Electronic Colloquium on Computational Complexity, Report No. 94 (2018)

Lower Bounds for Tolerant Junta and Unateness Testing via Rejection Sampling of Graphs

Amit Levi[*] Erik Waingarten[†]

May 2, 2018

Abstract

We introduce a new model for testing graph properties which we call the *rejection sampling model*. We show that testing bipartiteness of n-nodes graphs using rejection sampling queries requires complexity $\widetilde{\Omega}(n^2)$. Via reductions from the rejection sampling model, we give three new lower bounds for tolerant testing of Boolean functions of the form $f\colon \{0,1\}^n \to \{0,1\}$:

- Tolerant k-junta testing with *non-adaptive* queries requires $\widetilde{\Omega}(k^2)$ queries.
- Tolerant unateness testing requires $\widetilde{\Omega}(n)$ queries.
- Tolerant unateness testing with *non-adaptive* queries requires $\widetilde{\Omega}(n^{3/2})$ queries.

Given the $\widetilde{O}(k^{3/2})$-query non-adaptive junta tester of Blais [Bla08], we conclude that non-adaptive tolerant junta testing requires more queries than non-tolerant junta testing. In addition, given the $\widetilde{O}(n^{3/4})$-query unateness tester of Chen, Waingarten, and Xie [CWX17b] and the $\widetilde{O}(n)$-query non-adaptive unateness tester of Baleshzar, Chakrabarty, Pallavoor, Raskhodnikova, and Seshadhri [BCP$^+$17b], we conclude that tolerant unateness testing requires more queries than non-tolerant unateness testing, in both adaptive and non-adaptive settings. These lower bounds provide the first separation between tolerant and non-tolerant testing for a natural property of Boolean functions.

[*] University of Waterloo. Email: amit.levi@uwaterloo.ca.

[†] Columbia University. Email: eaw@cs.columbia.edu.

ISSN 1433-8092

Contents

1 Introduction

Over the past decades, property testing has emerged as an important line of research in sublinear time algorithms. The goal is to understand randomized algorithms for approximate decision making, where the algorithm needs to decide (with high probability) whether a huge object has some property by making a few queries to the object. Many different types of objects and properties have been studied from this property testing perspective (see the surveys by Ron [Ron08, Ron10] and the recent textbook by Goldreich [Gol17] for overviews of contemporary property testing research). This paper deals with property testing of Boolean functions $f\colon \{0,1\}^n \to \{0,1\}$ and property testing of graphs with vertex set $[n]$.

In this paper we describe a new model of graph property testing, which we call the *rejection sampling model.* For $n \in \mathbb{N}$ and a subset $\mathcal{P}$ of graphs on the vertex set $[n]$, we say a graph G on vertex set $[n]$ has property $\mathcal{P}$ if $G \in \mathcal{P}$ and say G is ε-far from having property $\mathcal{P}$ if all graphs $H \in \mathcal{P}$ differ on at least εn^2 edges[1]. The problem of ε-testing $\mathcal{P}$ with *rejection sampling queries* is the following task:

> Given some $\varepsilon > 0$ and access to an unknown graph $G = ([n], E)$, output "accept" with probability at least $\frac{2}{3}$ if G has property $\mathcal{P}$, and output "reject" with probability at least $\frac{2}{3}$ if G is ε-far from having property $\mathcal{P}$. The access to G is given by the following oracle queries: given a query set $L \subseteq [n]$, the oracle samples an edge $(\boldsymbol{i}, \boldsymbol{j}) \sim E$ uniformly at random and returns $\{\boldsymbol{i}, \boldsymbol{j}\} \cap L$.

We measure the complexity of algorithms with rejection sampling queries by considering the sizes of the queries. The complexity of an algorithm making queries $L_1, \dots, L_t \subset [n]$ is $\sum_{i=1}^{t} |L_i|$.

The rejection sampling model allows us to study testers which rely on random sampling of edges, while providing the flexibility of making lower-cost queries. This type of query access strikes a delicate balance between simplicity and generality: queries are constrained enough for us to show high lower bounds, and at the same time, the flexibility of making queries allows us to reduce the rejection sampling model to Boolean function testing problems. Specifically, we reduce to tolerant junta testing and tolerant unateness testing (see Subsection 1.1).

Our main result in the rejection sampling model is regarding *non-adaptive* algorithms. These algorithms need to fix their queries in advance and are not allowed to depend on answers to previous queries (in the latter case we say that the algorithm is *adaptive*). We show a lower bound on the complexity of testing whether an unknown graph G is bipartite using non-adaptive queries.

Theorem 1. *There exists a constant $\varepsilon > 0$ such that any non-adaptive ε-tester for bipartiteness in the rejection sampling model has cost $\widetilde{\Omega}(n^2)$*[2].

More specifically, Theorem 1 follows from applying Yao's principle to the following lemma.

Lemma 1.1. *Let $\mathcal{G}_1$ be the uniform distribution over the union of two disjoint cliques of size $n/2$, and let $\mathcal{G}_2$ be the uniform distribution over complete bipartite graphs with each part of size $n/2$.*

[1] The distance definition can be modified accordingly when one considers bounded degree or sparse graphs.

[2] We use the notations $\widetilde{O}, \widetilde{\Omega}$ to hide polylogarithmic dependencies on the argument, i.e. for expressions of the form $O(f \log^c f)$ and $\Omega(f/\log^c f)$ respectively (for some absolute constant c).

Any deterministic non-adaptive algorithm that can distinguish between $\mathcal{G}_1$ and $\mathcal{G}_2$ with constant probability using rejection sampling queries, must have complexity $\widetilde{\Omega}(n^2)$.

We discuss a number of applications of the rejection sampling model (specifically, of Lemma 1.1) in the next subsection. In particular, we obtain new lower bounds in the *tolerant testing framework* introduced by Parnas, Ron, and Rubinfeld in [PRR06] for two well-studied properties of Boolean functions (specifically, k-juntas and unateness; see the next subsection for definitions of these properties). These lower bounds are obtained by a reduction from the rejection sampling model; we show that too-good-to-be-true Boolean function testers for these properties imply the existence of rejection sampling algorithms which distinguish $\mathcal{G}_1$ and $\mathcal{G}_2$ with $\tilde{o}(n^2)$ complexity. Therefore, we may view the rejection sampling model as a useful abstraction in studying the hard instances of tolerant testing k-juntas and unateness.

1.1 Applications to Tolerant Testing: Juntas and Unateness

Given $n \in \mathbb{N}$ and a subset $\mathcal{P}$ of n-variable Boolean functions, a Boolean function $f\colon \{0,1\}^n \to \{0,1\}$ has property $\mathcal{P}$ if $f \in \mathcal{P}$. The distance between Boolean functions $f, g\colon \{0,1\}^n \to \{0,1\}$ is $\mathrm{dist}(f,g) = \mathbf{Pr}_{\boldsymbol{x}\sim\{0,1\}^n}[f(\boldsymbol{x}) \neq g(\boldsymbol{x})]$. The distance of f to the property $\mathcal{P}$ is $\mathrm{dist}(f,\mathcal{P}) = \min_{g\in\mathcal{P}} \mathrm{dist}(f,g)$. We say that f is ε-close to $\mathcal{P}$ if $\mathrm{dist}(f,\mathcal{P}) \leq \varepsilon$ and f is ε-far from $\mathcal{P}$ if $\mathrm{dist}(f,\mathcal{P}) > \varepsilon$. The problem of *tolerant property testing* [PRR06] of $\mathcal{P}$ asks for query-efficient randomized algorithms for the following task:

> Given parameters $0 \leq \varepsilon_0 < \varepsilon_1 < 1$ and black-box query access to a Boolean function $f\colon \{0,1\}^n \to \{0,1\}$, accept with probability at least $\frac{2}{3}$ if f is ε_0-close to $\mathcal{P}$ and reject with probability at least $\frac{2}{3}$ if f is ε_1-far from $\mathcal{P}$.

An algorithm which performs the above task is an $(\varepsilon_0,\varepsilon_1)$-tolerant tester for $\mathcal{P}$. A $(0,\varepsilon_1)$-tolerant tester is a *standard* property tester or a *non-tolerant* tester. As noted in [PRR06], tolerant testing is not only a natural generalization, but is also very often the desirable attribute of testing algorithms. This motivates the high level question: how does the requirement of being tolerant affect the complexity of testing the properties studied? We make progress on this question by showing query-complexity separations for two well-studied properties of Boolean functions: k-juntas, and unate functions.

- (k-junta) A function $f\colon \{0,1\}^n \to \{0,1\}$ is a *k-junta* if it depends on at most k of its variables, i.e., there exists k distinct indices $i_1, \dots i_k \in [n]$ and a k-variable function $g\colon \{0,1\}^k \to \{0,1\}$ where $f(x) = g(x_{i_1}, \dots, x_{i_k})$ for all $x \in \{0,1\}^n$.
- (unateness) A function $f\colon \{0,1\}^n \to \{0,1\}$ is *unate* if f is either non-increasing or non-decreasing in every variable. Namely, there exists a string $r \in \{0,1\}^n$ such that the function $f(x \oplus r)$ is monotone with respect to the bit-wise partial order on $\{0,1\}^n$.

The next theorem concerns non-adaptive tolerant testers for k-juntas.

Theorem 2. *For any $\alpha < 1$, there exists constants $0 < \varepsilon_0 < \varepsilon_1 < 1$ such that for any $k = k(n) \leq \alpha n$, any non-adaptive $(\varepsilon_0,\varepsilon_1)$-tolerant k-junta tester must make $\widetilde{\Omega}(k^2)$ queries.*

We give a noteworthy consequences of the Theorem 2. In [Bla08], Blais gave a non-adaptive $\widetilde{O}(k^{3/2})$-query tester for (non-tolerant) testing of k-juntas, which was shown to be optimal for non-adaptive algorithms by Chen, Servedio, Tan, Waingarten and Xie in [CST^{+}17]. Combined with Theorem 2, this shows a polynomial separation in the query complexity of non-adaptive tolerant junta testing and non-adaptive junta testing.

The next two theorems concern tolerant testers for unateness.

Theorem 3. *There exists constants $0 < \varepsilon_0 < \varepsilon_1 < 1$ such that any (possibly adaptive) $(\varepsilon_0, \varepsilon_1)$-tolerant unateness tester must make $\widetilde{\Omega}(n)$ queries.*

Theorem 4. *There exists constant $0 < \varepsilon_0 < \varepsilon_1 < 1$ such that any non-adaptive $(\varepsilon_0, \varepsilon_1)$-tolerant unateness tester must make $\widetilde{\Omega}(n^{3/2})$ queries.*

A similar separation in tolerant and non-tolerant testing occurs for the property of unateness as a consequence of Theorem 3 and Theorem 4. Recently, in [BCP^{+}17b], Baleshzar, Chakrabarty, Pallavoor, Raskhodnikova, and Seshadhri gave a non-adaptive $\widetilde{O}(n)$-query tester for (non-tolerant) unateness testing, and Chen, Waingarten and Xie [CWX17a] gave an (adaptive) $\widetilde{O}(n^{3/4})$-query tester for (non-tolerant) unateness testing. We thus, conclude that by Theorem 3 and Theorem 4, tolerant unateness testing is polynomially harder than (non-tolerant) unateness testing, in both adaptive and non-adaptive settings.

1.2 Related Work

The properties of k-juntas and unateness have received much attention in property testing research ([FKR^{+}04, CG04, Bla08, Bla09, BGSMdW13, STW15, CST^{+}17, BCE^{+}18] study k-juntas, and [GGL^{+}00, KS16, CS16, BCP^{+}17b, CWX17a, CWX17b] study unateness). We briefly review the current state of affairs in (non-tolerant) k-junta testing and unateness testing, and then discuss tolerant testing of Boolean functions and the rejection sampling model.

Testing k-juntas. The problem of testing k-juntas, introduced by Fischer, Kindler, Ron, Safra, and Samorodnitsky [FKR^{+}04], is now well understood up to poly-logarithmic factors. Chockler and Gutfreund [CG04] show that any tester for k-juntas requires $\Omega(k)$ queries (for a constant ε_1). Blais [Bla09] gave a junta tester that uses $O(k \log k + k/\varepsilon_1)$ queries, matching the bound of [CG04] up to a factor of $O(\log k)$ for constant ε_1. When restricted to non-adaptive algorithms, [FKR^{+}04] gave a non-adaptive tester making $\widetilde{O}(k^2/\varepsilon_1)$ queries, which was subsequently improved in [Bla08] to $\widetilde{O}(k^{3/2})/\varepsilon_1$. In terms of lower bounds, Buhrman, Garcia-Soriano, Matsliah, and de Wolf [BGSMdW13] gave a $\Omega(k \log k)$ lower bound for $\varepsilon = \Omega(1)$, and Servedio, Tan, and Wright [STW15] gave a lower bound which showed a separation between adaptive and non-adaptive algorithms for $\varepsilon_1 = \frac{1}{\log k}$. These results were recently improved in [CST^{+}17] to $\widetilde{\Omega}(k^{3/2}/\varepsilon_1)$, settling the non-adaptive query complexity of the problem up to poly-logarithmic factors.

Testing unateness. The problem of testing unateness was introduced alongside the problem of testing monotonicity in Goldreich, Goldwasser, Lehman, Ron, and Samorodnitsky [GGL^{+}00], where they gave the first $O(n^{3/2}/\varepsilon_1)$-query non-adaptive tester. Khot and Shinkar [KS16] gave the first improvement by giving a $\widetilde{O}(n/\varepsilon_1)$-query adaptive algorithm. A non-adaptive algorithm with $\widetilde{O}(n/\varepsilon_1)$ queries was given in [CC16, BCP$^+$17b]. Recently, [CWX17a, BCP$^+$17a] show that $\widetilde{\Omega}(n)$ queries are necessary for non-adaptive one-sided testers. Subsequently, [CWX17b] gave an adaptive algorithm testing unateness with query complexity $\widetilde{O}(n^{3/4}/\varepsilon_1^2)$. The current best lower bound for general adaptive testers appears in [CWX17a], where it was shown that any adaptive two-sided tester must use $\widetilde{\Omega}(n^{2/3})$ queries.

Tolerant testing. Once we consider tolerant testing, i.e., the case $\varepsilon_0 > 0$, the picture is not as clear. In the paper introducing tolerant testing, [PRR06] observed that standard algorithms whose queries are uniform (but not necessarily independent) are inherently tolerant to some extent. Nevertheless, achieving $(\varepsilon_0, \varepsilon_1)$-tolerant testers for constants $0 < \varepsilon_0 < \varepsilon_1$, can require applying different methods and techniques (see e.g, [GR05, PRR06, FN07, ACCL07, KS09, MR09, FR10, CGR13, BRY14, BMR16, Tel16]).

By applying the observation from [PRR06] to the unateness tester in [BCP$^+$17b], the tester accepts functions which are $O(\varepsilon_1/n)$-close to unate with constant probability. We similarly obtain weak guarantees for tolerant testing of k-juntas. Diakonikolas, Lee, Matulef, Onak, Rubinfeld, Servedio, and Wan [DLM$^+$07] observed that one of the (non-adaptive) junta testers from [FKR$^+$04] accepts functions that are poly$(\varepsilon_1, 1/k)$-close to k-juntas. Chakraborty, Fischer, Garcia-Soríano, and Matsliah [CFGM12] noted that the analysis of the junta tester of Blais [Bla09] implicitly implies an $\exp(k/\varepsilon_1)$-query complexity tolerant tester which accepts functions that are ε_1/c-close to some k-junta (for some constant $c > 1$) and rejects functions that are ε_1-far from every k-junta. Recently, Blais, Canonne, Eden, Levi and Ron [BCE$^+$18] showed that when required to distinguish between the cases that f is $\varepsilon_1/10$-close to a k-junta, or is ε_1-far from a $2k$-junta, poly$(k, 1/\varepsilon_1)$ queries suffice.

For general properties of Boolean functions, tolerant testing could be much harder than standard testing. Fischer and Fortnow [FF06] used PCPs in order to construct a property of Boolean functions $\mathcal{P}$ which is $(0, \varepsilon_1)$-testable with a constant number of queries (depending on ε_1), but any $(1/4, \varepsilon_1)$-tolerant test for $\mathcal{P}$ requires n^c queries for some $c > 0$. While [FF06] presents a strong separation between tolerant and non-tolerant testing, the complexity of tolerant testing of many natural properties remains open. We currently neither have a poly$(k, \frac{1}{\varepsilon_1})$-query tester which $(\varepsilon_0, \varepsilon_1)$-tests k-juntas, nor a poly$(n, \frac{1}{\varepsilon_1})$-query tester that $(\varepsilon_0, \varepsilon_1)$-tests unateness or monotonicity when $\varepsilon_0 = \Theta(\varepsilon_1)$.

Testing graphs with rejection sampling queries. Even though the problem of testing graphs with rejection sampling queries has not been previously studied, the model shares characteristics with previous studied frameworks. These include sample-based testing studied by Goldreich, Goldwasser, and Ron in [GGR98, GR16], where the oracle receives random samples from the input. One crucial difference between rejection sampling algorithms (which always query $[n]$) and sample-based testers is the fact that rejection sampling algorithms only receive *positive* examples (in the form of edges), as opposed to random positions in the adjacency matrix (which may be a *negative* example indicated the non-existence of an edge).

The rejection sampling model for graph testing also bears some resemblance to the conditional sampling framework for distribution testing introduced in Canonne, Ron, and Servedio, as well as Chakraborty, Fischer, Goldhirsh, and Matsliah [CRS15, CFGM16], where the algorithm specifies a query set and receives a sample conditioned on it lying in the query set.

1.3 Techniques and High Level Overview

We first give an overview of how the lower bound in the rejection sampling model (Lemma 1.1) implies lower bounds for tolerant testing of k-juntas and unateness, and then we give an overview of how Lemma 1.1 is proved.

Reducing Boolean Function Testing to Rejection Sampling This work should be considered alongside some recent works showing lower bounds for testing the properties of monotonicity, unateness, and juntas in the standard property testing model [BB16, CWX17a, CST$^+$17]. The lower bounds in [BB16, CWX17a] and [CST$^+$17] may be reinterpreted as following the same general paradigm. We discuss this general view next, followed by an overview of this work. At a high level, one may view the lower bounds from [BB16, CWX17a, CST$^+$17] as proceeding in three steps:

1. First, design a randomized indexing function $\mathbf{\Gamma}\colon \{0,1\}^n \to [N]$ that partitions the Boolean cube $\{0,1\}^n$ into roughly equal parts in a way compatible with the property (either monotonicity, unateness, or junta). We want to ensure that algorithms that make few queries cannot learn too much about $\mathbf{\Gamma}$, and that queries falling in the same part are close in Hamming distance.
2. Second, define two distributions over sub-functions $\boldsymbol{h}_i\colon \{0,1\}^n \to \{0,1\}$ for each $i \in [N]$. The hard functions are defined by $\boldsymbol{f}(x) = \boldsymbol{h}_{\mathbf{\Gamma}(x)}(x)$, so that one distribution corresponds to functions with the property, and the other distribution corresponds to functions far from the property.
3. Third, show that any testing algorithm for the property is actually solving some algorithmic task (determined by the distributions of $\boldsymbol{h}_i$) which is hard when queries are close in Hamming distance.

Belovs and Blais [BB16] used a construction of Talagrand [Tal96], known as the Talagrand function, to implement a randomized partition in a monotone fashion. The Talagrand function is a randomized DNF of $2^{\sqrt{n}}$ monotone terms of size $\sqrt{n}$, and one may define $\mathbf{\Gamma}\colon \{0,1\}^n \to [2^{\sqrt{n}}]$ to output the index of the first term of a Talagrand function which satisfies input $x \in \{0,1\}^n$. One can show that any two queries $z, z' \in \{0,1\}^n$ which are semi-balanced[3] with Hamming distance more than $\widetilde{\Omega}(n^{3/4})$ will fall in different parts with high probability. The sub-functions $\boldsymbol{h}_i\colon \{0,1\}^n \to \{0,1\}$ are then given by random dictators or random anti-dictators, so the algorithmic task is simple: determine whether the distribution over functions $\boldsymbol{h}_i$ is supported on dictators or anti-dictators when queries in the same part are at distance at most $\widetilde{O}(n^{3/4})$ from each other. An argument in the spirit of the one-sided error monotonicity lower bound from [FLN$^+$02] gives an $\Omega(n^{1/4})$ lower bound for monotonicity testing. [CWX17a] further refined the idea by designing improved randomized partitions $\mathbf{\Gamma}\colon \{0,1\}^n \to [N]$, which they called two-level Talagrand functions. The improved construction $\mathbf{\Gamma}$ partitions $\{0,1\}^n$ in a monotone fashion, but has the property that queries $z, z' \in \{0,1\}^n$ which are semi-balanced with Hamming distance $\widetilde{\Omega}(n^{2/3})$ fall into different parts with high probability, thus bringing the lower bound to $\widetilde{\Omega}(n^{1/3})$ using the same algorithmic task as [BB16].

[3] We will say $z \in \{0,1\}^n$ is *semi-balanced* if $|z| \approx \frac{n}{2} \pm \sqrt{n}$.

Higher lower bounds for unateness are possible because the unateness property allows for reductions to harder algorithmic tasks. Specifically, [CWX17a] consider the following algorithmic task: there are two classes of distributions supported on $[n] \times \{+,-\}$, and the task is to distinguish two classes with random samples. One class of distributions consists of the uniform distribution μ over $[n] \times \{+,-\}$, the other class of distributions is uniform over the support, but each μ satisfies the property that each $j \in [n]$ has either $\mu(j,+) = 0$ or $\mu(j,-) = 0$. Each sub-function $\boldsymbol{h}_i$ is specified by a random sample of μ, where $\boldsymbol{h}_i$ is a dictator in variable j if $(j,+)$ was sampled, and an anti-dictator in variable j if $(j,-)$ was sampled. The first key observation is that the distance of the functions $\boldsymbol{f}(x) = \boldsymbol{h}_{\boldsymbol{\Gamma}(x)}(x)$ from unateness, depends on whether μ comes from the first or second case. The second key observation is that multiple random samples are required to distinguish the two classes of distributions.[4]

For the case of k-juntas, [CST$^+$17] used a simple indexing function $\boldsymbol{\Gamma}\colon \{0,1\}^n \to [2^{n/2}]$ that partitions $\{0,1\}^n$ according to projections on randomly chosen $\frac{n}{2}$ variables. The second and third step also follows the above strategy. In their case, they define the SSSQ and SSEQ (for Set-Size-Set-Queries and Set-Size-Element-Queries) problems as the hard algorithmic task, which give the lower bounds.

Our lower bounds for tolerant testing follow the same paradigm. For the randomized indexing function, we use the construction from [CST$^+$17] for the junta lower bound and a Talagrand-based construction (similar to [CWX17a], but somewhat simpler) for the unateness lower bounds. The hard algorithmic task we embed is distinguishing between the distributions $\mathcal{G}_1$ and $\mathcal{G}_2$ with access to a rejection sampling oracle.

At a high level, our reductions show that the class of functions which are close to k-juntas and the class of functions which are close to unate have much richer structure than k-juntas and unate functions. In particular, the distance of the functions drawn from our hard distributions to k-junta and unateness will depend on a global parameter of an underlying graph used to define the functions[5]. Thus, tolerant testing algorithms for k-juntas and unateness must explore the relationships between different variables to gain some information about the underlying graph. This lies in stark contrast to the algorithms of [Bla08], [CWX17b], and [BCP$^+$17b] which test k-juntas (non-adaptively) and unateness, since these three algorithms treat the variables independently.

The distributions and the reductions themselves are quite involved, so we defer a high level overview of the reductions to those corresponding sections (Sections 4 and 5).

Distinguishing $\mathcal{G}_1$ and $\mathcal{G}_2$ with Rejection Sampling Queries In order to prove Lemma 1.1, one needs to rule out any deterministic non-adaptive algorithm which distinguishes between $\mathcal{G}_1$ and $\mathcal{G}_2$ with rejection sampling queries of complexity $\widetilde{o}(n^2)$. In order to keep the discussion at a high level, we identify three possible "strategies" for determining whether an underlying graph is a complete bipartite graph, or a union of two disjoint cliques:

[4] For example, in order to distinguish whether a distribution μ belongs to the first or second class with one-sided error, an algorithm must observe two samples $(j,+)$ and $(j,-)$ from μ, which would indicate that μ is uniform over the whole set $[n] \times \{+,-\}$. In fact, the adaptive algorithm for unateness testing in [CWX17b] can be interpreted as one based on solving this algorithmic task with a "rejection sampling"-style oracle.

[5] The relevant graph parameter in k-juntas and unateness will be different. Luckily, both graph parameters will have gaps in their value depending on the distribution the graphs were drawn from (either $\mathcal{G}_1$ or $\mathcal{G}_2$). This allows us to reuse the work of proving Lemma 1.1 to obtain Theorem 2, Theorem 3, and Theorem 4.

1. One approach is for the algorithm to sample edges and consider the subgraph obtained from edges returned by the oracle. For instance, the algorithm may make all rejection sampling queries to be $[n]$. These queries are expensive in the rejection sampling model, but they guarantee that an edge from the graph will be observed. If the algorithm is lucky, and there exists a triangle in the subgraph observed, the graph must not be bipartite, so it must come from $\mathcal{G}_2$.

2. Another sensible approach is for the algorithm to forget about the structure of the graph, and simply view the distribution on the edges generated by the randomness in the rejection sampling oracle as a distribution testing problem. Suppose for simplicity that the algorithm makes rejection sampling queries $[n]$. Then, the corresponding distributions supported on edges from $\mathcal{G}_1$ and $\mathcal{G}_2$ will be $\Omega(1)$-far from each other, so a distribution testing algorithm can be used.

3. A third, more subtle, approach is for the algorithm to use the fact that $\mathcal{G}_1$ and $\mathcal{G}_2$ correspond to a complete bipartite graph and the union of two cliques, and extract knowledge about the non-existence of edges when making queries which return either $\emptyset$ or a single vertex. More specifically, an algorithm may query a random subset $L \subset [n]$ of size $\frac{n}{2}$. The subset L will be split among the two sides of the graph (in the case of $\mathcal{G}_1$ and $\mathcal{G}_2$), and when an edge sampled by the oracle is incident on only one vertex of L, the rejection sampling oracle will return this one vertex. At this point, the algorithm may extract some information about how L is divided in the underlying graph, and eventually distinguish between $\mathcal{G}_1$ and $\mathcal{G}_2$.

The three strategies mentioned above all fail to give $\widetilde{o}(n^2)$ rejection sampling algorithms. The first approach fails because with a budget of $\widetilde{o}(n^2)$, rejection sampling algorithms will observe subgraphs which consist of various trees of size at most $\log n$, thus we will not observe cycles. The second approach fails since the distributions are supported on $\Omega(n^2)$ edges, so distribution testing algorithms will require $\Omega(n)$ edges (which costs $\Omega(n^2)$) to distinguish between $\mathcal{G}_1$ and $\mathcal{G}_2$. Finally, the third approach fails since algorithms will only observe $o(n)$ responses from the oracle corresponding to lone vertices which will be split roughly evenly among the unknown parts of the graph, so these observations will not be enough to distinguish between $\mathcal{G}_1$ and $\mathcal{G}_2$.

Our lower bound rules out the three strategies sketched above when the complexity is $\widetilde{o}(n^2)$, and shows that if the above three strategies do not work (in any possible combination with each other as well), then no non-adaptive algorithm of complexity $\widetilde{o}(n^2)$ will work. The main technical challenge is to show that the above strategies are the *only* possible strategies to distinguish $\mathcal{G}_1$ and $\mathcal{G}_2$. In Section 6, we give a more detailed, yet still high-level discussion of the proof of Lemma 1.1.

Finally, the analysis of Lemma 1.1 is tight; there is a non-adaptive rejection sampling algorithm which distinguishes $\mathcal{G}_1$ and $\mathcal{G}_2$ with complexity $\widetilde{O}(n^2)$. The algorithm (based on the first approach mentioned above) is simple: make $\widetilde{O}(n)$ queries $L = [n]$, and if we observe an odd-length cycle, we output "$\mathcal{G}_1$", otherwise, output "$\mathcal{G}_2$".

2 Preliminaries

We use boldfaced letters such as $\mathbf{A}, \mathbf{M}$ to denote random variables. Given a string $x \in \{0,1\}^n$ and $j \in [n]$, we write $x^{(j)}$ to denote the string obtained from x by flipping the j-th coordinate. An edge along the j-th direction in $\{0,1\}^n$ is a pair (x, y) of strings with $y = x^{(j)}$. In addition, for $\alpha \in \{0,1\}$ we use the notation $x^{(j \to \alpha)}$ to denote the string x where the jth coordinate is set to α. Given $x \in \{0,1\}^n$ and $S \subseteq [n]$, we use $x|_S \in \{0,1\}^S$ to denote the projection of x on S. For a distribution $\mathcal{D}$ we write $\boldsymbol{d} \sim \mathcal{D}$ to denote an element d drawn according to the distribution. We sometimes write $a \approx b \pm c$ to denote $b - c \leq a \leq b + c$.

Throughout this paper, we extensively use a generalization of Chernoff bounds for *negatively correlated* random variables.

Definition 2.1. *Let $\mathbf{X}_1, \ldots, \mathbf{X}_n \in \{0,1\}$ be random variables. We say that $\mathbf{X}_1, \ldots, \mathbf{X}_n$ are* negatively correlated *if for all $I \subset [n]$ the following hold:*

$$\mathbf{Pr}\left[\forall i \in I \;:\; \mathbf{X}_i = 0\right] \leq \prod_{i \in I} \mathbf{Pr}\left[\mathbf{X}_i = 0\right]$$
$$\mathbf{Pr}\left[\forall i \in I \;:\; \mathbf{X}_i = 1\right] \leq \prod_{i \in I} \mathbf{Pr}\left[\mathbf{X}_i = 1\right] \ .$$

Theorem 5 (Theorem 1.16 from [Doe11])**.** *Let $\mathbf{X}_1, \ldots, \mathbf{X}_n$ be negatively correlated binary random variables. Let $a_1, \ldots, a_n \in [0,1]$ and $\mathbf{X} = \sum_{i=1}^{n} a_i \mathbf{X}_i$. Then, for $\delta \in [0,1]$,*

$$\mathbf{Pr}\left[\mathbf{X} \geq (1+\delta)\, \mathbf{E}\left[\mathbf{X}\right]\right] \leq \exp(-\delta^2\, \mathbf{E}[\mathbf{X}]/2)$$
$$\mathbf{Pr}\left[\mathbf{X} \leq (1-\delta)\, \mathbf{E}\left[\mathbf{X}\right]\right] \leq \exp(-\delta^2\, \mathbf{E}[\mathbf{X}]/3) \ .$$

In addition, some of our proofs will use *hyper-geometric* random variables. Consider a population of size N that consists of K objects of a special type. Suppose n objects are picked without replacement. Let $\mathbf{X}$ be a random variable that counts the number of special objects picked in the sample. Then, we say that $\mathbf{X}$ is a hyper-geometric random variable, and we denote $\mathbf{X} \sim \mathrm{HG}(N, K, n)$. These hyper-geometric random variables enjoy tight concentration inequities (which are similar to Chernoff type bounds).

Theorem 6 ([Hoe63])**.** *Let $\mathbf{X} \sim \mathrm{HG}(N, K, n)$ and $\mu = K/N$. Then for any $t > 0$*

$$\mathbf{Pr}\left[\mathbf{X} \leq (\mu - t)n\right] \leq \exp(-2t^2 n)$$
$$\mathbf{Pr}\left[\mathbf{X} \geq (\mu + t)n\right] \leq \exp(-2t^2 n) \ .$$

3 The Rejection Sampling Model

In this section, we define the rejection sampling model and the distributions over graphs we will use throughout this work. We define the rejection sampling model tailored to our specific application of proving Lemma 1.1.

Definition 3.1. *Consider two distributions, $\mathcal{G}_1$ and $\mathcal{G}_2$ supported on graphs with vertex set $[n]$. The problem of distinguishing $\mathcal{G}_1$ and $\mathcal{G}_2$ with a rejection sampling oracle aims to distinguish between the following two cases with a specific kind of query:*

- *Cases: We have an unknown graph $\mathbf{G} \sim \mathcal{G}_1$ or $\mathbf{G} \sim \mathcal{G}_2$.*
- *Rejection Sampling Oracle: Each query is a subset $L \subset [n]$; an oracle samples an edge $(\boldsymbol{j}_1, \boldsymbol{j}_2)$ from $\mathbf{G}$ uniformly at random, and the oracle returns $\boldsymbol{v} = \{\boldsymbol{j}_1, \boldsymbol{j}_2\} \cap L$. The complexity of a query L is given by $|L|$.*

We say a non-adaptive algorithm Alg for this problem is a sequence of query sets $L_1, \dots, L_q \subset [n]$, as well as a function $\text{Alg}\colon ([n] \cup ([n] \times [n]) \cup \{\emptyset\})^q \to \{\text{“}\mathcal{G}_1\text{”}, \text{“}\mathcal{G}_2\text{”}\}$. The algorithm sends each query to the oracle, and for each query L_i, the oracle responds $\boldsymbol{v}_i \in [n] \cup ([n] \times [n]) \cup \{\emptyset\}$, which is either a single element of $[n]$, an edge in $\mathbf{G}$, or $\emptyset$. The algorithm succeeds if:

$$\mathop{\mathbf{Pr}}_{\substack{\mathbf{G} \sim \mathcal{G}_1, \\ \boldsymbol{v}_1, \dots, \boldsymbol{v}_q}} [\text{Alg}(\boldsymbol{v}_1, \dots, \boldsymbol{v}_q) \text{ outputs “}\mathcal{G}_1\text{”}] - \mathop{\mathbf{Pr}}_{\substack{\mathbf{G} \sim \mathcal{G}_2, \\ \boldsymbol{v}_1, \dots, \boldsymbol{v}_q}} [\text{Alg}(\boldsymbol{v}_1, \dots, \boldsymbol{v}_q) \text{ outputs “}\mathcal{G}_1\text{”}] \geq \frac{1}{3}.$$

The complexity of Alg is measured by the sum of the complexity of the queries, so we let $\text{cost}(\text{Alg}) = \sum_{i=1}^{q} |L_i|$.

While our interest in this work is primarily on lower bounds for the rejection sampling model, an interesting direction is to explore upper bounds of various natural graph properties with rejection sampling queries. Our specific applications only require ruling out non-adaptive algorithms, but one may define adaptive algorithms in the rejection sampling model and study the power of adaptivity in this setting as well.

3.1 The Distributions $\mathcal{G}_1$ and $\mathcal{G}_2$

Let $\mathcal{G}_1$ and $\mathcal{G}_2$ be two distributions supported on graphs with vertex set $[n]$ defined as follows. Let $\mathbf{A} \subset [n]$ be a uniform random subset of size $\frac{n}{2}$.

$$\begin{aligned}
\mathcal{G}_1 &= \left\{ K_{\mathbf{A}} \cup K_{\overline{\mathbf{A}}} : \mathbf{A} \subset [n] \text{ random subset size } \frac{n}{2} \right\} \\
\mathcal{G}_2 &= \left\{ K_{\mathbf{A}, \overline{\mathbf{A}}} : \mathbf{A} \subset [n] \text{ random subset size } \frac{n}{2} \right\},
\end{aligned}$$

where for a subset A, K_A is the complete graph on vertices in A and $K_{A,\overline{A}}$ is the complete bipartite graph whose sides are A and $\overline{A}$.

4 Tolerant Junta Testing

In this section, we will prove that distinguishing the two distributions $\mathcal{G}_1$ and $\mathcal{G}_2$ using a rejection sampling oracle reduces to distinguishing two distributions $\mathcal{D}_{\text{yes}}$ and $\mathcal{D}_{\text{no}}$ over Boolean functions, where $\mathcal{D}_{\text{yes}}$ is supported on functions that are close to k-juntas and $\mathcal{D}_{\text{no}}$ is supported on functions that are far from any k-junta with high probability.

4.1 High Level Overview

We start by providing some intuition of how our constructions and reduction implement the plan set forth in Subsection 1.3 for the property of being a k-junta. We define two distributions supported on Boolean functions, $\mathcal{D}_{\text{yes}}$ and $\mathcal{D}_{\text{no}}$, so that functions in $\mathcal{D}_{\text{yes}}$ are ε_0-close to being k-juntas and functions in $\mathcal{D}_{\text{no}}$ are ε_1-far from being k-juntas (where ε_0 and ε_1 are appropriately defined constants and $k = \frac{3n}{4}$).

As mentioned in the introduction, our distributions are based on the indexing function used in [CST$^+$17]. We draw a uniform random subset $\mathbf{M} \subset [n]$ of size $n/2$ and our function $\mathbf{\Gamma} = \Gamma_{\mathbf{M}} \colon \{0,1\}^n \to [2^{n/2}]$ projects the points onto the variables in $\mathbf{M}$. Thus, it remains to define the sequence of functions $\mathbf{H} = (\boldsymbol{h}_i \colon \{0,1\}^n \to \{0,1\} : i \in [2^{n/2}])$.

We will sample a graph $\mathbf{G} \sim \mathcal{G}_1$ (in the case of $\mathcal{D}_{\text{yes}}$), and a graph $\mathbf{G} \sim \mathcal{G}_2$ (in the case of $\mathcal{D}_{\text{no}}$) supported on vertices in $\overline{\mathbf{M}}$. Each function $\boldsymbol{h}_i \colon \{0,1\}^n \to \{0,1\}$ is given by first sampling an edge $(\boldsymbol{j}_1, \boldsymbol{j}_2) \sim \mathbf{G}$ and letting $\boldsymbol{h}_i$ be a parity (or a negated parity) of the variables $x_{\boldsymbol{j}_1}$ and $x_{\boldsymbol{j}_2}$. Thus, a function $\boldsymbol{f}$ from $\mathcal{D}_{\text{yes}}$ or $\mathcal{D}_{\text{no}}$ will have all variables being relevant, however, we will see that functions in $\mathcal{D}_{\text{yes}}$ have a group of $\frac{n}{4}$ variables which can be eliminated efficiently[6].

We think of the sub-functions $\boldsymbol{h}_i$ defined with respect to edges from $\mathbf{G}$ as implementing a sort of *gadget*: the gadget defined with respect to an edge (j_1, j_2) will have the property that if $\boldsymbol{f}$ eliminates the variable j_1, it will be "encouraged" to eliminate variable j_2 as well. In fact, each time an edge $(\boldsymbol{j}_1, \boldsymbol{j}_2) \sim \mathbf{G}$ is used to define a sub-function $\boldsymbol{h}_i$, any k-junta $g \colon \{0,1\}^n \to \{0,1\}$ where variable $\boldsymbol{j}_1$ or $\boldsymbol{j}_2$ is irrelevant will have to change half of the corresponding part indexed by $\mathbf{\Gamma}$. Intuitively, a function $\boldsymbol{f} \sim \mathcal{D}_{\text{yes}}$ or $\mathcal{D}_{\text{no}}$ (which originally depends on all n variables) wants to eliminate its dependence of $n - k$ variables in order to become a k-junta. When $\boldsymbol{f}$ picks a variable $j \in \overline{\mathbf{M}}$ to eliminate (since variables in $\mathbf{M}$ are too expensive), it must change points in parts where the edge sampled is incident on j. The key observation is that when $\boldsymbol{f}$ needs to eliminate multiple variables, if $\boldsymbol{f}$ picks the variables j_1 and j_2 to eliminate, whenever a part samples the edge (j_1, j_2), the function changes the points in one part and eliminates two variables. Thus, $\boldsymbol{f}$ eliminates two variables by changing the same number of points when there are edges between j_1 and j_2.

At a high level, the gadgets encourage the function $\boldsymbol{f}$ to remove the dependence of variables within a group of edges, i.e., the closest k-junta will correspond to a function g which eliminates groups of variables with edges within each other and few outgoing edges. More specifically, if we wants to eliminate $\frac{n}{4}$ variables from $\boldsymbol{f}$, we must find a bisection of the graph $\mathbf{G}$ whose cut value is small; in the case of $\mathcal{G}_1$, one of the cliques will have cut value 0, whereas any bisection of a graph from $\mathcal{G}_2$ will have a high cut value, which makes functions in $\mathcal{D}_{\text{yes}}$ closer to $\frac{3n}{4}$-juntas than functions in $\mathcal{D}_{\text{no}}$.

The reduction from rejection sampling is straight-forward. We consider all queries which are indexed to the same part, and if two queries indexed to the same part differ on a variable j, then we the algorithm "explores" direction j. Each part $i \in [2^{n/2}]$ where some query falls in has a corresponding rejection sampling query L_i, which queries the variables explored by the Boolean function testing algorithm.

[6] We say that a variable is eliminated if we change the function to remove the dependence of the variable.

4.2 The Distributions $\mathcal{D}_{\text{yes}}$ and $\mathcal{D}_{\text{no}}$

The goal of this subsection is to define the two distributions $\mathcal{D}_{\text{yes}}$ and $\mathcal{D}_{\text{no}}$, supported over Boolean functions with n variables. Functions $f \in \mathcal{D}_{\text{yes}}$ will be *close* to being a k-junta (for $k = \frac{3n}{4}$) with high probability, and functions $f \sim \mathcal{D}_{\text{no}}$ will be *far* from any k-junta with high probability.

Distribution $\mathcal{D}_{\text{yes}}$ A function f from $\mathcal{D}_{\text{yes}}$ is generated from a tuple of three random variables, $(\mathbf{M}, \mathbf{A}, \mathbf{H})$, and we set $f = f_{\mathbf{M},\mathbf{A},\mathbf{H}}$. The tuple is drawn according to the following randomized procedure:

1. Sample a uniformly random subset $\mathbf{M} \subset [n]$ of size $m \overset{\text{def}}{=} \frac{n}{2}$. Let $N = 2^m$ and $\Gamma_{\mathbf{M}} : \{0,1\}^n \to [N]$ be the function that maps $x \in \{0,1\}^n$ to a number encoded by $x|_{\mathbf{M}} \in [N]$.
2. Sample $\mathbf{A} \subset \overline{\mathbf{M}}$ of size $\frac{n}{4}$ uniformly at random, and consider the graph $\mathbf{G}$ defined on vertices $[\overline{\mathbf{M}}]$ with $\mathbf{G} = K_{\mathbf{A}} \cup K_{\overline{\mathbf{A}}}$, i.e., $\mathbf{G}$ is a uniformly random graph drawn according to $\mathcal{G}_1$.
3. Define a sequence of N functions $\mathbf{H} = \{\boldsymbol{h}_i \colon \{0,1\}^n \to \{0,1\} : i \in [N]\}$ drawn from a distribution $\mathcal{E}(\mathbf{G})$. For each $i \in \{1, \dots, N/2\}$, we let $\boldsymbol{h}_i(x) = \bigoplus_{\ell \in \mathbf{M}} x_\ell$.

   For each $i \in \{N/2+1, \dots, N\}$, we will generate $\boldsymbol{h}_i$ independently by sampling an edge $(\boldsymbol{j}_1, \boldsymbol{j}_2) \sim \mathbf{G}$ uniformly at random, as well as a uniform random bit $\boldsymbol{r} \sim \{0,1\}$. We let

   $$\boldsymbol{h}_i(x) = x_{\boldsymbol{j}_1} \oplus x_{\boldsymbol{j}_2} \oplus \boldsymbol{r}.$$
4. Using $\mathbf{M}, \mathbf{A}$ and $\mathbf{H}$, define $f_{\mathbf{M},\mathbf{A},\mathbf{H}} = \boldsymbol{h}_{\Gamma_{\mathbf{M}}(x)}(x)$ for each $x \in \{0,1\}^n$.

Distribution $\mathcal{D}_{\text{no}}$ A function f drawn from $\mathcal{D}_{\text{no}}$ is also generated by first drawing the tuple $(\mathbf{M}, \mathbf{A}, \mathbf{H})$ and setting $f = f_{\mathbf{M},\mathbf{A},\mathbf{H}}$. Both $\mathbf{M}$ and $\mathbf{A}$ are drawn using the same procedure; the only difference is that the graph $\mathbf{G} = K_{\mathbf{A},\overline{\mathbf{A}}}$, i.e., $\mathbf{G}$ is a uniformly random graph drawn according to $\mathcal{G}_2$. Then $\mathbf{H} \sim \mathcal{E}(\mathbf{G})$ is sampled from the modified graph $\mathbf{G}$.

We let

$$k \overset{\text{def}}{=} \frac{3n}{4} \qquad \varepsilon_0 \overset{\text{def}}{=} \frac{1}{8} \qquad \varepsilon_1 \overset{\text{def}}{=} \frac{3}{16}.$$

Consider a fixed subset $M \subset [n]$ which satisfies $|M| = \frac{n}{2}$, and a fixed subset $A \subset \overline{M}$ which satisfies $|A| = \frac{n}{4}$. Let G be a graph defined over vertices in $\overline{M}$, and for any subsets $S_1, S_2 \subset \overline{M}$, let

$$E_G(S_1, S_2) = \left|\{(j_1, j_2) \in G : j_1 \in S_1, j_2 \in S_2\}\right|,$$

be the number of edges between sets S_1 and S_2. Additionally, we let

$$\chi(G) = \min\left\{ \frac{E_G(S,S) + E_G(S,\overline{S})}{E_G(\overline{M},\overline{M})} : S \subset \overline{M}, |S| \geq \frac{n}{4} \right\} \tag{1}$$

be the minimum fraction of edges adjacent to a set S of size at least $\frac{n}{4}$. The following lemma relates the distance of a function $\boldsymbol{f} = f_{M,A,\mathbf{H}}$ where $\mathbf{H} \sim \mathcal{E}(G)$ to being a k-junta to $\chi(G)$. We then apply this lemma to the graph in $\mathcal{D}_{\text{yes}}$ and $\mathcal{D}_{\text{no}}$ to show that functions in $\mathcal{D}_{\text{yes}}$ are ε_0-close to being k-juntas, and functions in $\mathcal{D}_{\text{no}}$ are ε_1-far from being k-juntas.

Lemma 4.1. *Let G be any graph defined over vertices in A. If $\boldsymbol{f} = f_{M,A,\mathbf{H}}$, where $\mathbf{H} \sim \mathcal{E}(G)$, then*

$$\frac{1}{4} \cdot \chi(G) - o(1) \leq \mathrm{dist}(\boldsymbol{f}, k\text{-Junta}) \leq \frac{1}{4} \cdot \chi(G) + o(1)$$

with probability at least $1 - o(1)$.

Proof: We first show that $\mathrm{dist}(\boldsymbol{f}, k\text{-Junta}) \leq \frac{1}{4} \cdot \chi(G) + o(1)$. Let $S \subset \overline{M}$ with $|S| \geq \frac{n}{4}$ be the subset achieving the minimum in (1), and consider the indicator random variables $\mathbf{X}_i$ for $i \in \{N/2+1, \dots, N\}$ defined as:

$$\mathbf{X}_i = \begin{cases} 1 & \boldsymbol{h}_i(x) = x_{j_1} \oplus x_{j_2} \oplus r \text{ with } j_1 \in S \text{ or } j_2 \in S \\ 0 & \text{otherwise} \end{cases},$$

and note that the variables $\mathbf{X}_i$ are independent and equal 1 with probability $\chi(G)$. Consider the function $\boldsymbol{g} \colon \{0,1\}^n \to \{0,1\}$ is defined as:

$$\boldsymbol{g}(x) = \begin{cases} \boldsymbol{h}_{\Gamma_M(x)}(x) & \mathbf{X}_{\Gamma_M(x)} = 0 \\ 0 & \text{otherwise} \end{cases}.$$

Note that the function $\boldsymbol{g}$ is a k-junta, since $\boldsymbol{g}$ only depends on variables in $[n] \setminus S$, and $|S| \geq \frac{n}{4}$. In addition, we have that:

$$\mathrm{dist}(\boldsymbol{f}, k\text{-Junta}) \leq \mathrm{dist}(\boldsymbol{f}, \boldsymbol{g}) = \frac{1}{2^n} \sum_{i=N/2+1}^{N} \frac{2^{n-m}}{2} \cdot \mathbf{X}_i = \frac{1}{2 \cdot 2^m} \sum_{i=N/2+1}^{N} \mathbf{X}_i,$$

and by a Chernoff bound, we obtain the desired upper bound.

For the lower bound, let $T \subset [n]$ of size $\frac{n}{4}$. We divide the proof into two cases: 1) $M \cap T \neq \emptyset$, and 2) $M \cap T = \emptyset$.

We handle the first case first, and let $j \in M \cap T$.

- Suppose j is the highest order bit of M, so that $\Gamma_M(x^{(j \to 0)}) \in \{1, \dots, N/2\}$ and $\Gamma_M(x^{(j \to 1)}) \in \{N/2+1, \dots, N\}$. For $y \in \{0,1\}^{M \setminus \{j\}}$ and $\alpha \in \{0,1\}$, let $X_{y,\alpha} = \{x \in \{0,1\}^n : x_{|M \setminus \{j\}} = y, x_j = \alpha\}$, $X_y = X_{y,0} \cup X_{y,1}$. For every $x \in X_y$,

  $$\boldsymbol{f}(x) = \begin{cases} \bigoplus_{i \in M} x_i & x_j = 0 \\ x_{\boldsymbol{j}_1} \oplus x_{\boldsymbol{j}_2} \oplus \boldsymbol{r} & x_j = 1 \end{cases},$$

  for some $\boldsymbol{j}_1, \boldsymbol{j}_2 \in \overline{M}$ and $\boldsymbol{r} \in \{0,1\}$. Thus, for at least half of all points in $x \in X_{y,0}$, $\boldsymbol{f}(x) \neq \boldsymbol{f}(x^{(j)})$. Therefore, for any function $g \colon \{0,1\}^n \to \{0,1\}$ which does not depend on j, for each $x \in X_{y,0}$ where $\boldsymbol{f}(x) \neq \boldsymbol{f}(x^{(j)})$, either $\boldsymbol{f}(x) \neq g(x)$, or $\boldsymbol{f}(x^{(j)}) \neq g(x^{(j)})$, thus,

  $$\mathrm{dist}(\boldsymbol{f}, g) \geq \frac{1}{2^n} \sum_{y \in \{0,1\}^{M \setminus \{j\}}} \frac{1}{2} \cdot |X_{y,0}| \geq \frac{1}{4}.$$

- Suppose j is not the highest order bit of M. Then, if $\Gamma_M(x) \in \{1, \dots, N/2\}$, then $\Gamma_M(x^{(j)}) \in \{1, \dots, N/2\}$. We note that for each $y \in \{0,1\}^{M \setminus \{j\}}$ and $x \in X_{y,0}$ with $\Gamma_M(x) \in \{1, \dots, 2^{m-1}\}$, $\boldsymbol{f}(x) \neq \boldsymbol{f}(x^{(i)})$. Thus again, for any $g \colon \{0,1\}^n \to \{0,1\}$ which does not depend on j, $\mathrm{dist}(\boldsymbol{f}, g) \geq \frac{1}{4}$, since half of all points $x \in \{0,1\}^n$ satisfy $\Gamma_M(x) \in \{1, \dots, N/2\}$.

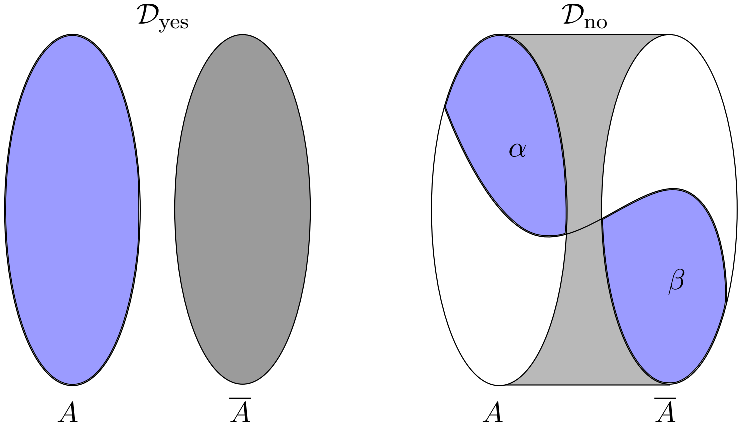

Figure 1: Example of graphs $\mathbf{G}$ from $\mathcal{D}_{\text{yes}}$ and $\mathcal{D}_{\text{no}}$. On the left, the graph $\mathbf{G}$ is the union of two cliques of size $\frac{n}{4}$, corresponding to $\mathcal{D}_{\text{yes}}$. We note that $\chi(G) = \frac{1}{2}$, since if we let $S = \mathbf{A}$ (pictured as the blue set), we see that S contains half of the edges. On the right, the graph $\mathbf{G}$ is the complete bipartite graph with side sizes $\frac{n}{4}$, corresponding to $\mathcal{D}_{\text{no}}$. We note that $\chi(G) = \frac{3}{4}$: consider any set $S \subset \overline{M}$ of size at least $\frac{n}{4}$ pictured in the blue region, and let $\alpha = |S \cap A|$ and $\beta = |S \cap \overline{A}|$, where $\alpha + \beta \geq \frac{n}{4}$, so $E(S, S) + E(S, \overline{S}) \geq (\frac{n}{4})^2 - \alpha\beta \geq (\frac{n}{4})^2(1 - \frac{1}{4})$.

Therefore, we may assume that $T \subset \overline{M}$. Again, consider the indicator random variables $\mathbf{X}_i$ for $i \in \{N/2 + 1, \dots, N\}$ given by

$$\mathbf{X}_i = \begin{cases} 1 & \boldsymbol{h}_i(x) = x_{j_1} \oplus x_{j_2} \oplus r \text{ with } j_1 \in T \text{ or } j_2 \in T \\ 0 & \text{otherwise} \end{cases},$$

and by the definition of $\chi(G)$, we have that $\mathbf{X}_i = 1$ with probability at least $\chi(G)$. Suppose $x \in \{0,1\}^n$ with $\Gamma_M(x) = i$ and $\mathbf{X}_i = 1$ with $\boldsymbol{h}_i(x) = x_{j_1} \oplus x_{j_2} \oplus r$ with $j_1 \in T$, then $\boldsymbol{f}(x) \neq \boldsymbol{f}(x^{(j_1)})$, which means that any function $g \colon \{0,1\}^n \to \{0,1\}$ which does not depend on variables in T, either $g(x) \neq \boldsymbol{f}(x)$ or $g(x^{(j_1)}) \neq \boldsymbol{f}(x^{(j_1)})$, thus, for all such functions g,

$$\text{dist}(\boldsymbol{f}, g) \geq \frac{1}{4 \cdot 2^{m-1}} \sum_{i=N/2+1}^{N} \mathbf{X}_i \geq \frac{1}{4} \cdot \chi(G) - \frac{1}{n}$$

with probability $1 - \exp\left(-\Omega(\frac{N}{n^2})\right)$ by a Chernoff bound. Thus, we union bound over at most $2^{n/2}$ possible subsets $T \subset \overline{M}$ with $|T| \geq \frac{n}{4}$ to conclude that $\text{dist}(\boldsymbol{f}, k\text{-Junta}) \geq \frac{1}{4} \cdot \chi(G) - \frac{1}{n}$ with probability $1 - o(1)$. ∎

Corollary 4.2. *We have that $\boldsymbol{f} \sim \mathcal{D}_{yes}$ has $\text{dist}(\boldsymbol{f}, k\text{-Junta}) \leq \varepsilon_0 + o(1)$ with probability $1 - o(1)$, and that $\boldsymbol{f} \sim \mathcal{D}_{no}$ has $\text{dist}(\boldsymbol{f}, k\text{-Junta}) \geq \varepsilon_1 - o(1)$ with probability $1 - o(1)$.*

Proof: For the upper bound in $\mathcal{D}_{\text{yes}}$, when $G = K_A \cup K_{\overline{A}}$, we have $\chi(G) \leq \frac{1}{2}$. For the lower bound in $\mathcal{D}_{\text{no}}$, when $G = K_{A,\overline{A}}$, $\chi(G) \geq \frac{3}{4}$ (see Figure 4.2). ∎

4.3 Reducing from Rejection Sampling

In this subsection, we will prove that distinguishing the two distributions $\mathcal{G}_1$ and $\mathcal{G}_2$ using rejection sampling oracle reduces to distinguishing the two distributions $\mathcal{D}_{\text{yes}}$ and $\mathcal{D}_{\text{no}}$.

Lemma 4.3. *Suppose there exists a deterministic non-adaptive algorithm* Alg *making q queries to Boolean functions $f\colon \{0,1\}^{2n} \to \{0,1\}$. Then, there exists a deterministic non-adaptive algorithm* Alg$'$ *making rejection sampling queries to an n-vertex graph such that:*

$$\Pr_{\boldsymbol{f}\sim\mathcal{D}_{yes}}[\mathrm{Alg}(\boldsymbol{f}) \text{ “accepts”}] = \Pr_{\mathbf{G}\sim\mathcal{G}_1}[\mathrm{Alg}'(\mathbf{G}) \text{ outputs “}\mathcal{G}_1\text{”}], \qquad \text{and}$$
$$\Pr_{\boldsymbol{f}\sim\mathcal{D}_{no}}[\mathrm{Alg}(\boldsymbol{f}) \text{ “accepts”}] = \Pr_{\mathbf{G}\sim\mathcal{G}_2}[\mathrm{Alg}'(\mathbf{G}) \text{ outputs “}\mathcal{G}_1\text{”}].$$

and has $\mathrm{cost}(\mathrm{Alg}') = O(q\log n)$ *with probability* $1 - o(1)$ *over the randomness in* Alg$'$.

Proof: Consider an algorithm Alg making q queries to a Boolean function $\boldsymbol{f} = f_{\mathbf{M},\mathbf{A},\mathbf{H}}\colon \{0,1\}^{2n} \to \{0,1\}$ (sampled from either $\mathcal{D}_{\text{yes}}$ or $\mathcal{D}_{\text{no}}$). First, note that $\mathbf{M}$ and $\mathbf{A}$ is distributed in the same way in $\mathcal{D}_{\text{yes}}$ and $\mathcal{D}_{\text{no}}$. Therefore, a rejection sampling algorithm may generate $\mathbf{M}$ and $\mathbf{A}$ and utilize its randomness from the rejection sampling oracle to simulate $\mathbf{H}$.

Specifically, given the queries $z_1, \dots, z_1 \in \{0,1\}^{2n}$ of Alg, we will partition them into sets $\mathbf{Q}_1, \dots, \mathbf{Q}_t$, such that for all $z, z' \in \mathbf{Q}_i$, we have that $z|_{\mathbf{M}} = z'|_{\mathbf{M}}$. Given the above partition, we define our queries to the rejection sampling oracle $\mathbf{L}_1, \dots, \mathbf{L}_t \subset \overline{\mathbf{M}}$ such that for every $i \in [t]$ we let

$$\mathbf{L}_i \overset{\text{def}}{=} \{j \in \overline{\mathbf{M}} : \exists z, z' \in \mathbf{Q}_i, (z)_j \neq (z')_j\} \ .$$

Since $|\overline{\mathbf{M}}| = n$, we may associate each element of $\overline{\mathbf{M}}$ with an integer in $[n]$ and view the graphs in $\mathcal{G}_1$ and $\mathcal{G}_2$ as having vertex set $\overline{\mathbf{M}}$. In short, we let $\mathbf{L}_i$ is the set of indices with two queries in $\mathbf{Q}_i$ disagreeing in that index. Next, we claim that the cost of Alg$'$ is at most $O(q\log n)$ with probability $1 - o(1)$.

Consider the bad event which occurs if there exist two queries $z, z' \in \{0,1\}^{2n}$ such that $z|_{\mathbf{M}} = z'|_{\mathbf{M}}$ and $\|z - z'\| > 100\log(2n)$. Note that for any two queries z, z' such that $\|z - z'\| > 100\log(2n)$, the probability that $z|_{\mathbf{M}} = z'|_{\mathbf{M}}$ over the choice of $\mathbf{M}$ is at most $2^{-100\log(2n)} \ll \frac{1}{q^2}$, and thus we may use a union bound over all pairs of queries to get that the bad event occurs with probability $o(1)$. Therefore, we get that for any $i \in [t]$ and two queries $z, z' \in \mathbf{Q}_i$ we have that $\|z - z'\| \leq 100\log(2n)$ with probability $1 - o(1)$, which implies that the cost of Alg$'$ is $O(q\log n)$ with probability $1 - o(1)$.

Now, given the responses to the queries $\mathbf{L}_1, \dots, \mathbf{L}_t \subset [\overline{\mathbf{M}}]$, as well as the values of $\mathbf{M}, \mathbf{A}$, we will be able to simulate all the randomness in the construction of the two distributions $\mathcal{D}_{\text{yes}}$ and $\mathcal{D}_{\text{no}}$. More formally, Alg$'$ works in the following way.

1. Alg$'$ makes set queries $\mathbf{L}_1, \dots, \mathbf{L}_t$.
2. Once Alg$'$ receives the responses $\boldsymbol{v}_1, \dots, \boldsymbol{v}_t \in \overline{\mathbf{M}} \cup \big(\overline{\mathbf{M}} \times \overline{\mathbf{M}}\big) \cup \{\emptyset\}$ from the oracle, it will generate a Boolean string $(\boldsymbol{r}_1, \dots, \boldsymbol{r}_q) \in \{0,1\}^q$ which is distributed exactly as $(f_{\mathbf{M},\mathbf{A},\mathbf{H}}(z_1), \dots, f_{\mathbf{M},\mathbf{A},\mathbf{H}}(z_q))$, where $f_{\mathbf{M},\mathbf{A},\mathbf{H}} \sim \mathcal{D}_{\text{yes}}$ if $\mathbf{G} \sim \mathcal{G}_1$ and $f_{\mathbf{M},\mathbf{A},\mathbf{H}} \sim \mathcal{D}_{\text{no}}$ if $\mathbf{G} \sim \mathcal{G}_2$.
3. Then if Alg$(\boldsymbol{r}_1, \dots, \boldsymbol{r}_q)$ outputs “accept”, then Alg$'$ should output “$\mathcal{G}_1$”, if Alg$(\boldsymbol{r}_1, \dots, \boldsymbol{r}_q)$ outputs “reject”, then Alg$'$ should output “$\mathcal{G}_2$”.

Next, we will describe how to generate $(\boldsymbol{r}_1, \dots, \boldsymbol{r}_q) \in \{0,1\}^q$. We start with setting some notations. For $i \in [t]$, we denote $\mathbf{Q}_i = \{z^i_1, \dots, z^i_{|\mathbf{Q}_i|}\}$ and $\boldsymbol{r}^i_1, \dots, \boldsymbol{r}^i_{|\mathbf{Q}_i|}$.

We aim to show that the random variables $(f_{\mathbf{M},\mathbf{A},\mathbf{H}}(z^i_\ell) : \ell \in [|\mathbf{Q}_i|], i \in [t])$ when $f_{\mathbf{M},\mathbf{A},\mathbf{H}} \sim \mathcal{D}_{\text{yes}}$ is distributed exactly the same as $(\boldsymbol{r}^i_\ell : \ell \in [|\mathbf{Q}_i|], i \in [t])$ when $\mathbf{G} \sim \mathcal{G}_1$ and $\boldsymbol{v}_1, \dots, \boldsymbol{v}_t$ are sampled by the oracle (the complement case where $f_{\mathbf{M},\mathbf{A},\mathbf{H}} \sim \mathcal{D}_{\text{no}}$ and $\mathbf{G} \sim \mathcal{G}_2$ is similar).

We will proceed in t stages, each in stage $i \in [t]$, we will set the values of $\boldsymbol{r}^i_1, \dots, \boldsymbol{r}^i_{|\mathbf{Q}_i|}$ which will correspond to $f_{\mathbf{M},\mathbf{A},\mathbf{H}}(z^i_1), \dots, f_{\mathbf{M},\mathbf{A},\mathbf{H}}(z^i_{|\mathbf{Q}_i|})$.

If $\mathbf{Q}_i$ contains strings z such that $\Gamma_{\mathbf{M}}(z) \in \{1, \dots, 2^{n-1}\}$ then we let $\boldsymbol{r}^i_1, \dots, \boldsymbol{r}^i_{|\mathbf{Q}_i|}$ be given by $\boldsymbol{r}^i_\ell = \bigoplus_{j \in \mathbf{M}} (z^i_\ell)_j$ for $\ell \in [|\mathbf{Q}_i|]$. Otherwise $\Gamma_{\mathbf{M}}(z) \in \{2^{n-1}+1, \dots, 2^n\}$, the algorithm will use the response $\boldsymbol{v}_i$ to generate the values $\boldsymbol{r}^i_1, \dots, \boldsymbol{r}^i_{|\mathbf{Q}_i|}$: Alg$'$ samples a random bit $\boldsymbol{r}^i \sim \{0,1\}$ uniformly and generates $\boldsymbol{r}^i_1, \dots, \boldsymbol{r}^i_{|\mathbf{Q}_i|}$ according to three cases, corresponding to the three cases $\boldsymbol{v}_i$ can be in:

- If $\boldsymbol{v}_i = \emptyset$, then $\boldsymbol{r}^i_1 = \dots = \boldsymbol{r}^i_{|\mathbf{Q}_i|} = \boldsymbol{r}^i$.
- If $\boldsymbol{v}_i = \{j\} \subset \overline{\mathbf{M}}$, for each $\ell \in [|\mathbf{Q}_i|]$, $\boldsymbol{r}^i_\ell = \boldsymbol{r}^i$ if $(z^i_\ell)_j = 0$, and $\boldsymbol{r}^i_\ell = 1 - \boldsymbol{r}^i$ if $(z^i_\ell)_j = 1$.
- If $\boldsymbol{v}_i = \{j_1, j_2\} \subset \overline{\mathbf{M}}$, for each $\ell \in [|\mathbf{Q}_i|]$, $\boldsymbol{r}^i_\ell = \boldsymbol{r}^i$ if $(z^i_\ell)_{j_1} \oplus (z^i_\ell)_{j_2} = 0$, and $\boldsymbol{r}^i_\ell = 1 - \boldsymbol{r}^i$ if $(z^i_\ell)_{j_1} \oplus (z^i_\ell)_{j_2} = 1$.

We conclude with the following claim which is immediate from the definition of $\mathcal{D}_{\text{yes}}$, $\mathcal{D}_{\text{no}}$, $\mathcal{G}_1$ and $\mathcal{G}_2$, and the corresponding proof simply unravels the definitions of these distributions.

Claim 4.4. *If $\mathbf{G} \sim \mathcal{G}_1$, then $(\boldsymbol{r}_1, \dots, \boldsymbol{r}_q)$ is distributed exactly as $(f_{\mathbf{M},\mathbf{A},\mathbf{H}}(z_1), \dots, f_{\mathbf{M},\mathbf{A},\mathbf{H}}(z_q))$ when $f_{\mathbf{M},\mathbf{A},\mathbf{H}} \sim \mathcal{D}_{yes}$, and if $\mathbf{G} \sim \mathcal{G}_2$, then $(\boldsymbol{r}_1, \dots, \boldsymbol{r}_q)$ is distributed exactly as $(f_{\mathbf{M},\mathbf{A},\mathbf{H}}(z_1), \dots, f_{\mathbf{M},\mathbf{A},\mathbf{H}}(z_q))$ when $f_{\mathbf{M},\mathbf{A},\mathbf{H}} \sim \mathcal{D}_{no}$.*

Proof: We give the formal proof for $\mathcal{D}_{\text{yes}}$ and $\mathcal{G}_1$, as the case with $\mathcal{D}_{\text{no}}$ and $\mathcal{G}_2$ is the same argumentation. Recall from the definition of $\mathcal{D}_{\text{yes}}$, that $\mathbf{M}$ and $\mathbf{A}$ are uniform random sets of size n and $\frac{n}{2}$ respectively. Conditioned on $\mathbf{M}$ and $\mathbf{A}$, each sub-function $\boldsymbol{h}_i$ is picked independently. Thus, we have

$$\begin{aligned}
&\mathop{\mathbf{Pr}}_{f_{\mathbf{M},\mathbf{A},\mathbf{H}} \sim \mathcal{D}_{\text{yes}}} \left[\forall i \in [t], \forall \ell \in [|\mathbf{Q}_i|], f_{\mathbf{M},\mathbf{A},\mathbf{H}}(z^i_\ell) = y^i_\ell \right] \\
&\qquad = \binom{2n}{n}^{-1} \binom{n}{n/2}^{-1} \sum_{M \subset [2n]} \sum_{A \subset \overline{M}} \prod_{i=1}^{t} \mathop{\mathbf{Pr}}_{\boldsymbol{h}_i} \left[\forall \ell \in [|Q_i|], \boldsymbol{h}_i(z^i_\ell) = y^i_\ell \mid \mathbf{M} = M, \mathbf{A} = A \right].
\end{aligned}$$

We now turn to the graph problem. Recall from the definition of $\mathbf{G} \sim \mathcal{G}_1$, that conditioned on $\mathbf{M}$ and $\mathbf{A}$, the responses of the oracle, $\boldsymbol{v}_1, \dots, \boldsymbol{v}_t$ are independent, and $\boldsymbol{r}^1, \dots, \boldsymbol{r}^t$ are independent. Thus, we may write:

$$\mathop{\mathbf{Pr}}_{\substack{\mathbf{M},\mathbf{A},\boldsymbol{v}_1,\dots,\boldsymbol{v}_t \\ \boldsymbol{r}^1,\dots,\boldsymbol{r}^t}} \left[\forall j \in [q], \forall \ell \in [|Q_i|], \boldsymbol{r}^i_\ell = y^i_\ell \right] = \binom{2n}{n}^{-1} \binom{n}{n/2}^{-1} \sum_{M} \sum_{A} \prod_{i=1}^{t} \mathop{\mathbf{Pr}}_{\boldsymbol{v}_i, \boldsymbol{r}^i} \left[\forall \ell \in [|Q_i|], \boldsymbol{r}^i_\ell = y^i_\ell \right].$$

Therefore, it suffices to show that for any $M \subset [2n]$ of size n, $A \subset \overline{M}$ of size $\frac{n}{2}$ and any $i \in [t]$, the random variable $(\boldsymbol{h}_i(z^i_\ell) : \ell \in [|Q_i|])$ with $\boldsymbol{h}_i$ from $\mathcal{D}_{\text{yes}}$ with sets M and A is distributed as $(\boldsymbol{r}^i_1, \dots, \boldsymbol{r}^i_{|Q_i|})$ with oracle response $\boldsymbol{v}_i$ and bit $\boldsymbol{r}^i$.

Let $(\boldsymbol{j}_1, \boldsymbol{j}_2)$ be a uniform random edge from $K_A \cup K_{\overline{A}}$, and we let $\boldsymbol{h}_i \colon \{0,1\}^{2n} \to \{0,1\}$ be given by:

$$\boldsymbol{h}_i(x) = \begin{cases} x_{\boldsymbol{j}_1} \oplus x_{\boldsymbol{j}_2} & \text{with probability } \frac{1}{2} \\ \neg x_{\boldsymbol{j}_1} \oplus x_{\boldsymbol{j}_2} & \text{with probability } \frac{1}{2} \end{cases}$$

Assume that $\boldsymbol{v}_i = L_i \cap \{\boldsymbol{j}_1, \boldsymbol{j}_2\} = \emptyset$. Then $\boldsymbol{h}_i(z_1^i) = \cdots = \boldsymbol{h}_i(z_{|Q_i|}^i)$ is given by a uniform random bit. Similarly, given these values of $\boldsymbol{v}_i = \emptyset$, $\boldsymbol{r}_1^i = \cdots = \boldsymbol{r}_{|Q_i|}^i$ is also given by a uniform random bit.

Now, assume that $L_i \cap \{\boldsymbol{j}_1, \boldsymbol{j}_2\} = \{\boldsymbol{j}\}$. Then, for any two queries $z, z' \in Q_i$ such that $(z)_{\boldsymbol{j}} \neq (z')_{\boldsymbol{j}}$ we must have that $\boldsymbol{h}_i(z) \neq \boldsymbol{h}_i(z')$, but after this condition is set, the value of any particular $\boldsymbol{h}_i(z)$ is a uniform random bit. Likewise, these constraints are set by the procedure generating $\boldsymbol{r}_1^i, \dots, \boldsymbol{r}_{|Q_i|}^i$, and each $\boldsymbol{r}_\ell^i$ is a uniform random bit.

Finally, assume that $L_i \cap \{\boldsymbol{j}_1, \boldsymbol{j}_2\} = \{\boldsymbol{j}_1, \boldsymbol{j}_2\}$. Then, for any two queries $z, z' \in Q_i$ such that $(z)_{\boldsymbol{j}_1} \oplus (z)_{\boldsymbol{j}_2} \neq (z')_{\boldsymbol{j}_1} \oplus (z')_{\boldsymbol{j}_2}$ we have that $\boldsymbol{h}_i(z) \neq \boldsymbol{h}_i(z')$, and each value of $\boldsymbol{h}_i(z)$ is a uniform random bit. Finally, these constraints are also set forth in the definition of $\boldsymbol{r}_1^i, \dots, \boldsymbol{r}_{|Q_i|}^i$. ■

■

Therefore, we conclude with the following corollary.

Corollary 4.5. *Suppose* Alg *is a deterministic non-adaptive algorithm which distinguishes $\mathcal{D}_{yes}$ and $\mathcal{D}_{no}$ supported on Boolean functions of $2n$ variables with query complexity q, then there exists a non-adaptive algorithm* Alg′ *for distinguishing between $\mathcal{G}_1$ and $\mathcal{G}_2$ supported on graphs with n vertices such that with probability $1 - o(1)$ over the randomness of* Alg′ *it holds that* $\text{cost}(\text{Alg}') = O(q \log n)$.

Proof: We have:

$$\begin{aligned} &\mathop{\mathbf{Pr}}_{\mathbf{G} \sim \mathcal{G}_1}[\text{Alg}'(\mathbf{G}) \text{ outputs "}\mathcal{G}_1\text{"}] - \mathop{\mathbf{Pr}}_{\mathbf{G} \sim \mathcal{G}_2}[\text{Alg}'(\mathbf{G}) \text{ outputs "}\mathcal{G}_1\text{"}] \\ &\qquad = \mathop{\mathbf{Pr}}_{f_{\mathbf{M},\mathbf{A},\mathbf{H}} \sim \mathcal{D}_{\text{yes}}}[\text{Alg}(\boldsymbol{f}) \text{ "accepts"}] - \mathop{\mathbf{Pr}}_{f_{\mathbf{M},\mathbf{A},\mathbf{H}} \sim \mathcal{D}_{\text{no}}}[\text{Alg}(\boldsymbol{f}) \text{ "accepts"}] \geq \frac{1}{3} - o(1). \end{aligned}$$

We also have that with probability at least $1 - o(1)$, for each $i \in [t]$, if $Q_i = \{z_1^i, \dots, z_{|Q_i|}^i\}$, then $|L_i| \leq \sum_{j=2}^{|Q_i|} \|z_1^i - z_j^i\|_1 \leq |Q_i| \cdot 100 \log(2n)$. Therefore, $\text{cost}(\text{Alg}') = \sum_{i=1}^t |L_i| = O(q \log n)$ with probability at least $1 - o(1)$. ■

5 Tolerant Unateness Testing

In this section, we show how to reduce distinguishing distributions $\mathcal{G}_1$ and $\mathcal{G}_2$ to distinguishing between Boolean functions which are close to unate and Boolean functions which are far from unate. We start with a high level overview of the constructions and reduction, and then proceed to give formal definitions and the reductions for adaptive and non-adaptive tolerant testing.

5.1 High Level Overview

We now describe how our constructions and reduction implement the plan set forth in Subsection 1.3 for the property of unateness. Similarly to Section 4, we define two distributions $\mathcal{D}_{\text{yes}}$ and $\mathcal{D}_{\text{no}}$

supported on Boolean functions, so that functions in $\mathcal{D}_{\text{yes}}$ are ε_0-close to being unate, and functions in $\mathcal{D}_{\text{no}}$ are ε_1-far from being unate (where ε_0 and ε_1 are appropriately defined constants).

We will use a randomized indexing function $\mathbf{\Gamma}\colon \{0,1\}^n \to [N]$ based on the Talagrand-style constructions from [BB16, CWX17a] to partition $\{0,1\}^n$ in a unate fashion. Again, we will then use a graph $\mathbf{G} \sim \mathcal{G}_1$ or $\mathcal{G}_2$ to define the sequence of sub-function $\mathbf{H} = (\boldsymbol{h}_i\colon \{0,1\}^n \to \{0,1\} : i \in [N])$. The sub-functions $\boldsymbol{h}_i$ will be given by a parity (or negated parity) of three variables: two variables will correspond to the end points of an edge sampled $(\boldsymbol{j}_1, \boldsymbol{j}_2) \sim \mathcal{G}$, the third variable will be one of two pre-specified variables, which we call m_1 and m_2. Consider for simplicity the case when $\boldsymbol{h}_i(x) = x_{\boldsymbol{j}_1} \oplus x_{\boldsymbol{j}_2} \oplus x_{m_1}$, and assume that we require that variable m_1 is non-decreasing.

Similarly to Section 4, the functions $\boldsymbol{h}_i$ are thought of as gadgets. We will have that if $\boldsymbol{h}_i$ is defined with respect to an edge (j_1, j_2) and m_1, then the function $\boldsymbol{f}$ will be "encouraged" to make variables j_1 and j_2 have opposite directions, i.e., either j_1 is non-increasing and j_2 is non-decreasing, or j_1 is non-decreasing and j_2 is non-increasing. In order to see why the three variable parity implements this gadget, we turn our attention to Figure 5.1 and Figure 5.1.

Intuitively, the function $\boldsymbol{f}$ needs to change some of its inputs to be unate, and it must choose whether the variables j_1 and j_2 will be monotone (non-decreasing) or anti-monotone (non-increasing). Suppose $\boldsymbol{f}$ decides that the variable j_1 should be monotone and j_2 be anti-monotone, and m_1 will always be monotone (since it will be too expensive to make it anti-monotone). Then, when $\boldsymbol{h}_i(x) = x_{j_1} \oplus x_{j_2} \oplus x_{m_1}$, $\boldsymbol{h}_i$ will have some *violating edges*, i.e., edges in direction j_1 which are decreasing, or edges in direction j_2 which are increasing, or edges in direction m_1 which are decreasing (see Figure 5.1, where these violating edges are marked in red). In this case, there exists a way that $\boldsymbol{f}$ may change $\frac{1}{4}$-th fraction of the points and remove all violating edges (again, this procedure is shown in Figure 5.1).

In contrast, suppose that $\boldsymbol{f}$ decides that the variables j_1 and j_2 both should be monotone. Then, when $\boldsymbol{h}_i(x) = x_{j_1} \oplus x_{j_2} \oplus x_{m_1}$, the violating edges (shown in Figure 5.1) form vertex-disjoint cycles of length 6 in $\{0,1\}^n$, thus, the function $\boldsymbol{f}$ will have to change $\frac{3}{8}$-th fraction of the points in order to remove all violating edges. In other words, when there is an edge (j_1, j_2) sampled in $\boldsymbol{h}_i$, the function $\boldsymbol{f}$ is "encouraged" to make j_1 and j_2 have opposite directions, and "discouraged" to make j_1 and j_2 have the same direction. The other cases are presented in Figures 5.2, 5.2, and 5.2.

In order for $\boldsymbol{f}$ to become unate, it must first choose whether each variable will be monotone or anti-monotone. $\boldsymbol{f}$ will choose all variables in $\mathbf{M}$ to be monotone, the variable m_1 to be monotone, and m_2 to be anti-monotone, but will have to make a choice for each variable in $\overline{\mathbf{M}}$, corresponding to each vertex of the graph $\mathbf{G}$. As discussed above, for each edge (j_1, j_2) in the graph, $\boldsymbol{f}$ is encouraged to make these orientations opposite from each other, so $\boldsymbol{f}$ will want to look for the maximum cut on the graph, whose value will be different in $\mathcal{G}_1$ and $\mathcal{G}_2$.

Similarly to the case in Section 4, the reduction will follow by defining the rejection sampling queries L_i corresponding to variables explored in sub-function $\boldsymbol{h}_i$. The unate indexing functions $\mathbf{\Gamma}$ are not as strong as the indexing functions from the Section 4, so for each query in the Boolean function testing algorithm, our reduction will lose some cost in the rejection sampling algorithm. In particular, the adaptive reduction loses n cost for each Boolean function query, since adaptive algorithms can efficiently explore variables with a binary search; this gives the $\widetilde{\Omega}(n)$ lower bound for tolerant unateness testing. The non-adaptive reduction loses $O(\sqrt{n}\log n)$ cost for each Boolean function

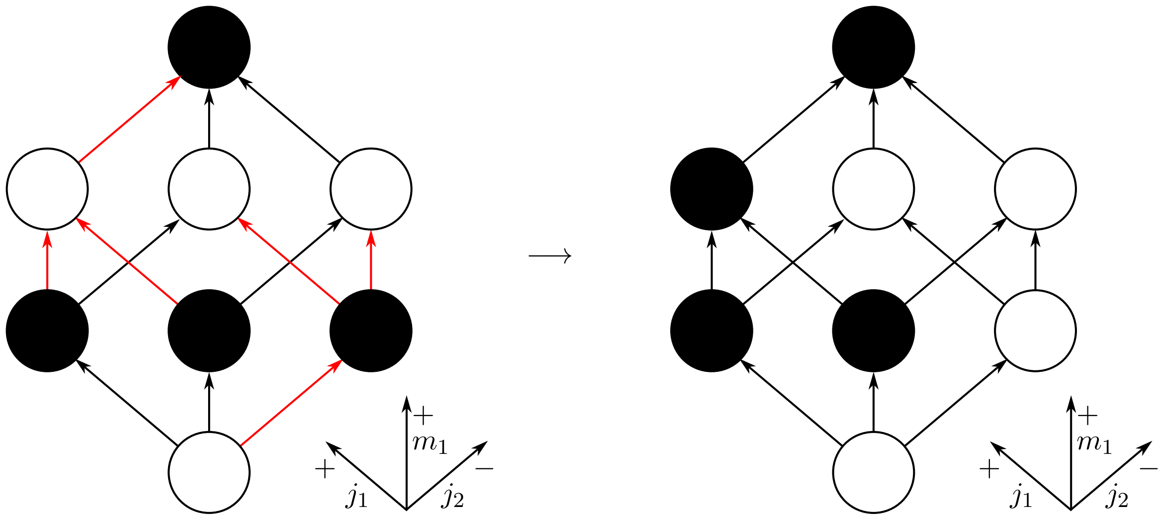

Figure 2: Example of a function $\boldsymbol{h}_i\colon \{0,1\}^n \to \{0,1\}$ with $\boldsymbol{h}_i(x) = x_{j_1} \oplus x_{j_2} \oplus x_{m_1}$ with variable j_1 (which ought to be monotone), j_2 (which ought to be anti-monotone), and m_1 (which is always monotone). The image on the left-hand side represents $\boldsymbol{h}_i$, and the red edges correspond to violating edges for variables j_1, j_2 and m_1. In other words, the red edges correspond to anti-monotone edges in variables j_1, monotone edges in variables j_2, and anti-monotone edges in direction m_1. On the right-hand side, we show how such a function can being "fixed" into a function $\boldsymbol{h}'_i\colon \{0,1\}^n \to \{0,1\}$ by changing $\frac{1}{4}$-fraction of the points.

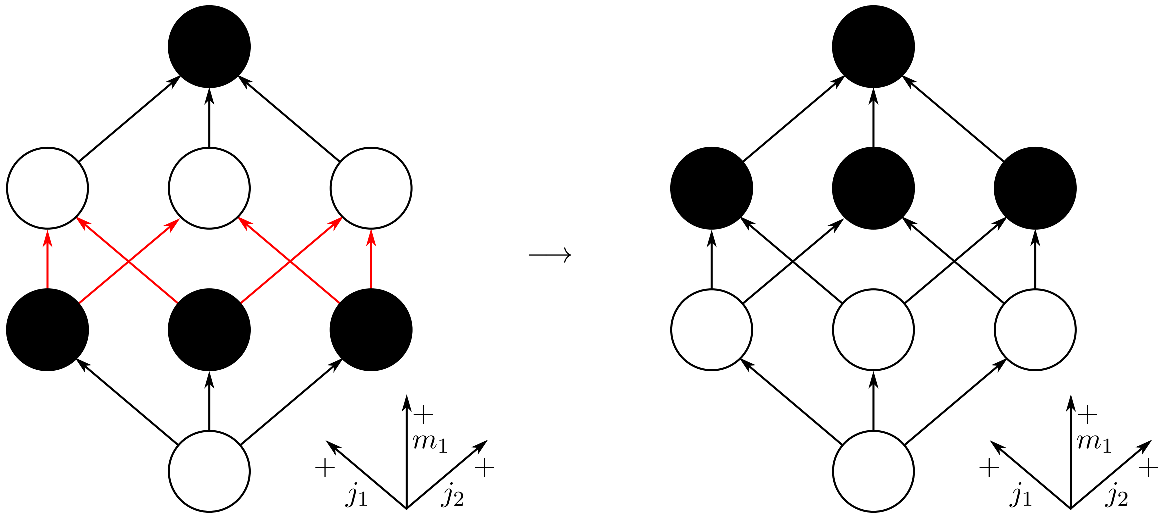

Figure 3: Example of a function $\boldsymbol{h}_i\colon \{0,1\}^n \to \{0,1\}$ with $\boldsymbol{h}_i(x) = x_{j_1} \oplus x_{j_2} \oplus x_{m_1}$ with variables j_1 and j_2 (which ought to be monotone), and m_1 (which ought to be monotone). On the left side, we indicate the violating edges with red arrows, and note that the functions in the left and right differ by $\frac{3}{8}$-fraction of the points. We also note that any function $\boldsymbol{h}'_i\colon \{0,1\}^n \to \{0,1\}$ which has j_1, j_2 and m_1 monotone must differ from $\boldsymbol{h}_i$ on at least $\frac{3}{8}$-fraction of the points because the violating edges of $\boldsymbol{h}_i$ form a cycle of length 6.

query since queries falling in the same part may be $\Omega(\sqrt{n})$ away from each other (the same scenario occurs in the non-adaptive monotonicity lower bound of [CWX17a]). The non-adaptive reduction is more complicated than the adaptive reduction since it is not exactly a black-box reduction (we require a lemma from Section 6). This gives the $\widetilde{\Omega}(n^{3/2})$ lower bound for non-adaptive tolerant unateness testing.

5.2 The Distributions $\mathcal{D}_{\text{yes}}$ and $\mathcal{D}_{\text{no}}$

We now turn to describing a pair of distributions $\mathcal{D}_{\text{yes}}$ and $\mathcal{D}_{\text{no}}$ supported on Boolean functions $f\colon \{0,1\}^n \to \{0,1\}$. These distributions will have the property that for some constants ε_0 and ε_1 with $0 < \varepsilon_0 < \varepsilon_1$,

$$\mathop{\mathbf{Pr}}_{\boldsymbol{f}\sim\mathcal{D}_{\text{yes}}}[\text{dist}(\boldsymbol{f}, \text{Unate}) \le \varepsilon_0] = 1 - o(1) \qquad \text{and} \qquad \mathop{\mathbf{Pr}}_{\boldsymbol{f}\sim\mathcal{D}_{\text{no}}}[\text{dist}(\boldsymbol{f}, \text{Unate}) \ge \varepsilon_1] = 1 - o(1).$$

We first define a function $\boldsymbol{f} \sim \mathcal{D}_{\text{no}}$, where we fix the parameter:

$$N = 2^{\sqrt{n}}.$$

1. Sample some set $\mathbf{M} \subset [n]$ of size $|\mathbf{M}| = \frac{n}{2}$ uniformly at random and let $\boldsymbol{m}_1, \boldsymbol{m}_2 \sim \mathbf{M}$ be two distinct indices.
2. We let $\mathbf{T} \sim \mathcal{E}(\mathbf{M}\backslash\{\boldsymbol{m}_1, \boldsymbol{m}_2\})$ (which we describe next). $\mathbf{T}$ is a sequence of terms $(\mathbf{T}_i : i \in [N])$ which is used to defined a multiplexer map $\Gamma_{\mathbf{T}}\colon \{0,1\}^n \to [N] \cup \{0^*, 1^*\}$.
3. We sample $\mathbf{A} \subset \overline{\mathbf{M}}$ of size $|\mathbf{A}| = \frac{n}{2}$ and define a graph as:
$$\mathbf{G} = K_{\mathbf{A}} \cup K_{\overline{\mathbf{A}}}.$$
4. We now define the distribution over sub-functions $\mathbf{H} = (\boldsymbol{h}_i : i \in [N]) \sim \mathcal{H}(\boldsymbol{m}_1, \boldsymbol{m}_2, \mathbf{G})$. For each function $\boldsymbol{h}_i\colon \{0,1\}^n \to \{0,1\}$, we generate $\boldsymbol{h}_i$ independently:
   - When $i \le 3N/4$, we sample $\boldsymbol{j} \sim \{\boldsymbol{m}_1, \boldsymbol{m}_2\}$ and we let:
   $$\boldsymbol{h}_i(x) = \begin{cases} x_{\boldsymbol{j}} & \boldsymbol{j} = \boldsymbol{m}_1 \\ \neg x_{\boldsymbol{j}} & \boldsymbol{j} = \boldsymbol{m}_2 \end{cases}.$$
   - Otherwise, if $i > 3N/4$, we sample an edge $(\boldsymbol{j}_1, \boldsymbol{j}_2) \sim \mathbf{G}$ and an index $\boldsymbol{j}_3 \sim \{\boldsymbol{m}_1, \boldsymbol{m}_2\}$ we let:
   $$\boldsymbol{h}_i(x) = \begin{cases} x_{\boldsymbol{j}_1} \oplus x_{\boldsymbol{j}_2} \oplus x_{\boldsymbol{j}_3} & \boldsymbol{j}_3 = \boldsymbol{m}_1 \\ \neg x_{\boldsymbol{j}_1} \oplus x_{\boldsymbol{j}_2} \oplus x_{\boldsymbol{j}_3} & \boldsymbol{j}_3 = \boldsymbol{m}_2 \end{cases}.$$

The function $\boldsymbol{f}\colon \{0,1\}^n \to \{0,1\}$ is given by $\boldsymbol{f}(x) = f_{\mathbf{T},\mathbf{A},\mathbf{H}}(x)$ where:

$$f_{\mathbf{T},\mathbf{A},\mathbf{H}}(x) = \begin{cases} 1 & |x_{|\mathbf{M}}| > \frac{n}{4} + \sqrt{n} \\ 0 & |x_{|\mathbf{M}}| < \frac{n}{4} - \sqrt{n} \\ 1 & \Gamma_{\mathbf{T}}(x) = 1^* \\ 0 & \Gamma_{\mathbf{T}}(x) = 0^* \\ \boldsymbol{h}_{\Gamma_{\mathbf{T}}(x)}(x) & \text{otherwise} \end{cases}. \tag{2}$$

We now turn to define the distribution $\mathcal{E}(M)$ supported on terms $\mathbf{T}$, as well as the multiplexer map $\Gamma_{\mathbf{T}}\colon \{0,1\}^n \to [N]$. As mentioned above, $\mathbf{T} \sim \mathcal{E}(M)$ will be a sequence of N terms $(\mathbf{T}_i : i \in [N])$, where each $\mathbf{T}_i$ is given by a DNF term:

$$\mathbf{T}_i(x) = \bigwedge_{j \in \mathbf{T}_i} x_j,$$

where the set $\mathbf{T}_i \subset M$ is a uniformly random $\sqrt{n}$-element subset. Given the sequence of terms $\mathbf{T}$, we let:

$$\Gamma_{\mathbf{T}}(x) = \begin{cases} 0^* & \forall i \in [N], \mathbf{T}_i(x) = 0 \\ 1^* & \exists i_1 \neq i_2 \in [N], \mathbf{T}_{i_1}(x) = \mathbf{T}_{i_2}(x) = 1 \\ i & \mathbf{T}_i(x) = 1 \text{ for a unique } i \in [N] \end{cases}.$$

It remains to define the distribution $\mathcal{D}_{\text{yes}}$ supported on Boolean functions. The function $\boldsymbol{f} \sim \mathcal{D}_{\text{yes}}$ will be defined almost exactly the same. We still have $\boldsymbol{f} = f_{\mathbf{T},\mathbf{A},\mathbf{H}}$ as defined above, however, the graph $\mathbf{G}$ will be different. In particular, we will let:

$$\mathbf{G} = K_{\mathbf{A},\overline{\mathbf{A}}}.$$

Fix any set $M \subset [n]$ of size $\frac{n}{2}$ and let $m_1, m_2 \in M$ be two distinct indices and $M' = M \setminus \{m_1, m_2\}$. For any $\mathbf{T} \sim \mathcal{E}(M')$, let $\mathbf{X} \subset \{0,1\}^n$ be the subset of points indexed to some subfunction $\boldsymbol{h}_i$:

$$\mathbf{X} \overset{\text{def}}{=} \left\{x \in \{0,1\}^n : |x_{|M}| \in [n/4 - \sqrt{n}, n/4 + \sqrt{n}] \text{ and } \Gamma_T(x) \in [N]\right\},$$

and define $\gamma \in (0,1)$ be the parameter:

$$\gamma \overset{\text{def}}{=} \mathop{\mathbf{E}}_{\mathbf{T} \sim \mathcal{E}(M')}\left[\frac{|\mathbf{X}|}{2^n}\right].$$

Claim 5.1. *With probability at least* $1 - \exp\left(-\Omega(N/n^2)\right)$ *over the draw* $\mathbf{T} \sim \mathcal{E}(M)$ *the set* $\mathbf{X}$ *has size* $|\mathbf{X}| = 2^n \gamma(1 \pm \frac{1}{n})$, *where* $\gamma = \Omega(1)$.

Proof: Note that:

$$\mathop{\mathbf{E}}_{\mathbf{T} \sim \mathcal{E}(M)}[|\mathbf{X}|] = \sum_{\substack{x \in \{0,1\}^n: \\ n/4 - \sqrt{n} \le |x_{|M}| \le n/4 + \sqrt{n}}} \mathop{\mathbf{Pr}}_{\mathbf{T} \sim \mathcal{E}(M)}[x \in \mathbf{X}].$$

Fix $x \in \{0,1\}^n$ such that $n/4 - \sqrt{n} \le |x_{|M}| \le n/4 + \sqrt{n}$. We can view the probability on the right hand side as a sequence of N disjoint events. Every event $j \in [N]$ correspond to the case where x satisfies the unique term $\mathbf{T}_j$. The probability of each such event is:

$$\begin{aligned} \mathop{\mathbf{Pr}}_{\mathbf{T} \sim \mathcal{E}(M)}[\Gamma_{\mathbf{T}}(x) = i] &\ge \left(\frac{1}{(n/2-2)^{\sqrt{n}}} \prod_{k=0}^{\sqrt{n}-1} (|x_{|M}| - k - 2)\right) \cdot \left(1 - \left(\frac{|x_{|M}|}{n/2-2}\right)^{\sqrt{n}}\right)^{N-1} \\ &\ge \left(\frac{n/4 - 2\sqrt{n}}{n/2}\right)^{\sqrt{n}} \cdot \left(1 - \left(\frac{n/4 + \sqrt{n}}{n/2-2}\right)^{\sqrt{n}}\right)^{N-1} = \Omega(1/N). \end{aligned}$$

Therefore, the probability that $x \in \mathbf{X}$ is at least $\Omega(1)$. Summing up all the x with $|x_{|M}| \approx \frac{n}{4} \pm \sqrt{n}$ gives $\mathbf{E}_{\mathbf{T}\sim\mathcal{E}(M)}[|\mathbf{X}|] = \Omega(2^n)$, so $\gamma = \Omega(1)$. In order to show that the random variable $|\mathbf{X}|$ is concentrated around the mean, let Ω be the space of all possible $\sqrt{n}$-sized terms with variables in $M \setminus \{m_1, m_2\}$, and let $c\colon \Omega^N \to \mathbb{Z}^{\geq 0}$ be the function on the independent terms which computes the size of $\mathbf{X}$:

$$c(\mathbf{T}_1, \dots, \mathbf{T}_N) = |\mathbf{X}|.$$

For every $j \in [N]$ and $T_1, \dots, T_N, T'_j \in \Omega$

$$\left|c(T_1, \dots, T'_j, \dots, T_N) - c(T_1, \dots, T_j, \dots, T_N)\right| \leq \frac{2^n}{N} \ ,$$

so by McDiarmid's inequality:

$$\Pr_{\mathbf{T}\sim\mathcal{E}(M)} \left[||\mathbf{X}| - \gamma 2^n| \geq 2^n/n \right] \leq \exp\left(-\frac{\Omega(2^{2n}/n^2)}{\sum_{i=1}^N 2^{2n}/N^2} \right) = \exp\left(-\Omega(N/n^2)\right) \ .$$

■

In addition, let $X_i \subset X$ be the subset of points $x \in X$ with $\Gamma_T(x) = i$, and note that the subsets $X_1, \dots, X_N$ partition X, where each $|X_i| \leq 2^{n-\sqrt{n}}$. With a similar argument as Claim 5.1, we conclude that with probability $1 - o(1)$ over the draw of $\mathbf{T} \sim \mathcal{E}(M)$, we have:

$$\sum_{i=1}^{3N/4} |\mathbf{X}_i| = 2^n \cdot \frac{3\gamma}{4}\left(1 \pm \frac{1}{n}\right) \qquad \text{and} \qquad \sum_{i=3N/4+1}^{N} |\mathbf{X}_i| = 2^n \cdot \frac{\gamma}{4}\left(1 \pm \frac{1}{n}\right). \tag{3}$$

Thus, we only consider functions $\boldsymbol{f} \sim \mathcal{D}_{\text{yes}}$ (or $\sim \mathcal{D}_{\text{no}}$) where the sets M, and T satisfy (3).

We consider any set $A \subset \overline{M}$ of size $\frac{n}{4}$. Now, consider any graph G defined over vertices in $\overline{M}$, and we let:

$$\chi(G) = \min\left\{ \frac{E_G(S,S) + E_G(\overline{S},\overline{S})}{E_G(\overline{M},\overline{M})} : S \subset \overline{M} \right\}.$$

In other words, we note that $\chi(G)$ is one minus the fractional value of the maximum cut, and the value of $\chi(G)$ is minimized for the set S achieving the maximum cut of G. The following lemma relates the distance to unateness of a function $\boldsymbol{f} = f_{T,A,\mathbf{H}}$ with $\mathbf{H} \sim \mathcal{H}(m_1, m_2, G)$, where G is an underlying graph defined on vertices in $\overline{M}$.

Lemma 5.2. *Let G be any graph defined over vertices in $\overline{M}$. If $\boldsymbol{f} = f_{T,A,\mathbf{H}}$ where $\mathbf{H} \sim \mathcal{H}(m_1, m_2, G)$, then*

$$\frac{\gamma}{16}\left(1 + \frac{1}{2} \cdot \chi(G)\right) - o(1) \leq \text{dist}(\boldsymbol{f}, \text{Unate}) \leq \frac{\gamma}{16}\left(1 + \frac{1}{2} \cdot \chi(G)\right) + o(1).$$

with probability $1 - o(1)$.

Proof: We first show that $\text{dist}(\boldsymbol{f}, \text{Unate}) \leq \frac{\gamma}{16}\left(1 + \frac{1}{2} \cdot \chi(G)\right) + o(1)$ with high probability. Consider the set $S \subset \overline{M}$ which achieves the minimum of $\chi(G)$, i.e.,

$$\chi(G) = \frac{E(S,S) + E(\overline{S},\overline{S})}{E(\overline{M},\overline{M})},$$

and let $\boldsymbol{g}\colon \{0,1\}^n \to \{0,1\}$ be the unate function which makes variables in M monotone, m_1 monotone, m_2 anti-monotone, S monotone, and $\overline{M} \setminus S$ anti-monotone. We defined $\boldsymbol{g}$ as follows:

$$\boldsymbol{g}(x) = \begin{cases} 1 & |x_{|M}| > \frac{n}{4} + \sqrt{n} \\ 0 & |x_{|M}| < \frac{n}{4} - \sqrt{n} \\ 1 & \Gamma_T(x) = 1^* \\ 0 & \Gamma_T(x) = 0^* \\ \boldsymbol{h}'_{\Gamma_T(x)}(x) & \text{otherwise} \end{cases},$$

where we define $\boldsymbol{h}'_i\colon \{0,1\}^n \to \{0,1\}$ as a Boolean function which depends on $\boldsymbol{h}_i$. In particular, if $i \leq 3N/4$, we let $\boldsymbol{h}'_i = \boldsymbol{h}_i$. Otherwise, suppose $\boldsymbol{h}_i$ is defined with respect to (j_1, j_2, j_3). There are two cases:

- (Directions of j_1 and j_2 disagree) If $j_1 \in S$ and $j_2 \notin S$, or $j_1 \notin S$ and $j_2 \in S$, then we let $\boldsymbol{h}'_i$ be the function on variables x_{j_1}, x_{j_2} and x_{j_3} with $\text{dist}(\boldsymbol{h}_i, \boldsymbol{h}'_i) = \frac{1}{4}$ (see Figure 5.1 for an example with $j_3 = m_1$ which needs to be monotone, $j_1 \in S$ and $j_2 \in \overline{S}$; Figure 5.2 and Figure 5.2 give the symmetric constructions when j_1 and j_2 are flipped, and when variable m_2 is used instead of m_1, respectively).
- (Directions of j_1 and j_2 agree) If $j_1 \in S$ and $j_2 \in S$, or $j_1 \notin S$ and $j_2 \notin S$, then we let $\boldsymbol{h}'_i$ be the function on variables x_{j_1}, x_{j_2} and x_{j_3} with $\text{dist}(\boldsymbol{h}_i, \boldsymbol{h}'_i) = \frac{3}{8}$ (see Figure 5.1 for an example with $j_3 = m_1$ which needs to be monotone, $j_1 \in S$ and $j_2 \in S$; Figure 5.2 gives the violating edges of the symmetric examples when variable m_2 is used, and either both j_1 and j_2 are monotone, or both anti-monotone).

Therefore, we define the indicator random variable $\mathbf{C}_i$ for each $i \in \{3N/4+1, \dots, N\}$ by

$$\mathbf{C}_i = \begin{cases} 1 & (\boldsymbol{j}_1, \boldsymbol{j}_2) \text{ from } \boldsymbol{h}_i \text{ is not cut by } S \\ 0 & \text{otherwise} \end{cases},$$

and we note that all $\mathbf{C}_i$ are independent and $\mathbf{Pr}_{\mathbf{H}}[\mathbf{C}_i] = \chi(G)$. By the two cases displayed above, we have that:

$$\text{dist}(\boldsymbol{f}, \boldsymbol{g}) = \frac{1}{2^n} \sum_{i=3N/4+1}^{N} |X_i| \left(\frac{1}{4} + \mathbf{C}_i \cdot \frac{1}{8} \right) \leq \frac{\gamma}{16} \left(1 + \frac{1}{2} \cdot \chi(G) \right) + o(1/n),$$

with probability at least $1 - \exp\left(-\Omega(N/n^2)\right)$ over the draw of all $\mathbf{C}_i$.

For the lower bound, consider any function $g\colon \{0,1\}^n \to \{0,1\}$ which is unate. Suppose variable x_{m_1} is anti-monotone in g, then let $\mathbf{C}_i$ for $i \in [3N/4]$ be the indicator random variable

$$\mathbf{C}_i = \begin{cases} 1 & \boldsymbol{h}_i(x) = x_{m_1} \\ 0 & \boldsymbol{h}_i(x) = \neg x_{m_2} \end{cases}.$$

We note that if $\mathbf{C}_i = 1$, then $\boldsymbol{f}$ and g differ on at least $|X_i|/2$ from X_i. Thus, we have $\text{dist}(\boldsymbol{f}, g) \geq \frac{3\gamma}{8}\left(1 - \frac{1}{n}\right) - o(1)$ with high probability over the draw of $\mathbf{C}_i$. Likewise, we may say that if x_{m_2} is monotone, then $\text{dist}(\boldsymbol{f}, g) \geq \frac{3\gamma}{8}\left(1 - \frac{1}{n}\right) - o(1)$. Thus, we may consider functions $g\colon \{0,1\}^n \to \{0,1\}$ with x_{m_1} being monotone and x_{m_2} being anti-monotone. In this case, consider a set $S \subset \overline{M}$, then

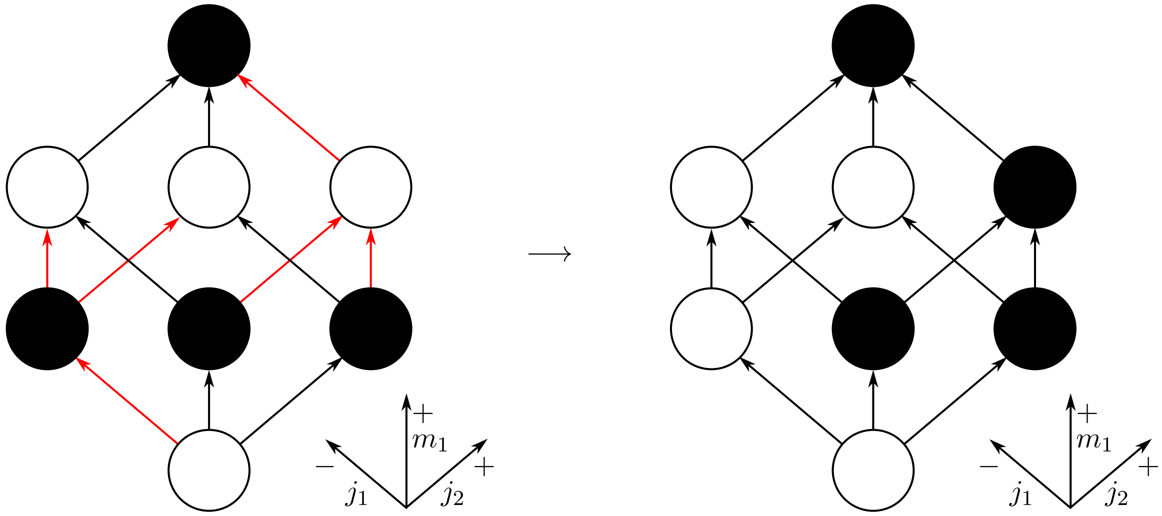

Figure 4: Similarly to Figure 5.1, this is an example of a function $\boldsymbol{h}_i\colon \{0,1\}^n \to \{0,1\}$ with $\boldsymbol{h}_i(x) = x_{j_1} \oplus x_{j_2} \oplus x_{m_1}$ variables j_1 (which ought to be anti-monotone), j_2 (which ought to be monotone), and m_1 (which is always monotone) being "fixed" into a function $\boldsymbol{h}'_i\colon \{0,1\}^n \to \{0,1\}$ defined on the right-hand side.

if g is any unate function with variables in S being monotone and variables in $\overline{M} \setminus S$ being anti-monotone, then we note that for each $i \in \{3N/4+1, \dots, N\}$, if $\boldsymbol{h}_i$ sampled an edge (j_1, j_2) which is cut by S, then X_i must differ on $\frac{1}{4}$th of the points in X_i (see Figure 5.1 for an example of the violating edges if j_1 and j_2 are oriented in opposite directions). On the other hand, if (j_1, j_2) is not cut by S, then X_i must differ on $\frac{3}{8}$ths of the points in X_i (see Figure 5.1 to see how the violating edges require $\frac{3}{8}$ths of the points being different). Thus, if we let the indicator random variable $\mathbf{C}_i$ be

$$\mathbf{C}_i = \left\{ \begin{array}{ll} 1 & (\boldsymbol{j}_1, \boldsymbol{j}_2) \text{ from } \boldsymbol{h}_i \text{ is not cut by } S \\ 0 & \text{otherwise} \end{array} \right. ,$$

we may write:

$$\mathrm{dist}(\boldsymbol{f}, g) \geq \frac{1}{2^n} \sum_{i=3N/4+1}^{N} |X_i| \left(\frac{1}{4} + \frac{1}{8} \cdot \mathbf{C}_i \right) \geq \frac{\gamma}{16} \left(1 + \frac{1}{2} \cdot \chi(G) \right) + O(1/n),$$

with probability $1 - \exp\left(-\Omega(N/n^2)\right)$ over the draw of $\mathbf{C}_i$, since $\mathbf{Pr}[\mathbf{C}_i = 1] \geq \chi(G)$. Thus, we may union bound over all $2^{n/2}$ subsets $S \subset \overline{M}$ to conclude the claim. ■

We consider the constants

$$\varepsilon_0 = \frac{\gamma}{16} \qquad \text{and} \qquad \varepsilon_1 = \frac{5\gamma}{64}.$$

Corollary 5.3. *We have that* $\boldsymbol{f} \sim \mathcal{D}_{yes}$ *has* $\mathrm{dist}(\boldsymbol{f}, \mathrm{Unate}) \leq \varepsilon_0 + o(1)$ *with high probability, and* $\boldsymbol{f} \sim \mathcal{D}_{no}$ *has* $\mathrm{dist}(\boldsymbol{f}, \mathrm{Unate}) \geq \varepsilon_1 - o(1)$ *with high probability.*

Proof: We simply note that when $\mathbf{G} = K_{\mathbf{A},\overline{\mathbf{A}}}$ (as is the case in $\mathcal{D}_{\text{yes}}$), we have $\chi(\mathbf{G}) = 0$, and when $\mathbf{G} = K_{\mathbf{A}} \cup K_{\overline{\mathbf{A}}}$, we have $\chi(\mathbf{G}) \to \frac{1}{2}$ as $n \to \infty$. ■

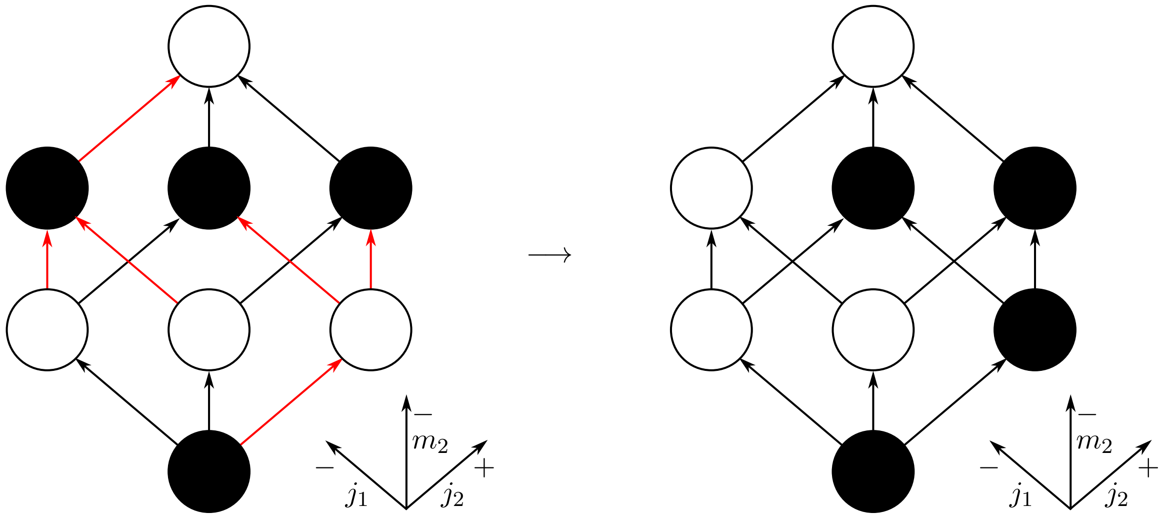

Figure 5: Similarly to Figure 5.1, this is an example of a function $\boldsymbol{h}_i \colon \{0,1\}^n \to \{0,1\}$ with $\boldsymbol{h}_i(x) = \neg x_{j_1} \oplus x_{j_2} \oplus x_{m_2}$ variables j_1 (which ought to be anti-monotone), j_2 (which ought to be monotone), and m_2 (which is always anti-monotone) being "fixed" into a function $\boldsymbol{h}'_i \colon \{0,1\}^n \to \{0,1\}$ defined on the right-hand side.

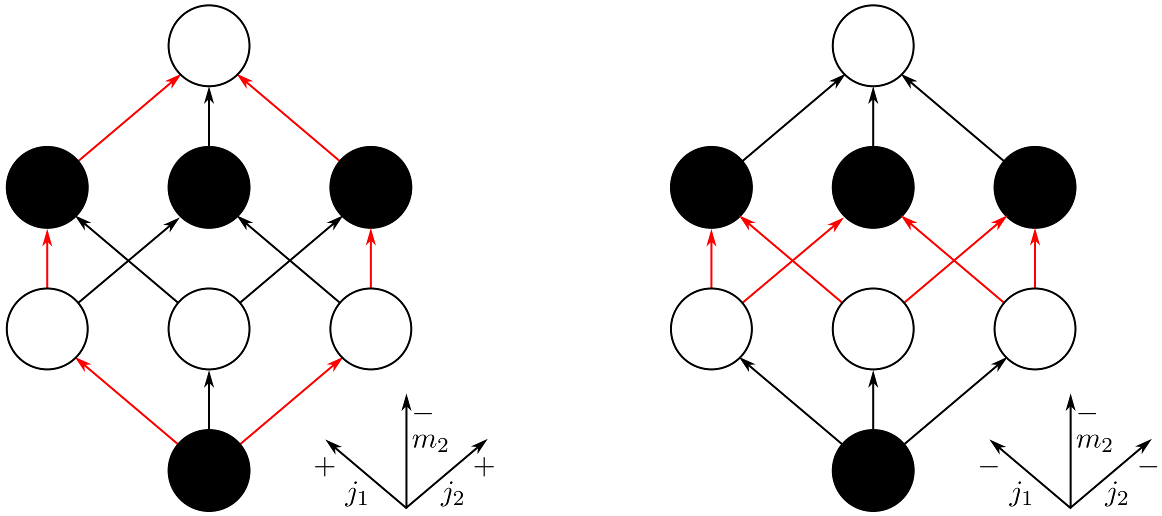

Figure 6: Examples of functions $\boldsymbol{h}_i \colon \{0,1\}^n \to \{0,1\}$ with orientations on the variables and violating edges. On the left-hand side, $\boldsymbol{h}_i(x) = \neg x_{j_1} \oplus x_{j_2} \oplus x_{m_2}$ with variables j_1 and j_2 (which ought to be monotone), and m_2 (which is always anti-monotone). On the right-hand side, $\boldsymbol{h}_i(x) = \neg x_{j_1} \oplus x_{j_2} \oplus x_{m_2}$ with variables j_1 and j_2 (which ought to be anti-monotone), and m_2 (which is always anti-monotone). We note that the violating edges form a cycle of length 6, so any unate function whose orientations on j_1 and j_2 are as indicated (both monotone on the left-hand side, and both anti-monotone on the right-hand side) must disagree on a $\frac{3}{8}$-fraction of the points.

5.3 Reducing from Rejection Sampling

The goal of this section is to prove the following two lemmas.

Lemma 5.4. *Suppose there exists a deterministic algorithm* Alg *making q queries to Boolean functions $f\colon \{0,1\}^{2n} \to \{0,1\}$. Then, there exists a deterministic non-adaptive algorithm* Alg$'$ *making rejection sampling queries to an n-vertex graph with* $\mathrm{cost}(\mathrm{Alg}') = qn$ *such that:*

$$\mathop{\mathbf{Pr}}_{\boldsymbol{f}\sim\mathcal{D}_{yes}}[\mathrm{Alg}(\boldsymbol{f}) \text{ ``accepts''}] = \mathop{\mathbf{Pr}}_{\mathbf{G}\sim\mathcal{G}_2}[\mathrm{Alg}'(\mathbf{G}) \text{ outputs ``}\mathcal{G}_2\text{''}], \qquad \text{and}$$
$$\mathop{\mathbf{Pr}}_{\boldsymbol{f}\sim\mathcal{D}_{no}}[\mathrm{Alg}(\boldsymbol{f}) \text{ ``accepts''}] = \mathop{\mathbf{Pr}}_{\mathbf{G}\sim\mathcal{G}_1}[\mathrm{Alg}'(\mathbf{G}) \text{ outputs ``}\mathcal{G}_2\text{''}].$$

Lemma 5.5. *Suppose there exists a deterministic non-adaptive algorithm* Alg *making q queries to Boolean functions $f\colon \{0,1\}^{2n} \to \{0,1\}$ where $q \leq \frac{n^{3/2}}{\log^8 n}$. Then, there exists a deterministic non-adaptive algorithm* Alg$'$ *making rejection sampling queries to an n-vertex graph such that:*

$$\mathop{\mathbf{Pr}}_{\boldsymbol{f}\sim\mathcal{D}_{yes}}[\mathrm{Alg}(\boldsymbol{f}) \text{ ``accepts''}] \approx \mathop{\mathbf{Pr}}_{\mathbf{G}\sim\mathcal{G}_2}[\mathrm{Alg}'(\mathbf{G}) \text{ outputs ``}\mathcal{G}_2\text{''}] \pm o(1), \qquad \text{and}$$
$$\mathop{\mathbf{Pr}}_{\boldsymbol{f}\sim\mathcal{D}_{no}}[\mathrm{Alg}(\boldsymbol{f}) \text{ ``accepts''}] \approx \mathop{\mathbf{Pr}}_{\mathbf{G}\sim\mathcal{G}_1}[\mathrm{Alg}'(\mathbf{G}) \text{ outputs ``}\mathcal{G}_2\text{''}] \pm o(1).$$

and has $\mathrm{cost}(\mathrm{Alg}') \leq q\sqrt{n}\log n$ *with probability* $1 - o(1)$ *over the randomness in* Alg$'$.

Combining Lemma 5.4 with Theorem 1, we conclude Theorem 3, and combining Lemma 5.5 with Theorem 1, we conclude Theorem 4.

5.4 Proof of Lemma 5.4

Consider an algorithm Alg making q queries to a Boolean function which receives access to a Boolean function $\boldsymbol{f} = f_{\mathbf{T},\mathbf{A},\mathbf{H}}\colon \{0,1\}^{2n} \to \{0,1\}$ (sampled from either $\mathcal{D}_{\mathrm{yes}}$ or $\mathcal{D}_{\mathrm{no}}$).

Since the values of $\mathbf{M}, \boldsymbol{m}_1, \boldsymbol{m}_2$ and $\mathbf{T}$ are distributed in the same way in $\mathcal{D}_{\mathrm{yes}}$ and $\mathcal{D}_{\mathrm{no}}$, a rejection sampling algorithm may generate $\mathbf{M}, \boldsymbol{m}_1, \boldsymbol{m}_2$ and $\mathbf{T}$, and utilize the randomness from rejection sampling to output values of $\mathbf{H}$. In particular, for each query in Alg, we will query the set $[n]$ in the rejection sampling algorithm. Then, given the edges sampled, as well as the values of $\mathbf{M}$, $\boldsymbol{m}_1$, $\boldsymbol{m}_2$ and $\mathbf{T}$, we will be able to simulate all the randomness in the construction of $\mathcal{D}_{\mathrm{yes}}$ and $\mathcal{D}_{\mathrm{no}}$. We give a formal description of a rejection sampling algorithm Alg$'$ which assumes access to an algorithm Alg testing Boolean functions.

1. We first sample $\mathbf{M} \subset [2n]$ of size n, and let $\boldsymbol{m}_1, \boldsymbol{m}_2 \sim \mathbf{M}$ be two distinct indices. Sample $\mathbf{T} \sim \mathcal{E}(\mathbf{M} \setminus \{\boldsymbol{m}_1, \boldsymbol{m}_2\})$. We may now view the hidden graph $\mathbf{G}$ (from rejection sampling) as a graph on vertex set $\overline{\mathbf{M}}$.
2. For each $t \in [q]$, perform the query $L_t = \overline{\mathbf{M}}$, which returns $(j_1^{(t)}, j_2^{(t)}) \in \mathbf{G}$, we sample $\boldsymbol{j}_3^{(t)} \sim \{\boldsymbol{m}_1, \boldsymbol{m}_2\}$ and $\boldsymbol{j}^{(t)} \sim \{\boldsymbol{m}_1, \boldsymbol{m}_2\}$. Intuitively, the values of $(j_1^{(t)}, j_2^{(t)}, \boldsymbol{j}_3^{(t)})$ will generate the t-th accessed subfunction $\boldsymbol{h}_i$ with $\Gamma_{\mathbf{T}}(x) > 3N/4$, and $\boldsymbol{j}^{(t)}$ will generate the t-th accessed subfunction $\boldsymbol{h}_i$ with $\Gamma_{\mathbf{T}}(x) \leq 3N/4$.

3. We simulate Alg by maintaining two q-tuples $p_1, p_2 \in (\{0\} \cup [N])^q$, which is initially $p_1 = p_2 = (0, 0, \dots 0)$ which will record the indices of the subfunctions accessed. We proceed as follows, where we assume that Alg makes the query $z \in \{0,1\}^{2n}$:
   - Suppose $|z_{|\mathbf{M}}| > \frac{n}{2} + \sqrt{2n}$, $|z_{|\mathbf{M}}| < \frac{n}{2} - \sqrt{2n}$, $\Gamma_{\mathbf{T}}(z) = 1^*$, or $\Gamma_{\mathbf{T}}(z) = 0^*$, report to Alg the appropriate value of $\boldsymbol{f}(x)$.
   - Otherwise, consider $\Gamma_{\mathbf{T}}(z) = i \in [N]$.
     - Suppose $i \leq \frac{3N}{4}$ and $(p_1)_t = i$ (if $(p_1)_t \neq i$ for all t, then find the first $t \in [q]$ with $(p_1)_t = 0$ and write $(p_1)_t = i$). In this case, report $z_{\boldsymbol{j}^{(t)}}$ if $\boldsymbol{j}^{(t)} = \boldsymbol{m}_1$ and $\neg z_{\boldsymbol{j}^{(t)}}$ if $\boldsymbol{j}^{(t)} = \boldsymbol{m}_2$.
     - If $i > \frac{3N}{4}$ and $(p_2)_t = i$ (again, if $(p_2)_t \neq i$ for all t, then find the first $t \in [q]$ with $(p_2)_t = 0$ and write $(p_2)_t = i$). In this case, we report $\neg z_{j_1^{(t)}} \oplus z_{j_2^{(t)}} \oplus z_{\boldsymbol{j}_3^{(t)}}$ if $\boldsymbol{j}_3^{(t)} = \boldsymbol{m}_1$ and $z_{j_1^{(t)}} \oplus z_{j_2^{(t)}} \oplus z_{\boldsymbol{j}_3^{(t)}}$ if $\boldsymbol{j}_3^{(t)} = \boldsymbol{m}_2$.
4. If Alg outputs "accept", then Alg$'$ outputs "$\mathcal{G}_2$", if Alg outputs "reject", then Alg$'$ outputs "$\mathcal{G}_1$".

Clearly, cost(Alg$'$) $= qn$. In addition, we may view Alg$'(\mathbf{G})$ as generating the necessary randomness for answering queries $\boldsymbol{f}(x)$ on the go, where $\mathbf{G}$ will determine whether $\boldsymbol{f} \sim \mathcal{D}_{\text{yes}}$ or $\boldsymbol{f} \sim \mathcal{D}_{\text{no}}$. When $\mathbf{G} = K_{\mathbf{A},\overline{\mathbf{A}}}$ (in the case $\mathbf{G} \sim \mathcal{G}_2$, the resulting function $\boldsymbol{f}$ is distributed as a function drawn from $\mathcal{D}_{\text{yes}}$; when $\mathbf{G} = K_{\mathbf{A}} \cup K_{\overline{\mathbf{A}}}$ (in the case $\mathbf{G} \sim \mathcal{G}_1$), the resulting function $\boldsymbol{f}$ is distributed as a function drawn from $\mathcal{D}_{\text{no}}$. Therefore, by the principle of deferred decisions, we have that Alg$'(\mathbf{G})$ perfectly simulates queries to a Boolean function $\boldsymbol{f} \sim \mathcal{D}_{\text{yes}}$ (if $\mathbf{G} \sim \mathcal{G}_2$) or $\boldsymbol{f} \sim \mathcal{D}_{\text{no}}$ (if $\mathbf{G} \sim \mathcal{G}_1$). We conclude that

$$\mathop{\mathbf{Pr}}_{\mathbf{G} \sim \mathcal{G}_1}[\text{Alg}'(\mathbf{G}) \text{ outputs ``}\mathcal{G}_2\text{''}] = \mathop{\mathbf{Pr}}_{\boldsymbol{f} \sim \mathcal{D}_{\text{yes}}}[\text{Alg}(\boldsymbol{f}) \text{ ``accepts''}], \qquad \text{and}$$
$$\mathop{\mathbf{Pr}}_{\mathbf{G} \sim \mathcal{G}_2}[\text{Alg}'(\mathbf{G}) \text{ outputs ``}\mathcal{G}_2\text{''}] = \mathop{\mathbf{Pr}}_{\boldsymbol{f} \sim \mathcal{D}_{\text{no}}}[\text{Alg}(\boldsymbol{f}) \text{ ``accepts''}].$$

Remark 7. *A close inspection of the proof of Lemma 5.4 reveals that the rejection sampling algorithm distinguishing $\mathcal{G}_1$ and $\mathcal{G}_2$ always makes queries $L_i = [n]$. This makes the lower bound simpler, as we can focus on proving lower bounds against algorithms which receive random edge samples.*

5.5 Proof of Lemma 5.5

Similarly to the proof of Lemma 5.4, we will proceed by generating the necessary randomness to generate the functions $\boldsymbol{f}$ from $\mathcal{D}_{\text{yes}}$ or from $\mathcal{D}_{\text{no}}$. However, unlike Lemma 5.4, this will not be a black box reduction, since we will not be able to simulate $\boldsymbol{f}$ exactly.

Consider a deterministic non-adaptive algorithm Alg which makes queries to a Boolean function $\boldsymbol{f} \colon \{0,1\}^{2n} \to \{0,1\}$ sampled from $\mathcal{D}_{\text{yes}}$ or $\mathcal{D}_{\text{no}}$ and outputs "accept" if Alg believes $\boldsymbol{f}$ was sampled from $\mathcal{D}_{\text{yes}}$, and outputs "reject" if Alg believes $\boldsymbol{f}$ was sampled from $\mathcal{D}_{\text{no}}$. Since Alg is non-adaptive and deterministic, all queries are determined, so consider the queries $z_1, \dots, z_q \in \{0,1\}^{2n}$, and let Alg: $\{0,1\}^q \to \{\text{``accept''}, \text{``reject''}\}$ be a function.

We will now define a non-adaptive algorithm Alg$'$ which makes rejection sampling queries to an unknown graph $\mathbf{G}$ on n vertices sampled from $\mathcal{G}_1$ or from $\mathcal{G}_2$. The algorithm Alg$'$ proceeds as follows:

1. Using some randomness and answers from rejection sampling queries to an unknown graph $\mathbf{G}$, we generate a sequence of r bits $(\boldsymbol{r}_1, \dots, \boldsymbol{r}_q)$ satisfying the following two conditions (we give the procedure to generate these random bits after)[7]:
   - If $\mathbf{G} \sim \mathcal{G}_1$, then $(\boldsymbol{r}_1, \dots, \boldsymbol{r}_q)$ will be roughly distributed as $(\boldsymbol{f}(z_1), \dots, \boldsymbol{f}(z_q))$ where $\boldsymbol{f}$ is a Boolean function $\boldsymbol{f} \sim \mathcal{D}_{\text{no}}$.
   - If $\mathbf{G} \sim \mathcal{G}_2$, then $(\boldsymbol{r}_1, \dots, \boldsymbol{r}_q)$ will be roughly distributed as a $(\boldsymbol{f}(z_1), \dots, \boldsymbol{f}(z_q))$ where $\boldsymbol{f}$ is a Boolean function $\boldsymbol{f} \sim \mathcal{D}_{\text{yes}}$.
2. Finally, if Alg$(\boldsymbol{r}_1, \dots, \boldsymbol{r}_q)$ outputs "accept", then Alg$'$ outputs "$\mathcal{G}_2$", and if Alg$(\boldsymbol{r}_1, \dots, \boldsymbol{r}_q)$ outputs "reject", then Alg$'$ outputs "$\mathcal{G}_1$".

In order to formalize the notion of "roughly distributed as" from above, let $\mathcal{V}_{\text{yes}}$ and $\mathcal{V}_{\text{no}}$ be the distributions supported on $\{0,1\}^q$ given by:

$$\begin{aligned} \boldsymbol{r} \sim \mathcal{V}_{\text{yes}} \qquad &\text{where} \qquad \forall i \in [q], \boldsymbol{r}_i = \boldsymbol{f}(z_i), \text{ and } \boldsymbol{f} \sim \mathcal{D}_{\text{yes}}. \\ \boldsymbol{r} \sim \mathcal{V}_{\text{no}} \qquad &\text{where} \qquad \forall i \in [q], \boldsymbol{r}_i = \boldsymbol{f}(z_i), \text{ and } \boldsymbol{f} \sim \mathcal{D}_{\text{no}}. \end{aligned}$$

Now, given the algorithm Alg$'$, we let $\mathcal{U}_{\text{yes}}, \mathcal{U}_{\text{no}}$ be the distributions supported in $\{0,1\}^q$ given by:

$$\begin{aligned} \boldsymbol{r} \sim \mathcal{U}_{\text{yes}} \qquad &\text{where} \qquad \text{Alg}'(\mathbf{G}) \text{ outputs } (\boldsymbol{r}_1, \dots, \boldsymbol{r}_q) \text{ when } \mathbf{G} \sim \mathcal{G}_2 \\ \boldsymbol{r} \sim \mathcal{U}_{\text{no}} \qquad &\text{where} \qquad \text{Alg}'(\mathbf{G}) \text{ outputs } (\boldsymbol{r}_1, \dots, \boldsymbol{r}_q) \text{ when } \mathbf{G} \sim \mathcal{G}_1 \end{aligned}$$

The following lemma is a simple consequence will allow us to conclude Lemma 5.5.

Lemma 5.6. *Suppose $\mathcal{V}_{yes}, \mathcal{V}_{no}, \mathcal{U}_{yes}$ and $\mathcal{U}_{no}$ satisfy:*

$$d_{TV}(\mathcal{V}_{yes}, \mathcal{U}_{yes}) = o(1) \qquad \textit{and} \qquad d_{TV}(\mathcal{V}_{no}, \mathcal{U}_{no}) = o(1).$$

Then, we have that:

$$\begin{aligned} \mathop{\mathbf{Pr}}_{\mathbf{G} \sim \mathcal{G}_1}[\text{Alg}'(\mathbf{G}) \textit{ outputs "}\mathcal{G}_1\textit{"}] &\approx \mathop{\mathbf{Pr}}_{\boldsymbol{f} \sim \mathcal{D}_{no}}[\text{Alg}(\boldsymbol{f}) \textit{ "rejects"}] \pm o(1). \\ \mathop{\mathbf{Pr}}_{\mathbf{G} \sim \mathcal{G}_2}[\text{Alg}'(\mathbf{G}) \textit{ outputs "}\mathcal{G}_2\textit{"}] &\approx \mathop{\mathbf{Pr}}_{\boldsymbol{f} \sim \mathcal{D}_{yes}}[\text{Alg}(\boldsymbol{f}) \textit{ "accepts"}] \pm o(1). \end{aligned}$$

Proof: We show the first inequality in the conclusion, as the argument is the same for the second inequality. Consider the set $R = \{r \in \{0,1\}^q : \text{Alg}(r) = \text{"reject"}\}$. Then, we have:

$$\begin{aligned} \mathop{\mathbf{Pr}}_{\boldsymbol{f} \sim \mathcal{D}_{\text{no}}}[\text{Alg}(\boldsymbol{f}) \text{ "rejects"}] &= \mathop{\mathbf{Pr}}_{\boldsymbol{r} \sim \mathcal{V}_{\text{no}}}[\boldsymbol{r} \in R] \\ &\approx \mathop{\mathbf{Pr}}_{\boldsymbol{r} \sim \mathcal{U}_{\text{no}}}[\boldsymbol{r} \in R] \pm o(1) \\ &\approx \mathop{\mathbf{Pr}}_{\mathbf{G} \sim \mathcal{G}_1}[\text{Alg}'(\mathbf{G}) \text{ outputs "accept"}] \pm o(1). \end{aligned}$$

[7] With a slight abuse of notation, we let Alg$'(\mathbf{G})$ correspond to to the output $(\boldsymbol{r}_1, \dots, \boldsymbol{r}_q)$ that Alg$'$ produces with rejection sampling access to graph $\mathbf{G}$.

■

Given Lemma 5.6, it remains to describe the randomized procedure Alg$'$ which given rejection sampling access to an unknown n-vertex graph $\mathbf{G}$ from $\mathcal{G}_1$ or $\mathcal{G}_2$ outputs a bit-string of length q such that:

$$d_{TV}(\mathcal{V}_{\text{yes}}, \mathcal{U}_{\text{yes}}) = o(1) \qquad \text{and} \qquad d_{TV}(\mathcal{V}_{\text{no}}, \mathcal{U}_{\text{no}}) = o(1).$$

The procedure will work as follows:

1. First, sample a random subset $\mathbf{M} \subset [2n]$ of size n, and let $\boldsymbol{m}_1, \boldsymbol{m}_2 \sim \mathbf{M}$ be two distinct random indices, and let $\mathbf{T} \sim \mathcal{E}(\mathbf{M} \setminus \{\boldsymbol{m}_1, \boldsymbol{m}_2\})$. This defines an indexing function[8] $\Gamma_{\mathbf{T}} \colon \{0,1\}^{2n} \to [N]$. We may view the unknown graph $\mathbf{G}$ as being defined over vertices in $\overline{\mathbf{M}}$ [9].

2. We now consider partitioning the queries $z_1, \dots, z_q \in \{0,1\}^{2n}$ into at most $t+4$ sets (where we will have $t \leq q$) $\mathbf{Q}_{\mathbf{M}}^{(+)}, \mathbf{Q}_{\mathbf{M}}^{(-)}, \mathbf{Q}_*^{(0)}, \mathbf{Q}_*(1)$ and $\mathbf{Q}_{\ell_1}, \dots, \mathbf{Q}_{\ell_t}$ non-empty sets where $\ell_1, \dots, \ell_t \subset [N]$:

$$\begin{aligned}
\mathbf{Q}_{\mathbf{M}}^{(-)} &= \left\{ z_i : |(z_i)_{|\mathbf{M}}| < \frac{n}{2} - \sqrt{2n} \right\}, \\
\mathbf{Q}_{\mathbf{M}}^{(+)} &= \left\{ z_i : |(z_i)_{|\mathbf{M}}| > \frac{n}{2} + \sqrt{2n} \right\}, \\
\mathbf{Q}_*^{(0)} &= \left\{ z_i : \Gamma_{\mathbf{T}}(z_i) = 0^* \wedge z_i \notin \mathbf{Q}_{\mathbf{M}}^{(-)} \cup \mathbf{Q}_{\mathbf{M}}^{(+)} \right\}, \\
\mathbf{Q}_*^{(1)} &= \left\{ z_i : \Gamma_{\mathbf{T}}(z_i) = 1^* \wedge z_i \notin \mathbf{Q}_{\mathbf{M}}^{(-)} \cup \mathbf{Q}_{\mathbf{M}}^{(+)} \right\}, \\
\mathbf{Q}_\ell &= \left\{ z_i : \Gamma_{\mathbf{T}}(z_i) = \ell \wedge z_i \notin \mathbf{Q}_{\mathbf{M}}^{(-)} \cup \mathbf{Q}_{\mathbf{M}}^{(+)} \right\}.
\end{aligned}$$

3. If $z_i \in \mathbf{Q}_M^{(-)}$, we let $\boldsymbol{r}_i = 0$, if $z_i \in \mathbf{Q}_M^{(+)}$, we let $\boldsymbol{r}_i = 1$. If $z_i \in \mathbf{Q}_*^{(0)}$, we let $\boldsymbol{r}_i = 0$, and if $z_i \in \mathbf{Q}_*^{(1)}$, we let $\boldsymbol{r}_i = 1$. We may thus only consider the queries in $\mathbf{Q}_{\ell_1}, \dots, \mathbf{Q}_{\ell_t}$, and for simplicity in the notation, we re-index the queries to let:

$$\mathbf{Q}_{\ell_i} = \left\{ z_1^{(i)}, z_2^{(i)}, \dots, z_{|\mathbf{Q}_{\ell_i}|}^{(i)} \right\}$$

   for each $i \in [t]$, and the corresponding bits $\boldsymbol{r}_1^{(i)}, \boldsymbol{r}_2^{(i)}, \dots, \boldsymbol{r}_{|\mathbf{Q}_{\ell_i}|}^{(i)}$.

4. We thus consider each $i \in [t]$ and independently set the values of $\boldsymbol{r}_1^{(i)}, \dots, \boldsymbol{r}_{|\mathbf{Q}_{\ell_i}|}^{(i)}$ as follows:

   (a) If $\ell_i \leq 3N/4$, sample some $\boldsymbol{j} \sim \{\boldsymbol{m}_1, \boldsymbol{m}_2\}$, and for every $\alpha \in [|\mathbf{Q}_{\ell_i}|]$, let:

$$\boldsymbol{r}_\alpha^{(i)} = \left\{ \begin{array}{cc} (z_\alpha^{(i)})_{\boldsymbol{j}} & \boldsymbol{j} = \boldsymbol{m}_1 \\ \neg(z_\alpha^{(i)})_{\boldsymbol{j}} & \boldsymbol{j} = \boldsymbol{m}_2 \end{array} \right. .$$

   (b) Otherwise, if $\ell_i > 3N/4$, consider the following sets

$$\mathbf{L}_i = \left\{ k \in \overline{\mathbf{M}} : \exists \alpha, \beta \in [|\mathbf{Q}_{\ell_i}|], (z_\alpha^{(i)})_k \neq (z_\beta^{(i)})_k \right\},$$

[8] Note that now, $N = 2^{\sqrt{2n}}$ since we are considering Boolean functions with $2n$ variables.

[9] We may assume this by picking an arbitrary mapping of the indices in $\overline{\mathbf{M}}$ to $[n]$.

and,

$$\overline{\mathbf{L}}_i^{(0)} = \left\{k \in \overline{\mathbf{M}} \setminus \mathbf{L}_i : z \in \mathbf{Q}_{\ell_i}, z_k = 0\right\} \qquad \overline{\mathbf{L}}_i^{(1)} = \left\{k \in \overline{\mathbf{M}} \setminus \mathbf{L}_i : z \in \mathbf{Q}_{\ell_i}, z_k = 1\right\} .$$

We make the query $\mathbf{L}_i$ if $|\mathbf{L}_i| \leq \frac{n}{\log n}$ and $\overline{\mathbf{M}}$ otherwise to the rejection sampling oracle and obtain a response $\boldsymbol{v} \in (\overline{\mathbf{M}} \times \overline{\mathbf{M}}) \cup \overline{\mathbf{M}} \cup \{\emptyset\}$. In addition, sample $\boldsymbol{j}_3 \sim \{\boldsymbol{m}_1, \boldsymbol{m}_2\}$. We now consider three cases:

i. If $\boldsymbol{v} = (\boldsymbol{j}_1, \boldsymbol{j}_2) \in \overline{\mathbf{M}} \times \overline{\mathbf{M}}$ is an edge, then for each $\alpha \in [|\mathbf{Q}_{\ell_i}|]$, we let:

$$\boldsymbol{r}_\alpha^{(i)} = \begin{cases} (z_\alpha^{(i)})_{\boldsymbol{j}_1} \oplus (z_\alpha^{(i)})_{\boldsymbol{j}_2} \oplus (z_\alpha^{(i)})_{\boldsymbol{j}_3} & \boldsymbol{j}_3 = \boldsymbol{m}_1 \\ \neg(z_\alpha^{(i)})_{\boldsymbol{j}_1} \oplus (z_\alpha^{(i)})_{\boldsymbol{j}_2} \oplus (z_\alpha^{(i)})_{\boldsymbol{j}_3} & \boldsymbol{j}_3 = \boldsymbol{m}_2 \end{cases} .$$

ii. If $\boldsymbol{v} = \boldsymbol{j}_2 \in \overline{\mathbf{M}}$ is a lone vertex, then let $w = \neg(z_1^{(i)})_{\boldsymbol{j}_2}$ and $p_v(\overline{\mathbf{L}}_i^{(w)}) = \frac{|\overline{\mathbf{L}}_i^{(w)}|}{|\overline{\mathbf{L}}_i|}$, we sample $\boldsymbol{b} \sim \mathrm{Ber}(p_v(\overline{\mathbf{L}}_i^{(w)}))$ and for each $\alpha \in [|\mathbf{Q}_{\ell_i}|]$, we let:

$$\boldsymbol{r}_\alpha^{(i)} \oplus (z_\alpha^{(i)})_{\boldsymbol{j}_2} \oplus (z_\alpha^{(i)})_{\boldsymbol{j}_3} = \begin{cases} \boldsymbol{b} \oplus (z_1^{(i)})_{\boldsymbol{j}_2} & \boldsymbol{j}_3 = \boldsymbol{m}_1 \\ \neg\boldsymbol{b} \oplus (z_1^{(i)})_{\boldsymbol{j}_2} & \boldsymbol{j}_3 = \boldsymbol{m}_2 \end{cases} .$$

iii. Lastly, if $\boldsymbol{v} = \emptyset$ is the empty set, then let $p_\emptyset(\overline{\mathbf{L}}_i) = \frac{2|\overline{\mathbf{L}}_i^{(0)}||\overline{\mathbf{L}}_i^{(1)}|}{|\overline{\mathbf{L}}_i|^2}$ and sample $\boldsymbol{b} \sim \mathrm{Ber}(p(\overline{\mathbf{L}}_i))$ and for each $\alpha \in [|\mathbf{Q}_{\ell_i}|]$, we let:

$$\boldsymbol{r}_\alpha^{(i)} \oplus (z_\alpha^{(i)})_{\boldsymbol{j}_3} = \begin{cases} \boldsymbol{b} & \boldsymbol{j}_3 = \boldsymbol{m}_1 \\ \neg\boldsymbol{b} & \boldsymbol{j}_3 = \boldsymbol{m}_2 \end{cases} .$$

Remark 8. *The procedure described above does not exactly simulate queries to a $\boldsymbol{f} \sim \mathcal{D}_{yes}$ or $\mathcal{D}_{no}$ (in the case of $\mathbf{G} \sim \mathcal{G}_2$ or $\mathbf{G} \sim \mathcal{G}_1$, respectively) as in the reductions of Lemma 5.4 and Lemma 4.3). Let us briefly explain why this happens by giving an illuminating example. Consider a one-query algorithm which makes query $z \in \{0,1\}^n$ and suppose $|z_{\mathbf{M}}| \approx \frac{n}{2} \pm \sqrt{2n}$ and $\Gamma_{\mathbf{T}}(z) = i > \frac{3N}{4}$ with $z_{m_1} = 0$ and $z_{m_2} = 1$. Then, the value $\boldsymbol{f}(z) = \boldsymbol{h}_i(z)$ will be 0 if $z_{\boldsymbol{j}_1} = z_{\boldsymbol{j}_2}$, and 1 if $z_{\boldsymbol{j}_1} \neq z_{\boldsymbol{j}_2}$, where $(\boldsymbol{j}_1, \boldsymbol{j}_2) \sim \mathbf{G}$ is the edge sampled for subfunction $\boldsymbol{h}_i$.*

We note that this probability is slightly different for $\mathbf{G} \sim \mathcal{G}_1$ and $\mathbf{G} \sim \mathcal{G}_2$ and depends on how $\mathbf{A}$ partitions the 0-variables and 1-variables of z. Despite this difference, Alg$'$ *always observes $\emptyset$ from the rejection sampling oracle, so the output bit $\boldsymbol{r} \in \{0,1\}$ which* Alg$'$ *produces will not simulate $\boldsymbol{f}(z)$ exactly. The bulk of the argument shows that* Alg$'$ *can sample a random bit whose distribution is close to $\boldsymbol{f}(z)$ in total variation distance, so that* Alg *cannot exploit the fact that the simulation is not exact.*

We first note the following lemma.

Lemma 5.7. *With probability $1-o(1)$ over the draw of $\mathbf{M} \subset [n]$, $\boldsymbol{m}_1, \boldsymbol{m}_2$ and $\mathbf{T} \sim \mathcal{E}(\mathbf{M} \setminus \{\boldsymbol{m}_1, \boldsymbol{m}_2\})$, we have that for all $i \in [t]$,*

$$|\mathbf{L}_i| \leq |\mathbf{Q}_{\ell_i}| \cdot 90\sqrt{n} \log n.$$

Proof: We will prove this by showing that for any two $z, z' \in \mathbf{Q}_{\ell_i}$, $\|z - z'\|_1 \leq 90\sqrt{n}\log n$ with probability $1 - \frac{1}{n^{10}}$, so that we may union bound over all possible pairs. More specifically, consider two queries $z, z' \in \{0,1\}^{2n}$ which differ by more than $90\sqrt{n}\log n$ indices. Note that the distribution of the random variable $\|(z - z')_{|\mathbf{M}}\|_1 \sim \mathrm{HG}(2n, |z - z'|, n)$. Then using Theorem 6 we have that with probability at least $1 - \frac{1}{n^{10}}$ over the draw of $\mathbf{M}$, $\|z_{|\mathbf{M}} - z'_{|\mathbf{M}}\|_1 \geq 30\sqrt{n}\log n$.

Next, if $|z_{|\mathbf{M}}| \approx \frac{n}{2} \pm \sqrt{2n}$ and $|z'_{|\mathbf{M}}| \approx \frac{n}{2} \pm \sqrt{2n}$ (if either of these conditions do not hold, then we know the strings are not in $\mathbf{Q}_{\ell_i}$ for any i), then there exists a set $\mathbf{P} \subset \mathbf{M}$ with $|\mathbf{P}| = 15\sqrt{n}\log n$ such that for all $k \in \mathbf{P}$, $z_k = 1$ and $z'_k = 0$. Thus, we have that:

$$\Pr_{\mathbf{T}}[\exists i \in [t], z, z' \in \mathbf{Q}_{\ell_i}] \leq \Pr_{\mathbf{T}}[z' \in \mathbf{Q}_{\ell_i} \mid z \in \mathbf{Q}_{\ell_i}] \leq \Pr_{\mathbf{T}_{\ell_i}}[\mathbf{T}_{\ell_i} \cap \mathbf{P} = \emptyset] \leq \left(1 - \frac{15\log n}{\sqrt{n}}\right)^{\sqrt{n}} \ll \frac{1}{n^{10}}.$$

So we may union bound over all pairs of queries to conclude that if $z, z' \in \mathbf{Q}_{\ell_i}$, then $\|z - z'\|_1 \leq 90\sqrt{n}\log n$ with high probability, which gives the desired claim. ■

Thus, given Lemma 5.7 as well as the fact that we query $[n]$ when $|\mathbf{L}_i| \geq \frac{n}{\log n}$, we conclude that if Alg makes q queries, then Alg′ has complexity at most $q \cdot O(\sqrt{n}\log^2 n)$ in the rejection sampling model.

Lemma 5.8. *If $q \leq \frac{n^{3/2}}{\log^8 n}$, then with probability $1 - o(1)$ over the draw of $\mathbf{M} \subset [n], \boldsymbol{m}_1, \boldsymbol{m}_2$, $\mathbf{T}$, and $\mathbf{A} \subset \overline{\mathbf{M}}$, we have that for every $i \in [t]$ where $|\mathbf{L}_i| \leq \frac{n}{\log n}$, the sets $|\overline{\mathbf{L}}_i^{(0)}|, |\overline{\mathbf{L}}_i^{(1)}|$ satisfy the following*

$$|\overline{\mathbf{L}}_i^{(0)}|, |\overline{\mathbf{L}}_i^{(1)}| = \Omega(n) ,$$

$$\left|\mathbf{A} \cap \overline{\mathbf{L}}_i^{(0)}\right| \approx \frac{\left|\overline{\mathbf{L}}_i^{(0)}\right|}{2} \pm \sqrt{n}\log n \qquad \textit{and} \qquad \left|\mathbf{A} \cap \overline{\mathbf{L}}_i^{(1)}\right| \approx \frac{\left|\overline{\mathbf{L}}_i^{(1)}\right|}{2} \pm \sqrt{n}\log n.$$

Proof: We first claim that with probability $1 - o(1)$ over the choice of $\mathbf{M}$, all the queries $z \in \{0,1\}^{2n}$ that are mapped to some $\mathbf{Q}_{\ell_i}$ are such that $|z| \approx n \pm 50\sqrt{2n}\log n$. Assume $z \in \{0,1\}^{2n}$ is such that $|z| > n + 50\sqrt{2n}\log n$, and consider the random variable $|z_{|\mathbf{M}}|$. Note that the distribution of $|z_{|\mathbf{M}}|$ is hyper-geometric with parameters $(2n, |z|, n)$. By using Theorem 6 on the tail bounds for hyper-geometric random variable, we get that for any $t > 0$

$$\Pr_{\mathbf{M}}\left[|z_{|\mathbf{M}}| < \left(\frac{|z|}{2n} - t\right)n\right] \leq e^{-2t^2 n} .$$

By choosing $t = \frac{50\log n}{\sqrt{2n}} - \frac{\sqrt{2}}{\sqrt{n}}$, and considering the complement event, we have that

$$\Pr_{\mathbf{M}}\left[|z_{|\mathbf{M}}| \geq \frac{|z|}{2} - \frac{50\sqrt{n}\log n}{\sqrt{2}} + \sqrt{2n}\right] \geq 1 - \frac{1}{n^{50}} .$$

Combining this with the fact that $|z| > n + 50\sqrt{2n}\log n$, we get that the probability that $|z_{|\mathbf{M}}| > n/2 + \sqrt{2n}$ is at least $1 - 1/n^{50}$.

Similarly, we get that when $|z| < n - 50\sqrt{2n}\log n$, we have that with probability $1 - 1/n^{50}$ over the choice of $\mathbf{M}$, $|z_{|\mathbf{M}}| < n/2 - \sqrt{2n}$. By using a union bound on the number of queries we get that with probability $1 - o(1)$ over the choice of $\mathbf{M}$, all the queries $z \in \{0,1\}^{2n}$ that are mapped to some $\mathbf{Q}_{\ell_i}$ are such that $|z| \approx n \pm 50\sqrt{2n}\log n$.

We henceforth condition on such $\mathbf{M} = M$. Consider any $T \sim \mathcal{E}(M)$ and all the indices $i \in [t]$ such that $|L_i| \leq \frac{n}{\log n}$. By definition, if $z \in \{0,1\}^{2n}$ is mapped to some Q_{ℓ_i}, then $|z|_M| \approx n/2 \pm \sqrt{2n}$, which implies that $|z|_{\overline{M}}| \approx n/2 \pm 49\sqrt{2n}\log n$. Therefore, by the fact that all queries in Q_{ℓ_i} must agree on all of the coordinates in $\overline{L}_i$, we can conclude that $|\overline{L}_i^{(0)}|$ and $|\overline{L}_i^{(1)}|$ are $\Omega(n)$.

Next, consider the random variable $|\mathbf{A} \cap \overline{L}_i^{(1)}|$, and note that its distribution is hyper-geometric with parameters $(n, |\overline{L}_i^{(1)}|, n/2)$. By using tail bounds for hyper-geometric random variable, we get that with probability at least $1 - o(1)$ over the choice of $\mathbf{A}$

$$|\mathbf{A} \cap \overline{L}_i^{(1)}| \approx \frac{|L_i^{(1)}|}{2} \pm \sqrt{n}\log n\ .$$

Using the same argument, we also get that with probability $1 - o(1)$ over the choice of $\mathbf{A}$ we have that

$$|\mathbf{A} \cap \overline{L}_i^{(0)}| \approx \frac{|L_i^{(0)}|}{2} \pm \sqrt{n}\log n\ .$$

By applying a union bound over all indices $i \in [t]$ the lemma follows. ■

Lemma 5.9. *If* $\mathrm{cost}(\mathrm{Alg}') \leq \frac{n^2}{\log^6 n}$ *which occurs with high probability over* $\mathbf{M}$*, with probability* $1 - o(1)$ *over the draw of* $\boldsymbol{v}$ *in Step 4(b), there are at most* $\frac{n}{\log^4 n}$ *responses* $\boldsymbol{v} \in \overline{\mathbf{M}}$ *which are lone vertices of case (ii).*

As discussed earlier, the proof of the above lemma is given in the lower bound for distinguishing $\mathcal{G}_1$ and $\mathcal{G}_2$ in Section 6 (Lemma 6.14). We assume its correctness for the rest of this section.

We note that since $\mathbf{M}, \boldsymbol{m}_1, \boldsymbol{m}_2$ and $\mathbf{T}$ are distributed in the same way in $\boldsymbol{f} \sim \mathcal{D}_{\text{yes}}$ and in Step 1 of Alg, we may consider the distribution $\mathcal{V}_{\text{yes}}(M, m_1, m_2, T)$ denoting $\mathcal{V}_{\text{yes}}$ conditioned on $\mathbf{M} = M, \boldsymbol{m}_1 = m_1, \boldsymbol{m}_2 = m_2$ and $\mathbf{T} = T$, and we analogously define $\mathcal{U}_{\text{yes}}(M, m_1, m_2, T)$, $\mathcal{V}_{\text{no}}(M, m_1, m_2, T)$ and $\mathcal{U}_{\text{no}}(M, m_1, m_2, T)$. In addition, we may denote the event $\boldsymbol{\mathcal{E}}_A$ to denote the event that the hidden subset $\mathbf{A}$ sampled in $\boldsymbol{f}$ or in the graph $\mathbf{G}$ satisfies the conditions of Lemma 5.8, and the event $\boldsymbol{\mathcal{E}}_V$ to be the event that there are at most $\frac{n}{\log^4 n}$ responses which are lone vertices from Lemma 5.9. We thus consider a fixed set M, m_1, m_2, and T satisfying the following conditions of Lemma 5.7 and consider the distribution $\mathcal{V}'_{\text{yes}}$ to be the distribution given by sampling $\boldsymbol{r} \sim \mathcal{V}_{\text{yes}}(M, m_1, m_2, T)$ conditioned on events $\boldsymbol{\mathcal{E}}_A$ and $\boldsymbol{\mathcal{E}}_V$. We analogously define $\mathcal{V}'_{\text{no}}$, $\mathcal{U}'_{\text{yes}}$ and $\mathcal{U}'_{\text{no}}$. We note it suffices to show $d_{TV}(\mathcal{V}'_{\text{yes}}, \mathcal{U}'_{\text{yes}}) = o(1)$ and $d_{TV}(\mathcal{V}'_{\text{no}}, \mathcal{U}'_{\text{no}}) = o(1)$.

We now note that conditioned on M, m_1, m_2 and T, the sets $\mathbf{Q}_{\mathbf{M}}^{(+)}, \mathbf{Q}_{\mathbf{M}}^{(-)}, \mathbf{Q}_*^{(1)}$ and $\mathbf{Q}_*^{(0)}$, as well as all $\mathbf{Q}_{\ell_1}, \ldots, \mathbf{Q}_{\ell_t}$ are no longer random. Furthermore, when $z \in \mathbf{Q}_{\mathbf{M}}^{(+)} \cup \mathbf{Q}_{\mathbf{M}}^{(-)} \cup \mathbf{Q}_*^{(1)} \cup \mathbf{Q}_*^{(0)}$ the values of $f_{T,\mathbf{A},\mathbf{H}}(z)$ from $\mathcal{D}_{\text{yes}}$ (and from $\mathcal{D}_{\text{no}}$) are fixed to their corresponding values according to (2), which match their settings in $\mathcal{U}'_{\text{yes}}$ and $\mathcal{U}'_{\text{no}}$. Likewise, when $z \in \mathbf{Q}_{\ell_i}$ with $\ell_i \leq \frac{3N}{4}$, $f_{T,\mathbf{A},\mathbf{H}}(z)$ is determined by a dictator or anti-dictator in $\{m_1, m_2\}$; by the principle of deferred decisions, the values of $f_{T,\mathbf{A},\mathbf{H}}(z)$ can be simulated exactly. Therefore, it remains to consider the values of $\boldsymbol{r}_\alpha^{(i)}$

corresponding to $f_{T,\mathbf{A},\mathbf{H}}(z_\alpha^{(i)})$ for each $i \in [t]$, where $\ell_i > \frac{3N}{4}$, so for simplicity, assume that every $\ell_i > \frac{3N}{4}$.

Consider a function $v\colon [t] \to \{\text{“edge”}, \text{“lone vertex”}, \text{“empty set”}\}$ which indicates whether the response of the ith rejection sampling query sampled in Step 4(b) falls into case (i) (when $\boldsymbol{v}_i$ is an edge), or case (ii) (when $\boldsymbol{v}_i$ is a lone vertex), or case (iii) (when $\boldsymbol{v}_i$ is $\emptyset$). In other words,

$$v(i) = \begin{cases} \text{“edge”} & \boldsymbol{v}_i \in \overline{\mathbf{M}} \times \overline{\mathbf{M}} \\ \text{“lone vertex”} & \boldsymbol{v}_i \in \overline{\mathbf{M}} \\ \text{“empty set”} & \boldsymbol{v}_i = \emptyset \end{cases}$$

We thus consider one fixed function $v\colon [t] \to \{\text{“edge”}, \text{“lone vertex”}, \text{“empty set”}\}$ and condition on the fact that v specifies the three cases of Step 4(b) (in the case of $\mathcal{U}_{\text{yes}}$ and $\mathcal{U}_{\text{no}}$) and whether the edge sampled $(\boldsymbol{j}_1, \boldsymbol{j}_2) \sim \mathbf{G}$ in the fourth step of generating $\mathcal{D}_{\text{yes}}$ and $\mathcal{D}_{\text{no}}$ for $\boldsymbol{h}_{\ell_i}$ either intersects $\mathbf{L}_i$ fully (in the case of an edge), or partially (in the case of a lone vertex), or it does not intersect at all (in the case of the empty set). Thus, again, we may consider the distributions conditioned on the edges sampled are specified correctly by v.

The following three lemmas give the distribution of $\boldsymbol{r}_1^{(i)} \sim \mathcal{V}'_{\text{yes}}$ and $\boldsymbol{r}_1^{(i)} \sim \mathcal{V}'_{\text{no}}$ in the cases when $\boldsymbol{v}_i$ is an edge, or a lone vertex, or the empty set. We note that the three lemmas indicate how to generate the bits $\boldsymbol{r}_\alpha^{(i)}$ in Step 4(b) of Alg$'$.

Lemma 5.10. *For every $i \in [t]$ with $v(i) =$ “edge”, we have that every $\alpha \in [|\mathbf{Q}_{\ell_i}|]$ has $\boldsymbol{r}_\alpha^{(i)}$ generated from* Alg$'$ *is distributed exactly as $\boldsymbol{f}(z_\alpha^{(i)})$.*

Proof: This simply follows from the principle of deferred decisions, since Alg$'$ generates all the necessary randomness to simulate a query to a function $\boldsymbol{f} \sim \mathcal{D}_{\text{yes}}$ or $\boldsymbol{f} \sim \mathcal{D}_{\text{no}}$ which indexes to the sub-function $\boldsymbol{h}_{\ell_i}$. ■

Lemma 5.11. *For every $i \in [t]$ with $v(i) =$ “empty set”, there exists $|\gamma_{yes}|, |\gamma_{no}| = O(\frac{\log^2 n}{n})$ such that for $\boldsymbol{r} \sim \mathcal{V}'_{yes}$ satisfies*

$$\boldsymbol{r}_1^{(i)} \oplus (z_1^{(i)})_{\boldsymbol{j}_3} \sim \begin{cases} \text{Ber}\left(p_\emptyset(\overline{\mathbf{L}}_i) + \gamma_{yes}\right) & \boldsymbol{j}_3 = m_1 \\ \text{Ber}(1 - p_\emptyset(\overline{\mathbf{L}}_i) - \gamma_{yes}) & \boldsymbol{j}_3 = m_2 \end{cases},$$

and $\boldsymbol{r} \sim \mathcal{V}'_{no}$ satisfies

$$\boldsymbol{r}_1^{(i)} \oplus (z_1^{(i)})_{\boldsymbol{j}_3} \sim \begin{cases} \text{Ber}\left(p_\emptyset(\overline{\mathbf{L}}_i) + \gamma_{no}\right) & \boldsymbol{j}_3 = m_1 \\ \text{Ber}\left(1 - p_\emptyset(\overline{\mathbf{L}}_i) - \gamma_{no}\right) & \boldsymbol{j}_3 = m_2 \end{cases}.$$

Proof: We recall that $\boldsymbol{h}_{\ell_i}$ is determined by $(\boldsymbol{j}_1, \boldsymbol{j}_2) \sim \mathbf{G}$ and $\boldsymbol{j}_3 \sim \{m_1, m_2\}$ in the fourth step of generating $\boldsymbol{f} \sim \mathcal{D}_{\text{yes}}$ or $\mathcal{D}_{\text{no}}$. Consider the case when $\boldsymbol{j}_3 = m_1$, and the case when $(z_1^{(i)})_{m_1} = 0$ (since the case $(z_1^{(i)})_{m_1} = 1$ is symmetric, except we flip the answer).

Recall that we condition on the fact that the edge $(\boldsymbol{j}_1, \boldsymbol{j}_2) \sim \mathbf{G}$ satisfies $\mathbf{L}_i \cap \{\boldsymbol{j}_1, \boldsymbol{j}_2\} = \emptyset$, as well as the conclusions from Lemma 5.8, so we may write:

$$
\begin{aligned}
&\mathop{\mathbf{Pr}}_{\boldsymbol{r}\sim\mathcal{V}_{\text{yes}}}\left[\boldsymbol{r}_1^{(i)} = 1 \mid v(i) = \text{“empty set”}\right] \\
&= \mathop{\mathbf{Pr}}_{\substack{\mathbf{G}\sim\mathcal{D}_{\text{no}} \\ (\boldsymbol{j}_1,\boldsymbol{j}_2)}}\left[\left(\boldsymbol{j}_1 \in \mathbf{A}\cap\overline{\mathbf{L}}_i^{(0)} \wedge \boldsymbol{j}_2 \in \overline{\mathbf{A}}\cap\overline{\mathbf{L}}_i^{(1)}\right) \vee \left(\boldsymbol{j}_1 \in \mathbf{A}\cap\overline{\mathbf{L}}_i^{(1)} \wedge \boldsymbol{j}_2 \in \overline{\mathbf{A}}\cap\overline{\mathbf{L}}_i^{(0)}\right) \mid v(i) = \text{“empty set”}\right], \\
&= \frac{1}{|\mathbf{A}\cap\overline{\mathbf{L}}_i| \cdot |\overline{\mathbf{A}}\cap\overline{\mathbf{L}}_i|} \cdot \left(|\mathbf{A}\cap\overline{\mathbf{L}}_i^{(0)}| \cdot |\overline{\mathbf{A}}\cap\overline{\mathbf{L}}_i^{(1)}| + |\mathbf{A}\cap\overline{\mathbf{L}}_i^{(1)}| \cdot |\overline{\mathbf{A}}\cap\overline{\mathbf{L}}_i^{(0)}|\right) \qquad (4)
\end{aligned}
$$

since the value of $\boldsymbol{f}(z_1^{(i)})$ in the case of $\boldsymbol{j}_3 = m_1$ will be a parity of the end points, so this parity will be 1 when the values of the variables $\boldsymbol{j}_1$ and $\boldsymbol{j}_2$ under $z_1^{(i)}$ disagree. In order to see this, we recall that $\mathbf{G}$ is the complete bipartite graph (in the case when $\boldsymbol{r}\sim\mathcal{V}_{\text{yes}}$) with sides $\mathbf{A}$ and $\overline{\mathbf{A}}$, so the edge $(\boldsymbol{j}_1,\boldsymbol{j}_2) \in \mathbf{A}\times\overline{\mathbf{A}}$ must have $(z_1^{(i)})_{\boldsymbol{j}_1} \neq (z_1^{(i)})_{\boldsymbol{j}_2}$, and $\boldsymbol{j}_1,\boldsymbol{j}_2 \in \overline{\mathbf{L}}_i$.

Since $v(i) =$ “empty set”, we note that $|\mathbf{L}_i| \leq \frac{n}{\log n}$, so $|\overline{\mathbf{L}}_i| = \Omega(n)$. In addition, by Lemma 5.8, let:

$$
|\mathbf{A}\cap\overline{\mathbf{L}}_i^{(0)}| = \frac{|\overline{\mathbf{L}}_i^{(0)}|}{2} + \xi_0 \qquad \text{and} \qquad |\mathbf{A}\cap\overline{\mathbf{L}}_i^{(1)}| = \frac{|\overline{\mathbf{L}}_i^{(1)}|}{2} + \xi_1, \qquad (5)
$$

where $|\xi_0|, |\xi_1| \leq \sqrt{n}\log n$, which in turn, implies:

$$
|\overline{\mathbf{A}}\cap\overline{\mathbf{L}}_i^{(0)}| = \frac{|\overline{\mathbf{L}}_i^{(0)}|}{2} - \xi_0 \qquad \text{and} \qquad |\overline{\mathbf{A}}\cap\overline{\mathbf{L}}_i^{(1)}| = \frac{|\overline{\mathbf{L}}_i^{(1)}|}{2} - \xi_1. \qquad (6)
$$

Therefore, combining (4) with (5) and (6),

$$
\begin{aligned}
&\mathop{\mathbf{Pr}}_{\boldsymbol{r}\sim\mathcal{V}_{\text{yes}}}\left[\boldsymbol{r}_1^{(i)} = 1 \mid v(i) = \text{“empty set”}\right] \\
&= \frac{1}{\left(\frac{|\overline{\mathbf{L}}_i|}{2} + \xi_0 + \xi_1\right)\left(\frac{|\overline{\mathbf{L}}_i|}{2} - \xi_0 - \xi_1\right)}\left(\left(\frac{|\overline{\mathbf{L}}_i^{(0)}|}{2} + \xi_0\right)\left(\frac{|\overline{\mathbf{L}}_i^{(1)}|}{2} - \xi_1\right) + \left(\frac{|\overline{\mathbf{L}}_i^{(1)}|}{2} + \xi_1\right)\left(\frac{|\overline{\mathbf{L}}_i^{(0)}|}{2} - \xi_0\right)\right) \\
&= \frac{2|\overline{\mathbf{L}}_i^{(0)}| \cdot |\overline{\mathbf{L}}_i^{(1)}| - 8\xi_0\xi_1}{|\overline{\mathbf{L}}_i|^2 - 4\xi_0^2 - 4\xi_1^2 - 8\xi_0\xi_1} = \frac{2|\overline{\mathbf{L}}_i^{(0)}| \cdot |\overline{\mathbf{L}}_i^{(1)}|}{|\overline{\mathbf{L}}_i|^2} + \gamma_{\text{yes}},
\end{aligned}
$$

where $|\gamma_{\text{yes}}| \leq O(\frac{\log^2 n}{n})$, since $|\overline{\mathbf{L}}_i|, |\overline{\mathbf{L}}_i^{(0)}|, |\overline{\mathbf{L}}_i^{(1)}| = \Omega(n)$.

The case when $\boldsymbol{r}\sim\mathcal{V}_{\text{no}}$ is analogous, except that now the underlying graph is the union of two cliques at $\mathbf{A}$ and $\overline{\mathbf{A}}$, so:

$$\begin{aligned}
&\mathop{\mathbf{Pr}}_{\boldsymbol{r}\sim\mathcal{V}_{\text{no}}}\left[\boldsymbol{r}_1^{(i)}=1 \mid v(i)=\text{“empty set”}\right]\\
&=\mathop{\mathbf{Pr}}_{\substack{\mathbf{G}\sim\mathcal{G}_1\\(\boldsymbol{j}_1,\boldsymbol{j}_2)}}\left[\left(\boldsymbol{j}_1\in\mathbf{A}\cap\overline{\mathbf{L}}_i^{(0)}\wedge\boldsymbol{j}_2\in\mathbf{A}\cap\overline{\mathbf{L}}_i^{(1)}\right)\vee\left(\boldsymbol{j}_1\in\overline{\mathbf{A}}\cap\overline{\mathbf{L}}_i^{(0)}\wedge\boldsymbol{j}_2\in\overline{\mathbf{A}}\cap\overline{\mathbf{L}}_i^{(1)}\right)\mid v(i)=\text{“empty set”}\right],\\
&=\frac{1}{\binom{|\mathbf{A}\cap\overline{\mathbf{L}}_i|}{2}+\binom{|\overline{\mathbf{A}}\cap\overline{\mathbf{L}}_i|}{2}}\cdot\left(|\mathbf{A}\cap\overline{\mathbf{L}}_i^{(0)}|\cdot|\mathbf{A}\cap\overline{\mathbf{L}}_i^{(1)}|+|\overline{\mathbf{A}}\cap\overline{\mathbf{L}}_i^{(0)}|\cdot|\overline{\mathbf{A}}\cap\overline{\mathbf{L}}_i^{(1)}|\right)\\
&=\frac{1}{\binom{\frac{|\overline{\mathbf{L}}_i|}{2}+\xi_0+\xi_1}{2}+\binom{\frac{|\overline{\mathbf{L}}_i|}{2}-\xi_0-\xi_1}{2}}\left(\left(\frac{|\overline{\mathbf{L}}_i^{(0)}|}{2}+\xi_0\right)\left(\frac{|\overline{\mathbf{L}}_i^{(1)}|}{2}+\xi_1\right)+\left(\frac{|\overline{\mathbf{L}}_i^{(0)}|}{2}-\xi_0\right)\left(\frac{|\overline{\mathbf{L}}_i^{(1)}|}{2}-\xi_1\right)\right)\\
&=\frac{2|\overline{\mathbf{L}}_i^{(0)}|\cdot|\overline{\mathbf{L}}_i^{(1)}|}{|\overline{\mathbf{L}}_i|^2}+\gamma_{\text{no}},
\end{aligned}$$

were again, $|\gamma_{\text{no}}|\leq O(\frac{\log^2 n}{n})$. ■

Lemma 5.12. *For every $i\in[t]$ with $v(i)=$ “lone vertex”, let $\boldsymbol{j}_2\in\overline{M}$ be the lone vertex observed and let $w=\neg(z_1^{(i)})_{\boldsymbol{j}_2}$. There exists $|\gamma'_{yes}|,|\gamma'_{no}|\leq O(\frac{\log n}{\sqrt{n}})$ such that for $\boldsymbol{r}\sim\mathcal{V}'_{yes}$ satisfies*

$$\boldsymbol{r}_1^{(i)}\oplus(z_1^{(i)})_{\boldsymbol{j}_3}\sim\begin{cases}\text{Ber}(p_v(\overline{\mathbf{L}}_i^{(w)})+\gamma'_{yes}) & \boldsymbol{j}_3=m_1\\ \text{Ber}(1-p_v(\overline{\mathbf{L}}_i^{(w)})-\gamma'_{yes}) & \boldsymbol{j}_3=m_2\end{cases},$$

and $\boldsymbol{r}\sim\mathcal{V}'_{no}$ satisfies

$$\boldsymbol{r}_1^{(i)}\oplus(z_1^{(i)})_{\boldsymbol{j}_3}\sim\begin{cases}\text{Ber}(p_v(\overline{\mathbf{L}}_i^{(w)})+\gamma'_{no}) & \boldsymbol{j}_3=m_1\\ \text{Ber}(1-p_v(\overline{\mathbf{L}}_i^{(w)})-\gamma'_{no}) & \boldsymbol{j}_3=m_2\end{cases}.$$

Proof: We follow a similar strategy to Lemma 5.11, where we know that we sample an edge $(\boldsymbol{j}_1,\boldsymbol{j}_2)\sim\mathbf{G}$ whose value of $\boldsymbol{j}_2\in\mathbf{L}_i$, and $\boldsymbol{j}_1\notin\mathbf{L}_i$. Consider for simplicity the case when $\mathbf{G}$ is a complete bipartite graph with sides $\mathbf{A}$ and $\overline{\mathbf{A}}$, and $\boldsymbol{j}_3=m_1$ and $(z_1^{(i)})_{m_1}=0$.

Similarly to (4), we have that in order for $\boldsymbol{r}_1^{(i)}=1$, we must have $(z_1^{(i)})_{\boldsymbol{j}_1}\neq(z_1^{(i)})_{\boldsymbol{j}_2}$. Suppose that $\boldsymbol{j}_2\in\mathbf{A}$ and $w=\neg(z_1^{(i)})_{\boldsymbol{j}_2}$, then in order for $\boldsymbol{r}_1^{(i)}=1$, $\boldsymbol{j}_1$ must have been sampled from $\overline{\mathbf{A}}\cap\overline{\mathbf{L}}_i^{(w)}$. Using Lemma 5.8, we have that there exists $|\xi_0|,|\xi_1|\leq\sqrt{n}\log n$ so:

$$\mathop{\mathbf{Pr}}_{\boldsymbol{r}\sim\mathcal{V}'_{\text{yes}}}[\boldsymbol{r}_1^{(i)}=1\mid v(i)=\text{“lone vertex”}]=\frac{|\overline{\mathbf{A}}\cap\overline{\mathbf{L}}_i^{(w)}|}{|\overline{\mathbf{A}}\cap\overline{\mathbf{L}}|}=\frac{|\overline{\mathbf{L}}_i^{(w)}|/2-\xi_w}{|\overline{\mathbf{L}}_i|/2-\xi_0-\xi_1}\approx\frac{|\overline{\mathbf{L}}_i^{(w)}|}{|\overline{\mathbf{L}}_i|}\pm O(\tfrac{\log n}{\sqrt{n}}),$$

where we used the fact that $|\mathbf{L}_i|,|\mathbf{L}_i^{(w)}|=\Omega(n)$. If $\boldsymbol{j}_2\in\overline{\mathbf{A}}$, then

$$\mathop{\mathbf{Pr}}_{\boldsymbol{r}\sim\mathcal{V}'_{\text{yes}}}[\boldsymbol{r}_1^{(i)}=1\mid v(i)=\text{“lone vertex”}]=\frac{|\mathbf{A}\cap\overline{\mathbf{L}}_i^{(w)}|}{|\mathbf{A}\cap\overline{\mathbf{L}}|}=\frac{|\overline{\mathbf{L}}_i^{(w)}|/2+\xi_w}{|\overline{\mathbf{L}}_i|/2+\xi_0+\xi_1}\approx\frac{|\overline{\mathbf{L}}_i^{(w)}|}{|\overline{\mathbf{L}}_i|}\pm O(\tfrac{\log n}{\sqrt{n}}).$$

In both cases, we have that $\boldsymbol{r}_1^{(i)}\sim\text{Ber}(p_v(\overline{\mathbf{L}}_i^{(w)})\pm O(\frac{\log n}{\sqrt{n}}))$, and when we have $(z_1^{(i)})_{m_1}=1$, we simply flip the answer. Likewise, when $\boldsymbol{j}_3=m_2$, we flip the answer once more.

In the case of $\mathbf{G}$ being the union of two cliques at $\mathbf{A}$ and $\overline{\mathbf{A}}$, when $\boldsymbol{j}_3 = m_1$ and $(z_1^{(i)})_{m_1} = 0$, we have that when $\boldsymbol{j}_2 \in \mathbf{A}$,

$$\underset{\boldsymbol{r}\sim\mathcal{V}'_{\text{no}}}{\mathbf{Pr}}[\boldsymbol{r}_1^{(i)} = 1 \mid v(i) = \text{“lone vertex”}] = \frac{|\mathbf{A} \cap \overline{\mathbf{L}}_i^{(w)}|}{|\mathbf{A} \cap \overline{\mathbf{L}}|} = \frac{|\overline{\mathbf{L}}_i^{(w)}|/2 + \xi_w}{|\overline{\mathbf{L}}_i|/2 + \xi_0 + \xi_1} \approx \frac{|\overline{\mathbf{L}}_i^{(w)}|}{|\overline{\mathbf{L}}_i|} \pm O(\tfrac{\log n}{\sqrt{n}}),$$

and when $\boldsymbol{j}_2 \in \overline{\mathbf{A}}$,

$$\underset{\boldsymbol{r}\sim\mathcal{V}'_{\text{no}}}{\mathbf{Pr}}[\boldsymbol{r}_1^{(i)} = 1 \mid v(i) = \text{“lone vertex”}] = \frac{|\overline{\mathbf{A}} \cap \overline{\mathbf{L}}_i^{(w)}|}{|\overline{\mathbf{A}} \cap \overline{\mathbf{L}}|} = \frac{|\overline{\mathbf{L}}_i^{(w)}|/2 - \xi_w}{|\overline{\mathbf{L}}_i|/2 - \xi_0 - \xi_1} \approx \frac{|\overline{\mathbf{L}}_i^{(w)}|}{|\overline{\mathbf{L}}_i|} \pm O(\tfrac{\log n}{\sqrt{n}}),$$

so we obtain the analogous conclusion. ■

We note that after defining $\boldsymbol{r}_1^{(i)}$ in the cases with $v(i) =$ “empty set”, we have that all values $\boldsymbol{r}_\alpha^{(i)}$ are determined by flipping the answer when $(z_\alpha^{(i)})_{\boldsymbol{j}_3} \neq (z_1^{(i)})_{\boldsymbol{j}_3}$. Likewise, after defining $\boldsymbol{r}_1^{(i)}$ in the cases with $v(i) =$ “lone vertex”, we have that all values $\boldsymbol{r}_\alpha^{(i)}$ are determined by flipping the answer when $(z_\alpha^{(i)})_{\boldsymbol{j}_3} \neq (z_1^{(i)})_{\boldsymbol{j}_3}$ and when $(z_\alpha^{(i)})_{\boldsymbol{j}_2} \neq (z_1^{(i)})_{\boldsymbol{j}_2}$.

Finally, consider the indices $i \in [t]$ of responses $\boldsymbol{r}_\alpha^{(i)}$ with $v(i) =$ “empty set”, and call these E. We have that for all $i \in E$, $\mathcal{U}'_{\text{yes}}$ and $\mathcal{U}'_{\text{no}}$ outputs bits which equal 1 with probability τ_i where $\tau_i = \Omega(1)$, and $\mathcal{V}'_{\text{yes}}$ and $\mathcal{V}'_{\text{no}}$ outputs bits which equal 1 with probability $\tau_i \pm O(\frac{\log^2 n}{n})$. Since these groups are independent and there at at most $q \ll n^{1.5}$ groups, we have that the bits $(\boldsymbol{r}_1^{(i)})_{i\in E} \sim \mathcal{U}'_{\text{yes}}$ (and also $\mathcal{U}'_{\text{no}}$) satisfy:

$$(\boldsymbol{r}_1^{(i)})_{i\in E} \sim \prod_{i\in E} \text{Ber}(\tau_i),$$

and for each $i \in E$, there exists $\gamma_{i,\text{yes}}$ and $\gamma_{i,\text{no}}$ with $|\gamma_{i,\text{yes}}|, |\gamma_{i,\text{no}}| = O(\frac{\log^2 n}{n})$ such that $(\boldsymbol{r}_1^{(i)})_{i\in E} \sim \mathcal{V}'_{\text{yes}}$ satisfies

$$(\boldsymbol{r}_1^{(i)})_{i\in E} \sim \prod_{i\in E} \text{Ber}(\tau_i + \gamma_{i,\text{yes}}),$$

and if $(\boldsymbol{r}_1^{(i)})_{i\in E} \sim \mathcal{V}'_{\text{no}}$ satisfies

$$(\boldsymbol{r}_1^{(i)})_{i\in E} \sim \prod_{i\in E} \text{Ber}(\tau_i + \gamma_{i,\text{no}}).$$

Thus, by [Roo01], we have that the distance in total variation between these two distributions is at most $o(1)$.

Similarly, we consider the indices $i \in [t]$ with $v(i) =$ “lone vertex”, and call these V. By Lemma 5.9, we have that $|V| \leq \frac{n}{\log^4 n}$ with probability $1 - o(1)$ if the cost of the rejection sampling algorithm is less than $\frac{n^2}{\log^6 n}$. So similarly to the case with the groups in E, these can only incur at most $o(1)$ in distance in total variation.

6 A lower bound for distinguishing $\mathcal{G}_1$ and $\mathcal{G}_2$ with rejection samples

In this section, we derive a lower bound for distinguishing $\mathcal{G}_1$ and $\mathcal{G}_2$ with rejection samples.

Lemma 6.1. *Any deterministic non-adaptive algorithm* Alg *with* $\mathrm{cost}(\mathrm{Alg}) \leq \frac{n^2}{\log^6 n}$*, has:*

$$\Pr_{\mathbf{G} \sim \mathcal{G}_1}[\text{Alg } \textit{outputs "}\mathcal{G}_1\textit{"}] \leq (1 + o(1)) \Pr_{\mathbf{G} \sim \mathcal{G}_2}[\text{Alg } \textit{outputs "}\mathcal{G}_1\textit{"}] + o(1).$$

We assume Alg is a deterministic non-adaptive algorithm with $\mathrm{cost}(\mathrm{Alg}) \leq \frac{n^2}{\log^6 n}$. Alg makes queries $L_1, \dots, L_t \subset [n]$ and the oracle returns $\boldsymbol{v}_1, \dots, \boldsymbol{v}_t$, some of which are edges, some are lone vertices, and some are $\emptyset$. Let $\mathbf{G}_o \subset \mathbf{G}$ be the graph observed by the algorithm by considering all edges in $\boldsymbol{v}_1, \dots, \boldsymbol{v}_t$. We let $|\mathbf{G}_o|$ be the number of edges.

Before going on to prove the lower bound, we use the following simplification. First, we assume that any algorithm Alg has all its queries $L_1, \dots, L_t$ satisfying that either $|L_i| \leq \frac{n}{\log n}$, or $L_i = [n]$. Thus, it suffices to show for this restricted class of algorithms, the cost must be at least $\frac{n^2}{\log^5 n}$.

6.1 High Level Overview

In this subsection, we will give a high level overview of the proof of Lemma 6.1.

The idea is that we will argue outcome-by-outcome; i.e., we consider the possible ways the algorithm can act, which depends on the responses to the queries the algorithm gets. Consider some responses $v_1, \dots, v_t \in [n] \cup ([n] \times [n]) \cup \{\emptyset\}$, where each v_i may be either a lone vertex, an edge, or $\emptyset$. Suppose that upon observing this outcome, the algorithm outputs "$\mathcal{G}_1$". There will be two cases:

- The first case is when the probability of observing this outcome from $\mathcal{G}_2$ is not too much lower than the probability of observing this outcome from $\mathcal{G}_1$. In these outcomes, we will not get too much advantage in distinguishing $\mathcal{G}_1$ and $\mathcal{G}_2$.
- The other case is when the probability of observing this outcome from $\mathcal{G}_2$ is substantially lower than the probability of observing this outcome from $\mathcal{G}_1$. These cases do help us distinguish between $\mathcal{G}_1$ and $\mathcal{G}_2$; thus, we will want to show that collectively, the probability that we observe these outcomes from $\mathcal{G}_1$ is $o(1)$.

We will be able to characterize the outcomes which fall into the first case and the second case by considering a sequence of events. In particular we define five events which depend on $v_1, \dots, v_t$, as well as the random choice of $\mathbf{A}$. Consider the outcome $v_1, \dots, v_t$ which together form components $C_1, \dots, C_\alpha$. The events are the following[10]:

1. $\mathcal{E}_T$ (Observe small trees): this is the event where the values of $v_1, \dots, v_t$ form components $C_1, \dots, C_\alpha$ which are all trees of size at most $\log n$.
2. $\mathcal{E}_F$ (Observe few non-empty responses): this is the event where the values of $v_1, \dots, v_t$ have at most $\frac{n}{\log^4 n}$ non-$\emptyset$ responses. This event implies that the total number of vertices in the responses $v_1, \dots, v_t$ is at most $\frac{n}{\log^4 n}$.

[10] We note that the first two event are not random and depends on the values $v_1, \dots, v_t$, and the rest are random variables depending on the partition $\mathbf{A}$ and the oracle responses $\boldsymbol{v}_1, \dots, \boldsymbol{v}_t$.

3. $\boldsymbol{\mathcal{E}}_{C,\text{yes}}$ and $\boldsymbol{\mathcal{E}}_{C,\text{no}}$ (Consistency condition of the components observed): these are the events where $\mathbf{A} \subset [n]$ partitions the components $C_1, \ldots, C_\alpha$ in a manner consistent with $\mathcal{G}_1$ in $\boldsymbol{\mathcal{E}}_{C,\text{yes}}$ or $\mathcal{G}_2$ in $\boldsymbol{\mathcal{E}}_{C,\text{no}}$. See Definition 6.5 for a formal definition of this event. These events are random variables that depend only on $\mathbf{A}$. It will become clear that in order to observe the outcome $v_1, \ldots, v_t$ in $\mathcal{G}_1$, event $\boldsymbol{\mathcal{E}}_{C,\text{yes}}$ must be triggered, and in $\mathcal{G}_2$, event $\boldsymbol{\mathcal{E}}_{C,\text{no}}$ must be triggered. See Figure 7 for an illustration.

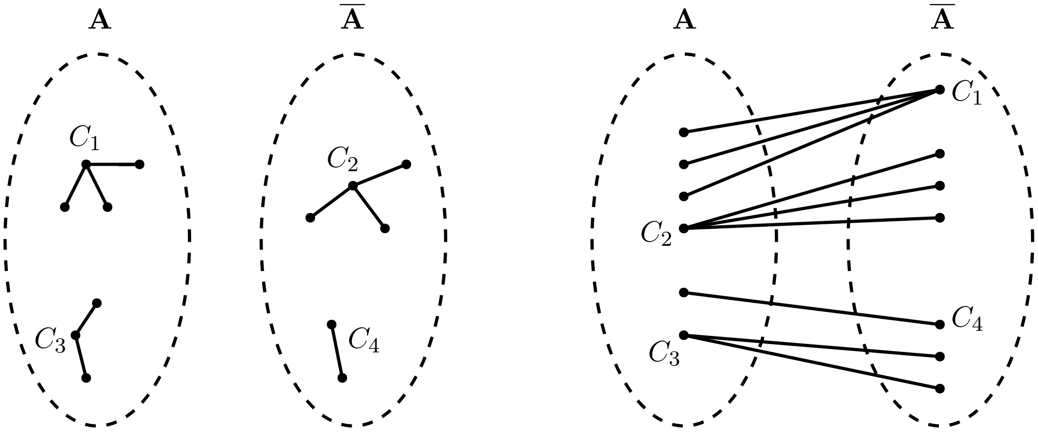

Figure 7: $\mathbf{A}$ consistently partition of the components C_1, C_2, C_3 and C_4 according to $\mathcal{G}_1$ (on the left) and $\mathcal{G}_2$ (on the right).

4. $\boldsymbol{\mathcal{E}}_O$ (Observe specific responses) : this event is over the randomness in $\mathbf{A}$, as well as the randomness in the responses of the oracle $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_t$. The event is triggered when the responses of the oracle are exactly those dictated by $v_1, \ldots, v_t$; i.e., for all $i \in [t]$, $\boldsymbol{v}_i = v_i$.

5. $\boldsymbol{\mathcal{E}}_B$ (Balanced lone vertices condition) : this event is over the randomness in $\mathbf{A}$, as well as the responses $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_t$. The event occurs when a particular quantity which depends on $\mathbf{A}$ and $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_t$ is bounded by some predetermined value. See Definition 6.15 for a formal definition.

Having defined these events, the lower bound follows by the following three lemmas. The first lemma says that for any outcomes satisfying $\mathcal{E}_T$ and $\mathcal{E}_F$, the probability over $\mathbf{A}$ of being consistent in $\mathcal{G}_1$ cannot be much higher than in $\mathcal{G}_2$. The second lemma says that the outcomes satisfying the events described above do not help in distinguishing $\mathcal{G}_1$ and $\mathcal{G}_2$. The third lemma says that good outcomes occur with high probability over $\mathcal{G}_1$.

Lemma 6.2 (Consistency Lemma)**.** *Consider a fixed $v_1, \ldots, v_t \in [n] \cup ([n] \times [n]) \cup \{\emptyset\}$ forming components $C_1, \ldots, C_\alpha$ where events $\mathcal{E}_T$ and $\mathcal{E}_F$ are satisfied. Then, we have:*

$$\mathop{\mathbf{Pr}}_{\substack{\mathbf{G} \sim \mathcal{G}_1 \\ \boldsymbol{v}_1, \ldots, \boldsymbol{v}_t}} [\boldsymbol{\mathcal{E}}_{C,yes}] \leq (1 + o(1)) \mathop{\mathbf{Pr}}_{\substack{\mathbf{G} \sim \mathcal{G}_2 \\ \boldsymbol{v}_1, \ldots, \boldsymbol{v}_t}} [\boldsymbol{\mathcal{E}}_{C,no}].$$

Lemma 6.3 (Good Outcomes Lemma)**.** *Consider a fixed $v_1, \ldots, v_t \in [n] \cup ([n] \times [n]) \cup \{\emptyset\}$ forming components $C_1, \ldots, C_\alpha$ where events $\mathcal{E}_T$ and $\mathcal{E}_F$ are satisfied. Then, we have:*

$$\mathop{\mathbf{Pr}}_{\substack{\mathbf{G} \sim \mathcal{G}_1 \\ \boldsymbol{v}_1, \ldots, \boldsymbol{v}_t}} [\boldsymbol{\mathcal{E}}_O \wedge \boldsymbol{\mathcal{E}}_B \mid \boldsymbol{\mathcal{E}}_{C,yes}] \leq (1 + o(1)) \mathop{\mathbf{Pr}}_{\substack{\mathbf{G} \sim \mathcal{G}_2 \\ \boldsymbol{v}_1, \ldots, \boldsymbol{v}_t}} [\boldsymbol{\mathcal{E}}_O \mid \boldsymbol{\mathcal{E}}_{C,no}].$$

Lemma 6.4 (Bad Outcomes Lemma). *We have that:*

$$\Pr_{\substack{\mathbf{G}\sim\mathcal{G}_1\\ \boldsymbol{v}_1,\dots,\boldsymbol{v}_t}}\left[\neg\boldsymbol{\mathcal{E}}_T \vee \neg\boldsymbol{\mathcal{E}}_F \vee \neg\boldsymbol{\mathcal{E}}_B\right] = o(1).$$

Assuming the above three lemmas, we may prove Lemma 6.1.

Proof: Let Λ be the set of outcomes of the algorithm which output "$\mathcal{G}_1$." Each outcome is a collection of responses $v_1, \dots, v_t$. We let

$$\Lambda_G = \{\ell \in \Lambda : \text{ responses } v_1, \dots, v_t \text{ satisfy } \mathcal{E}_T \wedge \mathcal{E}_F\},$$

and $\boldsymbol{\mathcal{E}}_{O,\ell}$ be the event that responses $\boldsymbol{v}_1, \dots, \boldsymbol{v}_t$ result in outcome ℓ. We have:

$$\begin{aligned}
\Pr_{\substack{\mathbf{G}\sim\mathcal{G}_1\\ \boldsymbol{v}_1,\dots,\boldsymbol{v}_t}}[\text{Alg outputs "}\mathcal{G}_1\text{"}] &\le \sum_{\ell\in\Lambda_G} \Pr_{\substack{\mathbf{G}\sim\mathcal{G}_1\\ \boldsymbol{v}_1,\dots,\boldsymbol{v}_t}}[\ell \text{ is observed by Alg} \mid \boldsymbol{\mathcal{E}}_{C,\text{yes}}] \Pr_{\substack{\mathbf{G}\sim\mathcal{G}_1\\ \boldsymbol{v}_1,\dots,\boldsymbol{v}_t}}[\boldsymbol{\mathcal{E}}_{C,\text{yes}}] + \Pr_{\substack{\mathbf{G}\sim\mathcal{G}_1\\ \boldsymbol{v}_1,\dots,\boldsymbol{v}_t}}[\neg\boldsymbol{\mathcal{E}}_T \vee \neg\boldsymbol{\mathcal{E}}_F] \\
&\le \sum_{\ell\in\Lambda_G} \Pr_{\substack{\mathbf{G}\sim\mathcal{G}_1\\ \boldsymbol{v}_1,\dots,\boldsymbol{v}_t}}[\boldsymbol{\mathcal{E}}_{O,\ell} \wedge \boldsymbol{\mathcal{E}}_B \mid \boldsymbol{\mathcal{E}}_{C,\text{yes}}] \Pr_{\substack{\mathbf{G}\sim\mathcal{G}_1\\ \boldsymbol{v}_1,\dots,\boldsymbol{v}_t}}[\boldsymbol{\mathcal{E}}_{C,\text{yes}}] + \Pr_{\substack{\mathbf{G}\sim\mathcal{G}_1\\ \boldsymbol{v}_1,\dots,\boldsymbol{v}_t}}[\neg\boldsymbol{\mathcal{E}}_T \vee \neg\boldsymbol{\mathcal{E}}_F \vee \neg\boldsymbol{\mathcal{E}}_B] \\
&\le (1+o(1)) \sum_{\ell\in\Lambda_G} \Pr_{\substack{\mathbf{G}\sim\mathcal{G}_2\\ \boldsymbol{v}_1,\dots,\boldsymbol{v}_t}}[\boldsymbol{\mathcal{E}}_{O,\ell} \mid \boldsymbol{\mathcal{E}}_{C,\text{no}}] \Pr_{\substack{\mathbf{G}\sim\mathcal{G}_2\\ \boldsymbol{v}_1,\dots,\boldsymbol{v}_t}}[\boldsymbol{\mathcal{E}}_{C,\text{no}}] + o(1) \\
&\le (1+o(1)) \Pr_{\substack{\mathbf{G}\sim\mathcal{G}_2\\ \boldsymbol{v}_1,\dots,\boldsymbol{v}_t}}[\text{Alg outputs "}\mathcal{G}_1\text{"}] + o(1),
\end{aligned}$$

where we used Lemma 6.2, Lemma 6.3, and Lemma 6.4 from the second to third line. ■

6.2 Proof of the Consistency Lemma: Lemma 6.2

We now turn to proving Lemma 6.2. We first give the formal definitions of events $\boldsymbol{\mathcal{E}}_{C,\text{yes}}$ and $\boldsymbol{\mathcal{E}}_{C,\text{no}}$. Next, we set up some definitions necessary for the proof and give two claims which imply the lemma. For the remainder of the section, we consider fixing the responses $v_1, \dots, v_t \in [n] \cup ([n] \times [n]) \cup \{\emptyset\}$. We assume the responses form the components $C_1, \dots, C_\alpha$ which satisfy events $\mathcal{E}_T$ and $\mathcal{E}_F$. For each $i \in [\alpha]$, let u_i be the minimum vertex in C_i with respect to the natural ordering of $[n]$, and consider rooting the trees C_i at u_i, forming a layered tree with at most $\log n$ layers. Namely, u_i will be in the first layer, all its neighbors in C_i will be in the second layer, and so on. We let $C_i(\text{even})$ be the set of vertices in even layers, and $C_i(\text{odd})$ be the set of vertices in odd layers.

Definition 6.5. *We let $\boldsymbol{\mathcal{E}}_{C,yes}$ be the event that $\mathbf{A} \subset [n]$ is consistent with the observations $v_1, \dots, v_t$ when $\mathbf{G} = K_{\mathbf{A}} \cup K_{\overline{\mathbf{A}}}$, and $\boldsymbol{\mathcal{E}}_{C,no}$ be the event that $\mathbf{A} \subset [n]$ is consistent with the observations $v_1, \dots, v_t$ when $\mathbf{G} = K_{\mathbf{A},\overline{\mathbf{A}}}$. In other words,*

- *In $\boldsymbol{\mathcal{E}}_{C,yes}$: for all $i \in [\alpha]$, either $C_i \subset \mathbf{A}$ or $C_i \subset \overline{\mathbf{A}}$.*
- *In $\boldsymbol{\mathcal{E}}_{C,no}$: for all $i \in [\alpha]$, either $C_i(odd) \subset \mathbf{A}$ and $C_i(even) \subset \overline{\mathbf{A}}$, or $C_i(odd) \subset \overline{\mathbf{A}}$ and $C_i(even) \subset \mathbf{A}$.*

For each $i \in [\alpha]$, let $\mathbf{Y}_i$ be the indicator random variable for $u_i \in \mathbf{A}$. Let:

$$\mathbf{W}_{A,\text{yes}} = \sum_{i=1}^{\alpha} \mathbf{Y}_i \cdot |C_i| \qquad \mathbf{W}_{A,\text{no}} = \sum_{i=1}^{\alpha} \left(\mathbf{Y}_i \cdot |C_i(\text{odd})| + (1 - \mathbf{Y}_i) \cdot |C_i(\text{even})|\right) \qquad V = \sum_{i=1}^{\alpha} |C_i|.$$

Definition 6.6. *We let $\boldsymbol{\mathcal{E}}_W$ be the event where:*

$$\frac{V}{2} - \sqrt{V} \log n \leq \mathbf{W}_{A,no} \leq \frac{V}{2} + \sqrt{V} \log n.$$

Lemma 6.2 follows from the next two claims.

Claim 6.7. *For $v_1, \dots, v_t$ satisfying events $\mathcal{E}_T$ and $\mathcal{E}_F$, we have:*

$$\mathop{\mathbf{Pr}}_{\substack{\mathbf{G} \sim \mathcal{G}_1 \\ \boldsymbol{v}_1, \dots, \boldsymbol{v}_t}} [\boldsymbol{\mathcal{E}}_{C,yes} \wedge \boldsymbol{\mathcal{E}}_W] \leq (1 + o(1)) \mathop{\mathbf{Pr}}_{\substack{\mathbf{G} \sim \mathcal{G}_2 \\ \boldsymbol{v}_1, \dots, \boldsymbol{v}_t}} [\boldsymbol{\mathcal{E}}_{C,no}].$$

Claim 6.8. *For $v_1, \dots, v_t$ satisfying events $\mathcal{E}_T$ and $\mathcal{E}_F$, we have:*

$$\mathop{\mathbf{Pr}}_{\substack{\mathbf{G} \sim \mathcal{G}_1 \\ \boldsymbol{v}_1, \dots, \boldsymbol{v}_t}} [\neg \boldsymbol{\mathcal{E}}_W \mid \boldsymbol{\mathcal{E}}_{C,yes}] = o(1).$$

Given Claim 6.7 and Claim 6.8, we proceed to proving Lemma 6.2.

Proof of Lemma 6.2: We simply compute the respective probabilities.

$$\begin{aligned}
\mathop{\mathbf{Pr}}_{\substack{\mathbf{G} \sim \mathcal{G}_1 \\ \boldsymbol{v}_1, \dots, \boldsymbol{v}_t}} [\boldsymbol{\mathcal{E}}_{C,\text{yes}}] &= \mathop{\mathbf{Pr}}_{\substack{\mathbf{G} \sim \mathcal{G}_1 \\ \boldsymbol{v}_1, \dots, \boldsymbol{v}_t}} [\boldsymbol{\mathcal{E}}_{C,\text{yes}} \wedge \boldsymbol{\mathcal{E}}_W] + \mathop{\mathbf{Pr}}_{\substack{\mathbf{G} \sim \mathcal{G}_1 \\ \boldsymbol{v}_1, \dots, \boldsymbol{v}_t}} [\neg \boldsymbol{\mathcal{E}}_W \mid \boldsymbol{\mathcal{E}}_{C,\text{yes}}] \mathop{\mathbf{Pr}}_{\substack{\mathbf{G} \sim \mathcal{G}_1 \\ \boldsymbol{v}_1, \dots, \boldsymbol{v}_t}} [\boldsymbol{\mathcal{E}}_{C,\text{yes}}] \\
&\leq (1 + o(1)) \mathop{\mathbf{Pr}}_{\substack{\mathbf{G} \sim \mathcal{G}_2 \\ \boldsymbol{v}_1, \dots, \boldsymbol{v}_t}} [\boldsymbol{\mathcal{E}}_{C,\text{no}}] + o(1) \mathop{\mathbf{Pr}}_{\substack{\mathbf{G} \sim \mathcal{G}_1 \\ \boldsymbol{v}_1, \dots, \boldsymbol{v}_t}} [\boldsymbol{\mathcal{E}}_{C,\text{yes}}], \qquad (7)
\end{aligned}$$

Where we applied both Claim 6.7 and Claim 6.8 in Line (7). Finally, this implies:

$$(1 - o(1)) \mathop{\mathbf{Pr}}_{\substack{\mathbf{G} \sim \mathcal{G}_1 \\ \boldsymbol{v}_1, \dots, \boldsymbol{v}_t}} [\boldsymbol{\mathcal{E}}_{C,\text{yes}}] \leq (1 + o(1)) \mathop{\mathbf{Pr}}_{\substack{\mathbf{G} \sim \mathcal{G}_2 \\ \boldsymbol{v}_1, \dots, \boldsymbol{v}_t}} [\boldsymbol{\mathcal{E}}_{C,\text{no}}],$$

which finishes the proof. ■

We now proceed to proving Claim 6.7, followed by the proof of Claim 6.8.

Proof of Claim 6.7: Note that $V \leq \frac{n}{\log^4 n}$ since event $\mathcal{E}_F$ is satisfied. Let $y \in \{0,1\}^\alpha$ be an assignment of $u_1, \dots, u_\alpha$ to $\mathbf{A}$; more formally, for a fixed $y \in \{0,1\}^\alpha$, we let $\boldsymbol{\mathcal{E}}_y$ be the event that for each $i \in [\alpha]$, $u_i \in \mathbf{A}$ if $y_i = 1$, and $u_i \in \overline{\mathbf{A}}$ if $y_i = 0$. Additionally, let

$$Y_G = \{y \in \{0,1\}^\alpha : \text{ if } \mathbf{A} \text{ satisfies } \boldsymbol{\mathcal{E}}_y, \text{ then } \boldsymbol{\mathcal{E}}_W \text{ is satisfied}\}.$$

Then,

$$\mathop{\mathbf{Pr}}_{\substack{\mathbf{G} \sim \mathcal{G}_1 \\ \boldsymbol{v}_1, \dots, \boldsymbol{v}_t}} [\boldsymbol{\mathcal{E}}_{C,\text{yes}} \wedge \boldsymbol{\mathcal{E}}_W] = \sum_{y \in Y_G} \mathop{\mathbf{Pr}}_{\substack{\mathbf{G} \sim \mathcal{G}_1 \\ \boldsymbol{v}_1, \dots, \boldsymbol{v}_t}} [\boldsymbol{\mathcal{E}}_{C,\text{yes}} \wedge \boldsymbol{\mathcal{E}}_y].$$

It suffices to show that for $y \in Y_G$:

$$\underset{\substack{\mathbf{G}\sim\mathcal{G}_1\\ \boldsymbol{v}_1,\dots,\boldsymbol{v}_t}}{\mathbf{Pr}}[\boldsymbol{\mathcal{E}}_{C,\text{yes}} \wedge \boldsymbol{\mathcal{E}}_y] \leq (1+o(1)) \underset{\substack{\mathbf{G}\sim\mathcal{G}_2\\ \boldsymbol{v}_1,\dots,\boldsymbol{v}_t}}{\mathbf{Pr}}[\boldsymbol{\mathcal{E}}_{C,\text{no}} \wedge \boldsymbol{\mathcal{E}}_y].$$

Note that if $\mathbf{A}$ satisfies $\boldsymbol{\mathcal{E}}_y$ and $\boldsymbol{\mathcal{E}}_{C,\text{yes}}$ is satisfied, there is precisely one choice for assigning each vertex in $C_1, \dots, C_\alpha$ to $\mathbf{A}$ or $\overline{\mathbf{A}}$. Likewise, if $\mathbf{A}$ satisfied $\boldsymbol{\mathcal{E}}_y$ and $\boldsymbol{\mathcal{E}}_{C,\text{no}}$, there is precisely one choice for assigning each vertex in $C_1, \dots, C_\alpha$ to $\mathbf{A}$ or $\overline{\mathbf{A}}$. The remaining vertices may be placed in $\mathbf{A}$ or $\overline{\mathbf{A}}$ so the resulting set $\mathbf{A}$ contains half of all vertices, therefore, we have:

$$\underset{\substack{\mathbf{G}\sim\mathcal{G}_1\\ \boldsymbol{v}_1,\dots,\boldsymbol{v}_t}}{\mathbf{Pr}}[\boldsymbol{\mathcal{E}}_{C,\text{yes}} \wedge \boldsymbol{\mathcal{E}}_y] \leq \frac{\binom{n-V}{\frac{n}{2}-\frac{V}{2}}}{\binom{n}{n/2}} \qquad \underset{\substack{\mathbf{G}\sim\mathcal{G}_2\\ \boldsymbol{v}_1,\dots,\boldsymbol{v}_t}}{\mathbf{Pr}}[\boldsymbol{\mathcal{E}}_{C,\text{no}} \wedge \boldsymbol{\mathcal{E}}_y] \geq \frac{\binom{n-V}{\frac{n}{2}-\frac{V}{2}-\sqrt{V}\log n}}{\binom{n}{n/2}}.$$

Taking the ratio, we have:

$$\begin{aligned}\frac{\mathbf{Pr}[\boldsymbol{\mathcal{E}}_{C,\text{yes}} \wedge \boldsymbol{\mathcal{E}}_y]}{\mathbf{Pr}[\boldsymbol{\mathcal{E}}_{C,\text{no}} \wedge \boldsymbol{\mathcal{E}}_y]} &\leq \frac{\binom{n-V}{\frac{n}{2}-\frac{V}{2}}}{\binom{n}{n/2}} \cdot \frac{\binom{n}{n/2}}{\binom{n-V}{\frac{n}{2}-\frac{V}{2}-\sqrt{V}\log n}} \leq \left(\frac{\frac{n}{2}-\frac{V}{2}+\sqrt{V}\log n}{\frac{n}{2}-\frac{V}{2}-\sqrt{V}\log n}\right)^{\sqrt{V}\log n} \\ &\leq \left(1+O\left(\frac{1}{\sqrt{n}\log n}\right)\right)^{\sqrt{n}/\log n} = 1+o(1).\end{aligned}$$

■

Proof of Claim 6.8: We let:

$$\mathbf{W}_{A,\text{no}}^{(O)} = \sum_{i=1}^{\alpha} \mathbf{Y}_i \cdot |C_i(\text{odd})| \qquad \text{and} \qquad \mathbf{W}_{A,\text{no}}^{(E)} = \sum_{i=1}^{\alpha}(1-\mathbf{Y}_i)\cdot|C_i(\text{even})|.$$

where $\mathbf{W}_{A,\text{no}}^{(O)} + \mathbf{W}_{A,\text{no}}^{(E)} = \mathbf{W}_{A,\text{no}}$ specifies the number of vertices in $\cup_{i\in[\alpha]} C_i$ assigned to $\mathbf{A}$. Conditioning on event $\boldsymbol{\mathcal{E}}_{C,\text{yes}}$, $\mathbf{A}$ and $\overline{\mathbf{A}}$ can be interchanged, so

$$\underset{\substack{\mathbf{G}\sim\mathcal{G}_1\\ \boldsymbol{v}_1,\dots,\boldsymbol{v}_t}}{\mathbf{Pr}}[\mathbf{Y}_i = 1 \mid \boldsymbol{\mathcal{E}}_{C,\text{yes}}] = \underset{\substack{\mathbf{G}\sim\mathcal{G}_1\\ \boldsymbol{v}_1,\dots,\boldsymbol{v}_t}}{\mathbf{Pr}}[\mathbf{Y}_i = 0 \mid \boldsymbol{\mathcal{E}}_{C,\text{yes}}] = \frac{1}{2}.$$

So,

$$\underset{\substack{\mathbf{G}\sim\mathcal{G}_1\\ \boldsymbol{v}_1,\dots,\boldsymbol{v}_t}}{\mathbf{E}}[\mathbf{W}_{A,\text{no}}^{O} \mid \boldsymbol{\mathcal{E}}_{C,\text{yes}}] = \frac{1}{2}\sum_{i\in[\alpha]}|C_i(\text{odd})| \qquad \text{and} \qquad \underset{\substack{\mathbf{G}\sim\mathcal{G}_1\\ \boldsymbol{v}_1,\dots,\boldsymbol{v}_t}}{\mathbf{E}}[\mathbf{W}_{A,\text{no}}^{E} \mid \boldsymbol{\mathcal{E}}_{C,\text{yes}}] = \frac{1}{2}\sum_{i\in[\alpha]}|C_i(\text{even})|.$$

Additionally, for any set of indices $I \subset [\alpha]$,

$$\underset{\substack{\mathbf{G}\sim\mathcal{G}_1\\ \boldsymbol{v}_1,\dots,\boldsymbol{v}_t}}{\mathbf{Pr}}[\forall i \in I, \mathbf{Y}_i = 1 \mid \boldsymbol{\mathcal{E}}_{C,\text{yes}}] \leq \frac{1}{2^{|I|}} \qquad \text{and} \qquad \underset{\substack{\mathbf{G}\sim\mathcal{G}_1\\ \boldsymbol{v}_1,\dots,\boldsymbol{v}_t}}{\mathbf{Pr}}[\forall i \in I, \mathbf{Y}_i = 0 \mid \boldsymbol{\mathcal{E}}_{C,\text{no}}] \leq \frac{1}{2^{|I|}},$$

which implies that the variables $\mathbf{Y}_i$, as well as the variables in $1-\mathbf{Y}_i$ are negatively correlated. We may apply Chernoff bounds (for negatively correlated variables) to obtain deviation bounds for $\mathbf{W}_{A,\text{no}}^{(O)}$ and $\mathbf{W}_{A,\text{no}}^{(E)}$. Then, a union bound gives the desired result for $\mathbf{W}_{A,\text{no}}$. ■

6.3 Proof of the Bad Outcomes Lemma: Lemma 6.4

In this section, we give a proof of Lemma 6.4, which says that the probability over $\mathbf{G} \sim \mathcal{G}_1$ and $\boldsymbol{v}_1, \dots, \boldsymbol{v}_t$ of not satisfying events $\boldsymbol{\mathcal{E}}_T$, $\boldsymbol{\mathcal{E}}_F$, as well as $\boldsymbol{\mathcal{E}}_B$ is $o(1)$. In order to prove this, we will show that individually, the probability of not satisfying each event is $o(1)$ and conclude with a union bound.

6.3.1 $\boldsymbol{\mathcal{E}}_T$: components observed are small trees

The goal of this section is to show that with high probability, the algorithm only sees edges which form various components of small trees.

Definition 6.9. *We let $\boldsymbol{\mathcal{E}}_T$ be the event that observed responses $\boldsymbol{v}_1, \dots, \boldsymbol{v}_t$ generate components $\mathbf{C}_1, \dots, \mathbf{C}_\alpha$ which are all trees of size less than $\log n$.*

Lemma 6.10. *We have that:*

$$\mathop{\mathbf{Pr}}_{\substack{\mathbf{G}\sim\mathcal{G}_1\\ \boldsymbol{v}_1,\dots,\boldsymbol{v}_t}} [\boldsymbol{\mathcal{E}}_T] \geq 1 - o(1).$$

We prove the above lemma by showing the following two claims.

Claim 6.11. *With probability $1 - o(1)$ over the draw of $\mathbf{G} \sim \mathcal{G}_1$ and the draw of $\boldsymbol{v}_1, \dots, \boldsymbol{v}_t$, $\mathbf{G}_o$ has no cycles.*

Proof: Recall that $L_1, \dots, L_t$ are the set queries made, and let $\boldsymbol{\mathcal{E}}_{\circ,\ell}$ be the event that $\mathbf{G}_o$ has a cycle of length ℓ. We have:

$$\begin{aligned}
\mathop{\mathbf{Pr}}_{\substack{\mathbf{G}\sim\mathcal{G}_1\\ \boldsymbol{v}_1,\dots,\boldsymbol{v}_t}} [\boldsymbol{\mathcal{E}}_{\circ,\ell}] &\leq \sum_{\substack{S\subset[t]\\ S=\{i_1,\dots,i_\ell\}}} \mathop{\mathbf{Pr}}_{\substack{\mathbf{G}\sim\mathcal{G}_1\\ \boldsymbol{v}_1,\dots,\boldsymbol{v}_t}} [\boldsymbol{v}_{i_1}, \dots, \boldsymbol{v}_{i_\ell} \text{ form cycle}] \\
&\leq \sum_{\substack{S\subset[t]\\ S=\{i_1,\dots,i_\ell\}}} \sum_{\substack{U\subset[n]\\ U=\{u_1,\dots,u_\ell\}\\ u_j\in L_{i_j}\cap L_{i_{j+1}}}} \mathop{\mathbf{Pr}}_{\substack{\mathbf{G}\sim\mathcal{G}_1\\ \boldsymbol{v}_1,\dots,\boldsymbol{v}_t}} [\forall j \in [\ell], \boldsymbol{v}_{i_j} = (u_j, u_{j+1})], \qquad (8)
\end{aligned}$$

where we think $j + 1 = 1$ when $j = \ell$. The above restriction of $u_j \in L_{i_j} \cap L_{i_{j+1}}$ is necessary if edges $\boldsymbol{v}_{i_j}$ and $\boldsymbol{v}_{i_{j+1}}$ will be the edges of the cycle incident on node u_j. Additionally, we may upper bound (8) by disregarding the effect of the partition $\mathbf{A}$ and $\overline{\mathbf{A}}$; in fact, the presence of $\mathbf{A}$ and $\overline{\mathbf{A}}$ make it harder to achieve a cycle, since if $u_j \in \mathbf{A}$ and $u_{j+1} \in \overline{\mathbf{A}}$, the probability of $\boldsymbol{v}_{i_j} = (u_j, u_{j+1})$ is 0. For any $S = \{i_1, \dots, i_\ell\}$, once we fix a set $U = \{u_1, \dots, u_\ell\}$ where $u_j \in L_{i_j} \cap L_{i_j+1}$,

$$\mathop{\mathbf{Pr}}_{\substack{\mathbf{G}\sim\mathcal{G}_1\\ \boldsymbol{v}_1,\dots,\boldsymbol{v}_t}} [\forall j \in [\ell], \boldsymbol{v}_{i_j} = (u_j, u_{j+1})] \leq \left(\frac{1}{2\binom{n/2}{2}}\right)^\ell.$$

Thus, we have:

$$\begin{aligned}\mathop{\mathbf{Pr}}_{\substack{\mathbf{G}\sim\mathcal{G}_1\\ \boldsymbol{v}_1,\dots,\boldsymbol{v}_t}}[\boldsymbol{\mathcal{E}}_{\circ,\ell}] &\leq \sum_{\substack{S\subset[t]\\ S=\{i_1,\dots,i_\ell\}}}\left(\prod_{j=1}^{\ell}|L_{i_j}\cap L_{i_{j+1}}|\right)\left(\frac{1}{2\binom{\frac{n}{2}}{2}}\right)^{\ell}\\ &\leq \left(\frac{1}{\Omega(n)}\right)^{2\ell}\sum_{\substack{S\subset[t]\\ S=\{i_1,\dots,i_\ell\}}}\prod_{j=1}^{\ell}|L_{i_j}|\\ &\leq \left(\frac{1}{\Omega(n)}\right)^{2\ell}\left(\sum_{i=1}^{t}|L_i|\right)^{\ell}\left(\frac{1}{t}\right)^{\ell}\binom{t}{\ell}\leq\left(O\left(\frac{1}{\log^5 n}\right)\right)^{\ell}.\end{aligned}$$

where we used the fact that $\sum_S\prod_{j=1}^{\ell}|L_{i_j}|$ is the elementary symmetric polynomial of degree ℓ, and $\sum_{i=1}^{t}|L_i|\leq\frac{n^2}{\log^5 n}$. Thus, we obtain:

$$\mathop{\mathbf{Pr}}_{\substack{\mathbf{G}\sim\mathcal{G}_1\\ \boldsymbol{v}_1,\dots,\boldsymbol{v}_t}}[\mathbf{G}_o\text{ contains a cycle}]\leq\sum_{\ell=1}^{t}\left(O\left(\frac{1}{\log^5 n}\right)\right)^{\ell}=o(1).$$

■

Claim 6.12. *With probability $1-o(1)$ over the draw of $\mathbf{G}\sim\mathcal{G}_1$ and the draw of $\boldsymbol{v}_1,\dots,\boldsymbol{v}_t$, we have $\mathbf{G}_o$ has all components of size at most $\log n$.*

Proof: This proof is very similar to the one above. Let $\boldsymbol{\mathcal{E}}_{T,\ell}$ be the event there exists a tree of ℓ edges. We note that there are at most $\exp(O(\ell))$ rooted trees of ℓ edges and $\ell+1$ vertices. We consider first picking a rooted tree, and we pick an arbitrary vertex to be the root of the tree. We then pick the ℓ edges of the tree to some responses, $\boldsymbol{v}_{i_1},\dots,\boldsymbol{v}_{i_\ell}$. We select the vertex on query of the edge going away from the root; this leaves the root, which we choose arbitrarily from $[n]$.

So we have:

$$\begin{aligned}\mathop{\mathbf{Pr}}_{\substack{\mathbf{G}\sim\mathcal{G}_1\\ \boldsymbol{v}_1,\dots,\boldsymbol{v}_t}}[\boldsymbol{\mathcal{E}}_{T,\ell}] &\leq \exp(O(\ell))\sum_{\substack{S\subset[t]\\ S=\{i_1,\dots,i_\ell\}}}\left(n\prod_{j=1}^{\ell}|L_{i_j}|\right)\left(\frac{1}{2\binom{n/2}{2}}\right)^{\ell}\\ &\leq n\cdot\left(O\left(\frac{1}{\log^5 n}\right)\right)^{\ell}=\left(O\left(\frac{1}{\log^5 n}\right)\right)^{\ell},\end{aligned}$$

when $\ell\geq\log n$. Thus, we sum over all $\ell\geq\log$ to get that there exists a tree of size $\log n$ or greater with probability $o(1)$. ■

6.3.2 $\boldsymbol{\mathcal{E}}_F$: few vertices are observed

The goal of this section is to show that the algorithm does not observe too many vertices from the responses $\boldsymbol{v}_1,\dots\boldsymbol{v}_t$ with high probability.

Definition 6.13. *We let $\boldsymbol{\mathcal{E}}_F$ be the event that the responses $\boldsymbol{v}_1, \dots, \boldsymbol{v}_t$ contain at most $\frac{n}{\log^4 n}$ values which are not $\emptyset$.*

Lemma 6.14. *We have:*

$$\mathop{\mathbf{Pr}}_{\substack{\mathbf{G}\sim\mathcal{G}_1\\ \boldsymbol{v}_1,\dots,\boldsymbol{v}_t}}[\boldsymbol{\mathcal{E}}_F] \geq 1 - o(1) \qquad \text{and} \qquad \mathop{\mathbf{Pr}}_{\substack{\mathbf{G}\sim\mathcal{G}_2\\ \boldsymbol{v}_1,\dots,\boldsymbol{v}_t}}[\boldsymbol{\mathcal{E}}_F] \geq 1 - o(1).$$

In other words, any rejection sampling algorithm with cost less than $\frac{n^2}{\log^6 n}$ will observe at most $\frac{n}{\log^4 n}$ non-$\emptyset$ responses in both $\mathcal{G}_1$ and $\mathcal{G}_2$ with high probability.

Proof: Simply note that for a query L_i, and any $G \in \mathcal{G}_1$, the probability of observing a response which is not $\emptyset$ is at most $\dfrac{|L_i| \cdot \frac{n}{2}}{2\binom{n/2}{2}} = O(|L_i|/n)$ (in the case of $\mathcal{G}_1$, and $\frac{|L_i|\cdot\frac{n}{2}}{n^2/4}$ in the case of $\mathcal{G}_2$). Therefore, the expected number of responses which are not $\emptyset$ is at most $O(n/\log^5 n)$, and via a Markov bound, we have the desired result. ■

6.3.3 $\boldsymbol{\mathcal{E}}_B$: vertices observed do not prefer any side too much

We now formally define the event $\boldsymbol{\mathcal{E}}_B$, and prove the event occurs with high probability over the draw of $\mathbf{G} \sim \mathcal{G}_1$ and $\boldsymbol{v}_1, \dots, \boldsymbol{v}_t$.

Definition 6.15. *Let $\mathbf{V}_L \subset [t]$ be the random variable corresponding to the set of indices of responses $\boldsymbol{v}_1, \dots, \boldsymbol{v}_t$ which correspond to observing lone vertices, and for $i \in \mathbf{V}_L$, we let $\boldsymbol{y}_i$ be the indicator random variable for $\boldsymbol{v}_i \in \mathbf{A}$. Let $\boldsymbol{\mathcal{E}}_B$ be the event where:*

$$\mathbf{B} = \sum_{i \in \mathbf{V}_L} (-1)^{\boldsymbol{y}_i} \left(|L_i \cap \mathbf{A}| - |L_i \cap \overline{\mathbf{A}}|\right) = O\left(\frac{n}{\log n}\right).$$

We start by giving some intuition. Fix some query L_i such that $|L_i| \leq \frac{n}{\log n}$. By using Chernoff bound we have that $||L_i \cap \mathbf{A}| - |L_i \cap \overline{\mathbf{A}}|| = O(\sqrt{|L_i|}\log n)$ with high probability. Now assume that *every* query we make is skewed toward $\overline{\mathbf{A}}$. This bad event will create a gap in the probabilities to see a lone vertex between the two distributions, and the algorithm might use it in order to distinguish $\mathcal{G}_1$ and $\mathcal{G}_2$. Hence, we would like to claim that *collectively* the probability of observing such bad events is extremely small. More precise details follows.

Definition 6.16. *Let $\boldsymbol{\mathcal{E}}_Q$ be the event that all queries $L_1, \dots, L_t$ satisfy:*

$$\left||L_i \cap \mathbf{A}| - |L_i \cap \overline{\mathbf{A}}|\right| = O\left(\sqrt{|L_i|}\log n\right).$$

Claim 6.17. *We have:*

$$\mathop{\mathbf{Pr}}_{\mathbf{G}\sim\mathcal{G}_1}[\boldsymbol{\mathcal{E}}_Q] \geq 1 - o(1).$$

Proof: This simply follows from a union bound over $2t$ applications of the Chernoff bound for negatively correlated random variables. In particular, for all $k \in [n]$, let $\mathbf{Y}_k$ be the indicator random variable for $k \in \mathbf{A}$. Then we note that for each $i \in [t]$,

$$|L_i \cap \mathbf{A}| = \sum_{k \in L_i} \mathbf{Y}_k \qquad \text{and} \qquad |L_i \cap \overline{\mathbf{A}}| = \sum_{k \in L_i} (1 - \mathbf{Y}_k).$$

In a similar way to the proof of Claim 6.8, we note that all $\mathbf{Y}_k$ are negatively correlated, and all $(1 - \mathbf{Y}_i)$ are negatively correlated, thus, we have that with probability at least $1 - n^{-10}$,

$$|L_i \cap \mathbf{A}| \leq \frac{|L_i|}{2} + \sqrt{|L_i|} \log n \qquad \text{and} \qquad |L_i \cap \overline{\mathbf{A}}| \leq \frac{|L_i|}{2} + \sqrt{|L_i|} \log n.$$

Thus, we may union bound over all $2t$ events, for the desired result. ■

Lemma 6.18. *We have that:*

$$\mathop{\mathbf{Pr}}_{\substack{\mathbf{G}\sim\mathcal{G}_1 \\ \boldsymbol{v}_1,\dots,\boldsymbol{v}_t}} \left[\neg \boldsymbol{\mathcal{E}}_B \wedge \boldsymbol{\mathcal{E}}_F\right] = o(1).$$

Proof: We first note that because of Claim 6.17, we have:

$$\mathop{\mathbf{Pr}}_{\substack{\mathbf{G}\sim\mathcal{G}_1 \\ \boldsymbol{v}_1,\dots,\boldsymbol{v}_t}} \left[\neg \boldsymbol{\mathcal{E}}_B \wedge \boldsymbol{\mathcal{E}}_F\right] = \sum_{\substack{A \subset [n] \\ \mathcal{E}_Q \text{ satisfied}}} \mathop{\mathbf{Pr}}_{\substack{\mathbf{G}\sim\mathcal{G}_1 \\ \boldsymbol{v}_1,\dots,\boldsymbol{v}_t}} \left[\mathbf{A} = A\right] \mathop{\mathbf{Pr}}_{\substack{\mathbf{G}\sim\mathcal{G}_1 \\ \boldsymbol{v}_1,\dots,\boldsymbol{v}_t}} \left[\neg \boldsymbol{\mathcal{E}}_B \wedge \boldsymbol{\mathcal{E}}_F \mid \mathbf{A} = A\right] + o(1).$$

So consider a fixed set $A \subset [n]$ which satisfies event $\mathcal{E}_Q$. Additionally, we have:

$$\mathop{\mathbf{Pr}}_{\substack{\mathbf{G}\sim\mathcal{G}_1 \\ \boldsymbol{v}_1,\dots,\boldsymbol{v}_t}} \left[\neg \boldsymbol{\mathcal{E}}_B \wedge \boldsymbol{\mathcal{E}}_F \mid \mathbf{A} = A\right] = \sum_{\substack{V_L \subset [t] \\ |V_L| \leq \frac{n}{\log^4 n}}} \mathop{\mathbf{Pr}}_{\substack{\mathbf{G}\sim\mathcal{G}_1 \\ \boldsymbol{v}_1,\dots,\boldsymbol{v}_t}} \left[\mathbf{V}_L = V_L \mid \mathbf{A} = A\right] \mathop{\mathbf{Pr}}_{\substack{\mathbf{G}\sim\mathcal{G}_1 \\ \boldsymbol{v}_1,\dots,\boldsymbol{v}_t}} \left[\neg \boldsymbol{\mathcal{E}}_B \mid \mathbf{A} = A, \mathbf{V}_L = V_L\right]$$

Thus, it suffices to prove that for all $A \subset [n]$ which satisfy $\mathcal{E}_Q$ and $V_L \subset [t]$ of size at most $\frac{n}{\log^4 n}$, $\mathbf{Pr}[\neg \boldsymbol{\mathcal{E}}_B \mid \mathbf{A} = A, \mathbf{V}_L = V_L] = o(1)$. In fact, once we condition on $\mathbf{A} = A$ and $\mathbf{V}_L = V_L$, we have:

$$\mathbf{B} = \sum_{i \in V_L} (-1)^{\boldsymbol{y}_i} \left(|L_i \cap A| - |L_i \cap \overline{A}|\right),$$

which is a sum of independent random variables. Additionally, since $\boldsymbol{y}_i$ is the indicator random variable for $\boldsymbol{v}_i \in A$ conditioned on $\boldsymbol{v}_i$ being a lone vertex, we have each $\boldsymbol{y}_i$ is independent and is 1 with probability p_i, where:

$$p_i = \frac{|L_i \cap A| \left(\frac{n}{2} - |L_i \cap A|\right)}{|L_i| \cdot \frac{n}{2} - |L_i \cap A|^2 - |L_i \cap \overline{A}|^2} = \frac{1}{2} \pm O\left(\frac{\log n}{\sqrt{n}}\right).$$

Thus, we have:

$$\mathop{\mathbf{E}}_{\substack{\mathbf{G}\sim\mathcal{G}_1 \\ \boldsymbol{v}_1,\dots,\boldsymbol{v}_t}} \left[\mathbf{B} \mid \mathbf{A} = A, \mathbf{V}_L = V_L\right] = |V_L| \cdot O(\log^2 n) = O\left(\frac{n}{\log^2 n}\right).$$

Additionally, each variable can contribute $O(\sqrt{|L_i|} \log n)$ to the sum, so via a standard Chernoff bound, noting the fact that $\sum_{i \in V_L} |L_i| \log^2 n \leq \frac{n^2}{\log^3 n}$, we have that $\boldsymbol{\mathcal{E}}_B$ is satisfied with high probability. ■

6.4 Proof of the Good Outcomes Lemma: Lemma 6.3

We may divide $v_1, \ldots, v_t$ into three sets: 1) V_E contain the indices $i \in [t]$ whose responses v_i which are edges, 2) V_L contain the indices $i \in [t]$ whose responses v_i are vertices, and 3) V_T contain the indices $i \in [t]$ whose responses v_i are $\emptyset$. We let:

$$\mathop{\mathbf{Pr}}_{\substack{\mathbf{G}\sim\mathcal{G}_1 \\ \boldsymbol{v}_1,\ldots,\boldsymbol{v}_t}}[\boldsymbol{\mathcal{E}}_O \wedge \boldsymbol{\mathcal{E}}_B \mid \boldsymbol{\mathcal{E}}_{C,\text{yes}}] = \mathcal{Y} \qquad \mathop{\mathbf{Pr}}_{\substack{\mathbf{G}\sim\mathcal{G}_2 \\ \boldsymbol{v}_1,\ldots,\boldsymbol{v}_t}}[\boldsymbol{\mathcal{E}}_O \mid \boldsymbol{\mathcal{E}}_{C,\text{no}}] = \mathcal{N}.$$

We note that for a fixed $\mathbf{A}$ the values of $\boldsymbol{v}_i$ are independent. Therefore, we may write:

$$\mathcal{Y} = \mathop{\mathbf{E}}_{\mathbf{A}}[\mathcal{Y}_E \cdot \mathcal{Y}_L \cdot \mathcal{Y}_T \cdot \boldsymbol{\mathcal{E}}_B \mid \boldsymbol{\mathcal{E}}_{C,\text{yes}}] \qquad \mathcal{N} = \mathop{\mathbf{E}}_{\mathbf{A}}[\mathcal{N}_E \cdot \mathcal{N}_L \cdot \mathcal{N}_T \mid \boldsymbol{\mathcal{E}}_{C,\text{no}}]$$

$$\mathcal{Y}_E = \prod_{i\in V_E} \mathop{\mathbf{Pr}}_{\boldsymbol{v}_i}[\boldsymbol{v}_i = v_i \mid Y(\mathbf{A})] \qquad \mathcal{N}_E = \prod_{i\in V_E} \mathop{\mathbf{Pr}}_{\boldsymbol{v}_i}[\boldsymbol{v}_i = v_i \mid N(\mathbf{A})]$$

$$\mathcal{Y}_L = \prod_{i\in V_L} \mathop{\mathbf{Pr}}_{\boldsymbol{v}_i}[\boldsymbol{v}_i = v_i \mid Y(\mathbf{A})] \qquad \mathcal{N}_L = \prod_{i\in V_L} \mathop{\mathbf{Pr}}_{\boldsymbol{v}_i}[\boldsymbol{v}_i = v_i \mid N(\mathbf{A})]$$

$$\mathcal{Y}_T = \prod_{i\in V_T} \mathop{\mathbf{Pr}}_{\boldsymbol{v}_i}[\boldsymbol{v}_i = \emptyset \mid Y(\mathbf{A})] \qquad \mathcal{N}_T = \prod_{i\in V_T} \mathop{\mathbf{Pr}}_{\boldsymbol{v}_i}[\boldsymbol{v}_i = \emptyset \mid N(\mathbf{A})]$$

where we slightly abused notation to let $\mathbf{Pr}_{\boldsymbol{v}_i}[\boldsymbol{v}_i = v_i \mid Y(\mathbf{A})]$ denote the probability that the sampled response $\boldsymbol{v}_i$ is v_i conditioned on the graph $\mathbf{G}$ being from $\mathcal{G}_1$ with partition $\mathbf{A}$; i.e., $\mathbf{G} = K_{\mathbf{A}} \cup K_{\overline{\mathbf{A}}}$. Likewise, $\mathbf{Pr}_{\boldsymbol{v}_i}[\boldsymbol{v}_i = v_i \mid N(\mathbf{A})]$ denotes the probability that the sampled response $\boldsymbol{v}_i$ is v_i conditioned on the graph $\mathbf{G}$ being from $\mathcal{G}_2$ with partition $\mathbf{A}$; i.e., $\mathbf{G} = K_{\mathbf{A},\overline{\mathbf{A}}}$. We now simply go through the three products in to show each is at most $1 + o(1)$. We shall prove the following claims:

Claim 6.19. *For any* $\mathbf{A}$ *for which* $\boldsymbol{\mathcal{E}}_{C,yes}$ *occurs, we have* $\mathcal{Y}_E \leq (1 + o(1))\mathcal{N}_E$.

Proof: Note that for any choice of A for which $\boldsymbol{\mathcal{E}}_{C,\text{yes}}$ occurs, since the v_i's are specific edges:

$$\mathop{\mathbf{Pr}}_{\boldsymbol{v}_i}[\boldsymbol{v}_i = v_i \mid Y(A)] = \frac{1}{2\binom{n/2}{2}}$$

and for any choice of A for which $\boldsymbol{\mathcal{E}}_{C,\text{no}}$ occurs,

$$\mathop{\mathbf{Pr}}_{\boldsymbol{v}_i}[\boldsymbol{v}_i = v_i \mid N(A)] = \frac{1}{(n/2)^2}.$$

Thus,

$$\frac{\mathbf{Pr}_{\boldsymbol{v}_i}[\boldsymbol{v}_i = v_i \mid Y(A)]}{\mathbf{Pr}_{\boldsymbol{v}_i}[\boldsymbol{v}_i = v_i \mid N(A)]} = \frac{n^2}{4} \cdot \frac{4}{n^2 - 2n} = 1 + O\left(\frac{1}{n}\right),$$

and since $|V_E| \leq \frac{n}{\log^4 n}$, we get that $\dfrac{\mathcal{Y}_E}{\mathcal{N}_E} = 1 + o(1)$. ■

Claim 6.20. *For any* $\mathbf{A}$ *for which* $\boldsymbol{\mathcal{E}}_{C,yes}$ *occurs, we have* $\mathcal{Y}_T \leq \mathcal{N}_T$.

Proof: Here, we have that for any set A which satisfies $\boldsymbol{\mathcal{E}}_{C,\text{yes}}$, we have

$$\mathop{\mathbf{Pr}}_{\boldsymbol{v}_i}[\boldsymbol{v}_i = \emptyset \mid Y(A)] = \frac{2\binom{n/2}{2} - |L_i|\frac{n}{2}}{2\binom{n/2}{2}} = 1 - \frac{2|L_i|}{n-2}$$

and similarly, for any set A which satisfies $\boldsymbol{\mathcal{E}}_{C,\text{no}}$, we have

$$\Pr_{\boldsymbol{v}_i}[\boldsymbol{v}_i = \emptyset \mid N(A)] = \frac{(n/2)^2 - |L_i|\frac{n}{2} + |A \cap L_i||\overline{A} \cap L_i|}{(n/2)^2} \geq 1 - \frac{2|L_i|}{n}.$$

Which finishes the proof. ■

Thus, by Claims 6.19 and 6.20 we have:

$$\frac{\mathbf{E}_{\mathbf{A}}\left[\mathcal{Y}_E \cdot \mathcal{Y}_L \cdot \mathcal{Y}_T \cdot \boldsymbol{\mathcal{E}}_B \mid \boldsymbol{\mathcal{E}}_{C,\text{yes}}\right]}{\mathbf{E}_{\mathbf{A}}\left[\mathcal{N}_E \cdot \mathcal{N}_L \cdot \mathcal{N}_T \mid \boldsymbol{\mathcal{E}}_{C,\text{no}}\right]} \leq (1 + o(1)) \frac{\mathbf{E}_{\mathbf{A}}[\mathcal{Y}_L \cdot \boldsymbol{\mathcal{E}}_B \mid \boldsymbol{\mathcal{E}}_{C,\text{yes}}]}{\mathbf{E}_{\mathbf{A}}[\mathcal{N}_L \mid \boldsymbol{\mathcal{E}}_{C,\text{no}}]}.$$

Therefore, it suffices to prove the following:

$$\frac{\mathbf{E}_{\mathbf{A}}[\mathcal{Y}_L \cdot \boldsymbol{\mathcal{E}}_B \mid \boldsymbol{\mathcal{E}}_{C,\text{yes}}]}{\mathbf{E}_{\mathbf{A}}[\mathcal{N}_L \mid \boldsymbol{\mathcal{E}}_{C,\text{no}}]} \leq 1 + o(1).$$

Suppose $\mathbf{A} \subset [n]$ satisfies $\boldsymbol{\mathcal{E}}_{C,\text{yes}}$, then if v_i is a vertex response at query L_i. We have:

$$\begin{aligned}\Pr_{\boldsymbol{v}_i}[\boldsymbol{v}_i = v_i \mid Y(A)] &= \frac{2}{n-2}\left(1 - \frac{|L_i|}{n} + (-1)^{\mathbf{Y}_i}\left(\frac{|L_i \cap \mathbf{A}| - |L_i \cap \overline{\mathbf{A}}|}{n}\right)\right) \\ &= \frac{2}{n-2}\left(1 - \frac{|L_i|}{n}\right)(1 + \mathbf{Z}_i),\end{aligned}$$

where:

$$\mathbf{Z}_i = c_i(-1)^{\mathbf{Y}_i}\left(\frac{|L_i \cap \mathbf{A}| - |L_i \cap \overline{\mathbf{A}}|}{n}\right),$$

where $c_i = \dfrac{1}{1 - |L_i|/n} \leq 1 + o(1)$, since $|L_i| \ll \frac{n}{\log n}$, and $\mathbf{Y}_i$ is the indicator random variable for $v_i \in \mathbf{A}$. Thus, we may simplify:

$$\mathbf{E}_{\mathbf{A}}[\mathcal{Y}_L \cdot \boldsymbol{\mathcal{E}}_B \mid \boldsymbol{\mathcal{E}}_{C,\text{yes}}] = \left(\frac{2}{n-2}\right)^{|V_L|}\left(1 - \frac{|L_i|}{n}\right)^{|V_L|} \mathbf{E}_{\mathbf{A}}\left[\boldsymbol{\mathcal{E}}_B \prod_{i \in V_L}(1 + \mathbf{Z}_i) \mid \boldsymbol{\mathcal{E}}_{C,\text{yes}}\right].$$

Similarly, suppose $\mathbf{A} \subset [n]$ satisfies $\boldsymbol{\mathcal{E}}_{C,\text{no}}$, then if v_i is a vertex response at query L_i, we have:

$$\Pr_{\boldsymbol{v}_i}[\boldsymbol{v}_i = v_i \mid N(A)] = \frac{2}{n}\left(1 - \frac{|L_i|}{n}\right)(1 + \mathbf{S}_i),$$

where we let $\mathbf{S}_i$ be the random variable:

$$\mathbf{S}_i = c_i(-1)^{\mathbf{Y}_i}\left(\frac{|L_i \cap \overline{\mathbf{A}}| - |L_i \cap \mathbf{A}|}{n}\right),$$

Therefore, we have:

$$\mathbf{E}_{\mathbf{A}}[\mathcal{N}_L \mid \boldsymbol{\mathcal{E}}_{C,\text{no}}] = \left(\frac{2}{n}\right)^{|V_L|}\left(1 - \frac{|L_i|}{n}\right)^{|V_L|} \mathbf{E}_{\mathbf{A}}\left[\prod_{i \in V_L}(1 + \mathbf{S}_i) \mid \boldsymbol{\mathcal{E}}_{C,\text{no}}\right].$$

We note that since $|V_L| \leq \frac{n}{\log^4 n}$, we finish off the proof with the following two claims.

Claim 6.21.

$$\mathop{\mathbf{E}}_{\mathbf{A}}\left[\boldsymbol{\mathcal{E}}_B \prod_{i\in V_L}(1+\mathbf{Z}_i) \mid \boldsymbol{\mathcal{E}}_{C,yes}\right] \le 1+o(1).$$

Claim 6.22.

$$\mathop{\mathbf{E}}_{\mathbf{A}}\left[\prod_{i\in V_L}(1+\mathbf{S}_i) \mid \boldsymbol{\mathcal{E}}_{C,no}\right] \ge 1-o(1)$$

Proof of Claim 6.21:

$$\begin{aligned}\mathop{\mathbf{E}}_{\mathbf{A}}\left[\boldsymbol{\mathcal{E}}_B \prod_{i\in V_L}(1+\mathbf{Z}_i) \mid \boldsymbol{\mathcal{E}}_{C,\text{yes}}\right] &\le \mathop{\mathbf{E}}_{\mathbf{A}}\left[\boldsymbol{\mathcal{E}}_B \cdot e^{\sum_{i\in V_L}\mathbf{Z}_i} \mid \boldsymbol{\mathcal{E}}_{C,\text{yes}}\right] \\ &\le e^{\frac{1}{\log n}} = 1+o(1).\end{aligned}$$

Where the last inequality follows from the fact that $\boldsymbol{\mathcal{E}}_B$ occurs. ■

Proof of Claim 6.22: Recall that

$$\mathbf{S}_i = c_i(-1)^{\mathbf{Y}_i}\left(\frac{|L_i\cap\overline{\mathbf{A}}| - |L_i\cap\mathbf{A}|}{n}\right),$$

therefore, by Chernoff bound (for negative correlations) we have that with probability at least $1-\frac{1}{n^{10}}$, $|\mathbf{S}_i| \le O\left(\frac{\log n}{\sqrt{n}}\right)$. We let $\mathbf{S}'_i$ be the random variable which is equal to $\mathbf{S}_i$ when $|\mathbf{S}_i| \le O(\frac{\log n}{\sqrt{n}})$ and $-2n$ otherwise. Via a very similar analysis to Claim A.1 from [CWX17a], we have:

$$\mathop{\mathbf{E}}_{\mathbf{A}}\left[\prod_{i\in V_L}(1+\mathbf{S}_i) \mid \boldsymbol{\mathcal{E}}_{C,\text{no}}\right] \ge (1-o(1))\left(1+\sum_{i\in V_L}\mathop{\mathbf{E}}_{\mathbf{A}}[\mathbf{S}'_i \mid \boldsymbol{\mathcal{E}}_{C,\text{no}}]\right).$$

We now evaluate each $\mathbf{E}_{\mathbf{A}}[\mathbf{S}'_i \mid \boldsymbol{\mathcal{E}}_{C,\text{no}}]$ for $i \in V_L$ individually. We have:

$$\begin{aligned}\mathop{\mathbf{E}}_{\mathbf{A}}[\mathbf{S}'_i \mid \boldsymbol{\mathcal{E}}_{C,\text{no}}] &\ge \mathop{\mathbf{E}}_{\mathbf{A}}[\mathbf{S}_i \mid \boldsymbol{\mathcal{E}}_{C,\text{no}}] + (-2n-c_i)\mathop{\mathbf{Pr}}_{\mathbf{A}}\left[|\mathbf{S}_i| > O\left(\frac{\log n}{\sqrt{n}}\right) \mid \boldsymbol{\mathcal{E}}_{C,\text{no}}\right] \\ &\ge \mathop{\mathbf{E}}_{\mathbf{A}}[\mathbf{S}_i \mid \boldsymbol{\mathcal{E}}_{C,\text{no}}] - O\left(\frac{1}{n^9}\right).\end{aligned}$$

Assume that v_i is in component C_j, and note that since $\mathbf{A}$ and $\overline{\mathbf{A}}$ are inter-changeable,

$$\mathop{\mathbf{Pr}}_{\mathbf{A}}[v_i \in \mathbf{A} \mid \boldsymbol{\mathcal{E}}_{C,\text{no}}] = \mathop{\mathbf{Pr}}_{\mathbf{A}}[v_i \in \overline{\mathbf{A}} \mid \boldsymbol{\mathcal{E}}_{C,\text{no}}] = \frac{1}{2}.$$

Now we have that,

$$\begin{aligned}\mathop{\mathbf{E}}_{\mathbf{A}}[\mathbf{S}_i \mid \boldsymbol{\mathcal{E}}_{A,\text{no}}] &\ge \frac{c_i}{n}\sum_{k\in L_i\setminus C_j}\mathop{\mathbf{E}}_{\mathbf{A}}\left[(-1)^{\mathbf{Y}_i}(-1)^{\mathbf{Y}_k} \mid \boldsymbol{\mathcal{E}}_{C,\text{no}}\right] - O\left(\frac{\log n}{n}\right) \\ &= \frac{c_i}{n}\sum_{k\in L_i\setminus C_j}\left(2\mathop{\mathbf{Pr}}_{\mathbf{A}}[\mathbf{Y}_k = 1 \mid \mathbf{Y}_i = 1; \boldsymbol{\mathcal{E}}_{C,\text{no}}] - 1\right) - O\left(\frac{\log n}{n}\right),\end{aligned}$$

where we used the fact that $|C_i| \leq \log n$, as well as the fact that $\mathbf{A}$ and $\overline{\mathbf{A}}$ are interchangeable. Since $|V_L| \leq \frac{n}{\log^4 n}$ and $|L_i| \leq \frac{n}{\log n}$ for each $i \in V_L$ (otherwise, we would have observed an edge), it suffices to prove that $\mathbf{Pr}_{\mathbf{A}}[\mathbf{Y}_k = 1 \mid \mathbf{Y}_i = 1; \boldsymbol{\mathcal{E}}_{C,\text{no}}] \geq \frac{1}{2} - \frac{\log^4 n}{n}$. This is indeed true, since $\sum_{i=1}^{\alpha} |C_i| \leq \frac{n}{\log^4 n}$ and $|C_i| \leq \log n$ (see Lemma A.1). ■

Putting everything together, we have:

$$\frac{\mathbf{E}_{\mathbf{A}}[\mathcal{Y}_L \cdot \boldsymbol{\mathcal{E}}_B \cdot \boldsymbol{\mathcal{E}}_Q \mid \boldsymbol{\mathcal{E}}_{C,\text{yes}}]}{\mathbf{E}_{\mathbf{A}}[\mathcal{N}_L \mid \boldsymbol{\mathcal{E}}_{C,\text{no}}]} \leq \left(\frac{n}{n-2}\right)^{|V_L|} \frac{1+o(1)}{1-o(1)} \leq 1 + o(1).$$

Acknowledgments

We thank Eric Blais, Rocco Servedio and Xi Chen for countless discussions and suggestions. We also thank Clément Canonne, Nathan Harms, Dor Minzer and Sofya Raskhodnikova for useful comments on an earlier version of this manuscript. This work is supported in part by the NSF Graduate Research Fellowship under Grant No. DGE-16-44869, CCF-1703925, CCF-1563155, CCF-1420349 and the David R. Cheriton Graduate Scholarship.

References

[ACCL07] Nir Ailon, Bernard Chazelle, Seshadhri Comandur, and Ding Liu. Estimating the distance to a monotone function. *Random Structures and Algorithms*, 31(3):371–383, 2007.

[BB16] Aleksandrs Belovs and Eric Blais. A polynomial lower bound for testing monotonicity. In *Proceedings of the 48th ACM Symposium on the Theory of Computing (STOC '2016)*, pages 1021–1032, 2016.

[BCE+18] Eric Blais, Clément L Canonne, Talya Eden, Amit Levi, and Dana Ron. In *Proceedings of the 29th ACM-SIAM Symposium on Discrete Algorithms (SODA '2018)*, pages 2113–2132. SIAM, 2018.

[BCP+17a] Roksana Baleshzar, Deeparnab Chakrabarty, Ramesh Krishnan S. Pallavoor, Sofya Raskhodnikova, and C. Seshadhri. A lower bound for nonadaptive, one-sided error testing of unateness of boolean functions over the hypercube. *arXiv preprint arXiv:1706.00053*, 2017.

[BCP+17b] Roksana Baleshzar, Deeparnab Chakrabarty, Ramesh Krishnan S. Pallavoor, Sofya Raskhodnikova, and C. Seshadhri. Optimal unateness testers for real-values functions: Adaptivity helps. In *Proceedings of the 44th International Colloquium on Automata, Languages and Programming (ICALP '2017)*, 2017.

[BGSMdW13] Harry Buhrman, David Garcıa-Soriano, Arie Matsliah, and Ronald de Wolf. The non-adaptive query complexity of testing k-parities. *Chicago Journal of Theoretical Computer Science*, 6:1–11, 2013.

[Bla08] Eric Blais. Improved bounds for testing juntas. In *Approximation, Randomization and Combinatorial Optimization. Algorithms and Techniques*, pages 317–330. Springer, 2008.

[Bla09] Eric Blais. Testing juntas nearly optimally. In *Proceedings of the 41st ACM Symposium on the Theory of Computing (STOC '2009)*, pages 151–158, 2009.

[BMR16] Piotr Berman, Meiram Murzabulatov, and Sofya Raskhodnikova. Tolerant testers of image properties. In *Proceedings of the 43th International Colloquium on Automata, Languages and Programming (ICALP '2016)*, pages 90:1–90:14, 2016.

[BRY14] Piotr Berman, Sofya Raskhodnikova, and Grigory Yaroslavtsev. L_p-testing. In *Proceedings of the 46th ACM Symposium on the Theory of Computing (STOC '2014)*, 2014.

[CC16] Deeparnab Chakrabarty and Seshadhri Comandur. An o(n) monotonicity tester for boolean functions over the hypercube. *SIAM Journal on Computing*, 45(2):461–472, 2016.

[CFGM12] Sourav Chakraborty, Eldar Fischer, David García-Soriano, and Arie Matsliah. Junto-symmetric functions, hypergraph isomorphism and crunching. In *Proceedings of the 27th Conference on Computational Complexity (CCC '2012)*, pages 148–158. IEEE, 2012.

[CFGM16] Sourav Chakraborty, Eldar Fischer, Yonatan Goldhirsh, and Arie Matsliah. On the power of conditional samples in distribution testing. *SIAM Journal on Computing*, 45(4):1261–1296, 2016.

[CG04] Hana Chockler and Dan Gutfreund. A lower bound for testing juntas. *Information Processing Letters*, pages 301–305, 2004.

[CGR13] Andrea Campagna, Alan Guo, and Ronitt Rubinfeld. Local reconstructors and tolerant testers for connectivity and diameter. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 411–424. Springer, 2013.

[CRS15] Clément L. Canonne, Dana Ron, and Rocco A. Servedio. Testing probability distributions using conditional samples. *SIAM Journal on Computing*, 44(3):540–616, 2015.

[CS16] Deeparnab Chakrabarty and C. Seshadhri. A $\widetilde{O}(n)$ non-adaptive tester for unateness. *arXiv preprint arXiv:1608.06980*, 2016.

[CST$^+$17] Xi Chen, Rocco A. Servedio, Li-Yang Tan, Erik Waingarten, and Jinyu Xie. Settling the query complexity of non-adaptive junta testing. In *Proceedings of the 32nd Conference on Computational Complexity (CCC '2017)*, 2017.

[CWX17a] Xi Chen, Erik Waingarten, and Jinyu Xie. Beyond talagrand functions: new lower bounds for testing monotonicity and unateness. In *Proceedings of the 49th ACM Symposium on the Theory of Computing (STOC '2017)*, 2017.

[CWX17b] Xi Chen, Erik Waingarten, and Jinyu Xie. Boolean unateness testing with $\widetilde{O}(n^{3/4})$ adaptive queries. In *Proceedings of the 58th Annual IEEE Symposium on Foundations of Computer Science (FOCS '2017)*, 2017.

[DLM$^+$07] Ilias Diakonikolas, Homin K Lee, Kevin Matulef, Krzysztof Onak, Ronitt Rubinfeld, Rocco A Servedio, and Andrew Wan. Testing for concise representations. In *Proceedings of the 48th Annual IEEE Symposium on Foundations of Computer Science (FOCS '2007)*, pages 549–558. IEEE, 2007.

[Doe11] Benjamin Doerr. Analyzing randomized search heuristics: Tools from probability theory. *Theory of randomized search heuristics*, 1:1–20, 2011.

[FF06] Eldar Fischer and Lance Fortnow. Tolerant versus intolerant testing for boolean properties. *Theory of Computing*, 2(9):173?–183, 2006.

[FKR$^+$04] Eldar Fischer, Guy Kindler, Dana Ron, Shmuel Safra, and Alex Samorodnitsky. Testing juntas. *Journal of Computer and System Sciences*, 68(4):753–787, 2004.

[FLN$^+$02] Eldar Fischer, Eric Lehman, Ilan Newman, Sofya Raskhodnikova, Ronitt Rubinfeld, and Alex Samorodnitsky. Monotonicity testing over general poset domains. In *Proceedings of the 34th ACM Symposium on the Theory of Computing (STOC '2002)*, pages 474–483, 2002.

[FN07] Eldar Fischer and Ilan Newman. Testing versus estimation of graph properties. *SIAM Journal on Computing*, 37(2):482–501, 2007.

[FR10] Shahar Fattal and Dana Ron. Approximating the distance to monotonicity in high dimensions. *ACM Transactions on Algorithms*, 6(3):52, 2010.

[GGL$^+$00] Oded Goldreich, Shafi Goldwasser, Eric Lehman, Dana Ron, and Alex Samordinsky. Testing monotonicity. *Combinatorica*, 20(3):301–337, 2000.

[GGR98] Oded Goldreich, Shafi Goldwasser, and Dana Ron. Property testing and its connection to learning and approximation. *Journal of the ACM*, 45(4):653–750, 1998.

[Gol17] Oded Goldreich. *Introduction to property testing.* Cambridge University Press, 2017.

[GR05] Venkatesan Guruswami and Atri Rudra. Tolerant locally testable codes. In *Approximation, Randomization and Combinatorial Optimization. Algorithms and Techniques*, pages 306–317. Springer, 2005.

[GR16] Oded Goldreich and Dana Ron. On sample-based testers. *ACM Transactions on Computation Theory*, 8(2), 2016.

[Hoe63] Wassily Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American statistical association*, 58(301):13–30, 1963.

[KS09] Swastik Kopparty and Shubhangi Saraf. Tolerant linearity testing and locally testable codes. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 601–614. Springer, 2009.

[KS16] Subhash Khot and Igor Shinkar. An $\widetilde{O}(n)$ queries adaptive tester for unateness. In *Approximation, Randomization and Combinatorial Optimization. Algorithms and Techniques*, pages 37:1–37:7, 2016.

[MR09] Sharon Marko and Dana Ron. Approximating the distance to properties in bounded-degree and general sparse graphs. *ACM Transactions on Algorithms*, 5(2):22, 2009.

[PRR06] Michal Parnas, Dana Ron, and Ronitt Rubinfeld. Tolerant property testing and distance approximation. *Journal of Computer and System Sciences*, 72(6):1012–1042, 2006.

[Ron08] Dana Ron. Property testing: A learning theory perspective. *Foundations and Trends® in Machine Learning*, 1(3):307–402, 2008.

[Ron10] Dana Ron. Algorithmic and analysis techniques in property testing. *Foundations and Trends® in Theoretical Computer Science*, 5(2):73–205, 2010.

[Roo01] Bero Roos. Binomial approximation to the poisson binomial distribution: The krawtchouk expansion. *Theory of Probability & Its Applications*, 45(2):258–272, 2001.

[STW15] Rocco A Servedio, Li-Yang Tan, and John Wright. Adaptivity helps for testing juntas. In *Proceedings of the 30th Conference on Computational Complexity (CCC '2015)*, pages 264–279, 2015.

[Tal96] Michel Talagrand. How much are increasing sets positively correlated? *Combinatorica*, 16(2):243–258, 1996.

[Tel16] Roei Tell. A note on tolerant testing with one-sided error. In *Electronic Colloquium on Computational Complexity (ECCC)*, volume 23, page 32, 2016.

A A Useful Claim

Consider any set of trees $C_1, \dots, C_\alpha \subset [n]$ with roots $u_1, \dots, u_\alpha$ satisfying the following conditions:

- Each $|C_i| \leq \log n$ for $i \in [\alpha]$,
- We have $\sum_{i=1}^{\alpha} |C_i| \leq \frac{n}{\log^4 n}$.

Recall that $\boldsymbol{\mathcal{E}}_{C,\text{no}}$ is the event that the components $C_1, \dots, C_\alpha$ is consistent with the partition $\mathbf{A} \subset [n]$. More formally, for each $i \in [\alpha]$, we consider layering the tree C_i with root u_i. We let $|C_i(\text{odd})|$ be the odd layers and $|C_i(\text{even})|$ be the even layers. Then, we have event $\boldsymbol{\mathcal{E}}_{C,\text{no}}$ is satisfied if for each $i \in [\alpha]$, either $C_i(\text{odd}) \subset \mathbf{A}$ and $C_i(\text{even}) \subset \overline{\mathbf{A}}$ or $C_i(\text{even}) \subset \mathbf{A}$ and $C_i(\text{odd}) \subset \overline{\mathbf{A}}$.

The following lemma is the last necessary step of Claim 6.22.

Lemma A.1. *Then, for any two indices j, k, which do not lie in the same component, we have:*

$$\Pr_{\mathbf{A}}[k \in \mathbf{A} \mid j \in \mathbf{A}, \boldsymbol{\mathcal{E}}_{C,no}] \geq \frac{1}{2} - \frac{\log^4 n}{n}.$$

Proof: The proof is very straight-forward, we simply count the number of possible partitions $\mathbf{A}$ for which $j \in \mathbf{A}$ and are consistent with $C_1, \dots, C_\alpha$ and divide by the total number of such partitions. For simplicity, assume that j lies in $C_1(\text{odd})$ and k lies in $C_2(\text{odd})$; the other cases, when $j \in C_1(\text{even})$ or $k \in C_2(\text{even})$ follow from very similar arguments.

We let X be the number of partitions $A \subset [n]$ of size $\frac{n}{2}$ which trigger event $\boldsymbol{\mathcal{E}}_{C,\text{no}}$ and have $C_1(\text{odd}) \subset A$ and $C_2(\text{odd}) \subset A$. In order to count these, we first choose which roots $u_3, \dots, u_\alpha$ will be included in A, and then we pick from the remaining vertices to include in A. For a subset $S \subset \{3, \dots, \alpha\}$, we define the quantities:

- $Q = \sum_{i=3}^{\alpha} |C_i|$ is the total vertices assigned from components.
- $S_A = \sum_{i \in S} |C_i(\text{odd})| + \sum_{i \in [\alpha] \setminus S} |C_i(\text{even})|$ is the total vertices assigned from components to A if we included the roots of components in S in A.
- $S_{\overline{A}} = Q - S_A$.

Note that for all subsets $S \subset \{3, \dots, \alpha\}$, we have $S_A \leq \frac{n}{\log^4 n}$.

Then we have:

$$X = \sum_{\ell=0}^{\alpha-2} \sum_{\substack{S \subset [3;\alpha] \\ |S|=\ell}} \binom{n - Q - |C_1| - |C_2|}{\frac{n}{2} - S_A - |C_1(\text{odd})| - |C_2(\text{odd})|}.$$

Let Y be the number of partitions $A \subset [n]$ of size $\frac{n}{2}$ which trigger event $\boldsymbol{\mathcal{E}}_{C,\text{no}}$ and have $C_1(\text{odd}) \subset A$ and $C_2(\text{even}) \subset A$. Similarly, we have:

$$Y = \sum_{\ell=0}^{\alpha-2} \sum_{\substack{S \subset [3;\alpha] \\ |S|=\ell}} \binom{n - Q - |C_1| - |C_2|}{\frac{n}{2} - S_A - |C_1(\text{odd})| - |C_2(\text{even})|}.$$

For a particular fixed $S \subset [3; \alpha]$ of size ℓ, we consider the ratio of the summand in X and in Y:

$$\begin{aligned}
\frac{\binom{n - Q - |C_1| - |C_2|}{\frac{n}{2} - S_A - |C_1(\text{odd})| - |C_2(\text{odd})|}}{\binom{n - Q - |C_1| - |C_2|}{\frac{n}{2} - S_A - |C_1(\text{odd})| - |C_2(\text{even})|}} &= \frac{\left(\frac{n}{2} - S_A - |C_1(\text{odd})| - |C_2(\text{even})|\right)!}{\left(\frac{n}{2} - S_A - |C_1(\text{odd})| - |C_2(\text{odd})|\right)!} \\
&\quad \times \frac{\left(\frac{n}{2} - S_{\overline{A}} - |C_1(\text{even})| - |C_2(\text{odd})|\right)!}{\left(\frac{n}{2} - S_{\overline{A}} - |C_1(\text{even})| - |C_2(\text{even})|\right)!} \\
&= \left(1 \pm O\left(\frac{\log n}{n}\right)\right)^{\log n} \left(1 \pm O\left(\frac{\log n}{n}\right)\right)^{\log n} \\
&= 1 \pm O\left(\frac{\log^2 n}{n}\right),
\end{aligned}$$

where we used the fact that $|C_2(\text{even})|, |C_2(\text{odd})| \leq \log n$, and $\frac{n}{2} - S_A - |C_1(\text{odd})| = \Omega(n)$ and $\frac{n}{2} - S_{\overline{A}} - |C_1(\text{odd})| = \Omega(n)$. Thus, we have:

$$\frac{X}{Y} = 1 \pm O\left(\frac{\log^2 n}{n}\right),$$

and since:

$$\Pr_{\mathbf{A}}[k \in \mathbf{A} \mid j \in \mathbf{A}, \boldsymbol{\mathcal{E}}_{C,\text{no}}] = \frac{X}{X+Y},$$

we get the desired claim. ■

B Reducing to the case $k = \frac{3}{4}n$

Claim B.1. *For $\varepsilon < \frac{1}{2}$, let $f\colon \{0,1\}^n \to \{0,1\}$ have* $\text{dist}(f, k\text{-Junta}) = \varepsilon < \frac{1}{2}$*. Then, $g\colon \{0,1\}^n \times \{0,1\} \to \{0,1\}$ given by $g(x,y) = f(x) \oplus y$ has* $\text{dist}(g, (k+1)\text{-Junta}) = \varepsilon$*.*

Proof: For the upper bound, suppose $h\colon \{0,1\}^n \to \{0,1\}$ had $\text{dist}(f,h) = \varepsilon$. Then, we have that $h'\colon \{0,1\}^n \times \{0,1\} \to \{0,1\}$ given by $h'(x,y) = h(x) \oplus y$ has $\text{dist}(h', g) = \varepsilon$. Thus, we have $\text{dist}(g, (k+1)\text{-Junta}) \leq \text{dist}(f, k\text{-Junta})$.

For the lower bound, suppose for the sake of contradiction that $h'\colon \{0,1\}^n \times \{0,1\} \to \{0,1\}$ is a $(k+1)$-junta with $\text{dist}(g, h') = \text{dist}(g, (k+1)\text{-Junta}) < \text{dist}(f, k\text{-Junta})$. We note that since $\varepsilon < \frac{1}{2}$, the last variable must be influential in h'. Then, consider the functions $h_0, h_1\colon \{0,1\}^n \to \{0,1\}$ given by $h_0(x) = h'(x, 0)$ and $h_1(x) = h(x, 1)$. Since y is influential in h', h_0 and h_1 are both k-juntas, and therefore

$$\text{dist}(h', g) = \frac{\text{dist}(h_0, f) + \text{dist}(h_1, \neg f)}{2} \geq \text{dist}(f, k\text{-Junta}),$$

which is a contradiction. ■

Claim B.2. *Let $f\colon \{0,1\}^n \to \{0,1\}$ have* $\text{dist}(f, k\text{-Junta}) = \varepsilon$*. Then $g\colon \{0,1\}^n \times \{0,1\} \to \{0,1\}$ given by $g(x,y) = f(x)$ has* $\text{dist}(g, k\text{-Junta}) = \varepsilon$*.*

Proof: For the upper bound, we have that if $h\colon \{0,1\}^n \to \{0,1\}$ has $\text{dist}(f,h) = \varepsilon$, then if $h'\colon \{0,1\}^n \times \{0,1\} \to \{0,1\}$ is given by $h(x,y) = h(x)$, then $\text{dist}(h', g) = \varepsilon$. Thus, we have $\text{dist}(g, (k+1)\text{-Junta}) \leq \text{dist}(f, k\text{-Junta})$.

For the lower bound, suppose for the sake of contradiction that $h'\colon \{0,1\}^n \times \{0,1\} \to \{0,1\}$ is a k-junta with $\text{dist}(g, h') = \text{dist}(g, k\text{-Junta}) < \text{dist}(f, k\text{-Junta})$. Then, similarly to above, the functions $h_0, h_1\colon \{0,1\}^n \to \{0,1\}$ given by $h_0(x) = h'(x,0)$ and $h_1(x) = h'(x,1)$ are k-juntas with

$$\text{dist}(g, k\text{-Junta}) = \text{dist}(g, h') = \frac{\text{dist}(f, h_0) + \text{dist}(f, h_1)}{2} \geq \varepsilon,$$

which is a contradiction. ■

Lemma B.3. *For $0 < \varepsilon_0 < \varepsilon_1 < \frac{1}{2}$, let B be a non-adaptive $(\varepsilon_0, \varepsilon_1)$-tolerant k-junta tester for $n(k)$ variable functions making $q(k)$ queries, where $k \leq \alpha n(k)$. Then, there exists a non-adaptive $(\varepsilon_0, \varepsilon_1)$-tolerant $\frac{3n}{4}$-junta tester making $q(O(n))$ queries.*

Proof: We give an algorithm which on input $f\colon \{0,1\}^n \to \{0,1\}$, determines whether f is ε_0-close from being a $\frac{3n}{4}$-junta or is ε_1-far from being a $\frac{3n}{4}$-junta. The algorithm works as follows: on input $f\colon \{0,1\}^n \to \{0,1\}$, we let $g\colon \{0,1\}^n \times \{0,1\}^{n'} \to \{0,1\}$ be given by:

$$g(x,y) = f(x) \oplus \bigoplus_{j=1}^{n'} y_j,$$

where $n' = \max\{\frac{(4\alpha-3)n}{4(1-\alpha)}, 0\}$. Note that if we let $m = n + n'$ (the number of variables in g), by Claim B.1, if f is ε_0-close from being a $\frac{3n}{4}$-junta, then g is ε_0-close to being an αm-junta, and if f is ε_1-far from being a $\frac{3n}{4}$-junta, then g is ε_1-far from being an αm-junta. Finally, we run the tester B with $k = \alpha m$ on f, where we add $n(k) - m$ dummy variables.

The query complexity is given by $q(O(n))$, since $k = O(n)$ when $\alpha < 1$ is a constant. ■

ECCC ISSN 1433-8092

https://eccc.weizmann.ac.il