REJECT" if the input graph is C_{2r+1}-free.

So, suppose that G is (p, r) admissible and ϵ-far from being C_{2r+1}-free. Let $\gamma = \lceil 16rp^r/\epsilon \rceil$, $\delta = 256r^2p^{r^2}\gamma^3$, $\xi_1 = \lceil 16\gamma/(\epsilon p) \rceil$, $\xi_2 = r^2 + r + 1$ and $\xi_3 = 12\gamma\delta r$. The last part of the claim follows from Lemma 21, since ξ_1, ξ_2 and ξ_3 depend only on p,r and ϵ.

Let $\hat{G}$ be the graph created by Trimming with parameters γ and δ. Since $\gamma > 8rp^r/\epsilon$, by Lemma 19, $\hat{G}$ is $\epsilon/2$-far from being C_{r+1}-free.

Since, $\hat{G}$ is $\epsilon/2$-far from being C_{2r+1}-free, by Lemma 24, with probability at least

$$1 - r(1 - \gamma^{-1})^{\xi_3} - (1 - \epsilon p/(2\gamma))^{\xi_1} > 19/20$$

(the inequality follows because $\xi_1 = \lceil 16\gamma/(\epsilon p) \rceil$ and $\xi_3 = 12\gamma\delta r$), at the end of iteration r of Outer Loop, the knowledge-graph includes a set of vertices D, such that for some C_{2r+1}-subgraph H of $\hat{G}$ we have $D \subseteq V(H)$ and every subpath of H that does not include a vertex from D has length at most r.

By Lemma 27, given that a set of vertices such as D is in the knowledge-graph at the end of iteration r of Outer Loop, with probability at least

$$1 - 2r \cdot \max\{r(1-\gamma^{-1})^{\xi_3}, (1-\delta^{-1})^{\xi_3} + (r-1)(1-\gamma^{-1})^{\xi_3}, r(\delta/(8\gamma^2))(1-\gamma^{-1})^{\xi_3} + e^{-\frac{\delta}{16\gamma^3}}\} > 19/20$$

(the inequality follows because $\gamma = \lceil 16rp^r/\epsilon \rceil$ and $\xi_3 = 12\gamma\delta r$), at the end of iteration $r^2 + r$ of Outer Loop, all the vertices of some C_{2r+1}-subgraph of $\hat{G}$ are in the knowledge-graph.

By Lemma 18, if at the end of iteration $r^2 + r$ of Outer Loop all the vertices of some C_{2r+1}-subgraph of $\hat{G}$ are in the knowledge-graph, then at the end of iteration $\xi_2 = r^2 + r + 1$ of Outer Loop, the knowledge-graph is not C_{2r+1}-free.

The previous three observations imply that, in this case, Algorithm 1 rejects with probability at least $2/3$. ◀

7 Lower bounds for testing C_{2r+1}-freeness in $(2, r-1)$-admissible graphs for $r \geq 2$

before and after neighbourhood, $N^-_{\mathbb{G}}(u)$, $N^+_{\mathbb{G}}(u)$, $\Delta^+(\mathbb{G})$, $\Delta^-(\mathbb{G})$

For this section we need some extra notation. For an ordered graph $\mathbb{G}$ we write $N^-_{\mathbb{G}}(u) := \{v \in N(u) \mid v <_{\mathbb{G}} u\}$ for the *before neighbourhood* and $N^+_{\mathbb{G}}(u) := \{v \in N(u) \mid v >_{\mathbb{G}} u\}$ for the *after neighbourhood* of a vertex $u \in \mathbb{G}$. We also use $\deg^-_{\mathbb{G}}(u) := |N^-_{\mathbb{G}}(u)|$ and $\deg^+_{\mathbb{G}}(u) := |N^+_{\mathbb{G}}(u)|$, as well as $\Delta^-(\mathbb{G}) := \max_{u \in \mathbb{G}} \deg^-_{\mathbb{G}}(u)$ and $\Delta^+(\mathbb{G}) := \max_{u \in \mathbb{G}} \deg^+_{\mathbb{G}}(u)$. We omit the subscript if the context implies it.

▶ **Theorem 29.** *For every integers $p, r \geq 2$ and sufficiently large $n \in \mathbb{N}$, every two-sided property tester for C_{2r+1}-freeness has query complexity $\Omega(n^{1/4})$, on $(2, r-1)$-admissible input graphs of size n.*

The following proofs are for testing C_4-freeness in $(2,1)$-admissible graphs. In the end, we explain how this implies the above theorem.

We prove the lower-bound theorems using Yao's minimax principle [13], which allows us to prove lower bounds for randomized property testers in the following manner.

In the theorem for the one-sided error case, we show that there exists a distribution $\mathcal{D}$ over input graphs that are $1/4$-far from C_4-freeness which further satisfy the following property: every deterministic one-sided property-tester for C_4-freeness—when given the parameters n, oracle access to a random graph picked from $\mathcal{D}$—with probability at least $2/3$ its knowledge-graph will not have a C_4-subgraph after using up to $n^{1/4}/16$ queries. Since a one-sided property tester can only accept if its knowledge-graph has a C_4-subgraph and must reject with probability at least $2/3$, the above implies that it has query complexity $\Omega(n^{1/4})$.

In the theorem for the two-sided error case, we design two distributions $\mathcal{D}_\mathcal{P}$, $\mathcal{D}_\mathcal{N}$ over $(2,1)$-admissible graphs. The support of $\mathcal{D}_\mathcal{P}$ is over C_4-free graphs, and the support of $\mathcal{D}_\mathcal{N}$ graphs is over graphs that are $1/6$-far being C_4-free. These distributions are constructed so that any two-sided deterministic error tester that uses $o(n^{1/4})$ queries, has a very low probability of computing which one of the two distributions is the origin of the input graph. If such a property tester rejects a graph from $\mathcal{D}_\mathcal{N}$ with probability at least $2/3$, then it accepts graphs from $\mathcal{D}_\mathcal{P}$ with probability strictly less than $2/3$.

We now define the graph T^n that is used to construct the distribution $\mathcal{D}$ and later, with some modifications, $\mathcal{D}_\mathcal{P}$ and $\mathcal{D}_\mathcal{N}$. For every $n \in \mathbb{N}$, we define T^n to be the graph that consists of the following vertices and edges: $V(T^n)$ is the union of two sets $Q = [\sqrt{n}]$ and $W = ([\sqrt{n}] \times [\sqrt{n}]) \setminus \{(i,i)\}_{i \in [\sqrt{n}]}$. $E(T^n)$ consists of all edges $\{\{u,(u,v)\} \mid u \in Q, (u,v) \in W\} \cup \{\{u,(v,u)\} \mid u \in Q, (v,u) \in W\}$. T^n

We sample a graph from $\mathcal{D}$ by applying a random permutation of the names of the vertices in Q and the same for the names of the in W. $\mathcal{D}$

▶ **Proposition 30.** *For every $n \in \mathbb{N}$, the graph T^n is $(2,1)$-admissible and $1/6$-far from C_4-freeness.*

Proof. To see that T^n is $(2,1)$-admissible, pick an arbitrary ordering $\mathbb{T}^n$ of the vertices of T^n where all the vertices of W are larger than all the vertices of Q. Clearly $\Delta^-(\mathbb{T}^n) \leq 2$, so every vertex in T^n participates in at most two 1-admissible paths.

For every pair of vertices $u, v \in Q$, T^n has exactly one C_4 that includes both vertices. We note that all these C_4 subgraphs are edge disjoint and $E(T^n)$ is the union of all their edges. So, to turn T^n into a C_4-free graph, at least $\binom{\sqrt{n}}{2}$ edges must be removed from T^n. Since $(2,1)$-admissible graphs have less than $2n$ edges, for a large enough value of n, T^n is $1/6$-far from being C_4-free. ◀

In order to simplify the proofs, we will make the following assumptions about the oracle: if the algorithm queries the identity of a vertex $v \in N(u)$ for $u \in Q$, then it also receives the identity of v's neighbour that is different from u (since $v \in W$ it has degree two) for the price of one query. Similarly, if it queries the neighbour of a vertex $v \in W$, it receives the identity of both of v's neighbours. Furthermore, we let the algorithm know all degrees (and thus the sets Q and W) in advance. Therefore, we can assume that the algorithm never asks for edges between vertices u, v if either $u, v \in Q$ or $u, v \in W$ and never uses degree queries. Thus, we may also assume that algorithm uses only adjacency queries between a vertex in w

and a vertex in Q. In the case that the algorithm asks such a query between the vertices $u \in W$ and $v \in Q$ and that uv exists, then we also give the algorithm the identity of the second neighbour of W. Since all these modified queries provide at least as much information as queries in the original setting, the lower bound naturally applies to the latter.

▶ **Theorem 31.** *For every sufficiently large integer $n \in \mathbb{N}$, every one-sided property tester for C_4-freeness of* adm_1*-graphs on n vertices has query complexity $\Omega(n^{1/4})$.*

Proof. Fix $\mathcal{T}$ to be an arbitrary deterministic one-sided property tester for C_4-freeness and suppose that it receives as input n, $p = 2$, $\epsilon > 0$ and oracle access to a random graph generated by $\mathcal{D}$.

For $i \in [t]$, let p_i be the probability that after i queries the knowledge-graph does not contain a C_4 subgraph conditioned on the event that the same holds after $i-1$ queries, where the probabilities are over the choice of the input graph according to $\mathcal{D}$. The probability that after t queries $\mathcal{T}$'s knowledge-graph does not contain a C_4 is $\prod_{i=1}^{t} p_i$. Thus, to conclude the proof, we only need to show that for $t \le n^{1/4}/16$ and every $i \in [t]$, we have $p_i \ge 1 - 2i/\sqrt{n}$, since this implies that

$$\prod_{i\in[t]} p_i \ge \prod_{i\in[t]} \left(1 - \frac{2i}{\sqrt{n}}\right) > \left(1 - \frac{2t}{\sqrt{n}}\right)^t > \left(1 - \frac{2t^2}{\sqrt{n}}\right) > 2/3.$$

Let G_{i-1} be the knowledge-graph after $i-1$ queries and suppose that G_{i-1} does not contain a C_4. Note that after a neighbour query at most 3 vertices are added to the knowledge-graph, one of degree 2 and two of degree $\sqrt{n}$. The same holds for adjacency queries. Therefore, G_{i-1} contains at most $2(i-1)$ vertices from the set Q and at most $i-1$ vertices of the set W.
Observe that for every $v \in V(G_i) \cap W$, all edges incident on v are in $E(G_i)$. So, the only type of queries that can result in G_i not being C_4-free are:

- a neighbour query to one of the vertices in $V(G_{i-1}) \cap Q$ that returns a vertex in $W \setminus V(G_{i-1})$ and a vertex in $V(G_{i-1}) \cap Q$,
- a neighbour query to a vertex in $W \setminus V(G_{i-1})$ that returns two vertices in $V(G_{i-1}) \cap Q$, and
- an adjacency query with a vertex in $V(G_{i-1}) \cap Q$ and a vertex in $W \setminus V(G_{i-1})$ that returns a vertex in $V(G_{i-1}) \cap Q$.

The probability that the first case does not occur is at least $1 - 2(i-1)/\sqrt{n}$. The probability that the second case does not occur is at least $1 - 4(i-1)^2/n$, which is strictly larger than $1 - 2i/\sqrt{n}$, when $i \le t$. Finally, the probability that the third case occurs is at least $1 - 2(i-1)/(n - \sqrt{n})$, which is strictly larger than $1 - 2i/\sqrt{n}$, when $i \le t$. ◀

T_N^n, T_P^n We now define the graphs T_P^n and T_N^n and the distributions $\mathcal{D}_{\mathcal{P}}$ and $\mathcal{D}_{\mathcal{N}}$. T_N^n consists of two disjoint copies of the graph T^n, and we distinguish the vertices of the copy with a prime (e.g. i' and W'). We construct T_P^n by first setting $T_P^n = T_N^n$ and then for every distinct $i, j \in [\sqrt{n}]$, we remove from T_P^n the edge $\{i, (i,j)\}$ and $\{i', (i', j')\}$ and add the edges $\{i, (i', j')\}$ and $\{i', (i,j)\}$. We sample a graph from $\mathcal{D}_{\mathcal{P}}$ by applying a random permutation of the names of the degree-$\sqrt{n}$ vertices and doing the same for the names of the degree-two vertices. The distribution $\mathcal{D}_{\mathcal{N}}$ is defined in the same way.

Note that here we also set Q to be the set of all degree $\sqrt{n}$ vertices and W to be the rest of the vertices. for a two sided-error algorithm we make the same assumptions on the queries as we did for a one-sided error algorithm.

▶ **Proposition 32.** *For every $n \in \mathbb{N}$, the graphs T_N^n and T_P^n are $(2,1)$-admissible. Moreover, T_N^n is $1/6$ far from C_4 freeness, and T_P^n is C_4 free.*

Proof. The proof of the first part of the claim follows from Proposition 30. For the second part, note that in T^n all the C_4 subgraphs are of the form $(i,(i,j),j,(j,i),i)$. Thus, when we constructed T_P^n from T_N^n, we "disconnected" all the C_4s that were in T_N^n, and we did not introduce any new C_4s. ◀

▶ **Theorem 33.** *For every sufficiently large integer $n \in \mathbb{N}$, every two-sided property tester for C_4-freeness of adm_1-bounded graphs on n vertices has query complexity $\Omega(n^{1/4})$.*

Proof. Fix $\mathcal{T}$ to be an arbitrary deterministic two-sided property tester for C_4-freeness and suppose that it receives as input n, $p = 2$, $\epsilon > 0$ and oracle access to a random graph generated by selecting u.a.r. one of the distributions $\mathcal{D}_\mathcal{P}$ and $\mathcal{D}_\mathcal{N}$ and then selecting a graph according to the chosen distribution.

Next, we show that, given that the input is selected from only one of the two distributions, with probability at least $9/10$ the answers that are vertices of degree $\sqrt{n}$ were drawn one by one u.a.r. without returns (after a vertex is drawn, it cannot be drawn again) from the vertices of degree $\sqrt{n}$. The same applies to vertices of degree 2.

This means that $\mathcal{T}$, with probability $9/10$, will behave exactly the same way, regardless of whether the input was drawn from $\mathcal{D}_\mathcal{P}$ or $\mathcal{D}_\mathcal{N}$. So, if given an input that is drawn from $\mathcal{D}_\mathcal{N}$, $\mathcal{T}$ rejects, with probability at least $2/3$, then with probability at least $2/3 - 1/10$ it rejects an input that is drawn from $\mathcal{D}_\mathcal{P}$, that is, it accepts such an input with probability strictly less than $2/3$. This is a contradiction to $\mathcal{T}$ being a property tester, since it must reject an input drawn from $\mathcal{D}_\mathcal{N}$ with probability at least $2/3$ and must accept an input drawn from $\mathcal{D}_\mathcal{P}$ with probability at least $2/3$.

We note that to prove this, we only need to show with probability at least $9/10$, every query $\mathcal{T}$ uses returns only vertices that are not already in its knowledge-graph. The proof follows the exact same lines as in the one-sided case. ◀

Fix r to be an integer larger than or equal to 2. To get the one-sided error lower bounds for C_{2r} freeness and C_{2r} freeness $(2, r-1)$-admissible for graphs work by creating a new graph T^n from the graph T^n as follows:

- For C_{2r}, for every vertex $(x,y) \in W$, where W is as in the construction of T^n, replace the edge between (x,y) and $\min\{x,y\}$ with a path of length $r-1$.
- For C_{2r+1}, for every vertex $(x,y) \in W$, where W is as in the construction of T^n, if $x < y$ replace the edge $\{x,(x,y)\}$ with a path of length r and otherwise replace the edge $\{y,(x,y)\}$ with a path of length $r-1$.

Note that the graph created according to the first item is $(2, r-1)$-admissible and has cycles of length $2r$ where the graph T^n had cycles of length 4. The graph created according to the second item is also $(2, r-1)$-admissible and has cycles of length $2r+1$ where the graph T^n had cycles of length 4.

To obtain the lower two-sided error lower bound for C_{2r} freeness and C_{2r+1} freeness of $(2, r-1)$ admissible graphs, apply the same changes used for originally constructing these graphs, to construct the new T_N^n and T_P^n from T^n with the following change: treat the paths that were added to T^n as if they were edges. Thus, Theorem 29 holds.

References

1 N. Alon, T. Kaufman, M. Krivelevich, and D. Ron. Testing triangle-freeness in general graphs. *SIAM Journal on Discrete Mathematics*, 22(2):786–819, 2008.

2 N. Alon, R. Yuster, and U. Zwick. Finding and counting given length cycles. *Algorithmica*, 17(3):209–223, 1997.

3 C. Awofeso, P. Greaves, O. Lachish, and F. Reidl. H-freeness testing in graphs of bounded r-admissibility. In *42nd International Symposium on Theoretical Aspects of Computer Science, STACS 2025, March 4–7, 2025, Jena, Germany*, volume 327 of *LIPIcs*. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2025.

4 A. Czumaj and C. Sohler. A characterization of graph properties testable for general planar graphs with one-sided error (it's all about forbidden subgraphs). In *2019 IEEE 60th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 1525–1548. IEEE, 2019.

5 M. Dalirrooyfard, T. D. Vuong, and V. V. Williams. Graph pattern detection: Hardness for all induced patterns and faster non-induced cycles. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, pages 1167–1178, 2019.

6 Z. Dvořák. Constant-factor approximation of the domination number in sparse graphs. *European Journal of Combinatorics*, 34(5):833–840, July 2013.

7 T. Eden, R. Levi, and D. Ron. Testing c_k-freeness in bounded-arboricity graphs. In K. Bringmann, M. Grohe, G. Puppis, and O. Svensson, editors, *51st International Colloquium on Automata, Languages, and Programming, ICALP 2024, July 8-12, 2024, Tallinn, Estonia*, volume 297 of *LIPIcs*, pages 60:1–60:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2024.

8 O. Goldreich and D. Ron. Property testing in bounded degree graphs. In *Proceedings of the twenty-ninth annual ACM symposium on Theory of computing*, pages 406–415, 1997.

9 D. B. Johnson. Finding all the elementary circuits of a directed graph. *SIAM Journal on Computing*, 4(1):77–84, 1975.

10 R. Levi. Testing triangle freeness in the general model in graphs with arboricity $O(\sqrt{n})$. In *48th International Colloquium on Automata, Languages, and Programming, ICALP 2021, July 12-16, 2021, Glasgow, Scotland (Virtual Conference)*, volume 198 of *LIPIcs*, pages 93:1–93:13. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021.

11 J. Nešetřil and P. Ossona de Mendez. *Sparsity: Graphs, Structures, and Algorithms*, volume 28 of *Algorithms and Combinatorics*. Springer, 2012.

12 V. Vassilevska and R. Williams. Finding, minimizing, and counting weighted subgraphs. In *Proceedings of the forty-first annual ACM symposium on Theory of computing*, pages 455–464, 2009.

13 A. C.-C. Yao. Probabilistic computations: Toward a unified measure of complexity. In *18th Annual Symposium on Foundations of Computer Science (sfcs 1977)*, pages 222–227. IEEE Computer Society, 1977.

14 X. Zhu. Colouring graphs with bounded generalized colouring number. *Discrete Mathematics*, 309(18):5562–5568, 2009.