Tight Lower Bound on Equivalence Testing in Conditional Sampling Model

Diptarka Chakraborty* Sourav Chakraborty † Gunjan Kumar ‡

Abstract

We study the equivalence testing problem where the goal is to determine if the given two unknown distributions on $[n]$ are equal or ε-far in the total variation distance in the conditional sampling model (CFGM, SICOMP16; CRS, SICOMP15) wherein a tester can get a sample from the distribution conditioned on any subset. Equivalence testing is a central problem in distribution testing, and there has been a plethora of work on this topic in various sampling models.

Despite significant efforts over the years, there remains a gap in the current best-known upper bound of $\tilde{O}(\log \log n)$ [FJOPS, COLT 2015] and lower bound of $\Omega(\sqrt{\log \log n})$[ACK, RANDOM 2015, Theory of Computing 2018]. Closing this gap has been repeatedly posed as an open problem (listed as problems 66 and 87 at sublinear.info). In this paper, we completely resolve the query complexity of this problem by showing a lower bound of $\tilde{\Omega}(\log \log n)$. For that purpose, we develop a novel and generic proof technique that enables us to break the $\sqrt{\log \log n}$ barrier, not only for the equivalence testing problem but also for other distribution testing problems, such as uniblock property.

*National University of Singapore, Singapore. Supported in part by an MoE AcRF Tier 2 grant (MOE-T2EP20221-0009) and Google South & South-East Asia Research Award. Email: diptarka@comp.nus.edu.sg

†Indian Statistical Institute, Kolkata. Email: sourav@isical.ac.in

‡National University of Singapore, Singapore. Supported in part by National Research Foundation Singapore under its NRF Fellowship Programme[NRF-NRFFAI1-2019-0004]. Email: dcsgunj@nus.edu.sg

1 Introduction

Probability distributions play a central role in modern data science, and consequently, the past few years have witnessed sustained interest from theoreticians and practitioners alike in the broad field of distribution testing, wherein the central object of study is probability distribution(s). In this work, we focus on the discrete distributions over the domain of size n. Since the value of n is often too large for distributions of interest, it is impractical to specify such a distribution. Therefore, one is often interested in quantifying complexity through the lens of queries to the distributions. The goal in this scenario is to check whether the input distribution(s) has some particular property or is (are) "ε-far" from satisfying that property, and doing all these while trying to reduce the number of queries made to the distribution(s).

Initial studies in distribution property testing focused on the model (SAMP) wherein one can only sample from the given distribution(s). The SAMP model was discovered to be too weak, as evidenced by strong lower bounds of the form $\Omega(n^{1-c})$, for some constant $c \geq 0$, for testing some of the most exciting properties. Such strong lower bounds necessitated the need to allow more powerful queries, and over the past decade, several models have been proposed. Among such proposals, the conditional sampling model (COND) – that allows drawing samples from the input distribution(s) conditioned on any arbitrary subset – is the most well-studied model in theory as well as in practice. From a theoretical perspective, various other distribution testing problems have been studied under the COND model [FJO+15, KT19, Nar21] and certain variants of it like *subcube conditioning model* [BC18, CCK+21, CJLW21]. Furthermore, the COND model and its variants have recently found applications in the areas like formal methods and machine learning (e.g., [CM19, MPC20, GJM22]).

In this paper, we focus on the equivalence testing problem, one of the central problems in the field of distribution testing. In particular, we want to determine whether two distributions D_1 and D_2 are equal or ε-far from each other in the total variation distance under the COND model. Equivalence testing is arguably the most celebrated problem in distribution testing. In the SAMP model, the equivalence testing is well understood, and its query complexity is $\Theta(\max(n^{2/3}/\varepsilon^{4/3}, \sqrt{n}/\varepsilon^2))$ [CDVV14, BFR+13, Val11].

Analyzing the complexity of equivalence testing has turned out to be more challenging in the context of the COND model. Despite significant efforts over the years, there remains a quadratic gap between the current best-known upper bound of $\tilde{O}(\log \log n)$ [FJO+15] and the lower bound of $\Omega(\sqrt{\log \log n})$ [ACK18]. The challenge of closing the gap between the lower and upper bounds has been a recurring open problem, which has been raised multiple times at various workshops and conferences, including the 2014 Bertinoro Workshop on Sublinear Algorithms and the FOCS 2017 Frontiers of Distribution Testing. This problem has also been discussed on open problem forums such as sublinear.info (listed as problems 66 and 87).

The main difficulty in bridging the gap stems from the limitations inherent in the current approach that was used to establish the lower bound of $\Omega(\sqrt{\log \log n})$. This limitation was best highlighted by the authors of [ACK18]: "*There appear to be conceptual barriers to strengthening our result, which would require new ideas*".

The primary contribution of this paper is to develop a novel and generic technique that overcomes the limitations of previous proof techniques and enables us to go past the $\Omega(\sqrt{\log \log n})$ lower bound barrier for not just equivalence testing but for several other problems in distribution testing.

1.1 Our Lower Bound Result on Equivalence Testing

The main contribution of this paper is to prove an (almost) tight lower bound on the query complexity in the COND query model for the equivalence testing of distributions (see Definition 2.1 for the formal definition). In the COND query model, the tester can specify a subset $A \subseteq [n]$ and then samples each $j \in A$ according to the distribution D conditioned on the set A, i.e., with probability $D(j)/D(A)$ (see Definition 2.3). Let us now state our main result.

Theorem 1.1. *Any (randomized) adaptive tester for testing equivalence between two distributions over* $[n]$ *must make* $\tilde{\Omega}(\log \log n)$ COND *queries. (The tilde hides a* $\mathrm{poly}(\log \log \log n)$ *factor.)*

We prove this result by introducing a weaker query model called the WCOND query model that is easier to analyze, then proving the query lower bound in that weaker model, and finally showing that this weaker model is (roughly) equivalent to the COND model at least for the equivalence testing problem. We discuss this approach in more detail in Section 3. We believe that our proof technique is very generic and can be used for other distribution testing problems in the COND query model. Indeed, the same technique helps us establish a query complexity lower bound for another class of problems - testing label invariant properties.

1.2 Our Lower Bound Result on testing Label Invariant properties

A property of a distribution that is invariant under relabeling of the universe (on which the distribution is defined) is called *label-invariant*. For a label-invariant property $\mathcal{P}$, the goal is to check if a given distribution satisfies the property $\mathcal{P}$ or ε-far (in the total variation distance) from satisfying the property $\mathcal{P}$. One crucial difference between the problem of testing equivalence and the problem of testing any label invariant property is that in the former, two distributions are given as input, and both have to be accessed using queries, while in the latter, the input is only one distribution that has to be accessed using queries.

CFGM [CFGM16] showed a universal tester to test any label invariant property in $\text{poly}(\log n, 1/\varepsilon)$ queries. More precisely, there exists a tester that, for *any* label invariant property $\mathcal{P}$, given COND-access to a distribution D as input, makes $\text{poly}(\log n)$ queries and with high probability, returns ACCEPT if D satisfies the property $\mathcal{P}$ and REJECT if D is ε-far in total variation distance from any distribution having property $\mathcal{P}$. They also defined a label invariant property, called *even uniblock* property, and showed a lower bound of $\Omega(\sqrt{\log\log n})$ on the query complexity. The significance of this lower bound is that now we cannot hope to show a universal tester that can test any label invariant property in $o(\sqrt{\log\log n})$ queries.

In this paper, we improve this lower bound to $\tilde{\Omega}(\log\log n)$.

Theorem 1.2. *There exists a label invariant property such that any (randomized) adaptive tester for that property must make* $\tilde{\Omega}(\log\log n)$ COND*-queries.*

We follow the same approach that we take for showing a lower bound for the equivalence testing - via the WCOND query model. While this improves the lower bound by a quadratic factor, the exact bound on the query complexity of label-invariant properties remains unknown.

1.3 Related Work

The equivalence testing problem has been studied extensively [BFR$^+$00, Val11, CDVV14] and a tight bound of $\Theta(\max(n^{2/3}/\varepsilon^{4/3}, \sqrt{n}/\varepsilon^2))$ on the query complexity is known in the basic SAMP model. There has been considerable recent interest in several alternative powerful query models that allow tremendous savings in the number of required samples. [CRS15] showed an upper bound of $O(\log^5 n/\varepsilon^4)$ in the COND model which was subsequently improved to $\tilde{O}(\frac{\log\log n}{\varepsilon^5})$ by [FJO$^+$15]. On the other hand, [ACK18] showed that $\Omega(\sqrt{\log\log n})$ queries are necessary for the COND model, leaving a quadratic gap (in the dependence on the domain size) between the upper and lower bound, which we settle in this work. In terms of ε, a lower bound of $\Omega(1/\varepsilon^2)$ is known [CR14], and finding the true dependency on ε in the query complexity is an exciting open problem.

Onak and Sun [OS18] considered a natural extension of SAMP model called the *probability-revealing sample* model (in short, PR-SAMP), wherein in addition to returning a sample, the oracle also returns the exact probability of the sample. An even more powerful model of interest is DUAL [CR14, BDKR02] where we have access to two oracles – SAMP that, as mentioned earlier, provides a sample from the distribution, and EVAL that returns the exact probability of any specified element[1]. Cannone and Rubinfield [CR14] showed that $O(1/\varepsilon)$ queries are necessary and sufficient for the equivalence testing in the DUAL model.

Another fundamental distribution testing problem is the *support size estimation* problem, where given a distribution D, the goal is to estimate the support size $|\{x \mid D(x) > 0\}|$. Very recently, Chakraborty, Kumar, and Meel (CKM) [CKM23] showed a tight lower bound of $\Omega(\log\log n)$ on the query complexity for

[1]Note that the PR-SAMP oracle can only provide the probability of the sampled element, whereas the EVAL oracle can give the probability of any arbitrary specified element.

the support-size estimation problem in the COND model. However, the lower bound for the support size estimation does not imply any lower bound on the equivalence testing. Two distributions can have the same support size but can be arbitrarily far apart. Similarly, two distributions can be arbitrarily close to each other, but their support sizes can differ by an arbitrarily large value. For instance, consider distribution $P = (1, 0, \cdots, 0)$ and $Q = (1-\varepsilon, \varepsilon/K, \cdots, \varepsilon/K)$ where ε is any arbitrary small value and K is an arbitrarily large integer. It is straightforward to see that the total variation distance between P and Q is 2ε (a small value), but their support size differs by a factor of K (an arbitrarily large number). Furthermore, the proof technique of CKM does not yield any lower bound for the equivalence testing. It is also worth noting that the hard instance of CKM allows inverse exponential probability mass on an element in the distribution to achieve the lower bound, which is most often not allowed in the distribution testing setup (where we are concerned with additive error). We want to emphasize that by applying our new proof technique, we obtain a similar lower bound for the support size estimation problem, which, more importantly, holds even when it is promised that all the elements with non-zero probability have mass at least $1/n$ and thus strengthen the lower bound result of CKM (see Section 3.5 for a brief discussion on this).

Organisation of the paper. In Section 2, we provide the necessary notations, definitions, and important theorems that we use in the rest of the paper. In Section 3, we provide the technical overview of our results. We first discuss the previous techniques and how we improve upon them for our results and then present a sketch of our proof technique in this section. In Section 4 and in Section 5, we present the proofs of Theorem 1.1 and Theorem 1.2 respectively. We conclude with a discussion on the limitation of our technique in Section 6.

2 Notations and Preliminaries

In this paper, we assume that all the distributions are defined over the set $[n]$. For any distribution D over $[n]$ and for any $s \in [n]$, we will denote by $D(s)$ the probability weight of s according to the distribution D. Similarly, for any $A \subseteq [n]$, we denote by $D(A)$ the probability mass of the set A according to the distribution D. In other words,

$$D(A) = \sum_{a \in A} D(a).$$

The *total variation distance* between two distributions D_1 and D_2 over $[n]$, denoted by $\mathsf{d_{TV}}(D_1, D_2)$, is defined as $\frac{1}{2}\sum_{s\in[n]} |D_1(s) - D_2(s)|$. If the $\mathsf{d_{TV}}$ distance between two distributions is 0, then they are *equal*.

Definition 2.1 (Equivalence Testing)**.** *The* equivalence testing problem *is that, given sample access to two (unknown) distributions D_1 and D_2 over $[n]$, and an $\varepsilon > 0$, decide whether*

- YES : *D_1 and D_2 are equal, or*
- NO : *$\mathsf{d_{TV}}(D_1, D_2) \geq \varepsilon$*

while drawing as few samples as possible.

Throughout this paper, for the purpose of proving the lower bound, we consider $\varepsilon = 1/4$.

The problem of equivalence testing of distributions is one of the most fundamental problems in statistics and property testing and has been studied under various sampling models. In this paper, our main sampling model is the COND model. To define this formally, we first need to define the conditional distribution.

Definition 2.2. *For a distribution D over $[n]$ and a subset $A \subseteq [n]$, the conditional distribution over A, denoted by $D|_A$, is defined as the distribution over A where for each $a \in A$ the probability mass is set to be $D(a)/D(A)$ if $D(A) > 0$, and $1/|A|$ if $D(A) = 0$.*

Definition 2.3 (COND Query Model)**.** *In the* COND *query model (or simply* COND *model), the sampling algorithm/tester specifies a subset $S \subseteq [n]$ and draws a sample according to the conditional distribution $D|_A$.*

We denote such a conditional query by $\mathsf{COND}_D(A)$*. Note that, in the case of an adaptive algorithm, at any point in time, the subset* A *may depend on the samples the algorithm has previously obtained.*

In this paper, we deal with adaptive algorithms.

Hypergeometric Distribution

The *hypergeometric distribution*(n, K, N) is a probability distribution that describes the number of successes (drawn item has a specified feature) when n items are drawn without replacement from a population of size N containing K objects with that feature. Note that when items are drawn with replacement, the distribution becomes $Binomial(n, K, N)$.

If $X \sim Hypergeometric(n, K, N)$ then like binomial distribution, we have $\mathbb{E}[X] = \frac{nK}{N}$. Further, Chernoff bound for hypergeometric distribution holds similar to binomial distributions.

Lemma 2.4. *Let* $X \sim Hypergeometric(n, K, N)$ *then* $\mu = \mathbb{E}[X] = \frac{nK}{N}$ *and*

$$\Pr\left[|X - \mu| \geq \lambda\mu\right] < 2exp\left(-\frac{\lambda^2\mu}{3}\right), \quad \textit{for any } 0 \leq \lambda \leq 1.$$

Extension of Yao's Lemma

One useful tool for proving the lower bound on the query complexity of various problems is the extension of Yao's lemma (formally proved in [Fis01]). Let $\mathcal{I}_{\mathsf{YES}}$ and $\mathcal{I}_{\mathsf{NO}}$ be two distributions over the YES-instances and NO-instances respectively. We use the notation $x \in_R \mathcal{I}_{\mathsf{YES}}$ (resp., $x \in_R \mathcal{I}_{\mathsf{NO}}$) to denote that x is drawn uniformly at random from $\mathcal{I}_{\mathsf{YES}}$ (resp., $\mathcal{I}_{\mathsf{NO}}$). Let a single query return an element from the set $[n]$. For a deterministic query algorithm $\mathcal{A}$ that makes q adaptive queries[2] note that all the answers to the q queries is an element of $[n]^q$. From now on, we consider a tiny constant $\delta = 1/100$.

Theorem 2.5 ([Fis01])**.** *If for a deterministic algorithm* $\mathcal{A}$ *that makes* q *adaptive queries to test a property* $\mathcal{P}$*, and for an event* $\mathsf{Bad}(\mathcal{A}, x)$ *(that depends on the algorithm* $\mathcal{A}$ *and the input) the following holds*

1. $\Pr_{x \in_R \mathcal{I}_{\mathsf{YES}}}[\mathsf{Bad}(\mathcal{A}, x)] + \Pr_{x \in_R \mathcal{I}_{\mathsf{NO}}}[\mathsf{Bad}(\mathcal{A}, x)] \leq \delta/2$
2. *For all* $\sigma \in [n]^q$*,*

$$\Pr_{x \in_R \mathcal{I}_{\mathsf{YES}}}\left[\textit{ the answers to the } q \textit{ queries made by } \mathcal{A}(x) \textit{ is } \sigma \mid \overline{\mathsf{Bad}(\mathcal{A}, x)}\right] \leq \frac{3}{2} \Pr_{x \in_R \mathcal{I}_{\mathsf{NO}}}\left[\textit{ the answers to the } q \textit{ queries made by } \mathcal{A}(x) \textit{ is } \sigma \mid \overline{\mathsf{Bad}(\mathcal{A}, x)}\right],$$

then the adaptive query complexity of the property $\mathcal{P}$ *is* $\Omega(q)$*.*

3 Technical Overview

3.1 Previous Approach

In the context of distribution testing, the COND model is known to be significantly more powerful than the SAMP model. Consequently, establishing a lower bound for the COND model is an immense challenge. One of the reasons it is so difficult to capture a tester's power in this model is that it allows the conditioning of arbitrary-sized sets in an adaptive manner. To overcome this challenge, [CFGM16] introduced the concept

[2]We can assume without loss of generality that an adaptive algorithm that makes at most q queries actually makes exactly q queries.

of *core-adaptive testers* for label-invariant properties. Roughly speaking, these testers do not consider the samples' labels into account when making decisions; instead, they rely on relations between the samples, such as whether two samples are the same or different. Quite surprisingly, they showed that the class of core-adaptive testers is as powerful as general testers in terms of testing label invariant properties.

Later, [ACK18] further built upon this idea by considering two classes of pairs of distributions. The first class consists of pairs in which both distributions are identical (YES-instance), while the second class consists of pairs that are far apart in terms of the total variation distance (NO-instance). The authors proved that any core-adaptive tester must make at least $q = \Omega(\sqrt{\log \log n})$ queries; otherwise, the distributions from the YES and NO instances become indistinguishable from the tester's point of view. The idea of the core-adaptive tester essentially helps in upper bounding the size of the corresponding decision tree R by $2^{O(q^2)}$. [ACK18] argued that to distinguish between the YES and NO instances, the number of nodes present in R must be $\Omega(\sqrt{\log n})$, and as a consequence, $q \geq \Omega(\sqrt{\log \log n})$. While this lower bound does not match the current best upper bound, it is optimal with respect to the proof technique, which is also emphasized in the survey [Can20] as:

"*The fact that both the lower bounds are similar is not a coincidence, but rather inherent to the technique used. Indeed, the core adaptive tester approaches both proofs rely on cannot get past this* $\sqrt{\log \log n}$ *barrier, which derives from the size of the decision tree representing the tester (namely,* $2^{O(q^2)}$ *for a* q*-query tester)*".

Our primary contribution lies in developing a novel and generic technique that overcomes the limitations of previous proof techniques and enables us to break the $\sqrt{\log \log n}$ barrier in the COND model. Our technique does not only apply to the equivalence testing problem, but we believe it can be applied to various other problems as well. For instance, we show that it also provides $\tilde{\Omega}(\log \log n)$ lower bound to the problem of testing another label-invariant property called the *even-uniblock property*, introduced by [CFGM16].

3.2 Core-adaptive testers

As in previous approaches, core-adaptive testers also play a crucial role in our proof technique. Thus let us start by describing the core-adaptive testers. Any general algorithm/tester for equivalence testing between two distributions D_1 and D_2 that makes at most q COND queries, at any step $1 \leq i \leq q$, chooses $k \in \{1, 2\}$, a set $A_i \subseteq [n]$ and places the conditional query $\mathsf{COND}_{D_k}(A_i)$, and then receives a sample $s_i \in A_i$ drawn according to the conditional distribution $D_k|_{A_i}$. Note that if the algorithm queries both distributions D_1 and D_2 on a set A, then we count it as two separate queries.

In [CFGM16], it is shown that without loss of generality, a general algorithm/tester for a label invariant property (such as equivalence testing) can be assumed to belong to a smaller class of testers called *core-adaptive testers*. To formally define these testers, we first give a few definitions.

Definition 3.1 (Atom)**.** *Given a family of sets* $\mathcal{A} = \{A_1, \ldots, A_i\}$*, the* atoms *generated by* $\mathcal{A}$*, denoted by* $\mathsf{At}(\mathcal{A})$*, are (at most)* 2^i *distinct sets of the form* $\cap_{j=1}^{i} C_j$ *where* $C_j \in \{A_j, [n] \setminus A_j\}$*.*

For example, if $i = 2$, then $\mathsf{At}(A_1, A_2) = \{A_1 \cap A_2, A_1 \setminus A_2, A_2 \setminus A_1, \overline{A_1 \cup A_2}\}$. Given a sequence of query-sample pairs $((A_1, s_1), \ldots, (A_i, s_i))$, all the label invariant information about the sample s_i can be captured by the *configuration* of s_i, defined below.

Definition 3.2 (Configuration of s_i)**.** *Given a sequence of query-sample pairs* $((A_1, s_1), \ldots, (A_i, s_i))$*, a* configuration *of* s_i *with respect to* $((A_1, s_1), \ldots, (A_i, s_i))$*, denoted by* c_i*, consists of* $2(i-1)$ *bits, indicating for each* $1 \leq \ell < i$ *whether*

1. $s_i = s_\ell$ *or* $s_i \neq s_\ell$*, and*
2. $s_i \in A_\ell$ *or not.*

Note that a configuration of s_i contains all the label-invariant information about s_i – whether collisions have happened (and if yes, then with which sample) and which unique atom in $\mathsf{At}(\mathcal{A})$ contains the sample s_i.

Definition 3.3 (Core-Adaptive Tester)**.** *A* core-adaptive tester *for a pair of distributions is an algorithm T that does the following:*

1. *Fixes $k \in \{1,2\}$ (fixes the distribution D_1 or D_2 on which to perform the conditional sampling query).*
2. *To make i-th query, based only on its own internal randomness and the configuration of the previous samples $(c_1, \ldots, c_{i-1})$, T provides:*
   (a) *A (non-negative) integer k_i^A for each $A \in \mathsf{At}(A_1, \ldots, A_{i-1})$ between 0 and $|A \setminus \{s_1, \ldots, s_{i-1}\}|$ (how many fresh – not already seen – elements of each particular atom should be included in the next query),*
   (b) *A set $O_i \subseteq \{s_1, \ldots, s_{i-1}\}$ (which of the samples $s_1, \ldots, s_{i-1}$ will be included in the next query).*
3. *Based on these specifications, the tester T constructs the i-th query set A_i by*
   (a) *Drawing uniformly at random, a set U_i from the set*

$$\left\{U \subseteq [n] \setminus \{s_1, \ldots, s_{i-1}\} \mid \forall A \in \mathsf{At}(A_1, \ldots, A_{i-1}), |U_i \cap A| = k_i^A\right\} \tag{1}$$

   i.e., among all the sets containing only "fresh elements", whose intersection with each atom contains exactly as many elements as T specifies (at Step 2a).
   (b) *$A_i := O_i \cup U_i$.*
4. *Samples from $\mathsf{COND}_{D_k}(A_i)$.*

After $q = q(\varepsilon, n)$ queries, the tester T returns ACCEPT or REJECT based on the configurations $(c_1, \ldots, c_q)$.

From now on, for brevity, we denote

$$\mathsf{At}(U_i) := \{U_i \cap A \mid A \in \mathsf{At}(\mathcal{A}_{i-1})\}.$$

It is easy to observe that the conditional sampling queries made by a core-adaptive tester can be viewed in an equivalent way as follows.

Observation 3.4. *For any distribution D_k ($k \in \{1,2\}$) and i-th query set $A_i = O_i \cup U_i$ ($i \in [q]$), the sample obtained from $\mathsf{COND}_{D_k}(A_i)$ can be viewed as:*

1. *First, pick an element $e \in O_i \cup \{U_i\}$ (where $O_i \cup \{U_i\}$ is the set consisting the elements of O_i and the set U_i itself) such that each $j \in O_i$ is picked with probability $\frac{D_k(j)}{D_k(O_i \cup U_i)}$ and $\{U_i\}$ is picked with probability $\frac{D_k(U_i)}{D_k(O_i \cup U_i)}$.*
2. *If $e \in O_i$, then the sample obtained from $\mathsf{COND}_{D_k}(A_i)$ is e.*
3. *Otherwise (i.e., if $e = \{U_i\}$), then pick $s' \sim \mathsf{COND}_{D_k}(U_i)$, and s' is the sample obtained from $\mathsf{COND}_{D_k}(A_i)$. Note that this is equivalent to*
   (a) *First picking an atom $V \in \mathsf{At}(U_i)$ with probability $\frac{D_j(V)}{D_j(U_i)}$, and*
   (b) *Then returning a sample $s' \sim \mathsf{COND}_{D_k}(V)$.*

It was shown in [CFGM16] that general testers are equivalent to core-adaptive testers in terms of testing a label invariant property.

Theorem 3.5 ([CFGM16])**.** *If there exists a q-query general tester for any label invariant property, then there also exists a q-query core adaptive tester.*

A core adaptive tester T, for any $i \leq q$, maps a sequence of configurations $(c_1, \ldots, c_{i-1})$ of the received samples so far, via a function f_T, to a pair (O_i, U_i) (where $O_i \subseteq \{s_1, \ldots, s_{i-1}\}$ and $U_i \subseteq [n] \setminus \{s_1, \ldots, s_{i-1}\}$) which determines the i-th query set $A_i = O_i \cup U_i$.

In a natural way, a tester T can be fully described by a decision tree R. Formally, (the edges of) a path (from root) to any node v at depth i is associated with a sequence of configurations $(c_1, \ldots, c_i)$, and the node v is labeled with a pair $(O_v, U_v) = f_T(c_1, \ldots, c_{i-1})$ (where $O_v \subseteq \{s_1, \ldots, s_{i-1}\}$ and $U_v \subseteq [n] \setminus \{s_1, \ldots, s_{i-1}\}$) which determines the next query set $A_v = O_v \cup U_v$. Further, for every possible value of the configuration c_i of the sample from A_v, there is a corresponding child of the node v, and the corresponding edge is labeled by the value of the configuration c_i. Finally, the leaves of R are labeled by either ACCEPT or REJECT.

We now define a few notations that we will use throughout the paper. We will use (O_v, U_v) for the label of a node v and $A_v = O_v \cup U_v$ for the corresponding query set. Let the nodes of the path from the root to the node v be $v_1, \ldots, v_i$ where v_1 is the root, and v_i is the node v. We will use $\boldsymbol{A}_v = (A_{v_1}, \ldots, A_{v_i})$ for the sequence of the query sets corresponding to the nodes in this path and $\boldsymbol{S}_v = (s_{v_1}, \ldots, s_{v_i})$ to denote the set of samples obtained in the node from the root to v. Further, $\mathsf{At}(\boldsymbol{A}_v) := \mathsf{At}(A_{v_1}, \ldots, A_{v_i})$. Note that the set of all the relevant atoms for the tester (described by a decision tree R) is $\cup_{v \in R} \mathsf{At}(\boldsymbol{A}_v)$, which we will denote by $\mathsf{At}(R)$.

3.3 High-level framework

To demonstrate a lower bound, one of the standard approaches (using Yao's minimax lemma) is, to begin with, two sets of pairs of distributions. The first set, YES instances, contains pairs of identical distributions. The second set, called NO instances, has pairs of distributions separated by a (total variation) distance of at least $1/4$ in the total variation distance. The YES and NO instances that we consider in this paper (formally defined in Section 4.1, informally later in this subsection) are slight modifications of the instances considered in [ACK18].[3] To prove a lower bound on the number of COND queries of $\Omega(\log \log n)$, it suffices to show that no tester can successfully differentiate between whether an input pair of distributions are drawn from the YES and NO instances (with a high probability) unless the tester makes at least $\Omega(\log \log n)$ queries. In other words, we need to show that if the input pair of distributions are drawn from the YES and NO instances, the respective distributions on the leaves of the tester are close to each other in the total variation distance if the number of queries made is $q < o(\log \log n)$.

Recall, as mentioned in Section 3.2, we only focus on proving lower bounds concerning the core-adaptive testers. We need to understand how the path traversed by the tester on the (corresponding) decision tree R depends on the input pair of distributions from YES and NO cases. Note the height of R is the number of queries made, i.e., q. At any given node v, the next node the tester reaches depends on the outcome of the COND query associated with that node. Suppose we can prove that for any node v, the distributions (for YES and NO cases respectively) over the outcomes are η-close in total variation distance. In that case, the overall total variation distance can be at most $q\eta$ since q is the height of the decision tree. We refer to a node v as *good* if the distributions over the outcomes are close in total variation corresponding to YES and NO cases, and *bad* otherwise (more details on good and bad nodes are provided later in this subsection).

As highlighted in Section 3.1, by [ACK18], the probability that there exists a bad node in R when input is drawn from either YES or NO instances is $\frac{|R|\sqrt{\log n}}{\log n} = o(1)$ only when $|R| = o(\sqrt{\log n})$. Since we have $|R| = 2^{q^2}$, if $q = o(\sqrt{\log \log n})$, then there are no bad nodes (with high probability). If there are no bad nodes, indistinguishability follows.

Thus, the bottleneck to show a stronger lower bound will be to reduce $|R|$ (in terms of q), which is impossible since the bound is already tight. This is where our novel idea of "decision tree specification" using weaker COND queries comes into play.

[3] The change is to simplify our proof, and our proof technique works for their exact instances as well, albeit with certain modifications.

Decision Tree Sparsification using WCOND query

Let us introduce a new weaker query model called WCOND query.

Definition 3.6 (WCOND query model)**.** *For any i, given the i-th query set $A_i = O_i \cup U_i$ and $k \in \{1, 2\}$, the weak conditioning query $\mathsf{WCOND}_{D_k}(A_i)$ does the following:*

1. *It picks an element e in $O_i \cup \{U_i\}$ (elements of O_i and the set U_i itself) with probability $\frac{D_j(e)}{D_j(O_i \cup U_i)}$.*
2. *If $e \in O_i$, then the i-th sample $s_i = e$.*
3. *Otherwise (i.e., if $e = \{U_i\}$), then*
   (a) *An atom $V \in \mathsf{At}(U_i)$ is picked with probability $\frac{|V|}{|U_i|}$.*
   (b) *The sample s_i is generated by the $s_i \sim \mathsf{COND}_{D_k}(V)$.*

Notice the difference between a WCOND and COND query is only in Step 3a (see Observation 3.4). In COND, an atom $V \in \mathsf{At}(U_i)$ is picked with probability $\frac{D_k(V)}{D_k(U_i)}$ whereas in WCOND, it is picked with probability $\frac{|V|}{|U_i|}$ (which is independent of input distributions).

Fix a decision tree R. Let $T(\mathsf{COND})$ and $T(\mathsf{WCOND})$ denote the tester T that uses the decision tree R (to determine the next node/query) when given access to COND and WCOND queries, respectively. We compare the behaviors of these two testers. Before doing so, we would like to emphasize that whether a node is good or bad depends solely on R and the input distributions, it does not depend on oracle access of tester T (we describe the good and bad nodes in the next subsection).

For $T(\mathsf{COND})$, the randomness in picking at atom $V \in \mathsf{At}(U_i)$ (in Step 3a) is external (i.e., depends on the distribution), whereas, for $T(\mathsf{WCOND})$, this randomness is internal. Thus a randomized tester with WCOND oracle can simulate this internal randomness. Any randomized algorithm can be seen as a distribution over its (deterministic) instantiations (obtained by fixing its random choices). Hence, the tester $T(\mathsf{WCOND})$ can be seen as a distribution over a set of deterministic algorithms such that in each deterministic algorithm, the tester picks a fixed atom $V \in \mathsf{At}(U_i)$ for each query $i \in [q]$. The crucial aspect is that now, the number of possible outcomes (at each node) is at most $|O_i| \leq q$ (as compared to the previous 2^q that has COND access, one for each possible atom in U_i) resulting in at most q^q nodes (instead of 2^{q^2}) in the decision tree. Thus, for each decision tree, we can show that if $q = o(\log \log n)$, there are no bad nodes with high probability. Since the original tester is a distribution over these deterministic counterparts, it is easy to argue that the $T(\mathsf{WCOND})$ does not pass through a bad node either with high probability if $q = o(\log \log n)$. Note that this does not contradict what we said earlier that the previous analysis is tight — we show that "the tester does not pass through a bad node with high probability" as opposed to "there are no bad nodes".

However, so far, we argue the results for $T(\mathsf{WCOND})$, but ultimately we want it to hold for $T(\mathsf{COND})$. Surprisingly we show that the distributions of the run of a tester are close in total variation distance for both cases. The following theorem is the heart of our proof.

Lemma 3.7 (Informal Statement of Lemma 4.10)**.** *For the instances we consider, the distributions on leaves of the decision tree R are close in total variation distance, for the cases when the tester is provided access to* COND *queries and* WCOND *queries respectively, unless $q = \tilde{\Omega}(\log \log n)$.*

The above theorem is pivotal because, by only an additional $o(1)$ probability, we can show that even $T(\mathsf{COND})$ does not pass through a bad node. Finally, in Lemma 4.9, we show that conditioned on the event that the tester does not pass through a bad node, the YES and NO instances are indistinguishable unless $q = \tilde{\Omega}(\log \log n)$.

Lemma 3.8 (Informal Statement of Lemma 4.9)**.** *The distributions on leaves of the decision tree R are close in total variation distance for* YES *and* NO *instances in $T(\mathsf{COND})$ unless $q = \tilde{\Omega}(\log \log n)$.*

On our YES and NO instances

We informally explain our YES and NO instances (formally defined in Section 4.1). For YES instances, we consider a pair of identical distributions (Q_1, Q_1). On the other hand, for NO instances, we consider a pair of 1/4-far distributions (Q_1, Q_2). For both Q_1 and Q_2, the support of the distributions is randomly partitioned into $\Theta(\sqrt{\log n})$ buckets, with each successive bucket increasing geometrically (more specifically, by a factor of exponential in $\sqrt{\log n}$) in size; however, the probability mass of each bucket remains the same (Figure 1). In other words, if we compare the probability of two elements from two different buckets, they differ (slightly) sub-polynomially (in n). We leverage this property crucially in our lower-bound argument. The distributions Q_1 and Q_2 differ within each bucket. Inside any bucket, Q_1 is uniform on its elements. In contrast, in Q_2, a random half of elements have probability a constant factor times the other random half of elements (see Figure 2). Let us refer to these two random partitions of buckets as sub-buckets.

It is important to note that if we could get two random samples, say s and s', from any particular bucket, it would be easy to distinguish between the two cases. The reason is that s and s' would be in two different sub-buckets with constant probability. Then for Q_1, we have $Q_1(s) = Q_1(s')$ whereas for Q_2, we have $Q_2(s)/Q_2(s') = \Omega(1)$ so a $\mathsf{COND}(s, s')$ query would detect whether the input pair of distributions is (Q_1, Q_1) or (Q_1, Q_2). Indeed, if the support size is known to the tester, then in $O(1)$-query, it is possible to pick two random samples from the same bucket. That is why our instances' support size is also set randomly.

Good and Bad nodes

Now we explain the good and bad nodes in detail. Ideally, we want to say a node is *good* if the outcomes are close in the total variation distance for YES and NO instances. It happens if three conditions are satisfied (see Definition 4.3), which we summarize below.

In any atom $A \in \mathsf{At}(R)$, the expected number of items from the j-th bucket is $\frac{|A|b\rho^j}{n}$. We would like that the actual number is concentrated tightly around this expectation for all atoms and all buckets. By an application of standard concentration inequalities, this would happen if this expected value is either too large or too small compared to 1 (see Lemma 4.5). Next, we also want that an atom to either intersects a large number of buckets (at least $\mathrm{poly}(\log \log n)$) – such atoms are called *large* atoms – or not intersect with any bucket (such atoms are called *small* atoms). This condition would be pivotal to argue that with high probability, no two samples can be from the same bucket (first item of Definition 4.7). Finally, we want the probability mass of U_v (at a node v) to be too heavy or too light compared to the probability mass of any single sampled element (second item of Definition 4.7). If none of the above three conditions holds, we call such a node a *bad* node.

It is worth noting that our definition of a node being good or bad is (almost) equivalent to that of ACK [ACK18]. Our main ingenuity lies in (i) considering the event "the tester does not pass through a bad node with high probability" as opposed to "there are no bad nodes", and (ii) introducing the weaker query model WCOND (and showing equivalence to COND) to analyze the event "the tester does not pass through a bad node with high probability".

Indistinguishability between YES and NO

Assuming that the tester does not pass through a bad node, we show that the input pair of distributions from the YES and NO instances are indistinguishable. As we argued before, we need to show that the induced distributions on the leaves of the decision tree are close in the total variation distance for both cases.

The main challenge is that even if we assume the testers do not pass through a bad node, the decision to choose the next node by the tester depends not only on the previous traversed nodes but also on the bucket distribution of the seen samples (that is, which buckets the seen samples belong to). Note that the bucket distributions are not part of configurations. That is why we argue that the bucket distributions, as well as the distributions of the next configuration for both YES and NO cases, are close in total variation distance at every step. We formally argue this in Section 4.5.

3.4 Applying our technique to testing label-invariant property

Firstly, note that our lower bound of $\Omega(\log \log n)$ for equivalence testing will not directly give a lower bound on a label invariant property. In the equivalence testing problem, the input is a pair of distributions, while the testing of a label invariant property has only one distribution as input. One may fix one of the distributions in the equivalence testing and regard the other distribution as the input. Interestingly, in this case, the number of required queries becomes independent of n [FJO$^+$15].

To obtain a lower bound on testing a label invariant property, we proceed along the path already charted by CFGM [CFGM16]. They defined a property called *even uniblock property* and used it to prove the lower bound of $\sqrt{\log \log n}$. In fact, for our lower bounds, we even use the same hard instances as defined in [CFGM16]. The crucial point where we do better than the previous attempts is that, like in the equivalence testing problem, our "Bad event" is when a run of the algorithm passes through a "Bad node" compared to the previous attempts where the "Bad event" was defined as when there is some "Bad node" in the decision tree. Of course, this crucial difference is significant as we can show via the WCOND model that if the number of queries is $\Omega(\log \log n)$, the Bad event happens with very low probability. Our proof, as in the case of the lower bound of equivalence testing, has three parts: (a) showing that a run of the WCOND model algorithm does not pass through a bad node with high probability if $q = o(\log \log n)$, (b) showing that the total variation distance between the distribution on leaves of the decision tree for the WCOND model and the COND model is small, (c) assuming that a run of the algorithm does not pass through a bad node, the total variation distance over leaves is small for YES and NO cases.

3.5 Corollary to the support size estimation

We show that if we can know the support size of the given distribution, then in $O(1)$ queries, we can distinguish between our YES and NO instances constructed for the equivalence testing problem (as we mentioned before, it is for this reason that we randomly set the support size in our instances). Therefore, the lower bound for the equivalence testing in Theorem 1.1 also holds for the support size estimation.

We quickly recap the informal construction of our YES and NO instances (see section 4.1 for the formal definition). For YES instances, we consider a pair of identical distributions (Q_1, Q_1). On the other hand, for NO instances, we consider a pair of 1/4-far distributions (Q_1, Q_2). For both Q_1 and Q_2, the support of the distributions are the same and randomly partitioned into $\Theta(\sqrt{\log n})$ buckets, with each successive bucket increasing geometrically (by a factor of exponential in $\sqrt{\log n}$, to be precise) in size. The distributions Q_1 and Q_2 differ within each bucket. Inside any bucket, Q_1 is uniform on its elements. In contrast, in Q_2, a random half of elements have probability a constant factor times the other random half of elements (see Figure 2). These two random partitions of buckets are called sub-buckets.

Let (D_1, D_2) be the pair of distributions provided to the tester (which could be either (Q_1, Q_1) or (Q_1, Q_2)). Note that by our construction, distributions D_1 and D_2 have the same support size (say s). Assume that the support size s is known. Let c be a large constant. We construct a set S by including each element $i \in [n]$ with probability c/s. Thus on expectation, there will be c elements from the support in S. Note that the size of each successive bucket differs by a factor of $2^{\sqrt{\log n}}$. Hence on expectation, there will be $O(1)(\approx c/2)$ elements from both the sub-buckets of the largest bucket (as both the sub-buckets are of equal size) and $o(1)$ elements from the rest of the buckets. Therefore, with a high probability, there will be $O(1)$ elements from both the sub-buckets of the largest bucket and no elements from the rest of the buckets. Since there are only $O(1)$ elements from the support in S, it is possible to identify these elements using only $O(1)$ queries. Finally, for each pair s, s' among these elements, we perform $\mathsf{COND}_{D_2}(\{s, s'\})$ queries. We note that in the YES instance, for each pair s, s', $\mathsf{COND}_{D_2}(\{s, s'\})$ is a uniform distribution on $\{s, s'\}$ whereas in the NO instance, for at least one pair s, s' (when s and s' belongs to different sub-buckets), $\mathsf{COND}_{D_2}(s, s')$ assigns probability 3/4 to one and 1/4 to another. Hence, using overall $O(1)$ queries (assuming the support size is known), one can distinguish the YES and NO instances.

Corollary 3.9. *Any (randomized) adaptive tester for estimating the support size of a distribution to a 4/3-multiplicative factor must make $\tilde{\Omega}(\log \log n)$ COND queries even when it is promised that the probability of any element in the support is at least $1/n$.*

4 $\tilde{\Omega}(\log\log n)$ lower bound for Equivalence testing

In this section, we prove Theorem 1.1. We will eventually use Theorem 2.5 to prove our theorem. For that purpose, we start with defining the hard-to-distinguish YES and NO instances in Section 4.1. Then we define the Good event in Section 4.2. The proof of Theorem 1.1 follows from two crucial lemmas - namely that the Good event happens with high probability and that if Good event happens, then the YES and NO instances are hard to distinguish. Lemma 4.8 proves that the Good event happens with high probability, and this is proved in Section 4.4. Lemma 4.9 states that if the Good event happens, then YES and NO instances are indistinguishable, and this is proved in Section 4.5.

4.1 Distributions over the YES and NO instances

Let us start by describing a distribution $\mathcal{I}_{\mathsf{YES}}$ over the YES-instances and a distribution $\mathcal{I}_{\mathsf{NO}}$ over the NO-instances. We describe them by describing the process using which an element of the YES-instances (and NO-instances respectively) is produced.

We will now present a randomized procedure to generate a pair of distributions (Q_1, Q_2). The YES instance will have both distributions as Q_1, whereas the NO instance will have the two distributions as Q_1 and Q_2.

1. An integer $\kappa \in \left\{0, 1, \ldots, \left\lfloor \frac{\log n}{2} \right\rfloor\right\}$ is chosen uniformly at random. We set $b = 2^{\kappa}$, $\rho = 2^{\sqrt{\log n}}$, $\tau = \frac{\sqrt{\log n}}{4}$, and let $m = b(\rho + \rho^2 + \cdots + \rho^{\tau})$.
2. A pair of distributions (Q_1, Q_2) is constructed as follows (see Figure 1 and 2):
   - We randomly choose a subset of size m from $[n]$ that forms the support of both distributions Q_1 and Q_2.
   - We then randomly partition the support into τ buckets (for both Q_1 and Q_2) $B_1, \ldots, B_{\tau}$ such that $|B_j| = b\rho^j$ and assign the probability $1/\tau$ to each bucket.
   - In distribution Q_1, the probability distribution is uniform in each bucket, i.e., for all $j \in [\tau]$, and $i \in B_j$, we have $Q_1(i) = \frac{1}{\tau b \rho^j}$.
   - In distribution Q_2, each bucket B_j is randomly partitioned into two subbuckets B_j^h and B_j^{ℓ} of equal size $(= \frac{|B_j|}{2})$ such that $Q_2(B_j^h) = \frac{3}{4\tau}$ and $Q_2(B_j^{\ell}) = \frac{1}{4\tau}$ and distribution is uniform in each sub-buckets. In other words, for all $j \in [\tau]$, we have

$$Q_2(i) = \begin{cases} \frac{3}{2\tau b \rho^j} & \text{if } i \in B_j^h \\ \frac{1}{2\tau b \rho^j} & \text{if } i \in B_j^{\ell} \end{cases}$$

By the construction of the NO instance (Q_1, Q_2), it follows immediately that

Observation 4.1. $\mathsf{d}_{\mathsf{TV}}(Q_1, Q_2) = \frac{1}{2}\sum_{i \in [\tau]} \frac{1}{2\tau b \rho^j} \cdot b\rho^j = \frac{1}{4}$.

4.2 The Good Event

For any adaptive algorithm $\mathcal{A}$ and input x (which is a pair of distributions D_1, D_2; recall, from the previous section that for YES instance $D_1 = D_2 = Q_1$, and for NO instance $D_1 = Q_1$ and $D_2 = Q_2$), let us first define the Good event that we will use with Theorem 2.5 to prove Theorem 1.1.

We will assume (from Theorem 3.5) the algorithm $\mathcal{A}$ is a core-adaptive algorithm. We will define events $\mathsf{Good}_1(\mathcal{A}, x)$, $\mathsf{Good}_2(\mathcal{A}, x)$ and $\mathsf{Good}_3(\mathcal{A}, x)$ and finally we will have an event $\mathsf{Good}(\mathcal{A}, x) = \mathsf{Good}_1(\mathcal{A}, x) \wedge \mathsf{Good}_2(\mathcal{A}, x) \wedge \mathsf{Good}_3(\mathcal{A}, x)$.

Throughout the rest of the paper, we will assume

$$\gamma = (\log\log n)^9, \qquad \alpha = (\log n)^{(\log\log n)^2}, \qquad \phi = (\log\log n)^{20}.$$

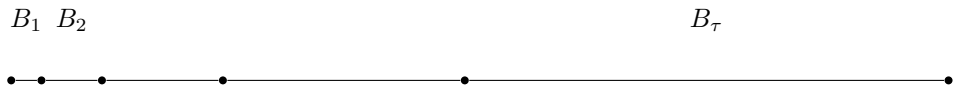

Figure 1: For both Q_1 and Q_2, $|B_j| = b\rho^j$ and $Q_1(B_j) = Q_2(B_j) = \frac{1}{\tau}$

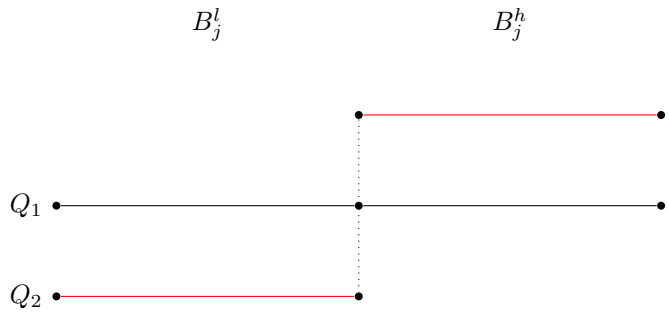

Figure 2: B_j^l and B_j^h are random partition of B_j such that $|B_j^l| = |B_j^h|$. We have $Q_1(i) = \frac{1}{\tau b\rho^j}$ for all $i \in B_j$ whereas $Q_2(i) = \frac{3}{2\tau b\rho^j}$ for $i \in B_j^h$ and $Q_2(i) = \frac{1}{2\tau b\rho^j}$ for $i \in B_j^l$.

Further, recall that as mentioned in Section 4.1,

$$\rho = 2^{\sqrt{\log n}}, \qquad \tau = \frac{\sqrt{\log n}}{4}, \qquad b = 2^\kappa$$

where κ is chosen uniformly at random from $\left\{0, 1, \ldots, \left\lfloor \frac{\log n}{2} \right\rfloor\right\}$.

The $\mathsf{Good}_1(\mathcal{A}, x)$ event: We start with defining ϕ_A associated with each atom A.

Definition 4.2. *For any atom A, ϕ_A is the smallest integer $\triangle \in \{0, 1, \ldots, \tau - 1\}$ such that $\frac{|A|b\rho^{\tau-\triangle}}{n} < 1/\alpha$ and if such an integer does not exist then $\phi_A = \tau$.*

Note that the expected size of the intersection of atom A with bucket $B_{\tau-\triangle}$ is $\frac{|A|b\rho^{\tau-\triangle}}{n}$. Thus, ϕ_A is the number of buckets with which A has a large intersection in expectation.

Definition 4.3 (Good and Bad node)**.** *For a node v of the decision tree R, recall that $\boldsymbol{A}_v = (A_{v_0}, \ldots, A_{v_i})$ is the set of all queries made in the path from the root to the node v in the decision tree. The node v is called* good *for x if it satisfies all of the following conditions.*

1. *For any atom $A \in \mathsf{At}(\boldsymbol{A}_v)$, we have for every bucket $j \in [\tau]$, either $\frac{|A|b\rho^j}{n} \geq \alpha$ or $\frac{|A|b\rho^j}{n} \leq \frac{1}{\alpha}$.*
2. *For any atom $A \in \mathsf{At}(\boldsymbol{A}_v)$, we have either $\frac{|A|b\rho^{\tau-\phi}}{n} \geq \alpha$ or $\frac{|A|b\rho^{\tau}}{n} \leq 1/\alpha$. If the former condition holds, we say atom A is large, and if the latter condition holds, we say A is small,*
3. *For all U_{v_ℓ} $(\ell \leq i)$, we have for all $j \in [\tau]$, either $\frac{\sum_{A \in \mathsf{At}(U_{v_\ell})} \phi_A |A|}{\tau n} \geq \frac{\gamma}{\tau b\rho^j}$ or $\frac{\sum_{A \in \mathsf{At}(U_{v_\ell})} \phi_A |A|}{\tau n} \leq \frac{1}{\gamma\tau b\rho^j}$.*

A node v is called bad *for x if it is not good for x.*

In simple terms, if the algorithm $\mathcal{A}$ is currently at a good node v, then so far (1) for all atoms, the expected intersection size with all buckets is either large (at least α) or small (at most $1/\alpha$). This will help in showing tight concentration bounds (Lemma 4.5) as, by Chernoff bound, the actual value is tightly concentrated to its expected value if the expectation is large or small, (2) any atom either has a large intersection with at

least ϕ buckets (such atoms are large atoms) or a small intersection with all buckets (such atoms are small atoms), (3) the expected probability mass of any unseen query set is 'incomparable' to the probability of any element in the entire distribution, that is, their ratio is either at least γ or at most $1/\gamma$.

Definition 4.4. *The event* $\mathsf{Good}_1(\mathcal{A}, x)$ *is defined as "the run of the algorithm* $\mathcal{A}$ *on input* x *does not pass through a bad node."*

Before we give the definition of the events Good_2 and Good_3, we prove concentration bounds for the good nodes.

Lemma 4.5. *With probability at least* $1 - \frac{1}{\mathrm{poly}(\log\log n)}$*, for all good nodes* v *and for all atoms* $A \in \mathsf{At}(\boldsymbol{A}_v)$ *,*

$$\text{if } j \le \tau - \phi_A \text{ then } |B_j \cap A| = 0, \text{ and,} \tag{2}$$

$$\text{if } j \ge \tau - \phi_A + 1 \text{ then } |A \cap B_j| \in \left[1 - \frac{1}{\gamma}, 1 + \frac{1}{\gamma}\right] \frac{|A| b\rho^j}{n}. \tag{3}$$

Proof. If $\frac{|A|b\rho^j}{n} \le \frac{1}{\alpha}$ for any atom A and $j \in [\tau]$ then by Markov's inequality $\Pr[|A \cap B_j| > 0] \le \frac{1}{\alpha}$. Taking union bound over all atoms corresponding to all good nodes (which can be at most $2^{q+q^2} \le (\log n)^{\log\log n}$) and all $j \in [\tau]$, with probability at least $1 - \frac{1}{\mathrm{poly}(\log\log n)}$,

$$\text{for any atom } A \text{ and } j \le \tau - \phi_A, \text{ we have } |B_j \cap A| = 0. \tag{4}$$

Now note that if $j \ge \tau - \phi_A + 1$ then $\mathbb{E}\left[|B_j^h \cap A|\right] = \frac{|A|b\rho^j}{2n} \ge \alpha/2$. Therefore, by Lemma 2.4,

$$\Pr\left[|A \cap B_j^h| \notin \left[1 - \frac{1}{\gamma}, 1 + \frac{1}{\gamma}\right] \frac{|A|b\rho^j}{2n}\right] \le O(exp(-\alpha/\gamma^2)).$$

Similarly,

$$\Pr\left[|A \cap B_j^\ell| \notin \left[1 - \frac{1}{\gamma}, 1 + \frac{1}{\gamma}\right] \frac{|A|b\rho^j}{2n}\right] \le O(exp(-\alpha/\gamma^2)).$$

Hence

$$\Pr\left[|A \cap B_j| \notin \left[1 - \frac{1}{\gamma}, 1 + \frac{1}{\gamma}\right] \frac{|A|b\rho^j}{n}\right] \le O(exp(-\alpha/\gamma^2)). \tag{5}$$

□

From Lemma 4.5, we deduce the following useful lemma.

Lemma 4.6. *With probability at least* $1 - o(1)$*, for all good nodes* v *and for all atoms* $A \in \mathsf{At}(\boldsymbol{A}_v)$*,*

$$\text{if } j \le \tau - \phi_A \text{ then } D_1(A \cap B_j) = D_2(A \cap B_j) = 0, \text{ and,} \tag{6}$$

$$\text{if } j \ge \tau - \phi_A + 1 \text{ then } D_1(A \cap B_j), D_2(A \cap B_j) \in \left[1 - \frac{1}{\gamma}, 1 + \frac{1}{\gamma}\right] \frac{|A|}{\tau n}. \tag{7}$$

Hence, $D_1(A), D_2(A) \in [1 - 1/\gamma, 1 + 1/\gamma]\frac{\phi_A|A|}{\tau n}$*. Therefore, for any query set* U_{v_ℓ} $(\ell \le i)$*, we have*

$$D_1(U_{v_\ell}), D_2(U_{v_\ell}) \in \left[1 - \frac{1}{\gamma}, 1 + \frac{1}{\gamma}\right] \frac{\sum_{A \in \mathsf{At}(U_{v_\ell})} \phi_A|A|}{\tau n}.$$

Proof. From Equation 2 and 3 in Lemma 4.5 we have

$$D_k(A) = \sum_j \frac{|A \cap B_j|}{b\tau\rho^j} \in \left[1 - \frac{1}{\gamma}, 1 + \frac{1}{\gamma}\right] \sum_{j : \ge r - \phi_A + 1} \frac{|A \cap B_j|}{b\tau\rho^j} = \left[1 - \frac{1}{\gamma}, 1 + \frac{1}{\gamma}\right] \frac{\phi_A|A|}{\tau n}.$$

□

The $\mathsf{Good}_2(\mathcal{A}, x)$ event: The input x (we are concerned about) is drawn according to the $\mathcal{I}_{\mathsf{NO}}$ or $\mathcal{I}_{\mathsf{YES}}$. Let $b(s_i)$ denote the index of the bucket to which the seen element s_i belongs to. Recall the notation $\boldsymbol{S}_i = (s_1, \ldots, s_i)$. We will use $\boldsymbol{B}(\boldsymbol{S}_i)$ for $(b(s_1), \ldots, b(s_i))$.

Definition 4.7. *Let $\mathsf{Good}_2(\mathcal{A}, x)$ be the event defined as:*

1. *For all $i \in [q]$, for any $s, s' \in \boldsymbol{S}_i$ with $D_k(s), D_k(s') \neq 0$ for any $k \in \{1, 2\}$, we have $b(s) \neq b(s')$.*
2. *For all $i \in [q]$, the ratio $\frac{D_k(U_i)}{D_k(s)}$ for any $k \in \{1, 2\}$ and $s \in \boldsymbol{S}_i$ is not in $[\frac{1}{3\gamma}, 3\gamma]$.*

The $\mathsf{Good}_3(\mathcal{A}, x)$ event: An $e_h \in O_i \cup \{U_i\}$ is called the unique heaviest element in $O_i \cup \{U_i\}$ if $D_k(e_h) > D_k(e)$ for all $k \in \{1, 2\}$ and $e \in O_i \cup \{U_i\}$ such that $e \neq e_h$. Let Good_3 be the event that for all $i \in [q]$, the outcome of $\mathsf{COND}_{D_k}(O_i \cup \{U_i\})$ is the unique heaviest element in $O_i \cup \{U_i\}$ for all $i \in [q]$.

The $\mathsf{Good}(\mathcal{A}, x)$ event: We define the event

$$\mathsf{Good} = \mathsf{Good}_1 \wedge \mathsf{Good}_2 \wedge \mathsf{Good}_3. \tag{8}$$

4.3 Proof of Theorem 1.1

We will use Theorem 2.5 to prove Theorem 1.1. The event $\mathsf{Bad}(\mathcal{A}, x)$ is the event $\overline{\mathsf{Good}(\mathcal{A}, x)}$, where $\mathsf{Good}(\mathcal{A}, x)$ is the event defined in Definition 8. So we note that Theorem 1.1 follows from the following two lemmas (Lemma 4.8 and 4.9).

Lemma 4.8. *For any deterministic core-adapative tester $\mathcal{A}$ that makes $q \leq \frac{\log\log n}{100 \log\log\log n}$ queries,*

1. $\Pr_{x \in_R \mathcal{I}_{\mathsf{YES}}}[\mathsf{Good}(\mathcal{A}, x)] \geq 1 - \delta/2$,
2. $\Pr_{x \in_R \mathcal{I}_{\mathsf{NO}}}[\mathsf{Good}(\mathcal{A}, x)] \geq 1 - \delta/2$.

Lemma 4.9. *For any deterministic core-adapative tester $\mathcal{A}$ that makes $q \leq \frac{\log\log n}{100 \log\log\log n}$ queries, for any $\sigma \in [n]^q$*

$$\Pr_{x \in_R \mathcal{I}_{\mathsf{YES}}}[\text{ the answers to the } q \text{ queries made by } \mathcal{A}(x) \text{ is } \sigma \mid \mathsf{Good}(\mathcal{A}, x)]$$
$$\leq \frac{3}{2} \Pr_{x \in_R \mathcal{I}_{\mathsf{NO}}}[\text{ the answers to the } q \text{ queries made by } \mathcal{A}(x) \text{ is } \sigma \mid \mathsf{Good}(\mathcal{A}, x)].$$

We will prove Lemma 4.9 in Section 4.5 and the proof of Lemma 4.8 in Section 4.4.

4.4 Proof of Lemma 4.8

Each of the parts of Lemma 4.8 can be proved in three steps. First, we will prove that Good_1 happens with probability at least $(1-\delta/6)$. Then we show that assuming Good_1, the event Good_2 happens with probability at least $(1-\delta/6)$. and finally assuming Good_1 and Good_2 then the event Good_3 will happen with probability at least $(1-\delta/6)$.

The lower bound on the probability of Good_1 is a main technical lemma, and we present the precise statement given in Lemma 4.10. The proof of Lemma 4.10 is presented in Section 4.4.1.

Lemma 4.10. *If the number of adaptive queries made by the algorithm $\mathcal{A}$ is $q \leq \frac{\log\log n}{100 \log\log\log n}$, then*

1. $\Pr_{x \in_R \mathcal{I}_{\mathsf{YES}}}[\mathsf{Good}_1(\mathcal{A}, x)] \geq 1 - \frac{\delta}{6}$,
2. $\Pr_{x \in_R \mathcal{I}_{\mathsf{NO}}}[\mathsf{Good}_1(\mathcal{A}, x)] \geq 1 - \frac{\delta}{6}$.

We will now prove that conditioned on the event Good_1 holds the event Good_2 holds with high probability. Let us consider the case where x is drawn according to $\mathcal{I}_{\mathsf{NO}}$. We note that the equivalent statement holds for the case when x is drawn according to $\mathcal{I}_{\mathsf{YES}}$.

We will bound the probability that s_i falls in the same bucket corresponding to some previous samples. The seen element (with non-zero probability) can only come from the query $\mathsf{COND}_{D_k}(W)$ for some large atom W for any $k \in \{1,2\}$. Note that for any large atom W, we have for all $k \in \{1,2\}$, $D_k(W \cap B_j) \in [1-\frac{1}{\gamma}, 1+\frac{1}{\gamma}]\frac{|W|}{\tau n}$ for all $j \geq r - \phi_W + 1$ and $D_j(W \cap B_j) = 0$ for $j \leq r - \phi_W$. Therefore,

$$\Pr[\text{ the sample from } \mathsf{COND}_{D_k}(W) \text{ is in bucket } B_j] = \begin{cases} \in [1-1/\gamma, 1+1/\gamma]\frac{1}{\phi_W} & \text{if } j \geq r-\phi_W+1 \\ = 0 & \text{if } j \leq r-\phi_W \end{cases}$$

Consider the inductive hypothesis - for all $i \in [q]$, with probability at least $1-\frac{i^2}{\phi}$, we have $b(s) \neq b(s')$ for all $s, s' \in \boldsymbol{S}_i$. Assuming the inductive hypothesis true for $i-1$, the probability that the i-th sample s_i falls into any bucket in $\boldsymbol{B}(\boldsymbol{S}_{i-1})$ is at most $(1+1/\gamma)\frac{i-1}{\phi_W} \leq (1+1/\gamma)\frac{i-1}{\phi} \leq \frac{2(i-1)}{\phi}$. In other words, with probability at least $1-\frac{(i-1)^2}{\phi} - \frac{2(i-1)}{\phi} \geq 1-\frac{i^2}{\phi}$, we have $b(s) \neq b(s')$ for all $s, s' \in \boldsymbol{S}_i$, i.e., the induction hypothesis is true for i as well. Hence the first point of event Good_2 happens with probability at least $1-q^2/\phi$.

Now we show that the second point happens with high probability assuming that the first point happens. Assuming Good_1 happens, for all U_i, we have for all $j \in [\tau]$,

- Either $\frac{\sum_{A \in \mathsf{At}(U_i)} \phi_A |A|}{\tau n} \geq \frac{\gamma}{\tau b \rho^j} \geq \frac{2D_k(s_i)}{3\gamma}$,
- Or $\frac{\sum_{A \in \mathsf{At}(U_i)} \phi_A |A|}{\tau n} \leq \frac{1}{\gamma \tau b \rho^j} \leq 2\gamma D_k(s_i)$.

Further by Lemma 4.6, we have

$$D_k(U_i) \in \left[1-\frac{1}{\gamma}, 1+\frac{1}{\gamma}\right] \sum_{A \in \mathsf{At}(U_i)} \frac{\phi_A |A|}{\tau n}.$$

Therefore,

$$D_k(U_i) \leq (1+\tfrac{1}{\gamma}) \cdot 2\gamma D_k(s_i) \leq 3\gamma D_k(s_i) \text{ or } D_k(U_i) \geq (1-\tfrac{1}{\gamma}) \cdot \frac{2D_k(s_i)}{3\gamma} \geq \frac{D_k(s_i)}{3\gamma}. \tag{9}$$

Thus, we have

$$\Pr_{x \in_R \mathcal{I}_{\mathsf{NO}}}[\mathsf{Good}_2(\mathcal{A}, x) \mid \mathsf{Good}_1(\mathcal{A}, x)] \geq 1 - \frac{\delta}{6} \text{ and } \Pr_{x \in_R \mathcal{I}_{\mathsf{YES}}}[\mathsf{Good}_2(\mathcal{A}, x) \mid \mathsf{Good}_1(\mathcal{A}, x)] \geq 1 - \frac{\delta}{6}. \tag{10}$$

Now let us assume that the event Good_2 happens. Note that $\rho >> \gamma$ and thus if $b(s) \neq b(s')$, we have

$$\text{either } \frac{D_k(s)}{D_k(s')} \geq \rho \geq \gamma \text{ or } \frac{D_k(s)}{D_k(s')} \leq 1/\rho \leq 1/\gamma. \tag{11}$$

Let e_h be the unique heaviest element in $O_i \cup \{U_i\}$. From Equation 9 and 11 we note that

$$\sum_{e \in (O_i \cup \{U_i\}) \setminus e_h} D_k(e) \leq \frac{D_k(e_h)}{\gamma} + \frac{D_k(e_h)}{\gamma^2} + \cdots \leq 2D_k(e_h)/\gamma.$$

Therefore, with probability at least $(1-2/\gamma)$, the outcome of the conditional query $\mathsf{COND}_{D_k}(O_i \cup \{U_i\})$ is $e_h \in O_i \cup \{U_i\}$. Thus,

$$\Pr_{x \in_R \mathcal{I}_{\mathsf{NO}}}[\mathsf{Good}_3(\mathcal{A}, x) \mid \mathsf{Good}_1(\mathcal{A}, x) \wedge \mathsf{Good}_2(\mathcal{A}, x)], \Pr_{x \in_R \mathcal{I}_{\mathsf{YES}}}[\mathsf{Good}_3(\mathcal{A}, x) \mid \mathsf{Good}_1(\mathcal{A}, x) \wedge \mathsf{Good}_2(\mathcal{A}, x)] \geq 1 - \frac{\delta}{6}. \tag{12}$$

From Lemma 4.10 and Equation 10 and 12 we have Lemma 4.8.

4.4.1 Proof of Lemma 4.10

In this subsection, we prove Lemma 4.10, i.e., $\mathsf{Good}_1(\mathcal{A}, x)$ happens with high probability (when $\mathcal{A}$ has an access to a COND query model).

Informally speaking, we first argue that if $\mathcal{A}$ had access to a WCOND oracle (instead of COND), then $\mathsf{Good}_1(\mathcal{A}, x)$ would happen with high probability. Let us consider the event G_1 that denotes the run of an algorithm $\mathcal{A}$ having access to WCOND oracle, on input x does not pass through a bad node. It is worth highlighting the difference between the Good_1 and G_1 event: The first one is defined for $\mathcal{A}$ having access to COND oracle, whereas the latter is defined for $\mathcal{A}$ having access to WCOND oracle. In Lemma 4.11, we claim that G_1 happens with a high probability. Then we show that given the event G_1 happens, Good_1 happens with a high probability. As a consequence, we conclude that Good_1 happens with a high probability, which completes the proof of Lemma 4.10.

Lemma 4.11. *If an algorithm $\mathcal{A}$ makes $q \leq \frac{\log\log n}{100\log\log\log n}$ adaptive WCOND queries, then*

1. $\Pr_{x\in_R\mathcal{I}_{\mathsf{YES}}}[\mathsf{G}_1(\mathcal{A}, x)] \geq 1 - o(1)$,
2. $\Pr_{x\in_R\mathcal{I}_{\mathsf{NO}}}[\mathsf{G}_1(\mathcal{A}, x)] \geq 1 - o(1)$.

Proof. First, observe that for any WCOND query, Step 3a (of Definition 3.6) does not depend on the input distribution (on which the WCOND query is placed). Thus a randomized algorithm can simulate this step without accessing the input distribution (by picking an atom $V \in \mathsf{At}(U_i)$ with probability $|V|/|U_i|$). Let $\mathcal{A}'$ be the new (randomized) algorithm that simulates Step 3a. One can think of the randomized algorithm $\mathcal{A}'$ as a distribution over a set of algorithms $\mathcal{A}'_1, \mathcal{A}'_2, \cdots$, where each $\mathcal{A}'_r$ is an instantiation of $\mathcal{A}'$ by fixing internal randomness that is used to simulate Step 3a. Thus each $\mathcal{A}'_r$ can be represented as a decision tree where each node denotes access (of the form either Step 3a or Step 3b) to the input distribution. Since the original algorithm $\mathcal{A}$ makes q WCOND queries, each decision tree has a height at most $2q$. Recall, by our assumption, $\mathcal{A}$ is a core-adaptive tester. Thus for any node at the i-th level, the number of children is $|O_i| + 1$ (follows from Step 1), which in turn is upper bounded by $q + 1$ (since $\mathcal{A}$ makes q WCOND queries, by the definition of O_i in Definition 3.3, $|O_i| \leq q$). So the number of nodes present in each such decision tree is at most $(q+1)^{2q}$.

We now use the following claim that helps us in bounding the probability of a node being bad (see Definition 4.3) in a decision tree.

Claim 4.12. *Irrespective of whether the input $x \in_R \mathcal{I}_{\mathsf{YES}}$ or $x \in_R \mathcal{I}_{\mathsf{NO}}$, the probability that a node of a decision tree is bad is at most $O\left(\frac{2^q(\log\log n)^{20}}{\sqrt{\log n}}\right)$.*

The proof of the above claim is an adaptation of the arguments used in [ACK18] and is provided at the end of this subsection. For now, we assume the above claim holds and continue with the proof. Now, by taking a union bound over all the nodes in the decision tree, the probability that at least one of the nodes is bad is at most $(q+1)^{2q} \cdot O\left(\frac{2^q(\log\log n)^{20}}{\sqrt{\log n}}\right)$. Thus when $q \leq \frac{\log\log n}{100\log\log\log n}$, for each $\mathcal{A}'_r$,

$$\Pr_{x\in_R\mathcal{I}_{\mathsf{YES}}}[\mathcal{A}'_r \text{ reaches a bad node on input } x] = o(1).$$

Since the randomized algorithm $\mathcal{A}'$ can be thought of as a distribution over the set of algorithms $\mathcal{A}'_1, \mathcal{A}'_2, \cdots$, we get that

$$\Pr_{x\in_R\mathcal{I}_{\mathsf{YES}}}[\mathcal{A}' \text{ reaches a bad node on input } x] = o(1).$$

By the construction of $\mathcal{A}'$, for every input x, the probability of reaching a bad node by $\mathcal{A}$ and $\mathcal{A}'$ are the same. Hence, $\Pr_{x\in_R\mathcal{I}_{\mathsf{YES}}}[\mathsf{G}_1(\mathcal{A}, x)] \geq 1 - o(1)$. A similar argument holds for $x \in_R \mathcal{I}_{\mathsf{NO}}$. This concludes the proof. □

From now, throughout the rest of this subsection, we use $\mathcal{A}_{\mathsf{COND}}$ and $\mathcal{A}_{\mathsf{WCOND}}$ to denote the core-adaptive tester $\mathcal{A}$ (with the corresponding decision tree R) having access to the COND and WCOND oracle respectively. Next, for any $i \leq q$, let us consider the following two distributions: $L_{\mathsf{COND}}(i)$ and $L_{\mathsf{WCOND}}(i)$ are the distributions of nodes at the i-th level of the decision tree associated with $\mathcal{A}_{\mathsf{COND}}$ and $\mathcal{A}_{\mathsf{WCOND}}$ respectively.

Let G_2 be the event of Lemma 4.6, i.e., for all good nodes v and for all atoms $A \in \mathsf{At}(\boldsymbol{A}_v)$, we have $D_1(A), D_2(A) \in [1-1/\gamma, 1+1/\gamma]\frac{\phi_A|A|}{\tau n}$. Now, we upper bound the total variation distance between the distribution over the nodes of R at any depth $i \leq q$, for $\mathcal{A}_{\mathsf{COND}}$ and $\mathcal{A}_{\mathsf{WCOND}}$, conditioned on event G_2.

Lemma 4.13. *Given the event $\mathsf{G}_1 \wedge \mathsf{G}_2$ happens, for each $i \leq q$, the total variation distance between the two distributions $L_{\mathsf{COND}}(i)$ and $L_{\mathsf{WCOND}}(i)$ is at most $4i/\gamma$.*

Proof. Consider an $i \leq q$. We denote the total variation distance between the two distributions $L_{\mathsf{COND}}(i)$ and $L_{\mathsf{WCOND}}(i)$ (conditioning on the event $\mathsf{G}_1 \wedge \mathsf{G}_2$) by $\mathsf{tv}(i)$. Note that if a node is bad, then all its descendants are also bad. For any node v, let $R(v)$ denote the subtree (of R) rooted at the node v. Let R' be the tree obtained from R by iteratively pruning $R(v) \setminus v$ (i.e., removing the subtree rooted at v while only keeping the node v) for all the bad nodes v in R, starting from the lowest level. Observe, in R', no internal node is bad. Furthermore, the probability of reaching a bad node in R' is the same as that in R. So it suffices to argue with the distribution of nodes at any level i of the tree R'.

Consider any good node v at depth i. We upper bound the total variation distance between distributions over children of v for $\mathcal{A}_{\mathsf{COND}}$ and $\mathcal{A}_{\mathsf{WCOND}}$, conditioned on the fact that the run of the tester is currently at the node v. It is not hard to see that if, for each node v, the total variation distance between distributions over children of v for $\mathcal{A}_{\mathsf{COND}}$ and $\mathcal{A}_{\mathsf{WCOND}}$ is upper bounded by η (for some $\eta \geq 0$), then

$$\mathsf{tv}(i+1) \leq \mathsf{tv}(i) + \eta. \tag{13}$$

Now, since $\mathcal{A}_{\mathsf{WCOND}}$ only differs at Step 3a (of the WCOND oracle) with $\mathcal{A}_{\mathsf{COND}}$ (see Observation 3.4), it suffices to upper bound the total variation distance for distributions over the atoms of U_v. If U_v has no large atom, then $\phi_{U_v} = 0$, and thus by Lemma 4.6, we have $D_1(U_v) = D_2(U_v) = 0$ (conditioned on the event G_2). In this case, the total variation distance is zero. So let us assume that U_v has at least one large atom. Let V_m be the largest atom in $\mathsf{At}(U_v)$ (breaking ties arbitrarily). Hence,

$$\phi_{V_m} \geq \phi. \tag{14}$$

Also, by Lemma 4.6, conditioning on G_2, for all $k \in \{1,2\}$ and $V \in \mathsf{At}(U_v)$, we have

$$\frac{D_k(V)}{D_k(U_v)} = \theta_V \cdot \frac{\phi_V|V|}{\sum_{W \in \mathsf{At}(U)} \phi_W|W|}$$

where $\theta_V \in [1-2/\gamma, 1+2/\gamma]$. Thus the total variation distance is equal to

$$\begin{aligned}
&\sum_{V \in \mathsf{At}(U_v)} \left| \frac{\theta_V \phi_V |V|}{\sum_{W \in \mathsf{At}(U_v)} \phi_W |W|} - \frac{|V|}{|U_v|} \right| \\
&= \sum_{V \in \mathsf{At}(U_v) : |V| < \frac{V_m}{\rho}} \left| \frac{\theta_V \phi_V |V|}{\sum_{W \in \mathsf{At}(U_v)} \phi_W |W|} - \frac{|V|}{|U_v|} \right| + \sum_{V \in \mathsf{At}(U_v) : |V| \geq \frac{V_m}{\rho}} \left| \frac{\theta_V \phi_V |V|}{\sum_{W \in \mathsf{At}(U_v)} \phi_W |W|} - \frac{|V|}{|U_v|} \right|
\end{aligned} \tag{15}$$

Now, since V_m is the largest atom in $\mathsf{At}(U_v)$ for any V such that $|V| < \frac{V_m}{\rho}$, both $\frac{\phi_V|V|}{\sum_{W \in \mathsf{At}(U_v)} \phi_W|W|}$ and $\frac{|V|}{|U_v|}$ are in $\left[0, \frac{1}{\rho}\right]$. Since $\mathsf{At}(U_v)$ contains at most $2^q \leq \log n$ atoms,

$$\sum_{V \in \mathsf{At}(U_v) : |V| < \frac{V_m}{\rho}} \left| \frac{\theta_V \phi_V |V|}{\sum_{W \in \mathsf{At}(U_v)} \phi_W |W|} - \frac{|V|}{|U_v|} \right| \leq \frac{(1+1/\gamma)\log n}{\rho}. \tag{16}$$

Now, it directly follows from the Definition 4.2 that

$$\text{For any } V, W \in \mathsf{At}(U_v) \text{ such that } \frac{|V_m|}{\rho} \leq |V|, |W| \leq |V_m|, \qquad |\phi_V - \phi_W| \in \{0, 1\}.$$

Therefore,

$$\left(\frac{\phi_{V_m} - 1}{\phi_{V_m}}\right) \cdot \frac{|V|}{|U_v|} \leq \frac{\phi_V |V|}{\sum_{W \in \mathsf{At}(U_v)} \phi_W |W|} \leq \left(\frac{\phi_{V_m}}{\phi_{V_m} - 1}\right) \cdot \frac{|V|}{|U_v|}$$

and hence

$$\left| \frac{\theta_V \phi_V |V|}{\sum_{W \in \mathsf{At}(U_v)} \phi_W |W|} - \frac{|V|}{|U_v|} \right| \leq \left(\frac{1}{\gamma} + \frac{1}{\phi_{V_m}}\right) \cdot \frac{|V|}{|U_v|} \leq \frac{2}{\gamma} \cdot \frac{|V|}{|U_v|} \tag{17}$$

where the last inequality follows from Equation 14. Hence, by combining Equation 15, 16, and 17, the total variation distance is bounded above by

$$\frac{(1 + 1/\gamma) \log n}{\rho} + \frac{2}{\gamma} \leq \frac{4}{\gamma}.$$

Then by Equation 13,

$$\mathsf{tv}(i + 1) \leq \mathsf{tv}(i) + 4/\gamma.$$

So we conclude that for each $i \leq q$, $\mathsf{tv}(i) \leq 4i/\gamma$. □

Now, we are ready to finish the proof of Lemma 4.10.

Proof of Lemma 4.10. It directly follows from Lemma 4.13 that the probability that $\mathcal{A}_{\mathsf{COND}}$ reaches a bad node (on the input x) is at most

$$\sum_{i=1}^{q} 4i/\gamma + \Pr[\overline{\mathsf{G}_1}] + \Pr[\overline{\mathsf{G}_2}] \leq 2q^2/\gamma + \Pr[\overline{\mathsf{G}_1}] + \Pr[\overline{\mathsf{G}_2}].$$

Then by Lemma 4.11 and Lemma 4.6,

$$\Pr_{x \in_R \mathcal{I}_{\mathsf{YES}}} \left[\overline{\mathsf{Good}_1(\mathcal{A}, x)}\right] \leq 2q^2/\gamma + o(1) \leq \delta/6$$

where the last inequality follows for $q \leq \frac{\log \log n}{100 \log \log \log n}$. The same argument also holds for $\Pr_{x \in_R \mathcal{I}_{\mathsf{NO}}} \left[\overline{\mathsf{Good}_1(\mathcal{A}, x)}\right]$, and that concludes the proof. □

So it only remains to prove Claim 4.12.

Proof of Claim 4.12. Note that $b = 2^\kappa$. Fix a node v. Let

$$\mathsf{K}_1 = \bigcup_{j \in [\tau], A \in \mathsf{At}(\boldsymbol{A}_v)} \left\{ \kappa : \log\left(\frac{n}{\alpha |A| \rho^j}\right) < \kappa < \log\left(\frac{n\alpha}{|A| \rho^j}\right) \right\}$$

be all the possible values of κ for which the first item of Definition 4.3 can get violated. We have

$$|\mathsf{K}_1| \leq \tau \cdot |\mathsf{At}(\boldsymbol{A}_v)| \cdot \log \frac{\frac{n\alpha}{|A|\rho^j}}{\frac{n}{\alpha |A| \rho^j}} \leq \tau 2^{q+1} \log \alpha.$$

Let

$$\mathsf{K}_2 = \bigcup_{A \in \mathsf{At}(\boldsymbol{A}_v)} \left\{ \kappa : \log\left(\frac{n}{\alpha |A| \rho^\tau}\right) < \kappa < \log\left(\frac{n\alpha\rho^\phi}{|A|}\right) \right\}$$

be all the possible values of κ for which the second item of Definition 4.3 can get violated. We have

$$|\mathsf{K}_2| \leq |\mathsf{At}(\boldsymbol{A}_v)| \cdot \log \frac{\frac{n\alpha\rho^\phi}{|A|}}{\frac{n}{\alpha|A|\rho^\tau}} \leq 2^q(2\log\alpha + \phi\log\rho).$$

Let

$$\mathsf{K}_3 = \bigcup_{i'\in[i], j\in[\tau]} \left\{ \kappa : \frac{n}{\gamma\rho^j} < 2^\kappa \sum_{A\in At(U_{v_{i'}})} \phi_A|A| < \frac{\gamma n}{\rho^j} \right\}$$

be all the possible values of κ for which the third item of Definition 4.3 can get violated. Observe that

$$\left| \left\{ \kappa : \frac{n}{\gamma\rho^j} < 2^\kappa \sum_{A\in At(U_{v_{i'}})} \phi_A|A| < \frac{\gamma n}{\rho^j} \right\} \right| \leq \left| \left\{ \kappa : \frac{n}{\gamma\rho^j} < 2^\kappa < \frac{\gamma n}{\rho^j} \right\} \right|.$$

So we have $|\mathsf{K}_3| \leq 2q\tau\log\gamma$. Therefore,

$$|\mathsf{K}_1| + |\mathsf{K}_2| + |\mathsf{K}_3| \leq \left(2^{q+1}\log\alpha + 2^q\left(2\log\alpha + \phi\log\rho\right) + 2q\tau\log\gamma\right).$$

In both YES and NO instances, since κ is drawn uniformly from $\{0, 1, \ldots, \lfloor\frac{\log n}{2}\rfloor\}$, the probability that any given node v is bad is at most

$$\frac{\left(2^{q+1}\log\alpha + 2^q\left(2\log\alpha + \phi\log\rho\right) + 2q\tau\log\gamma\right)}{\log n} \leq O\left(\frac{2^q(\log\log n)^{20}}{\sqrt{\log n}}\right).$$

□

4.5 Proof of Lemma 4.9

Let V_i^{YES} be the node reached by $\mathcal{A}$ after i queries when the input is drawn from the YES instance. And let $B^{\mathsf{YES}}(s_i)$ be the bucket in which the ith sample belongs. Similarly, we define V_i^{NO} and $B^{\mathsf{NO}}(s_i)$.

Conditioned on the event Good we will prove that for any node v_i (at depth i) and $(b_1, \ldots, b_i) \in [\tau]^i$,

$$\Pr\left[V_q^{\mathsf{YES}} = v_q, (B^{\mathsf{YES}}(s_1), \ldots, B^{\mathsf{YES}}(s_q)) = (b_1, \ldots, b_q)\right] \leq \left(1 + \frac{100i}{\gamma}\right) \Pr\left[V_q^{\mathsf{NO}} = v_q, (B^{\mathsf{NO}}(s_1), \ldots, B^{\mathsf{NO}}(s_q)) = (b_1, \ldots, b_q)\right]$$

We prove it by induction on i. Let the parent of the node v_i be v_{i-1}. For brevity, we use $\mathcal{B}^{\mathsf{YES}}(\boldsymbol{S}_i)$ for $\left(B^{\mathsf{YES}}(s_1), \ldots, B^{\mathsf{YES}}(s_i)\right)$ and $\boldsymbol{B}_i$ for $(b_1, \ldots, b_i)$.

$$\begin{aligned}
&\Pr\left[V_i^{\mathsf{YES}} = v_i, \mathcal{B}^{\mathsf{YES}}(\boldsymbol{S}_i) = \boldsymbol{B}_i\right] \\
&= \Pr\left[V_i^{\mathsf{YES}} = v_i, V_{i-1}^{\mathsf{YES}} = v_{i-1}, \mathcal{B}^{\mathsf{YES}}(\boldsymbol{S}_i) = \boldsymbol{B}_i\right] \\
&= \Pr\left[V_i^{\mathsf{YES}} = v_i, B^{\mathsf{YES}}(s_i) = b_i | V_{i-1}^{\mathsf{YES}} = v_{i-1}, \mathcal{B}^{\mathsf{YES}}(\boldsymbol{S}_{i-1}) = \boldsymbol{B}_{i-1}\right] \cdot \Pr[V_{i-1}^{\mathsf{YES}} = v_{i-1}, \mathcal{B}^{\mathsf{YES}}(\boldsymbol{S}_{i-1}) = \boldsymbol{B}_{i-1}\right] \\
&\leq \Pr\left[V_i^{\mathsf{YES}} = v_i, B^{\mathsf{YES}}(s_i) = b_i | V_{i-1}^{\mathsf{YES}} = v_{i-1}, \mathcal{B}^{\mathsf{YES}}(\boldsymbol{S}_{i-1}) = \boldsymbol{B}_{i-1}\right] \cdot \\
&\qquad \left(1 + \frac{100(i-1)}{\gamma}\right) \Pr\left[V_{i-1}^{\mathsf{NO}} = v_{i-1}, \mathcal{B}^{\mathsf{NO}}(\boldsymbol{S}_{i-1}) = \boldsymbol{B}_{i-1}\right] \\
&= \left(1 + \frac{100(i-1)}{\gamma}\right) \Pr\left[V_i^{\mathsf{YES}} = v_i, B^{\mathsf{YES}}(s_i) = b_i \mid V_{i-1}^{\mathsf{YES}} = v_{i-1}, \mathcal{B}^{\mathsf{YES}}(\boldsymbol{S}_{i-1}) = \boldsymbol{B}_{i-1}\right] \cdot \\
&\qquad \Pr\left[V_{i-1}^{\mathsf{NO}} = v_{i-1}, \mathcal{B}^{\mathsf{NO}}(\boldsymbol{S}_{i-1}) = \boldsymbol{B}_{i-1}\right]
\end{aligned}$$

Fixing the node v_{i-1} fixes the next query $\mathsf{COND}_k(O_{v_{i-1}} \cup \{U_{v_{i-1}}\})$ irrespective of the YES or NO instances. Since we are conditioning on the event Good_3, the unique heaviest element $e \in O_{v_{i-1}} \cup \{U_{v_{i-1}}\}$ will be the outcome of this query and is same for both instances. We have two cases to consider.

1. If $e \in O_{v_{i-1}}$ then the value of V_i^{YES} and V_i^{NO} is same which is (with probability 1) the child of v^{i-1} with the corresponding edge labeled by e. Obviously, there is no new seen element in this case, i.e., as $e \in O_{v_{i-1}}$ so we have $s_i = s_j$ for some $j \in [i-1]$ and hence $B^{\mathsf{YES}}(s_i) = B^{\mathsf{NO}}(s_i) = b_j$. Hence, in this case, we have

$$\Pr\left[V_i^{\mathsf{YES}} = v_i, B^{\mathsf{YES}}(s_i) = b_i \mid V_{i-1}^{\mathsf{YES}} = v_{i-1},\ \mathcal{B}^{\mathsf{YES}}(\boldsymbol{S}_{i-1}) = \boldsymbol{B}_{i-1}\right] = \Pr\left[V_i^{\mathsf{NO}} = v_i, B^{\mathsf{NO}}(s_i) = b_i \mid V_{i-1}^{\mathsf{NO}} = v_{i-1}, \mathcal{B}^{\mathsf{NO}}(\boldsymbol{S}_{i-1}) = \boldsymbol{B}_{i-1}\right]$$

2. Now consider $e = \{U_{v_{i-1}}\}$. In this case, by Lemma 4.6 in both YES and NO instances, each atom $W \in \mathsf{At}(U_{v_{i-1}})$ is picked with probability whose value is in $[1-1/\gamma, 1+1/\gamma]\frac{|W|}{|U_{v_{i-1}}|}$. Note that each atom U in one to one manner corresponds to a child of the node v_{i-1}. Thus we have

$$\Pr\left[V_i^{\mathsf{YES}} = v_i \mid V_{i-1}^{\mathsf{YES}} = v_{i-1}, \mathcal{B}^{\mathsf{YES}}(\boldsymbol{S}_{i-1}) = \boldsymbol{B}_{i-1}\right] \leq \left(\frac{1+1/\gamma}{1-1/\gamma}\right)\Pr\left[V_i^{\mathsf{NO}} = v_i \mid V_{i-1}^{\mathsf{NO}} = v_{i-1}, \mathcal{B}^{\mathsf{NO}}(\boldsymbol{S}_{i-1}) = \boldsymbol{B}_{i-1}\right]$$

From Lemma 4.5, we have for all $k \in \{1,2\}$, $D_k(W \cap B_j) = 0$ if $j \leq r - \phi_W$ and $\left[1-\frac{1}{\gamma}, 1+\frac{1}{\gamma}\right]\frac{|W|}{\tau n}$ if $j \geq r - \phi_W + 1$. Therefore, for any $k \in \{1,2\}$, the index of the bucket $b(s_i)$, when $s_i \sim \mathsf{COND}_k(W)$, takes any particular value in $\{r-\phi_W+1, r-\phi_W+2, \ldots, r\}$ with probability $\left[1-\frac{1}{\gamma}, 1+\frac{1}{\gamma}\right]\frac{1}{\phi_W}$ and any value in $\{1, 2, \ldots, r-\phi_W\}$ with probability 0.

$$\Pr\left[B^{\mathsf{YES}}(s_i) = b_i \mid V_i^{\mathsf{YES}} = v_i, V_{i-1}^{\mathsf{YES}} = v_{i-1}, \mathcal{B}^{\mathsf{YES}}(\boldsymbol{S}_{i-1}) = \boldsymbol{B}_{i-1}\right] \leq \left(\frac{1+1/\gamma}{1-1/\gamma}\right)\Pr\left[B^{\mathsf{NO}}(s_i) = b_i \mid V_i^{\mathsf{NO}} = v_i, V_{i-1}^{\mathsf{NO}} = v_{i-1}, \mathcal{B}^{\mathsf{NO}}(\boldsymbol{S}_{i-1}) = \boldsymbol{B}_{i-1}\right]$$

Hence, in this case, we have

$$\Pr\left[V_i^{\mathsf{YES}} = v_i, B^{\mathsf{YES}}(s_i) = b_i \mid V_{i-1}^{\mathsf{YES}} = v_{i-1}, \mathcal{B}^{\mathsf{YES}}(\boldsymbol{S}_{i-1}) = \boldsymbol{B}_{i-1}\right] \leq \left(\frac{1+1/\gamma}{1-1/\gamma}\right)^2\Pr\left[V_i^{\mathsf{NO}} = v_i, B^{\mathsf{NO}}(s_i) = b_i \mid V_{i-1}^{\mathsf{NO}} = v_{i-1}, \mathcal{B}^{\mathsf{NO}}(\boldsymbol{S}_{i-1}) = \boldsymbol{B}_{i-1}\right]$$

Since $\left(\frac{1+1/\gamma}{1-1/\gamma}\right)^2(1+100(i-1)/\gamma) \leq (1+100i/\gamma)$, the induction hypothesis is true for i as well.
Fixing any v_q and summing the inequality in for all possible values of $(b_1, \ldots, b_q)$, we get the result.

5 $\Omega(\log\log n)$ lower bound for testing Label Invariant Property

In this section, we prove Theorem 1.2. We will prove that there is a label invariant property testing which takes $\tilde{\Omega}(\log\log n)$ COND queries. CFGM [CFGM16] defined a label invariant property (called Even uniblock property) and show a lower bound of $\Omega(\sqrt{\log\log n})$ on the query complexity. We will improve this lower bound to $\tilde{\Omega}(\log\log n)$.

Even uniblock property: A distribution on $[n]$ is called even uniblock if and only if it is uniform over some subset $U \subseteq [n]$ of size $2^{2\kappa}$ for some $\frac{\log n}{8} \leq \kappa \leq \frac{3\log n}{8}$. Note that a distribution D on $[n]$ is said to be uniform on set $S \subseteq [n]$ if we have $D(i) = 1/|S|$ for all $i \in S$ and 0 otherwise.

Odd uniblock property: A distribution on $[n]$ is called odd uniblock if and only if it is uniform over some subset $U \subseteq [n]$ of size $2^{2\kappa+1}$ for some $\frac{\log n}{8} \leq \kappa \leq \frac{3\log n}{8}$. Note that a distribution D on $[n]$ is said to be uniform on set $S \subseteq [n]$ if we have $D(i) = 1/|S|$ for all $i \in S$ and 0 otherwise.

Observation 5.1 ([CFGM16])**.** *For any distribution D_e satisfying even uniblock property and any distribution D_o satisfying odd uniblock property, we have $d_{TV}(D_e, D_o) \geq \frac{1}{2}$.*

To prove Theorem 1.2 we will use the same hard instances (explained in Section 5.1) as defined in [CFGM16]. Then, in Section 5.2, we define our Good event, and in Section 5.3, we prove (using the WCOND model) the Good event happens with high probability. Finally, in Section 5.4 we prove Theorem 1.2 using Theorem 2.5.

5.1 Distributions over Even uniblock and Odd uniblock properties

We now consider two distributions $\mathcal{I}_{\mathsf{EVEN}}$ and $\mathcal{I}_{\mathsf{ODD}}$, over distributions satisfying even uniblock property and over distributions satisfying odd uniblock property, respectively. Note that these are the same distributions considered in [CFGM16].

1. Uniformly choose an integer κ such that $\frac{\log n}{8} \leq \kappa \leq \frac{3\log n}{8}$.
2. Uniformly pick a set S_e of size $2^{2\kappa}$ and a set S_o of size $2^{2\kappa+1}$.
3. The distribution $\mathcal{I}_{\mathsf{EVEN}}$ is a uniform distribution on S_e while $\mathcal{I}_{\mathsf{ODD}}$ is a uniform distribution on S_o

5.2 The Good event

Definition 5.2 ([CFGM16])**.** *We call a number b large with respect to S_e if $\frac{b|S_e|}{n} \geq 2^{\sqrt{\log n}}$ and small with respect to S_e if $\frac{b|S_e|}{n} < \frac{1}{2^{\sqrt{\log n}}}$. We have an analogous definition for S_o. Note that $|S_e| = 2^{2\kappa}$ and $|S_o| = 2^{2\kappa+1}$.*

Definition 5.3. *A node v of the decision tree R (recall that $\boldsymbol{A}_v = (A_{v_0}, \ldots, A_{v_i})$ is the set of all queries made in the path from the root to the node v in the decision tree.) is called good if for all atoms $A \in At(\boldsymbol{A}_v)$, $|A|$ is large with respect to both S_e and S_o or small with respect to both S_e and S_o. A node v is called bad if it is not good.*

We now show the following lemma that helps us in bounding the probability of a node being bad in a decision tree. The proof of the following lemma is an adaptation of the arguments used in [CFGM16].

Lemma 5.4. *Irrespective of whether the input $x \in_R \mathcal{I}_{\mathsf{EVEN}}$ or $x \in_R \mathcal{I}_{\mathsf{ODD}}$, the probability that a node of a decision tree is bad is at most $\frac{8\cdot 2^q}{\sqrt{\log n}}$.*

Proof. An atom A is neither large nor small with respect to S_e if $n2^{-\sqrt{\log n}} < |A||S_e| < n2^{\sqrt{\log n}}$ where $|S_e| = 2^{2\kappa}$ and κ is chosen uniformly such that $\frac{\log n}{8} \leq \kappa \leq \frac{3\log n}{8}$. Therefore, for a fixed $|A|$, there are at most $\sqrt{\log n}$ values of κ, which will make it neither large nor small with respect to S_e. So there are at most $2\sqrt{\log n}$ values of κ, which will make it neither large nor small with respect to both S_e and S_o. Since the range of κ is $\log n/4$ (in both μ_e and μ_o), with probability at most $\frac{8}{\sqrt{\log n}}$, the atom A will be neither large nor small. Since there are at most 2^q atoms corresponding to a node, the statement of the lemma follows. □

From now on, fix

$$\beta = 2^{\frac{\sqrt{\log n}}{4}}.$$

Lemma 5.5. *With probability at least $1 - o(1)$, for all good nodes v and for all atoms $A \in \mathsf{At}(\boldsymbol{A}_v)$:*

1. *If $|A|$ is large, we have $|A \cap S_e| \in [1 - 1/\beta, 1 + 1/\beta]\frac{|A||S_e|}{n}$ and $|A \cap S_o| \in [1 - 1/\beta, 1 + 1/\beta]\frac{|A||S_o|}{n}$.*

2. *If $|A|$ is small, we have $|A \cap S_e| = |A \cap S_o| = 0$.*

Proof. If $\mathbb{E}[|S_e \cap A|] = \frac{|A||S_e|}{n} \leq \frac{1}{2^{\sqrt{\log n}}}$ for any atom A then by Markov's inequality $\Pr[|A \cap S_e| > 0] \leq \frac{1}{2^{\sqrt{\log n}}}$. Taking a union bound over all the atoms corresponding to all the good nodes (which can be at most $2^{q+q^2} \leq (\log n)^{\log\log n}$), with probability at least $1 - o(1)$,

$$\text{for any small atom } A, \text{ we have } |S_e \cap A| = 0. \tag{18}$$

Now note that for any large atom, $\mathbb{E}[|S_e \cap A|] = \frac{|A||S_e|}{n} \geq 2^{\sqrt{\log n}}$. Therefore, by Chernoff bound,

$$\Pr\left[|A \cap S_e| \notin \left[1 - \frac{1}{\beta}, 1 + \frac{1}{\beta}\right] \frac{|A||S_e|}{n}\right] \leq O\left(exp\left(-2^{\sqrt{\log n}/\beta^2}\right)\right) \leq O\left(exp\left(-2^{\sqrt{\log n}/2}\right)\right).$$

The proof analogously holds for S_o. □

Definition 5.6. *The event $\mathsf{Good}(\mathcal{A}, x)$ is defined as "the run of the algorithm $\mathcal{A}$ on input x does not pass through a bad node."*

5.3 The event $\mathsf{Good}(\mathcal{A}, x)$ happens with high probability

In this subsection, we will prove the following lemma.

Lemma 5.7. *For any core-adaptive tester $\mathcal{A}$ that makes $q \leq \frac{\log\log n}{100\log\log\log n}$ queries,*

1. $\Pr_{x \in_R \mathcal{I}_{\mathsf{EVEN}}}[\mathsf{Good}(\mathcal{A}, x)] \geq 1 - o(1)$
2. $\Pr_{x \in_R \mathcal{I}_{\mathsf{ODD}}}[\mathsf{Good}(\mathcal{A}, x)] \geq 1 - o(1)$

5.3.1 Proof for WCOND oracle

Let us consider the event G_1 that denotes the run of an algorithm $\mathcal{A}$ having access to WCOND oracle, on input x does not pass through a bad node. In Lemma 5.8, we claim that G_1 happens with a high probability. Then we show that given the event G_1 happens, Good happens with a high probability. As a consequence, we conclude that Good happens with a high probability, which completes the proof of Lemma 5.7.

Lemma 5.8. *If an algorithm $\mathcal{A}$ makes $q \leq \frac{\log\log n}{100\log\log\log n}$ adaptive WCOND queries, then*

1. $\Pr_{x \in_R \mathcal{I}_{\mathsf{EVEN}}}[\mathsf{G}_1(\mathcal{A}, x)] \geq 1 - o(1)$,
2. $\Pr_{x \in_R \mathcal{I}_{\mathsf{ODD}}}[\mathsf{G}_1(\mathcal{A}, x)] \geq 1 - o(1)$.

Proof. First, observe that for any WCOND query, Step 3a (of Definition 3.6) does not depend on the input distribution (on which the WCOND query is placed). Thus, a randomized algorithm can simulate this step without accessing the input distribution (by picking an atom $V \in \mathsf{At}(U_i)$ with probability $|V|/|U_i|$). Let $\mathcal{A}'$ be the new (randomized) algorithm that simulates Step 3a. One can think of the randomized algorithm $\mathcal{A}'$ as a distribution over a set of algorithms $\mathcal{A}'_1, \mathcal{A}'_2, \cdots$, where each $\mathcal{A}'_r$ is an instantiation of $\mathcal{A}'$ by fixing internal randomness that is used to simulate Step 3a. Thus, each $\mathcal{A}'_r$ can be represented as a decision tree where each node denotes access (of the form either Step 3a or Step 3b) to the input distribution. Since the original algorithm $\mathcal{A}$ makes q WCOND queries, each decision tree has a height at most $2q$. Recall, by our assumption, $\mathcal{A}$ is a core-adaptive tester. Thus for any node at the i-th level, the number of children is $|O_i| + 1$ (follows from Step 1), which in turn is upper bounded by $q + 1$ (since $\mathcal{A}$ makes q WCOND queries, by the definition of O_i in Definition 3.3, $|O_i| \leq q$). So the number of nodes present in each such decision tree is at most $(q+1)^{2q}$.

Now, by Lemma 5.4 and taking a union bound over all the nodes in the decision tree, the probability that at least one of the nodes is bad is at most $\frac{8 \cdot 2^q \cdot (q+1)^{2q}}{\sqrt{\log n}}$. Thus when $q \leq \frac{\log\log n}{100\log\log\log n}$, for each $\mathcal{A}'_r$,

$$\Pr_{x \in_R \mathcal{I}_{\mathsf{EVEN}}}[\mathcal{A}'_r \text{ reaches a bad node on input } x] = o(1).$$

Since the randomized algorithm $\mathcal{A}'$ can be thought of as a distribution over the set of algorithms $\mathcal{A}'_1, \mathcal{A}'_2, \cdots$, we get that

$$\Pr_{x \in_R \mathcal{I}_{\mathsf{EVEN}}} [\mathcal{A}' \text{ reaches a bad node on input } x] = o(1).$$

By the construction of $\mathcal{A}'$, for every input x, the probability of reaching a bad node by $\mathcal{A}$ and $\mathcal{A}'$ are the same. Hence, $\Pr_{x \in_R \mathcal{I}_{\mathsf{EVEN}}} [\mathsf{G}_1(\mathcal{A}, x)] \geq 1 - o(1)$. A similar argument holds for $x \in_R \mathcal{I}_{\mathsf{ODD}}$. This concludes the proof. □

5.3.2 The total variation distance is small for testers given access to COND and WCOND oracle

From now on, throughout the rest of this subsection, we use $\mathcal{A}_{\mathsf{COND}}$ and $\mathcal{A}_{\mathsf{WCOND}}$ to denote the core-adaptive tester $\mathcal{A}$ (with the corresponding decision tree R) having access to the COND and WCOND oracle respectively. Next, for any $i \leq q$, let us consider the following two distributions: $L_{\mathsf{COND}}(i)$ and $L_{\mathsf{WCOND}}(i)$ are the distributions of nodes at the i-th level of the decision tree associated with $\mathcal{A}_{\mathsf{COND}}$ and $\mathcal{A}_{\mathsf{WCOND}}$ respectively.

Let G_2 be the event of Lemma 5.5, i.e., for all good nodes v and for all atoms $A \in \mathsf{At}(\boldsymbol{A}_v)$, we have $|A \cap S_e| \in [1 - 1/\beta, 1 + 1/\beta] \frac{|A||S_e|}{n}$ and $|A \cap S_o| \in [1 - 1/\beta, 1 + 1/\beta] \frac{|A||S_o|}{n}$. Now, we upper bound the total variation distance between the distribution over the nodes of R at any depth $i \leq q$, for $\mathcal{A}_{\mathsf{COND}}$ and $\mathcal{A}_{\mathsf{WCOND}}$, conditioned on the event $\mathsf{G}_1 \wedge \mathsf{G}_2$.

Lemma 5.9. *Given the event $\mathsf{G}_1 \wedge \mathsf{G}_2$ happens, for each $i \leq q$, the total variation distance between the two distributions $L_{\mathsf{COND}}(i)$ and $L_{\mathsf{WCOND}}(i)$ is at most $2i/\beta$.*

Proof. Consider an $i \leq q$. We denote the total variation distance between the two distributions $L_{\mathsf{COND}}(i)$ and $L_{\mathsf{WCOND}}(i)$ (conditioning on the event $\mathsf{G}_1 \wedge \mathsf{G}_2$) by $\mathsf{tv}(i)$. Note that if a node is bad, then all its descendants are also bad. For any node v, let $R(v)$ denote the subtree (of R) rooted at the node v. Let R' be the tree obtained from R by iteratively pruning $R(v) \setminus v$ (i.e., removing the subtree rooted at v while only keeping the node v) for all the bad nodes v in R, starting from the lowest level. Observe in R' no internal node is bad. Furthermore, the probability of reaching a bad node in R' is the same as that in R. So, it suffices to argue with the distribution of nodes at any level i of the tree R'.

Consider any good node v at depth i. We upper bound the total variation distance between distributions over children of v for $\mathcal{A}_{\mathsf{COND}}$ and $\mathcal{A}_{\mathsf{WCOND}}$, conditioned on the fact that the run of the tester is currently at the node v. It is not hard to see that if, for each node v, the total variation distance between distributions over children of v for $\mathcal{A}_{\mathsf{COND}}$ and $\mathcal{A}_{\mathsf{WCOND}}$ is upper bounded by η (for some $\eta \geq 0$), then

$$\mathsf{tv}(i+1) \leq \mathsf{tv}(i) + \eta. \tag{19}$$

Now, since $\mathcal{A}_{\mathsf{WCOND}}$ only differs at Step 3a (of the WCOND oracle) with $\mathcal{A}_{\mathsf{COND}}$, it suffices to upper bound the total variation distance for distributions over the atoms of U_v. If U_v has no large atom, by Lemma 5.5, we have $|U_v \cap S_e| = |U_v \cap S_o| = 0$ (conditioned on the event G_2). In this case, the total variation distance is zero. So, let us assume that U_v has at least one large atom.

Further, as we are conditioning on G_2, we have both $\frac{D_e(V)}{D_e(U_v)}$ and $\frac{D_o(V)}{D_o(U_v)} = \theta_V \frac{|V|}{|U_v|}$ where $\theta_V \in [1 - 2/\beta, 1 + 2/\beta]$. In this case, the total variation distance is at most

$$\sum_{V \in At(U)} \left| \frac{\theta_V |V|}{|U|} - \frac{|V|}{|U|} \right| \leq \frac{2}{\beta}.$$

Then by Equation 19,

$$\mathsf{tv}(i+1) \leq \mathsf{tv}(i) + 2/\beta.$$

So we conclude that for each $i \leq q$, $\mathsf{tv}(i) \leq 2i/\beta$. □

5.3.3 Good happens with high probability

Proof of Lemma 5.7. It directly follows from Lemma 5.9 that the probability that $\mathcal{A}_{\mathsf{COND}}$ reaches a bad node (on the input x) is at most

$$\sum_{i=1}^{q} 2i/\beta + \Pr[\overline{\mathsf{G}_1}] + \Pr[\overline{\mathsf{G}_2}] \leq q^2/\beta + \Pr[\overline{\mathsf{G}_1}] + \Pr[\overline{\mathsf{G}_2}].$$

Then by Lemma 5.8 and Lemma 5.5,

$$\Pr_{x \in_R \mathcal{I}_{\mathsf{EVEN}}} \left[\overline{\mathsf{Good}(\mathcal{A}, x)}\right] \leq q^2/\beta + o(1) = o(1)$$

where the last inequality follows for $q \leq \frac{\log\log n}{100 \log\log\log n}$. The same argument also holds for $\Pr_{x \in_R \mathcal{I}_{\mathsf{ODD}}} \left[\overline{\mathsf{Good}(\mathcal{A}, x)}\right]$, and that concludes the proof. □

5.4 Proof of Theorem 1.2

Lemma 5.10 ([CFGM16])**.** *Given the event* Good *happens, consider the resulting distributions over the set of leaves reached by the algorithm. These two distributions, under $\mathcal{I}_{\mathsf{EVEN}}$ compared to under $\mathcal{I}_{\mathsf{ODD}}$, are at most $\frac{2^{3q+1}}{\sqrt{\log n}}$ apart (in the total variation distance) from each other.*

We want to point out that in [CFGM16], the event Good in the above lemma stands for "none of the nodes in the decision tree is bad". However, the proof only uses the fact that none of the nodes encountered by the run of the algorithm is bad. Hence, the lemma remains valid for our definition of the event Good (as in Definition 5.6) as well.

Now consider a decision tree of the $\mathcal{A}$ and feed to it either μ_e or μ_o. Unless the queries made by $\mathcal{A}$ is $\tilde{\Omega}(\log\log n)$, in both cases, the event Good happens with $1 - o(1)$ probability (Lemma 5.7). Further, conditioned on the event Good, the total variation distance between the resulting distribution over the leaves is at most $o(1)$ (Lemma 5.10). Hence, the (unconditional) total variation distance between the resulting distribution over the leaves is at most $o(1) + o(1) = o(1)$. This means the $\mathcal{A}$ cannot distinguish between μ_e and μ_o (unless $q = \tilde{\Omega}(\log\log n)$).

6 Conclusion

In this paper, we introduce the WCOND model and show how this model can be used to obtain improved lower bounds for the seemingly stronger COND model. The concept of the core-adaptive tester was crucially used in previous works to prove the previous best lower bounds. In our paper, we kind of show that the core-adaptive tester with COND queries is similar to the WCOND samplers. In this context, we want to leave with an important question:

Are core-adaptive testers equivalent to a further restricted class of testers?

We believe this question is important for understanding exactly where the power of COND lies. Suppose we restrict the definition of the configuration (Definition 3.2) of the i sample by removing the Condition 2, i.e., now the configuration c'_i is the information whether $s_i = s_j$ for which $j \leq i - 1$. Consider a restricted class of core-adaptive testers where, for any $i \in [q]$, the i-th query A_i is

1. Either of the form $\mathsf{COND}_{D_k}(U)$ where U is the set of unseen elements (and in this case, we get a sample $s_i \sim \mathsf{COND}_{D_k}(U)$),
2. Or of the form $\mathsf{COND}_{D_k}(O_i)$ where as before O_i is some subset of previously seen elements.

Further, the next query set A_{i+1} is decided by the configurations of the previous samples.

In simplest words, at any step, the tester from this restricted class has the power to (i) sample $s \sim \mathsf{COND}(U)$ for any set of unseen elements U, (ii) can determine the value of $j \leq i-1$ such that $s_j = s$ where $s \sim \mathsf{COND}(O)$. One can also say that this class of testers does not consider the unique atoms to which the samples belong for any decision. It is easy to see that this restricted class of testers is the same as the tester when given access to WCOND queries.

This seems like a big restriction, and indeed the size of the decision tree for testers in this class is q^q as compared to 2^{q^2} before. Interestingly, to the best of our knowledge, all the testers for any label invariant property in the literature belong to this restricted class. Further, our work shows that the power of testers from this restricted class is the same as that of general core-adaptive testers in the context of equivalence and even uniblock testing. This raises an important question.

For what subset of label invariant properties does this restricted class of testers have the same power as general core-adaptive testers?

References

[ACK18] Jayadev Acharya, Clément L Canonne, and Gautam Kamath. A chasm between identity and equivalence testing with conditional queries. *Theory of Computing*, 14(19):1–46, 2018.

[BC18] Rishiraj Bhattacharyya and Sourav Chakraborty. Property testing of joint distributions using conditional samples. *ACM Transactions on Computation Theory (TOCT)*, 10(4):1–20, 2018.

[BDKR02] Tuğkan Batu, Sanjoy Dasgupta, Ravi Kumar, and Ronitt Rubinfeld. The complexity of approximating entropy. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, pages 678–687, 2002.

[BFR+00] Tugkan Batu, Lance Fortnow, Ronitt Rubinfeld, Warren D Smith, and Patrick White. Testing that distributions are close. In *Proceedings 41st Annual Symposium on Foundations of Computer Science*, pages 259–269. IEEE, 2000.

[BFR+13] Tuğkan Batu, Lance Fortnow, Ronitt Rubinfeld, Warren D Smith, and Patrick White. Testing closeness of discrete distributions. *Journal of the ACM (JACM)*, 60(1):1–25, 2013.

[Can20] Clément L Canonne. A survey on distribution testing: Your data is big. but is it blue? *Theory of Computing*, pages 1–100, 2020.

[CCK+21] Clément L Canonne, Xi Chen, Gautam Kamath, Amit Levi, and Erik Waingarten. Random restrictions of high dimensional distributions and uniformity testing with subcube conditioning. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*. SIAM, 2021.

[CDVV14] Siu-On Chan, Ilias Diakonikolas, Paul Valiant, and Gregory Valiant. Optimal algorithms for testing closeness of discrete distributions. In *Proceedings of the twenty-fifth annual ACM-SIAM symposium on Discrete algorithms*, pages 1193–1203. SIAM, 2014.

[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.

[CJLW21] Xi Chen, Rajesh Jayaram, Amit Levi, and Erik Waingarten. Learning and testing junta distributions with sub cube conditioning. In *Conference on Learning Theory*, pages 1060–1113. PMLR, 2021.

[CKM23] Diptarka Chakraborty, Gunjan Kumar, and Kuldeep S. Meel. Support size estimation: The power of conditioning. In *Proceedings of the 48th International Symposium on Mathematical Foundations of Computer Science, MFCS 2023, August 28 - September 01, 2023 (to appear)*, 2023.

[CM19] Sourav Chakraborty and Kuldeep S Meel. On testing of uniform samplers. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33-01, pages 7777–7784, 2019.

[CR14] Clément Canonne and Ronitt Rubinfeld. Testing probability distributions underlying aggregated data. In *International Colloquium on Automata, Languages, and Programming*, pages 283–295. Springer, 2014.

[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.

[Fis01] Eldar Fischer. The art of uninformed decisions. *Bull. EATCS*, 75:97, 2001.

[FJO+15] Moein Falahatgar, Ashkan Jafarpour, Alon Orlitsky, Venkatadheeraj Pichapati, and Ananda Theertha Suresh. Faster algorithms for testing under conditional sampling. In *Conference on Learning Theory*, pages 607–636. PMLR, 2015.

[GJM22] Priyanka Golia, Brendan Juba, and Kuldeep S Meel. A scalable Shannon entropy estimator. In *Computer Aided Verification: 34th International Conference, CAV 2022, Haifa, Israel, August 7–10, 2022, Proceedings, Part I*, pages 363–384. Springer, 2022.

[KT19] Gautam Kamath and Christos Tzamos. Anaconda: A non-adaptive conditional sampling algorithm for distribution testing. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 679–693. SIAM, 2019.

[MPC20] Kuldeep S Meel, Yash Pralhad Pote, and Sourav Chakraborty. On testing of samplers. *Advances in Neural Information Processing Systems*, 33:5753–5763, 2020.

[Nar21] Shyam Narayanan. On tolerant distribution testing in the conditional sampling model. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 357–373. SIAM, 2021.

[OS18] Krzysztof Onak and Xiaorui Sun. Probability–revealing samples. In *International Conference on Artificial Intelligence and Statistics*. PMLR, 2018.

[Val11] Paul Valiant. Testing symmetric properties of distributions. *SIAM Journal on Computing*, 40(6):1927–1968, 2011.