B. Comp. Dissertation

Fast Splitting Algorithms for Noisy and Sparsity-Constrained Group Testing

By

Tan Thong Cai, Nelvin

Department of Computer Science

School of Computing

National University of Singapore

2020/2021 Semester 2

B. Comp. Dissertation

Fast Splitting Algorithms for Noisy and Sparsity-Constrained Group Testing

By

Tan Thong Cai, Nelvin

Department of Computer Science

School of Computing

National University of Singapore

2020/2021 Semester 2

Project No: H240070
Advisor: Asst. Prof. Jonathan Scarlett
Deliverables:
Report: 1 Volume

Abstract

In group testing, the goal is to identify a subset of defective items within a larger set of items based on tests whose outcomes indicate whether at least one defective item is present. This problem is relevant in areas such as medical testing, DNA sequencing, communication protocols, and many more. In this report, we study (i) a noisy version of the problem, where each test outcome is independently flipped with some constant probability, and (ii) a sparsity-constrained version of the problem, in which the testing procedure is subjected to one of the following two constraints: items are finitely divisible and thus may participate in at most γ tests; or tests are size-constrained to pool no more than ρ items per test. Under each of these settings, considering the for-each recovery guarantee with asymptotically vanishing error probability, we introduce a fast splitting algorithm and establish its near-optimality not only in terms of the number of tests, but also in terms of the decoding time. While the most basic formulations of our algorithms require $\Omega(n)$ storage for each algorithm, we also provide low-storage variants based on hashing, with similar recovery guarantees.

Subject Descriptors:
E.4 Coding and Information Theory
F.2 Analysis of Algorithms and Problem Complexity
G.3 Probability and Statistics
G.2 Discrete Mathematics

Keywords:
Group Testing, sparsity, sublinear-time decoding

Implementation Software and Hardware:
R

Acknowledgement

I am deeply indebted to my supervisor Prof. Jonathan Scarlett for advising me on this project. He is very supportive and patient with me during the entire course of this research (and in fact, for most of my undergraduate years). Under his guidance, I learnt a great deal about numerous aspects of academic research. It has truly been a great pleasure to study and research under him, and I could not have asked for a better role model, teacher, and supervisor.

I would also like to extend my gratitude to the amazing teachers who have helped developed my foundations in Computer Science and Statistics. In particular, I would like to thank Prof. Kim Cuc (Daisy) Pham and Prof. Wanjie Wang for their superb teaching and advice.

Thanks should also go to my bright and motivated friends who have supported me along the way, especially Andrew Tan, Sean Ng, Gary Goh, Nicholas Teh, Kang Tze Ng, Heng Guang Lim, Sebastian Lie, Kenneth Goh, and Yanxun Wang for stimulating discussions and a source of encouragement.

Lastly, my deepest gratitude goes to my family for their omnipresent warmth and support.

List of Figures

List of Tables

Table of Contents

Chapter 1

Introduction

In the group testing problem, the goal is to identify a small subset $\mathcal{S}$ of defective items of size k within a larger set of items of size n, based on a number T of tests. This problem is relevant in areas such as medical testing, DNA sequencing, and communication protocols [3, Sec. 1.7], and more recently, utility in testing for COVID-19 [4, 5, 6]. Some important distinctions [3] in group testing are as follows:

- **Adaptive vs. non-adaptive.** Under adaptive testing, the test pools are designed sequentially, and each one can depend on the previous test outcomes. Under non-adaptive testing, the test pools are designed in advance before the testing process. This makes parallel implementation of the tests more viable.

- **Noiseless vs. noisy testing.** Under noiseless testing, we are guaranteed that the test procedure works perfectly: We get a negative test outcome if all items in the testing pool are non-defective, and a positive outcome if at least one item in the pool is defective. Under noisy testing, errors can occur, either according to some specified random model or in an adversarial manner.

- **Known vs. unknown number of defectives.** This distinguishes algorithms that need to be given the true number of defectives (or require estimation of the number of defectives first), and those that do not.

- **For-all vs. for-each guarantees.** In *combinatorial (for-all)* group testing, one seeks to construct a test design that guarantees the recovery of all defective sets up to a certain size with zero error probability. In contrast, in *probabilistic (for-each)* group testing, the test design may be randomized, and the algorithm is allowed some small error probability.

In this report, we present algorithms for noisy and sparsity-constrained (bounded tests-per-item or items-per-test) variants of group testing with a near-optimal sublinear decoding time, building on techniques recently proposed for the unconstrained noiseless group testing problem [7, 8]. These extensions come with new challenges presented by the need to handle both false positive and false negative tests in the noisy setting, and the infeasibility of the designs in [7, 8] in the sparsity-constrained setting.

1.1 Problem Setup

Let n denote the number of items, which we label as $\{1, \dots, n\}$. Let $\mathcal{S} \subset \{1, \dots, n\}$ denote the fixed set of defective items, and let $k = |\mathcal{S}|$ be the number of defective items. To avoid cumbersome notation, we present our algorithms in a form that uses k directly; however, the analysis goes through unchanged when an upper bound $\bar{k} \geq k$ is used instead, and $\bar{k}$ replaces k in the number of tests and decoding time.

We are interested in asymptotic scaling regimes in which n is large and k is comparatively small, and thus assume that $k = o(n)$ throughout. We let $T = T(n)$ be the *number of tests* performed. In the noiseless setting, the i-th test takes the form

$$Y^{(i)} = \bigvee_{j \in \mathcal{S}} X_j^{(i)}, \tag{1.1}$$

where the test vector $X^{(i)} = \big(X_1^{(i)}, \dots, X_n^{(i)}\big) \in \{0,1\}^n$ indicates which items are are included in the test, and $Y^{(i)} \in \{0,1\}$ is the resulting observation, indicating whether at least one defective item was included in the test. The goal of group testing is to design a sequence of tests $X^{(1)}, \dots, X^{(T)}$, with T ideally as small as possible, such that the outcomes can be used to reliably recover the defective set $\mathcal{S}$ with probability close to one, while ideally also having a low-complexity decoding procedure. We focus on the non-adaptive setting, in which all tests $X^{(1)}, \dots, X^{(T)}$ must be designed prior to observing any outcomes.

We consider the *for-each* recovery guarantee; specifically, we seek to develop a randomized algorithm that, for any fixed defective set $\mathcal{S}$ of cardinality k, produces an estimate $\widehat{\mathcal{S}}$ such that the error probability $P_e := \mathbb{P}[\widehat{\mathcal{S}} \neq \mathcal{S}]$ is asymptotically vanishing as $n \to \infty$. For all of our algorithms, only the tests $\{X^{(i)}\}_{i=1}^{T}$ will be randomized, and the decoding procedure will be deterministic given the test outcomes.

Notation. Throughout the report, the function $\log(\cdot)$ has base e, and we make use of Bachmann-Landau asymptotic notation (i.e., O, o, Ω, ω, Θ), as well as the notation $\widetilde{O}(\cdot)$, which omits poly-logarithmic factors in its argument.

1.1.1 Noisy Setting

Generalizing (1.1), we consider the following widely-adopted symmetric noise model:

$$Y^{(i)} = \left(\bigvee_{j \in \mathcal{S}} X_j^{(i)} \right) \oplus Z, \tag{1.2}$$

where $Z \sim \text{Bernoulli}(p)$ for some $p \in (0, 1/2)$, and $\oplus$ denotes modulo-2 addition. While the symmetry assumption may appear to be restrictive, our results and analysis will hold with essentially no change under any non-symmetric random noise model where $0 \to 1$ flips and $1 \to 0$ flips both have probability at most p.

1.1.2 Sparsity-Constrained Setting

In the sparsity-constrained group testing problem [2], the testing procedure is subjected to one of two constraints:

- Items are *finitely divisible* and thus may participate in at most $\gamma = o(\log n)$ tests;
- Tests are *size-constrained* and thus contain no more than $\rho = o(n/k)$ items per test.

For instance, in the classical application of testing blood samples for a given disease [9], the γ-divisible items constraint may arise when there are limitations on the volume of blood provided by each individual, and the ρ-sized test constraint may arise when there are limitations on the number of samples that the machine can accept, or on the number that can be mixed together while avoiding undesirable dilution effects.

It is well known that if each test comprises of $\Theta(n/k)$ items, then $\Theta(\min\{n, k \log n\})$ tests suffice for group testing algorithms with asymptotically vanishing error probability [10, 11, 12, 13]. Moreover, this scaling is known to be optimal [14]. Hence, the parameter regime of primary interest in the size-constrained setting is $\rho = o(n/k)$. By a similar argument, the parameter regime of primary interest in the finitely divisible setting is $\gamma = o(\log n)$.

It was shown in [2] that in the case of finitely divisible items, if the tests are subject to random noise of the form in (1.2), then the error probability is bounded away from zero regardless of the total number of tests whenever $\gamma = o(\log k)$. Thus, at least in most scaling regimes of interest, handling noise and finite-divisibility constraints simultaneously would require changing the noise model and/or the recovery criteria, and we make no attempt to do so. On the other hand, for noisy size-constrained tests, schemes that attain asymptotically vanishing error probability do indeed exist [2]. However, the techniques that we introduce for the noisy setting and the size-constrained setting appear to be incompatible (see Chapter 6 for a brief explanation); we thus only consider the two separately, and leave their combination for future work.

Mathematical and Computational Assumptions. For all algorithms presented in this report, we assume a word-RAM model of computation; for instance, with n items and T tests, it takes $O(1)$ time to read a single integer in $\{1, \dots, n\}$ from memory, perform arithmetic operations on such integers, fetch a single test outcome indexed by $\{1, \dots, T\}$ and so on.

For simplicity of notation, we assume throughout the analysis that k, n, and ρ are powers of two. Our algorithm only requires an upper bound on the number of defectives, and hence, any other value of k can simply be rounded up to a power of two. In addition, the total number of items n can be increased to a power of two by adding "dummy" non-defective items, and ρ can be rounded down without impacting our final scaling laws (we do not seek to characterize the precise constants).

Reference	Number of tests	Decoding time	Construction
Lower Bound [15]	$\Omega\big(k \log \frac{n}{k}\big)$	–	–
Inan et al. [18]	$O(k \log n)$	$\Omega(n)$	Explicit
Inan et al. [19]	$O(k \log n)$	$O\big(k^3 \cdot \log k + k \log n\big)$	Explicit
NCOMP & NDD [10, 20, 17]	$O(k \log n)$	$\Omega(n)$	Randomized
GROTESQUE [21]	$O(k \cdot \log k \cdot \log n)$	$O\big(k(\log n + \log^2 k)\big)$	Randomized
SAFFRON [22]	$O(k \cdot \log k \cdot \log n)$	$O(k \cdot \log k \cdot \log n)$	Randomized
BMC [23]	$O(k \log n)$	$O(k^2 \cdot \log k \cdot \log n)$	Randomized
This report	$O(k \log n)$	$O\big(\big(k \log \frac{n}{k}\big)^{1+\epsilon}\big)$	Randomized

Table 1.1: Overview of existing noisy non-adaptive group testing results under the for-each guarantee. A construction is said to be explicit if its test matrix can be computed deterministically in polynomial time, and in the final row, ϵ is an arbitrarily small positive constant.

1.2 Related Work

While extensive works have studied the number of tests for various group testing strategies (see [3] for a survey), relatively fewer have sought efficient $\text{poly}(k \log n)$ decoding time. For the standard noiseless group testing problem, the most relevant existing results come from two recent concurrent works [7, 8], which showed that there exists a non-adaptive group testing algorithm that succeeds with $O(k \log n)$ tests and has $O(k \log n)$ decoding time. We will build on their splitting techniques to develop our algorithms in this report (see Figures 2.1, 3.1, and 4.1 below).

For noisy non-adaptive group testing under the noise model in (1.2), the most relevant existing results are summarized in Table 1.1. Under $\Omega(n)$-decoding time, we note that the references shown are only illustrative examples, and that several additional works also exist with $O(k \log n)$ scaling, e.g., [15, 16, 17]. More relevant to our work is the fundamental limitation that the works attaining $O(k \log n)$ scaling only attain a quadratic or worse dependence in k in the decoding time (or $\Omega(n)$). On the other hand, GROTESQUE and SAFFRON attain $k\,\text{poly}(\log n)$ decoding time, but fail to attain order-optimality in the number of tests.

For (noiseless) sparsity-constrained group testing, we present a summary of previous algorithms attaining the for-each recovery guarantee in Table 1.2. Our algorithm for finitely divisible items matches that of the COMP algorithm[1] in the number of tests when $\gamma = \omega(1)$ (and comes close more generally), while having much lower decoding time. Furthermore, our algorithm for size-constrained tests succeeds with an order-optimal $O(n/\rho)$ number of tests, with matching $O(n/\rho)$ decoding time.

In a distinct but related line of works, the for-all recovery guarantee (i.e., zero error probability) was considered [24, 25, 26, 27], with typical results for the unconstrained setting requiring $O(k^2 \log n)$ tests and $\mathrm{poly}(k \log n)$ decoding time. In the sparsity-constrained setup, [27] gives a lower bound of $\Omega\big(\min\big\{n, k^{\frac{2k}{\gamma-1+k}} n^{\frac{k}{\gamma-1+k}}\big\}\big)$ and an algorithm that requires $O\big(\min\big\{n, kn^{\frac{k}{\gamma-1+k}}\big\}\big)$ tests and runs in $\mathrm{poly}(k) + O(T)$ time for the γ-divisible items, and a lower bound of $\Omega\big(k\frac{n}{\rho}\big)$ and an algorithm that requires $T = O\big(k\frac{n}{\rho}\big)$ tests and runs in $\mathrm{poly}(k) + O(T)$ time for the ρ-sized tests. Under all variants of the group testing problem, the stronger zero-error-probability guarantee comes at the price of requiring considerably more tests, and we thus omit detailed comparisons to our results.

1.3 Summary of Results

Here we informally summarize our main results, formally stated in Theorems 2.2.1, 3.2.1, and 4.2.1.

- **Noisy setting:** For any parameters $t = O(1)$ and $\epsilon \in (1/t, 1)$, there exists a non-adaptive group testing algorithm that succeeds with probability $1 - O\big(\big(k \log \frac{n}{k}\big)^{1-\epsilon t}\big)$ using $O(k \log n)$ tests and $O\big(\big(k \log \frac{n}{k}\big)^{1+\epsilon}\big)$ decoding time.

- **Finitely-divisible items:** A special case of our result states that for any $\beta_n = \frac{1}{\mathrm{poly}(\log n)}$, there exists a non-adaptive group testing algorithm that succeeds with probability $1 - O(\beta_n)$ using $\widetilde{O}\big(\gamma k n^{1/\gamma}\big)$ tests and $O\big(\gamma k n^{1/\gamma}\big)$ decoding time provided that $\gamma = \omega(1)$. The

[1]The COMP algorithm simply labels any item in an negative test as non-defective, and all other items as defective.

	Reference	Number of tests	Decoding time	Construction
γ-divisible items	Lower Bound [28, 29, 1]	$\Omega\big(\gamma k \max\{k, \frac{n}{k}\}^{1/\gamma}\big)$	–	–
	Gandikota et al. [2]	$O\big(\gamma k^2 \big(\frac{n}{k^2}\big)^{1/\gamma}\big)$	$O\big(k^2 \log\big(\frac{n}{k^2}\big)\big)$	Explicit
	COMP [2]	$\widetilde{O}(\gamma k n^{1/\gamma})$	$\Omega(n)$	Randomized
	DD [29, 1]	$O\big(\gamma k \max\{k, \frac{n}{k}\}^{\frac{1+o(1)}{\gamma}}\big)$	$\Omega(n)$	Randomized
	This report	$\widetilde{O}(\gamma k n^{1/\gamma})$	$O(\gamma k n^{1/\gamma})$	Randomized
ρ-sized tests	Lower Bound [2, 1]	$\Omega\big(\frac{n}{\rho}\big)$	–	–
	Gandikota et al. [2]	$O\big(\max\{\frac{n}{\rho}\log\rho, k^2\log\big(\frac{n}{k^2}\big)\}\big)$	$O(T)$	Explicit
	COMP & DD [2, 1]	$O\big(\frac{n}{\rho}\big)$	$\Omega(n)$	Randomized
	This report	$O\big(\frac{n}{\rho}\big)$	$O\big(\frac{n}{\rho}\big)$	Randomized

Table 1.2: Overview of existing noiseless non-adaptive group testing results for non-adaptive sparsity-constrained group testing under the for-each guarantee. For entries containing $\widetilde{O}(\cdot)$ notation, the results correspond to $\frac{1}{\mathrm{poly}(\log n)}$ error probability, but more general variants are also available. A construction is said to be explicit if its test matrix can be computed deterministically in polynomial time; the results shown for explicit constructions additionally require $k = O(\sqrt{n})$.

case of finite γ will also be handled with only slightly worse scaling laws, and we will specify the precise dependence on β_n, without resorting to $\widetilde{O}(\cdot)$ notation.

- **Size-constrained tests:** For any $\zeta > 0$, there exists a non-adaptive group testing algorithm that succeeds with probability $1 - O\big(n^{-\zeta}\big)$ using $O\big(n/\rho\big)$ tests and $O\big(n/\rho\big)$ decoding time.

In the noisy setting, we significantly improve on the best previous known decoding time among any algorithm using an order-optimal $O(k \log n)$ number of tests. Specifically, [23] incurred a quadratic dependence on k, whereas we incur a near-linear dependence. Similarly, in the sparsity-constrained setting, we attain a decoding time that matches the number of tests where previous algorithms using the same number of tests incurred $\Omega(n)$ decoding time.

While our focus is on the number of tests and decoding time, another important practical consideration is the storage required. Naively, the algorithms attaining the above results require $\Omega(n)$ storage. However, in Chapter 5, we discuss storage reductions via hashing, attaining identical results with sublinear storage in the noisy and size-constrained settings, and similar (but slightly weaker) results in the finitely-divisible setting.

Chapter 2

Algorithm for the Noisy Setting

Our algorithm for the noisy setting resembles the non-adaptive binary splitting approach of [7, 8], where we first test large groups of items together, placing each group into a single randomly-chosen test among a sequence (i.e., subset) of tests. Afterwards, these groups are “split” into smaller sub-groups, while using the previous test outcomes to eliminate those believed to be non-defective. This process is repeated—with the elimination step ensuring that the number of groups under consideration does not grow too large—until a superset of $\mathcal{S}$ is found. Finally, $\mathcal{S}$ is deduced from the superset via a final sequence of tests. We can visualize this using a tree (see Figure 2.1), where each level of the tree represents a stage of the splitting process, each node represents a group of items, and each split creates two edges.

The main difference between our noisy algorithm and [7, 8] is that when deciding whether a given node is defective or not, we have to look several levels further down the tree (instead of only considering the single test outcome of the given node). This somewhat complicates the analysis, and leads to a small increase in the decoding time. Additionally, in order to reduce the effective noise level, each node in the tree is placed in multiple tests, rather than just one.

2.1 Description of the Algorithm

Following [7, 8], our algorithm considers a tree representation (see Figure 2.1), in which each node corresponds to a set of items. The levels of the tree are indexed by $l = \log_2 k, \dots, \log_2 n$

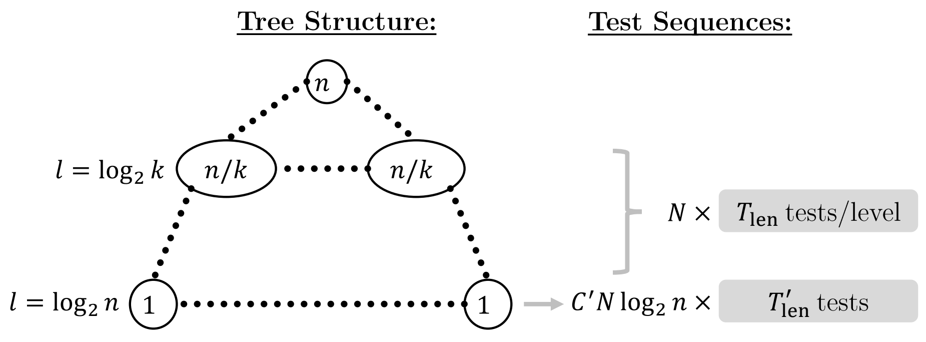

Figure 2.1: Tree structure of our algorithm for the noisy setting.

and the j-th node at the l-th level is denoted by $\mathcal{G}_j^{(l)} \subseteq \{1, \dots, n\}$. The algorithm works down the tree one level at a time, keeping a list of *possibly defective* ($\mathcal{PD}$) nodes, and performing tests to obtain such a list at the next level. When we perform tests at a given level, we treat each node as a "super-item"; including a node in a test amounts to including all of the items in the corresponding node $\mathcal{G}_j^{(l)}$. In addition, for the tree illustrated in Figure 2.1, we refer to nodes containing at least one defective item as defective nodes, to all other nodes as non-defective nodes, and to the sub-tree of defective nodes as the defective tree.

The testing is performed as follows: At each level of the tree, N sequences of tests are formed, each having length T_{len} (i.e., a total of NT_{len} tests per level). For each node and each of the N sequences, the node is placed into a single test, chosen uniformly at random among the T_{len} tests.

We make an important distinction between the *intermediate label* and *final label* of a given node:

- The intermediate label is formed via majority voting of the N tests that the node is included in.
- To obtain the final label of a given node, we look at the intermediate labels of all nodes up to r levels below the given node. If there exists any length-r path below the given node with more than $r/2$ positive intermediate labels, then we assign the node's final label to be positive. Otherwise, we assign it to be negative.

According to the tree structure in Figure 2.1, once we reach the later levels, there may be fewer

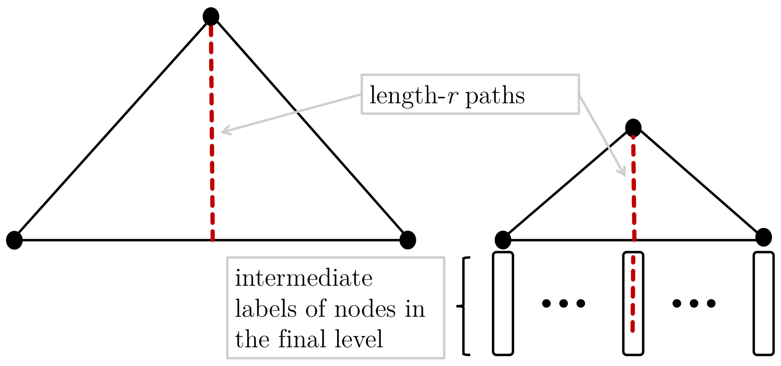

Figure 2.2: In the case of $l \leq \log_2 n - r$ (i.e., there are at least r levels below the node), the diagram on the left shows an example of a length-r path. Otherwise, we have $l > \log_2 n - r$ (i.e., there are less than r levels below the node), and the diagram on the right shows an example of a length-r path.

than r levels remaining. To account for such cases, we simply ensure that sufficiently many tests are performed at the final level so that a length-r "path" can be formed (here, no further branching is done, and each "node" is the same singleton); see Figure 2.2 for an illustration.

With the above notation and terminology in place, the overall test design is described in Algorithm 1, and the decoding procedure in Algorithm 2.

2.2 Algorithmic Guarantees

Theorem 2.2.1. (Algorithmic guarantees) *Let $\mathcal{S}$ be a (defective) subset of $\{1, \dots, n\}$ of cardinality $k = o(n)$. For any constants $\epsilon > 0$ and $t > 0$ satisfying $\epsilon t > 1$, there exist choices of $C, C', N = O(1)$ and $r = O(\log k + \log\log n)$ such that with $O\big(k \log n\big)$ tests, the preceding algorithm satisfies the following with probability at least $1 - O\big(\big(k \log \frac{n}{k}\big)^{1-\epsilon t}\big)$:*

- *The returned estimate $\widehat{\mathcal{S}}$ equals $\mathcal{S}$;*
- *The decoding time is $O\big(\big(k \log \frac{n}{k}\big)^{1+\epsilon}\big)$.*

Algorithm 1 Testing procedure for the noisy setting

```
Require: Number of items $n$, number of defective items $k$, and parameters $N$, $C$, and $C'$
1: Initialize $T_{\text{len}} = Ck$ and $T'_{\text{len}} = k$.
2: for each $l = \log_2 k, \dots, \log_2 n - 1$ do
3:     for each iteration in $\{1, \dots, N\}$ do
4:         for $j = 1, 2, \dots, 2^l$ do
5:             Place all items from each node $\mathcal{G}_j^{(l)}$ into a single test in a sequence of length $T_{\text{len}}$,
               chosen uniformly at random.
6: At level $l = \log_2 n$, form $C'N \log_2 n$ sequences of tests, each of length $T'_{\text{len}}$.
7: for each singleton at the final level do
8:     for each of the $C'N \log_2 n$ test sequences do
9:         Place the singleton in one of the corresponding $T'_{\text{len}}$ tests, chosen uniformly at random.
```

2.3 Analysis

The outline of the analysis is as follows:

- We first consider levels $l = \log_2 k, \dots, \log_2 n - 1$, and bound the probability that any node among three kinds—non-defective nodes at $l_{\min}$, defective nodes, and non-defective child nodes of defective nodes—are identified wrongly. Note that we do not have to consider other nodes, because if none of the nodes of these three kinds are identified wrongly, then the algorithm would not explore any of the other nodes when decoding.

- Conditioned on the correct identification of nodes of these three kinds, we consider the final level $l = \log_2 n$ and provide a bound for its error probability.

2.3.1 Analysis of Levels $l = \log_2 k, \dots, \log_2 n - 1$

We consider defective and non-defective nodes separately.

Defective nodes: Let $p_{\text{int}}^{(\text{d})}$ (respectively, $p_{\text{final}}^{(\text{d})}$) be the probability that the intermediate label (respectively, final label) of a given defective node is flipped from a one to a zero. Note that these may vary from node to node, but we will give upper bounds that hold uniformly.

For a given defective node, there are only two possible situations for each test it is in: A positive outcome due to no flip, or a negative test outcome due to a $1 \to 0$ flip. Hence, the number of negative tests that a given defective node participates in (i.e., the outcome is flipped) is distributed as Binomial(N, p). By the majority voting of N test outcomes at a given level, $p_{\text{int}}^{(\text{d})}$ is upper bounded by the probability that a given defective node participates in at least $N/2$ negative tests. Applying Hoeffding's inequality, we obtain

$$p_{\text{int}}^{(\text{d})} \le \exp\bigg(-2N\Big(\frac{1}{2} - p\Big)^2\bigg). \tag{2.1}$$

At this point, we introduce the variable t appearing in the theorem statement. Since $\exp\big(-2N(1/2-p)^2\big) \le \frac{2^{-2t}}{4} \Leftrightarrow N \ge \frac{2t\log 2+\log 4}{2(1/2-p)^2}$, we find that choosing $N \ge \frac{2t\log 2+\log 4}{2(1/2-p)^2}$ ensures that

$$p_{\text{int}}^{(\text{d})} \le \frac{2^{-2t}}{4}. \tag{2.2}$$

For the case that $l \le \log_2 n - r$, we consider the length-r paths below the defective node. The defective node will be labeled as negative if all 2^r paths below it have at least $r/2$ negative intermediate labels. The probability of this event is upper bounded by the probability that one particular *defective* path (i.e., every node along the path is defective) has at least $r/2$ negative intermediate labels, which is at most

$$\binom{r}{r/2}\big(p_{\text{int}}^{(\text{d})}\big)^{r/2} \le \big(4p_{\text{int}}^{(\text{d})}\big)^{r/2}, \tag{2.3}$$

where the left hand side (LHS) is by the union bound, and the right hand side (RHS) is by $\binom{r}{r/2} \le 2^r$. This gives $p_{\text{final}}^{(\text{d})} \le \big(4p_{\text{int}}^{(\text{d})}\big)^{r/2}$, and substituting (2.2) gives $p_{\text{final}}^{(\text{d})} \le 2^{-tr}$.

For the case that $l > \log_2 n - r$ (i.e., there are less than r levels below the given node), the probability of the defective path having at least $r/2$ negative intermediate labels remains unchanged, and hence, the preceding bound $p_{\text{final}}^{(\text{d})} \le 2^{-tr}$ still holds. Note that this step requires $C' \log_2 n \ge r$ in order to have enough intermediate labels per node in the final level to "pad" paths of length less than r (see Figure 2.2), and we will later set C' and r to ensure this.

Non-defective nodes: Let $p_{\text{int}}^{(\text{nd})}$ (respectively, $p_{\text{final}}^{(\text{nd})}$) be the probability that the intermediate label (respectively, final label) of a given non-defective node is flipped from a zero to a one. Again, these may vary from node to node, but we will give upper bounds that hold uniformly.

For a given non-defective node, there are four possible situations for each test: A negative outcome with no flip (i.e., no defectives), a negative outcome due to a $1 \to 0$ flip (i.e., at least one defective), a positive outcome with no flip (i.e., at least one defective), and a positive outcome due to a $0 \to 1$ flip (i.e., no defectives).

Focusing on one test sequence of length $T_{\text{len}} = Ck$ for now, let A be the event that a given non-defective node participates in a positive test, and let B be the event that the given node's test contains no defective item. We have

$$\mathbb{P}[A] = \mathbb{P}[B] \cdot \mathbb{P}[A|B] + \mathbb{P}[\neg B] \cdot \mathbb{P}[A|\neg B] \tag{2.4}$$

$$\overset{(a)}{\leq} \mathbb{P}[B] \cdot p + \frac{1}{C}(1-p) \tag{2.5}$$

$$\leq p + \frac{1}{C}, \tag{2.6}$$

where (a) holds since the probability of being in the same test as a given defective node is $1/T_{\text{len}} = 1/(Ck)$, and thus the union bound over k defective nodes gives $\mathbb{P}[\neg B] \leq 1/C$.

Equation (2.6) implies that for a given non-defective node, the number of positive tests that it participates in (out of N tests in total) is stochastically dominated by Binomial$(N, p + 1/C)$. Recalling that $p_{\text{int}}^{(\text{nd})}$ is the probability that a given non-defective node participates in at least $N/2$ positive tests, Hoeffding's inequality gives

$$p_{\text{int}}^{(\text{nd})} \leq \exp\left(-2N\left(\frac{1}{2} - p - \frac{1}{C}\right)^2\right), \tag{2.7}$$

where we require $1/2 - p - 1/C > 0 \Leftrightarrow C > 2/(1-2p)$. Hence, we set $C = \lceil 2/(1-2p) \rceil + 1$. Since $\exp\big(-2N(1/2 - p - 1/C)^2\big) \leq \frac{2^{-2t}}{16} \Leftrightarrow N \geq \frac{2t\log 2 + \log 16}{2(1/2-p-1/C)^2}$, we find that choosing $N \geq \frac{2t\log 2+\log 16}{2(1/2-p-1/C)^2}$ ensures that

$$p_{\text{int}}^{(\text{nd})} \leq \frac{2^{-2t}}{16}. \tag{2.8}$$

For the case that $l \leq \log_2 n - r$, we look at the length-r path below the non-defective node. The non-defective node will be labeled as positive if any of the 2^r paths below it has more than $r/2$ positive intermediate labels. By union bound over all 2^r paths, this probability is upper bounded as follows, similar to (2.3):

$$2^r \binom{r}{r/2} \big(p_{\text{int}}^{(\text{nd})}\big)^{r/2} \leq 2^r \big(4p_{\text{int}}^{(\text{nd})}\big)^{r/2} \leq \big(16p_{\text{int}}^{(\text{nd})}\big)^{r/2}. \tag{2.9}$$

This gives $p_{\text{final}}^{(\text{nd})} \le \big(16 p_{\text{int}}^{(\text{nd})}\big)^{r/2}$, and substituting (2.8) gives $p_{\text{final}}^{(\text{nd})} \le 2^{-tr}$.

Similarly to the defective nodes handled above, the case that $l > \log_2 n - r$ follows essentially unchanged; while the above analysis has an additional union bound over 2^r paths, the number of paths when $l > \log_2 n - r$ only gets smaller. Hence, the preceding bound on $p_{\text{final}}^{(\text{nd})} \le 2^{-tr}$ also holds in this case.

Combining the defective and non-defective cases: Taking the more stringent requirement on N in the above two cases, we set

$$N = \left\lceil \frac{2t \log 2 + \log 16}{2(1/2 - p - 1/C)^2} \right\rceil, \tag{2.10}$$

and we observe that regardless of the defectivity of a given node, the probability of the node's final label being wrong is at most 2^{-tr}.

Next, we upper bound the probability that any node among three groups—non-defective nodes at $l_{\min}$, defective nodes, and children nodes of all defective nodes—is identified wrongly. Note that if all such nodes are identified correctly, then the branching is only ever continued for defective nodes, and it follows that at most $2k$ nodes remain at the final level (analyzed below).

Since there are $\log_2(n/k)$ levels and k defectives, the number of non-defective children nodes of defective nodes is at most $k \log_2(n/k)$, and the number of non-defective nodes at $l_{\min}$ is at most k. Summing these up, we have at most $2k\log_2(n/k) + k$ nodes. By taking the union bound over all $2k\log_2(n/k) + k$ nodes, the probability of making an error in identifying any node in the mentioned three groups is at most $2^{-tr}(2k\log_2(n/k) + k)$. This can be upper bounded by a given target value β_n (approaching zero as $n \to \infty$), provided that

$$2^{-tr}\left(2k \log_2\left(\frac{n}{k}\right) + k\right) \le \beta_n, \tag{2.11}$$

which rearranges to give

$$r \ge \frac{1}{t}\log_2\left(\frac{2k}{\beta_n}\log_2\left(\frac{n}{k}\right) + \frac{k}{\beta_n}\right). \tag{2.12}$$

By choosing

$$r = \frac{1}{t}\log_2\left(\frac{3k}{\beta_n}\log_2\left(\frac{n}{k}\right)\right). \tag{2.13}$$

we deduce that the probability of any wrong decision is upper bounded by β_n.

2.3.2 Analysis of the Final Level

Recall from the analyses of (2.2) and (2.8) that given our choice of N in (2.10), regardless of the defectivity of a given node, the probability of a wrong intermediate label—let us call this p_{int}—is at most $2^{-2t}/4$. To get the final label of each node (singleton), we conduct a majority voting of $C' \log_2 n$ intermediate labels. Hence, a given node is labeled wrongly when it has at least $(C' \log_2 n)/2$ wrong intermediate labels. This gives the following upper bound on the probability of a wrong final label, denoted by p_{final}:

$$p_{\text{final}} \leq \binom{C' \log_2 n}{(C' \log_2 n)/2} \left(p_{\text{int}}\right)^{(C' \log_2 n)/2} \leq \left(4p_{\text{int}}\right)^{(C' \log_2 n)/2} \overset{(a)}{\leq} 2^{-tC' \log_2 n}, \tag{2.14}$$

where (a) uses $p_{\text{int}} \leq 2^{-2t}/4$. Taking the union bound over all n nodes at the final level, we get

$$n\left(2^{-tC' \log_2 n}\right) = n\left(n^{-tC'}\right) = O(n^{1-tC'}), \tag{2.15}$$

which approaches zero as $n \to \infty$ as long as $tC' > 1$. Note that while we show that all n nodes (singletons) at the final level will be correctly identified, only at most $2k$ will be used by the algorithm. This is because we have shown earlier in (2.11) that with probability $1 - \beta_n$, all non-defective nodes at $l_{\min}$, all nodes in the defective tree, and all children nodes of all defective nodes, are correctly identified, which implies that $|\mathcal{PD}^{(\log_2 n)}| \leq 2k$.

2.3.3 Number of Tests, Error Probability, and Decoding Time

For convenience, we restate all the values that we have assigned in our analysis above:

$$C = \left\lceil \frac{2}{1-2p} \right\rceil + 1 = O(1) \tag{2.16}$$

$$N = \left\lceil \frac{2t \log 2 + \log 16}{2(1/2 - p - 1/C)^2} \right\rceil = O(t) \tag{2.17}$$

$$r = \frac{1}{t} \log_2 \left(\frac{3k}{\beta_n} \log_2 \left(\frac{n}{k} \right) \right) = O\left(\frac{1}{t} \log \left(\frac{k \log(n/k)}{\beta_n} \right) \right), \tag{2.18}$$

where $p \in (0, 1/2)$ is the noise level. Now, we choose $t = O(1)$ and $\beta_n = \left(k \log_2(n/k)\right)^{1-\epsilon t}$, for some constant $\epsilon \in (1/t, 1)$. Choosing $\beta_n = \left(k \log_2(n/k)\right)^{1-\epsilon t}$ in (2.18) gives

$$r = \frac{1}{t} \log_2 \left(\frac{3k \log_2(n/k)}{\left(k \log_2(n/k)\right)^{1-\epsilon t}} \right) = \frac{1}{t} \log_2 \left(3\left(k \log_2 \left(\frac{n}{k}\right)\right)^{\epsilon t} \right). \tag{2.19}$$

Recall that we require $C' \log_2 n \geq r$, or equivalently $C' \geq r/\log_2 n$. Substituting (2.19) into $C' \geq r/\log_2 n$, we find that we require

$$C' \geq \frac{\frac{1}{t}\log_2\left(3\left(k\log_2\left(\frac{n}{k}\right)\right)^{\epsilon t}\right)}{\log_2 n}, \tag{2.20}$$

Since ϵ is constant, we choose a $C' = O(1)$ that is large enough to satisfy (2.20). With our choices of $C, C', N, t = O(1)$ and $\beta_n = \Theta\big((k\log n)^{1-\epsilon t}\big)$, we obtain the following:

- **Number of tests:** We used CNk tests per level for $l = \log_2 k, \ldots, \log_2 n - 1$. At the final level $l = \log_2 n$, we used $CC'Nk\log_2 n$ tests. Summing these together gives

$$T \leq CNk\log_2\left(\frac{n}{k}\right) + CC'Nk\log_2 n \overset{(a)}{=} O(k\log n), \tag{2.21}$$

  where (a) follows by substituting $C, C', N = O(1)$ and simplifying.

- **Error probability:** Combining the error probabilities from all levels, we have a total error probability of at most

$$\beta_n + O\big(n^{1-tC'}\big) = O\left(\left(k\log\left(\frac{n}{k}\right)\right)^{1-\epsilon t}\right), \tag{2.22}$$

  by substituting $\beta_n = \big(k\log_2(n/k)\big)^{1-\epsilon t}$ and choosing C' sufficiently large.

- **Decoding time:** The decoding time depends on the number of test outcome checks made. For $l = \log_2 k, \ldots, \log_2 n - 1$, w.h.p.,[1] we involved $O\big(k\log\frac{n}{k}\big)$ nodes in total. For each node involved, we checked at most $\sum_{i=1}^{r} 2^i = O(2^r) \overset{(2.18)}{=} O\big(\big(\frac{k\log(n/k)}{\beta_n}\big)^{1/t}\big)$ intermediate labels of other nodes to decide the final label of the given node. For each these nodes being checked, we checked $N = O(t)$ test outcomes to determine the intermediate label. Therefore, the decoding time for these levels is

$$O\left(k\log\left(\frac{n}{k}\right)\cdot\left(\frac{k\log(n/k)}{\beta_n}\right)^{1/t}\cdot t\right), \tag{2.23}$$

  At the final level $l = \log_2 n$, we have already shown that w.h.p., at most $2k$ nodes remain possibly defective. For each such node, we checked $C'\log_2 n$ intermediate labels to decide

[1]Here and subsequently, we write *with high probability* (w.h.p.) to mean holding under the high-probability events used in proving that the algorithm succeeds.

the final label of the given node. To decide each intermediate label, we checked $N = O(t)$ test outcomes. Therefore, the decoding time is $O(2k \cdot C' \log n \cdot t)$. Summing this with (2.23) gives us the total decoding time of

$$O\left(k \log\left(\frac{n}{k}\right) \cdot \left(\frac{k \log(n/k)}{\beta_n}\right)^{1/t} \cdot t\right) + O(2k \cdot C' \log n \cdot t) = O\big(\big(k \log \frac{n}{k}\big)^{1+\epsilon}\big), \tag{2.24}$$

by substituting $C', t = O(1)$ and $\beta_n = \big(k \log_2(n/k)\big)^{1-\epsilon t}$, and noting that the $O(k \log n)$ term is dominated by $O\big(\big(k \log \frac{n}{k}\big)^{1+\epsilon}\big)$ regardless of the scaling of k.

Algorithm 2 Decoding procedure for the noisy setting

```
Require: Outcomes of $T$ non-adaptive tests, number of items $n$, number of defective items $k$,
    and parameters $N$, $C$, $C'$, and $r$
 1: Initialize $\mathcal{PD}^{(l_{\min})} = \{\mathcal{G}_j^{(l_{\min})}\}_{j=1}^{k}$, where $l_{\min} = \log_2 k$.
 2: for $l = \log_2 k, \dots, \log_2 n - 1$ do
 3:   if $l \le \log_2 n - r$ (i.e., there are at least $r$ levels below the node) then
 4:     for each group $\mathcal{G} \in \mathcal{PD}^{(l)}$ do
 5:       Evaluate the intermediate labels of all nodes $r$ levels below $\mathcal{G}$.
 6:   else if $l > \log_2 n - r$ (i.e., there are fewer than $r$ levels below the node) then
 7:     for each group $\mathcal{G} \in \mathcal{PD}^{(l)}$ do
 8:       Evaluate the intermediate labels of all nodes all levels below $\mathcal{G}$ except the final level.
 9:       for each node reached at the final level do
10:         Iterate through the $C'N\log_2 n$ test outcomes in batches of size $N$: Conduct a
            majority vote for each batch to obtain an intermediate label for the node.
11:       Use intermediate labels from each node in the final level to make up paths of length
          $r$ (see Figure 2.2).
12:   If $\exists$ a path with more than $r/2$ positive intermediate labels, then assign $\mathcal{G}$'s final label to
      be positive. Otherwise, assign $\mathcal{G}$'s final label to be negative.
13:   If the final label of $\mathcal{G}$ is positive, then add both children of $\mathcal{G}$ to $\mathcal{PD}^{(l+1)}$.
14: At the final level, for each node (singleton), repeat step 10 to obtain $C'\log_2 n$ intermediate
    labels for the node, and conduct a majority vote for the node's intermediate labels to obtain
    its final label.
15: Return $\widehat{\mathcal{S}}$ containing the elements of singletons in $\mathcal{PD}^{(\log_2 n)}$ with a positive final label.
```

Chapter 3

Algorithm for Finitely Divisible Items

Under finitely-divisible items, we again build on the non-adaptive binary splitting approach of [7, 8], with the following main differences (see Figure 3.1):

- We use a shorter tree of height $\gamma' \leq \gamma$. This is because a given item is placed in a single test at each level, so the assumption $\gamma = o(\log n)$ prohibits us from having $O(\log n)$ levels. We consider $\gamma' \leq \gamma$ so that the remaining budget can be used at the final layer, and we later optimize γ' to minimize the number of tests.

- In view of the shorter height, we use *non-binary* splitting; this was considered under adaptive testing in [28, 29], and our algorithm can be viewed as a non-adaptive counterpart, in the same way that [7, 8] can be viewed as a non-adaptive counterpart of Hwang's binary splitting algorithm [30].

- In contrast to the unconstrained setting, we cannot readily use the idea of using N sequences of tests at each level while only increasing the number of tests by a factor of $N = O(1)$. Here, such an approach turns out to be highly wasteful in terms of its use of the limited γ budget, and we avoid it altogether.

- At the top level of the tree (excluding the root), we use individual testing (i.e., each node has its own test). This guarantees that no non-defective node from the second level can "continue" down the tree, which simplifies our analysis.

3.1 Description of the Algorithm

The levels of the tree, summarized in Figure 3.1, are indexed by $l = 0, 1, \dots, \gamma'$. Since testing at the root is not informative (we will always get a positive outcome), we start our testing procedure at $l = 1$ (the second level of nodes in Figure 3.1). We choose[1] $M = (n/k)^{\frac{\gamma'-1}{\gamma'}}$, $T_{\text{len}} = Ck(n/k)^{1/\gamma'}$ and $T'_{\text{len}} = \gamma' k(n/k)^{1/\gamma'}$, where C is a constant. Here the choice of M is taken to match the near-optimal adaptive splitting algorithm of [28], and the choices of T_{len} and T'_{len} are motivated by the goal of having a number of tests matching the COMP algorithm (see Table 1.2). Under these preceding choices, the total number of tests (excluding the last level) is given by

$$\underbrace{\frac{n}{M}}_{l=1} + \underbrace{\gamma' \cdot Ck\left(\frac{n}{k}\right)^{\frac{1}{\gamma'}}}_{l=2,\dots,\gamma'-2} + \underbrace{\gamma' k\left(\frac{n}{k}\right)^{\frac{1}{\gamma'}}}_{l=\gamma'-1} = O\left(\gamma' k\left(\frac{n}{k}\right)^{\frac{1}{\gamma'}}\right). \tag{3.1}$$

The overall testing procedure is described in Algorithm 3, and the decoding procedure is described in Algorithm 4. The j-th node at the l-th level is again written as $\mathcal{G}_j^{(l)}$.

Here and subsequently, we assume that $\gamma \geq 3$. We note that the case $\gamma = 1$ is trivial, and while $\gamma = 2$ could be handled by omitting the step at level $l = \gamma'$ containing T''_{len} tests, this variant is omitted for the sake of brevity.

3.2 Algorithmic Guarantees

Theorem 3.2.1. (Algorithmic guarantees) *Let $\mathcal{S}$ be a fixed (defective) subset of $\{1, \dots, n\}$ of cardinality k, let $\gamma = o(\log n)$ (with $\gamma \geq 3$) be the maximum number of times each item can be tested, and fix $\gamma' \in \{3, \dots, \gamma\}$ and any function β_n decaying as n increases. There exist choices*[2]

[1]Here and subsequently, we assume for notation convenience that $(n/k)^{1/\gamma}$ and $(n/k)^{1/\gamma'}$ are integers. Since we focus on scaling laws, the resulting effect of rounding has no impact on our results.

[2]Specifically, we will set $T_{\text{len}} = O\big(k(n/k)^{1/\gamma'}\big)$, $T'_{\text{len}} = \gamma' k(n/k)^{1/\gamma'}$, and $T''_{\text{len}} = k(k/\beta_n)^{\frac{1}{\gamma-\gamma'+1}}(n/k)^{\frac{1}{\gamma'(\gamma-\gamma'+1)}}$.

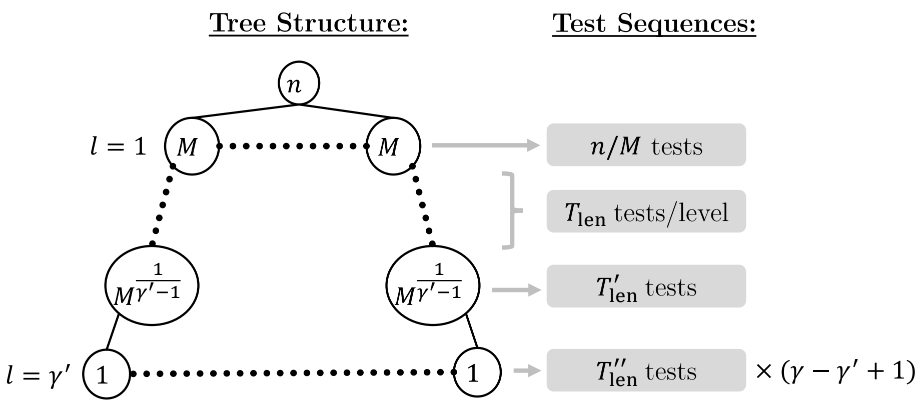

Figure 3.1: Tree structure of our algorithm. From the second level onwards, the branching factor is $M^{\frac{1}{\gamma'-1}}$.

of T_{len}, T'_{len}, and T''_{len} such that with

$$T = O\biggl(\gamma k \max\left\{\left(\frac{n}{k}\right)^{\frac{1}{\gamma'}}, \left(\frac{k}{\beta_n}\right)^{\frac{1}{\gamma-\gamma'+1}}\left(\frac{n}{k}\right)^{\frac{1}{\gamma'(\gamma-\gamma'+1)}}\right\}\biggr), \tag{3.2}$$

the preceding algorithm satisfies the following with probability at least $1 - O(\beta_n) - e^{-\Omega(k)}$:

- *The returned estimate $\widehat{\mathcal{S}}$ equals $\mathcal{S}$;*
- *The decoding time is*[3] $O\bigl(\gamma k (n/k)^{1/\gamma'}\bigr)$.

In order to better understand this bound on T, we consider $k = \Theta\bigl(n^\theta\bigr)$ for some $\theta \in [0, 1)$, and $\beta_n = \frac{1}{\text{poly}(\log n)}$. We use $\widetilde{O}(\cdot)$ notation to hide poly-logarithmic factors, and accordingly omit β_n and focus on the remaining terms. Substituting $k = \Theta\bigl(n^\theta\bigr)$ into (5.47), we obtain

$$T = \widetilde{O}\Bigl(\gamma k \max\Bigl\{n^{\frac{1-\theta}{\gamma'}}, n^{\frac{\theta}{\gamma-\gamma'+1}+\frac{1-\theta}{\gamma'(\gamma-\gamma'+1)}}\Bigr\}\Bigr). \tag{3.3}$$

Momentarily ignoring the integer constraint on γ', we obtain the optimal γ' by solving $\frac{1-\theta}{\gamma'} = \frac{\theta}{\gamma-\gamma'+1} + \frac{1-\theta}{\gamma'(\gamma-\gamma'+1)}$, which simplifies to $\gamma' = (1-\theta)\gamma$. Substituting $\gamma' = (1-\theta)\gamma$ back into (5.47) gives $T = \widetilde{O}\bigl(\gamma k n^{1/\gamma}\bigr)$. In addition, by the same substitution, we obtain $O\bigl(\gamma k n^{1/\gamma}\bigr)$ decoding time. In this case, the bound on T is the same as the bound for the COMP algorithm (see Table 1.2).

[3] In certain scaling regimes, this decoding time may be lower than the number of tests. This is because the algorithm sequentially decides which tests outcomes to observe, and does not necessarily end up observing every outcome.

Algorithm 3 Testing procedure for γ-divisible items

```
Require: Number of items $n$, number of defective items $k$, divisibility of each item $\gamma$, and
    parameters $\gamma'$, $M$, $T_{\text{len}}$, $T'_{\text{len}}$, and $T''_{\text{len}}$
 1: At $l = 1$, test each node separately in a single test (no randomization).
 2: for each $l = 2, 3, \dots, \gamma' - 1$ do
 3:   if $l = \gamma' - 1$ then form a sequence of tests of length $T'_{\text{len}}$.
 4:   else form a sequence of tests of length $T_{\text{len}}$.
 5:   for $j = 1, 2, \dots, \frac{n}{M}(M)^{(l-1)/(\gamma'-1)}$ do
 6:     Place all items from each group $\mathcal{G}_j^{(l)}$ into a single test, chosen uniformly at random.
 7: For $l = \gamma'$, form $\gamma - \gamma' + 1$ sequences of tests, each of length $T''_{\text{len}}$.
 8: for each singleton do
 9:   for each of the $\gamma - \gamma' + 1$ sequences of tests do
10:     Place the item in one of the corresponding tests, chosen uniformly at random.
```

In the case that $\gamma = \omega(1)$, it is straightforward to establish that the integer constraint on γ' does not impact the above findings. However, for $\gamma = O(1)$, we need to carefully account for the integer constraint. We start by noting that the function $\max\left\{\frac{1-\theta}{\gamma'}, \frac{\theta}{\gamma-\gamma'+1} + \frac{1-\theta}{\gamma'(\gamma-\gamma'+1)}\right\}$ is convex on $[3, \gamma]$; this is easily proved by computing the second derivative of each term in $\max\{.,.\}$. Since a convex function is monotone on either side of its minimum (in this case $(1-\theta)\gamma$), it follows that

$$\gamma' = \underset{\gamma' \in \{3,\dots,\gamma\}}{\operatorname{argmin}} \left(\gamma k \max\left\{n^{\frac{1-\theta}{\gamma'}}, n^{\frac{\theta}{\gamma-\gamma'+1} + \frac{1-\theta}{\gamma'(\gamma-\gamma'+1)}}\right\}\right) \tag{3.4}$$

$$= \begin{cases} 3 & \text{if } (1-\theta)\gamma < 3 \\ \underset{\gamma' \in \{\lfloor (1-\theta)\gamma \rfloor, \lceil (1-\theta)\gamma \rceil\}}{\operatorname{argmin}} \left(\max\left\{\frac{1-\theta}{\gamma'}, \frac{\theta}{\gamma-\gamma'+1} + \frac{1-\theta}{\gamma'(\gamma-\gamma'+1)}\right\}\right) & \text{otherwise.} \end{cases} \tag{3.5}$$

To see how the splitting algorithm compares to optimal behavior established in [29, 1] (i.e., an upper bound for the DD algorithm, and a matching algorithm-independent lower bound) and the COMP algorithm [2] for different values of γ, we introduce the following asymptotic

Algorithm 4 Decoding procedure for γ-divisible items

```
Require: Outcomes of $T$ non-adaptive tests, number of items $n$, number of defective items $k$,
    divisibility of each item $\gamma$, and parameters $\gamma'$, $M$, $T_{\text{len}}$, $T'_{\text{len}}$, and $T''_{\text{len}}$
1: Initialize $\mathcal{PD}^{(l_{\min})} = \{\mathcal{G}_j^{(l_{\min})}\}_{j=1}^{n/M}$, where $l_{\min} = 1$.
2: Place all nodes at $l = 1$ with a positive test outcome into $\mathcal{PD}^{(l_{\min})}$.
3: for $l = 2, 3, \dots, \gamma' - 1$ do
4:     for each group $\mathcal{G} \in \mathcal{PD}^{(l)}$ do
5:         Check whether the single test corresponding to $\mathcal{G}$ is positive or negative.
6:         if the test is positive then add all $M^{1/(\gamma'-1)}$ children of $\mathcal{G}$ to $\mathcal{PD}^{(l+1)}$
7: Let the estimate $\widehat{\mathcal{S}}$ of the defective set be the elements in $\mathcal{PD}^{(\gamma')}$ that are not included in
    any of the negative tests from the remaining $(\gamma - \gamma' + 1)T''_{\text{len}}$ tests.
8: Return $\widehat{\mathcal{S}}$.
```

quantity:

$$\eta = \lim_{n\to\infty} \frac{\log\left(\frac{n}{k}\right)}{\gamma \log\left(\frac{T}{\gamma k}\right)}. \tag{3.6}$$

Observe that for any fixed value of $\eta > 0$, re-arranging gives $T \sim \gamma k\big(\big(\frac{n}{k}\big)^{1/\gamma}\big)^{(1+o(1))/\eta}$. With η defined, we compare the performance in Figure 3.2. We observe that the splitting algorithm's curve quickly gets closer to the COMP algorithm's curve even for fairly low values of γ. On the other hand, matching the DD algorithm's curve with sublinear decoding time remains an interesting open challenge for future work.

3.3 Analysis

Throughout the analysis, the defective set $\mathcal{S}$ is fixed but otherwise arbitrary, and we condition on fixed placements of the defective items into tests (and hence, fixed test outcomes and a fixed defective tree). The test placements of the non-defective items are independent of those of the defective items, and our analysis will hold regardless of which particular tests the defectives were placed in. The defective test placements are written as $\mathcal{T}_{\mathcal{S}}$, and we write $\mathbb{P}[\cdot \mid \mathcal{T}_{\mathcal{S}}]$ to denote the conditioning.

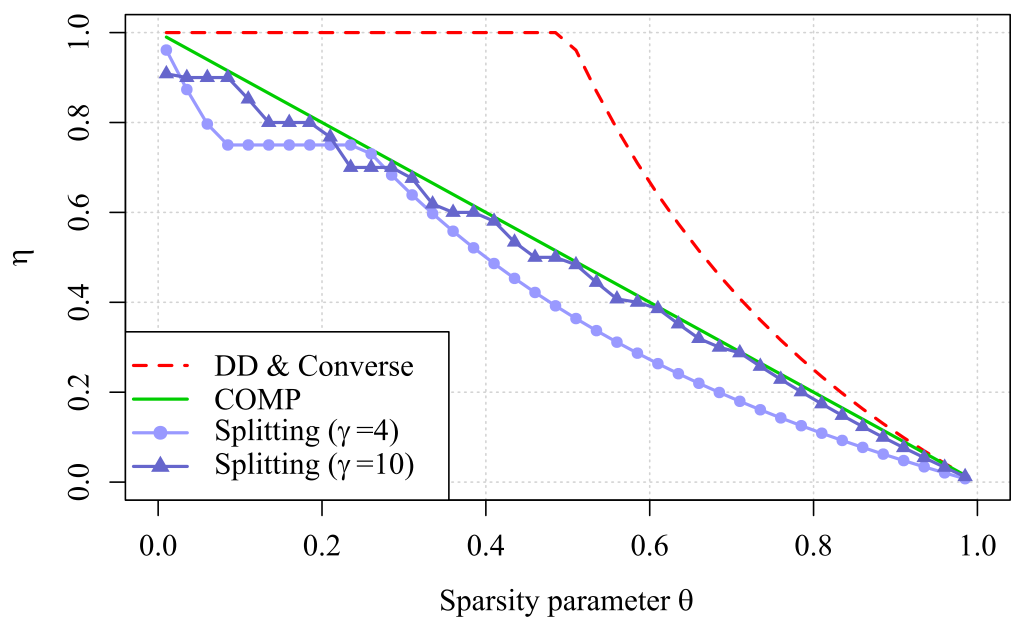

Figure 3.2: Plot of the asymptotic quantity η (see (3.6)) against the sparsity parameter θ for the converse (i.e., the lower bound) [1], the DD algorithm [1], the COMP algorithm [2], and our splitting algorithm (with $\gamma = 4$ and $\gamma = 10$).

We proceed with three lemmas that follow analogous steps to [8]. At level $l = 1$, the probability of a non-defective node being placed in a positive test is zero, because each node is placed in its own individual test. As for levels $l \in \{2, \dots, \gamma' - 2\}$, we proceed with the following simple lemma.

Lemma 3.3.1. (Probabilities of Non-Defectives Being in Positive Tests) *Under the above test design, the following holds at any given level* $l = 2, \dots, \gamma' - 2$*: Conditioned on any defective test placements* $\mathcal{T}_{\mathcal{S}}$*, any given non-defective node at level* l *has probability at most* $(1/C)(n/k)^{-1/\gamma'}$ *of being placed in a positive test.*

Proof. Since there are k defective items, at most k nodes at a given level can be defective. Hence, since each node is placed in a single test, at most k tests out of the $Ck(n/k)^{1/\gamma'}$ tests at the given level can be positive. Since the test placements are independent and uniform, it follows that for any non-defective node, the probability of being in a positive test is at most $k/T_{\mathrm{len}} = k/\big(Ck(n/k)^{1/\gamma'}\big) = (1/C)(n/k)^{-1/\gamma'}$. □

In view of this lemma, when starting at any non-defective child of any defective node, we can view any further branches down the non-defective sub-tree as "continuing" (i.e., the $M^{1/(\gamma'-1)}$ children are marked as possibility defective) with probability at most $(1/C)(n/k)^{-1/\gamma'}$, in particular implying the following.

Lemma 3.3.2. (Probability of Reaching a Non-Defective Node) *Under the setup of Lemma 3.3.1, any given non-defective node at distance Δ from the defective tree is reached (i.e., all of its ancestors are placed in positive tests, so the node is considered possibly defective) with probability at most $(1/C)^{\Delta-1}(n/k)^{(1-\Delta)/\gamma'}$.*

We will use the preceding lemmas to control the quantity N_{total}, defined to be the total number of non-defective nodes that are *reached*—in the sense of Lemma 3.3.2—among levels $l \in \{2, \dots, \gamma' - 1\}$. It will be useful to upper bound N_{total} for the purpose of controlling the overall decoding time and the number of items considered at the final level.

3.3.1 Bounding N_{total}

We first present a lemma bounding the average of N_{total}.

Lemma 3.3.3. (Bounding N_{total} on Average) *For any parameters $C > 1$ and $\gamma' > 1$, and any defective test placements $\mathcal{T}_{\mathcal{S}}$, under the choice $M = (n/k)^{\frac{\gamma'-1}{\gamma'}}$, we have*

$$\mathbb{E}[N_{\text{total}}|\mathcal{T}_{\mathcal{S}}] = O\biggl(\gamma' k \Bigl(\frac{n}{k}\Bigr)^{\frac{1}{\gamma'}}\biggr). \tag{3.7}$$

Proof. At level $l = 1$, we use n/M tests for individual nodes. This results in correct identification of the non-defective nodes, guaranteeing that they will not "continue" to branch. Hence, at level $l = 1$, we trivially upper bound the number of non-defective nodes by n/M.

For the remaining levels $l = 2, \dots, \gamma' - 1$, all splits are $\bigl(M^{1/(\gamma'-1)}\bigr)$-ary, and each defective node can have at most $M^{\Delta/(\gamma'-1)}$ descendants at distance Δ. Since there are at most $\gamma' k$ defective nodes in total among levels $l = 1, \dots, \gamma' - 1$, it follows that there are at most $\gamma' k M^{\frac{\Delta}{\gamma'-1}}$ non-defective nodes at distance Δ from defective nodes starting at those levels. Furthermore, we established in Lemma 3.3.2 that a distance of Δ gives a probability of at most

$\left(\frac{1}{C}\right)^{\Delta-1}\left(\frac{n}{k}\right)^{(1-\Delta)/\gamma'}$ of being reached. This gives

$$\mathbb{E}[N_{\text{total}}|\mathcal{T}_{\mathcal{S}}] \le \sum_{\Delta=1}^{\gamma'} \gamma' k M^{\frac{\Delta}{\gamma'-1}} \left(\frac{1}{C}\right)^{\Delta-1} \left(\frac{n}{k}\right)^{\frac{1-\Delta}{\gamma'}} + \frac{n}{M} \tag{3.8}$$

$$= \gamma' k M^{\frac{1}{\gamma'-1}} \sum_{\Delta=1}^{\gamma'} M^{\frac{\Delta-1}{\gamma'-1}} \left(\frac{1}{C}\right)^{\Delta-1} \left(\frac{n}{k}\right)^{\frac{1-\Delta}{\gamma'}} + \frac{n}{M} \tag{3.9}$$

$$\overset{(a)}{\le} \gamma' k M^{\frac{1}{\gamma'-1}} \frac{1}{1 - M^{\frac{1}{\gamma'-1}}\left(\frac{1}{C}\right)\left(\frac{n}{k}\right)^{-1/\gamma'}} + \frac{n}{M} \tag{3.10}$$

$$\overset{(b)}{=} \gamma' k \left(\frac{n}{k}\right)^{\frac{1}{\gamma'}} \frac{1}{1-1/C} + k\left(\frac{n}{k}\right)^{\frac{1}{\gamma'}}, \tag{3.11}$$

where (a) applies the geometric series formula (increasing the upper limit of the sum from γ' to ∞), and (b) follows by substituting $M = (n/k)^{\frac{\gamma'-1}{\gamma'}}$. □

We now wish to move from a characterization of the average to a high-probability characterization. At this point, we depart somewhat further from the analysis of [8], which is based on branching process theory, and appears to yield suboptimal results in the case that the tree's branching factor scales as $\omega(1)$.

We introduce the following definition, in which we refer to a *full* m-ary tree as a tree where every *internal* node has exactly m children.

Lemma 3.3.4. [31, Prop. 3.1] (Fuss-Catalan Numbers) *For natural integers $m, n \ge 2$, the order-m Fuss-Catalan number*

$$\text{Cat}_m^n = \frac{1}{(m-1)n+1}\binom{mn}{n} \le \binom{mn}{n} \le (em)^n, \tag{3.12}$$

is the number of full m-ary trees with exactly n internal nodes.

We note that the Catalan numbers also played an important role in the analysis of [7], but were used in a rather different manner that we were unable to extend to obtain a result comparable to Theorem 3.2.1. In the proof of the following lemma, these are used in a counting argument in order to establish the sub-exponential behavior of the random variable N_{total}.

Lemma 3.3.5. (High Probability Bound on N_{total}) *For any parameters $C \ge e^2$ and $\gamma' > 1$, and any defective test placements $\mathcal{T}_{\mathcal{S}}$, under the choice $M = (n/k)^{\frac{\gamma'-1}{\gamma'}}$, we have $N_{\text{total}} = O\big(\gamma' k (n/k)^{1/\gamma'}\big)$ with probability $1 - e^{-\Omega(\gamma' k)}$.*

Proof. Consider a single non-defective sub-tree following a defective node, and let N_b be the number of nodes in the sub-tree such that itself and all its ancestors only appear in positive tests (i.e., the number of nodes that lead to further branching). We have

$$\begin{aligned}
\mathbb{P}[N_b = n_b] &\leq \mathbb{P}[\exists \text{ a full } M^{1/(\gamma'-1)}\text{-tree reached with } n_b \text{ internal nodes}] && (3.13)\\
&\overset{(a)}{\leq} (\#\text{full } M^{1/(\gamma'-1)}\text{-trees with } n_b \text{ internal nodes}) \cdot \left(\frac{1}{C}\left(\frac{n}{k}\right)^{-\frac{1}{\gamma'}}\right)^{n_b} && (3.14)\\
&\overset{(b)}{\leq} \left(eM^{1/(\gamma'-1)}\right)^{n_b}\left(\frac{1}{C}\left(\frac{n}{k}\right)^{-\frac{1}{\gamma'}}\right)^{n_b} && (3.15)\\
&\overset{(c)}{=} \left(\frac{e}{C}\right)^{n_b} \overset{(d)}{\leq} e^{-n_b}, && (3.16)
\end{aligned}$$

where (a) applies Lemma 3.3.1 and the union bound, (b) applies Lemma 3.3.4, (c) is obtained by substituting $M = (n/k)^{\frac{\gamma'-1}{\gamma'}}$ and simplifying, and (d) holds since $C \geq e^2$. This implies that N_b is a sub-exponential random variable. Since we have at most $(\gamma'-1)k$ defective nodes in levels $l = 1, \dots, \gamma'-1$, we are adding together $O(\gamma' k)$ independent copies of such random variables (each corresponding to a different non-defective sub-tree following a defective node).[4] Letting $N_b^{(i)}$ denote the i-th copy, we can apply a standard concentration bound for sums of independent sub-exponential random variables [32, Prop. 5.16] to obtain

$$\mathbb{P}\big[N_b^{(1)} + \cdots + N_b^{(O(\gamma' k))} \geq \mathbb{E}[N_b^{(1)} + \cdots + N_b^{(O(\gamma' k))}] + t | \mathcal{T}_{\mathcal{S}}] \leq \exp\left(\Omega\Big(\min\Big\{\frac{t^2}{\gamma' k}, t\Big\}\Big)\right). \tag{3.17}$$

Setting $t = \Theta(\gamma' k)$, we get

$$\mathbb{P}[N_b^{(1)} + \cdots + N_b^{(O(\gamma' k))} \geq \mathbb{E}[N_b^{(1)} + \cdots + N_b^{(O(\gamma' k))}] + \Theta(\gamma' k) | \mathcal{T}_{\mathcal{S}}] \leq e^{-\Omega(\gamma' k)}. \tag{3.18}$$

Recall that each $N_b^{(i)}$ only counts "internal" nodes, whereas N_{total} also counts leaves, so passing from the former to the latter requires multiplying by the branching factor $M^{1/(\gamma'-1)} = (n/k)^{1/\gamma'}$. Multiplying on both sides inside the probability in (3.18) accordingly, we obtain

$$\mathbb{P}\left[N_{\text{total}} \geq \mathbb{E}[N_{\text{total}}] + \Theta\left(\gamma' k \left(\frac{n}{k}\right)^{1/\gamma'}\right) \Big| \mathcal{T}_{\mathcal{S}}\right] \leq e^{-\Omega(\gamma' k)}. \tag{3.19}$$

Substituting $\mathbb{E}[N_{\text{total}}] = O\big(\gamma' k (n/k)^{1/\gamma'}\big)$ (see Lemma 3.3.3) into (3.19), we obtain the desired result. □

[4] We do not consider the non-defective nodes at level $l = 1$, because they are guaranteed to be identified correctly as a result of individual testing of nodes.

We now briefly consider level $l = \gamma' - 1$, which uses $T'_{\text{len}} = \gamma' k (n/k)^{1/\gamma'}$ tests (see Figure 3.1). Since $|\mathcal{PD}^{(\gamma'-1)}| \le N_{\text{total}} + k$ holds trivially, Lemma 3.3.5 implies that $|\mathcal{PD}^{(\gamma'-1)}| = O\big(\gamma' k (n/k)^{1/\gamma'}\big)$ with probability $1 - e^{-\Omega(\gamma' k)}$. Using the same argument as Lemma 3.3.1, the probability of a non-defective node being in a positive test at level $l = \gamma' - 1$ is at most $k/T'_{\text{len}} = (1/\gamma')(n/k)^{-1/\gamma'}$. Hence, conditioned on $|\mathcal{PD}^{(\gamma'-1)}| = O\big(\gamma' k (n/k)^{1/\gamma'}\big)$, the number of non-defective nodes placed in a positive test is stochastically dominated by

$$\text{Binomial}\left(O\left(\gamma' k \left(\frac{n}{k}\right)^{1/\gamma'}\right), \frac{1}{\gamma'}\left(\frac{n}{k}\right)^{-1/\gamma'} \right). \tag{3.20}$$

By a multiplicative form of Chernoff bound, the number of such non-defective nodes in $\mathcal{PD}^{(\gamma'-1)}$ is $O(k)$ with probability at least $1 - e^{-\Omega(k)}$. Since the branching factor is $(n/k)^{1/\gamma'}$, it follows that the number of non-defective nodes in $\mathcal{PD}^{(\gamma')}$ behaves as $O(k(n/k)^{1/\gamma'})$.

3.3.2 Analysis of the Final Level

Recall that at the final level, we perform $\gamma - \gamma' + 1$ independent sequences of tests of length T''_{len}, with each item being randomly placed in one of these T''_{len} tests. Conditioned on the high probability event that $|\mathcal{PD}^{(\gamma')}| = O(k(n/k)^{1/\gamma'})$, we study the required T''_{len} for a vanishing error probability. Specifically, we upper bound the error probability by $O(\beta_n)$ for some decaying function $\beta_n \to 0$ as $n \to \infty$.

For a given non-defective item and a given sequence of T''_{len} tests, the probability of colliding with any defective item is at most k/T''_{len} by the same argument as Lemma 3.3.1. Due to the $\gamma - \gamma' + 1$ independent repetitions, the probability of a given non-defective item appearing only in positive tests is at most $(k/T''_{\text{len}})^{\gamma-\gamma'+1}$. By a union bound over $O(k(n/k)^{1/\gamma'})$ non-defective items at the final level, we find that the estimate $\widehat{\mathcal{S}}$ differs from $\mathcal{S}$ with (conditional) probability $O\big(k(n/k)^{1/\gamma'}(k/T''_{\text{len}})^{\gamma-\gamma'+1}\big)$. The error probability is thus upper bounded by $O(\beta_n)$ provided that

$$k\left(\frac{n}{k}\right)^{\frac{1}{\gamma'}}\left(\frac{k}{T''_{\text{len}}}\right)^{\gamma-\gamma'+1} \le \beta_n \tag{3.21}$$

$$\iff T''_{\text{len}} \ge k\left(\frac{k}{\beta_n}\right)^{\frac{1}{\gamma-\gamma'+1}}\left(\frac{n}{k}\right)^{\frac{1}{\gamma'(\gamma-\gamma'+1)}}. \tag{3.22}$$

Hence, we set $T''_{\text{len}} = k(k/\beta_n)^{\frac{1}{\gamma-\gamma'+1}}(n/k)^{\frac{1}{\gamma'(\gamma-\gamma'+1)}}$.

3.3.3 Number of Tests, Error Probability, and Decoding Time

- **Number of tests:** For $l = 1, \dots, \gamma' - 1$, we used a total of $n/M + C(\gamma' - 3)k(n/k)^{1/\gamma'} + \gamma' k(n/k)^{1/\gamma'}$ tests, which scales as $O\big(\gamma' k(n/k)^{1/\gamma'}\big)$ by substituting $M = (n/k)^{\frac{\gamma'-1}{\gamma'}}$ and $C = O(1)$. For the final level, we used $(\gamma - \gamma' + 1)T''_{\rm len} = O\big(\gamma k(k/\beta_n)^{\frac{1}{\gamma-\gamma'+1}}(n/k)^{\frac{1}{\gamma'(\gamma-\gamma'+1)}}\big)$ tests, since $T''_{\rm len} = k(k/\beta_n)^{\frac{1}{\gamma-\gamma'+1}}(n/k)^{\frac{1}{\gamma'(\gamma-\gamma'+1)}}$. Combining these, we obtain
$$T = O\bigg(\gamma k \max\bigg\{\Big(\frac{n}{k}\Big)^{\frac{1}{\gamma'}}, \Big(\frac{k}{\beta_n}\Big)^{\frac{1}{\gamma-\gamma'+1}}\Big(\frac{n}{k}\Big)^{\frac{1}{\gamma'(\gamma-\gamma'+1)}}\bigg\}\bigg). \tag{3.23}$$

- **Error probability:** The concentration bound on $N_{\rm total}$ (see Lemma 3.3.5) holds with probability $1 - e^{-\Omega(\gamma' k)}$, and at level $l = \gamma' - 1$, we incur $e^{-\Omega(k)}$ error probability. Furthermore, the final stage incurs $O(\beta_n)$ error (conditional) probability. In total, we incur $\beta_n + e^{-\Omega(\gamma' k)} + e^{-\Omega(k)} = O(\beta_n) + e^{-\Omega(k)}$ error probability.

- **Decoding time:** We claim that conditioned on the high-probability events above (in particular, $N_{\rm total} = O\big(\gamma' k(n/k)^{1/\gamma'}\big)$), the decoding time is $O\big(\gamma k(n/k)^{1/\gamma'}\big)$. Since we consider the word-RAM model, it takes constant time to check whether each defective node or non-defective node is in a positive or negative test. First considering the levels $l = 2, \dots, \gamma' - 1$, we reached $N_{\rm total} = O\big(\gamma' k(n/k)^{1/\gamma'}\big)$ non-defective nodes and $O(\gamma' k)$ defective nodes, which leads to a total of $O\big(\gamma' k(n/k)^{1/\gamma'}\big)$ decoding time. At level $l = 1$, we iterate through $\frac{n}{M} = O\big(k(n/k)^{1/\gamma'}\big)$ nodes, and at the final level $l = \gamma'$, for each of the $O\big(k(n/k)^{1/\gamma'}\big)$ relevant leaf nodes, we perform $\gamma - \gamma' + 1 = O(\gamma)$ checks of tests for a total time of $O\big(\gamma k(n/k)^{1/\gamma'}\big)$. Combining these terms, we deduce the desired claim.

Chapter 4

Algorithm for Size-Constrained Tests

In the case of size-constrained tests, we again modify the tree structure (see Figure 4.1), and the main differences from the standard noiseless algorithm [7, 8] are as follows:

- The first level after the root is chosen to have groups of size ρ, since testing even a single larger-sized group is impossible. In addition, at this level with nodes of size ρ, we test each node individually, thereby guaranteeing that we only "continue" down the tree for defective nodes at that level.

- We use non-binary splitting, geometrically decreasing the node size at each level until the final level with size one. We limit the number of levels to be $O(1)$, whereas binary splitting would require $O(\log \rho)$ levels, and (at least when using a similar level-by-level test design) would increase the number of tests by an $O(\log \rho)$ factor.

- We do not independently place nodes into tests, since doing so would cause a positive probability of violating the ρ-sized test constraint. Instead, at each level, we create a random testing sub-matrix with a column weight of exactly one, and a row weight exactly equal to to $\frac{\rho}{\text{node size}}$. A similar doubly-constant test design was also adopted in [2], but without the tree structure.

- Similarly to the noisy setting in Section 2, we use N independent sequences of tests at each level, which helps to reduce the error probability.

. We now proceed with a more detailed description.

4.1 Description of the Algorithm

Our algorithm works with a tree structure (see Figure 4.1) similar to previous sections. The j-th node at the l-th level is again denoted by $\mathcal{G}_j^{(l)}$. A distinction here as that the tree only has a constant depth, with the final index denoted by $C = O(1)$; hence, the splits are $\rho^{1/C}$-ary.[1] More importantly, there are key differences in the allocation of items to tests, which we describe as follows.

At each level l, we perform N independent iterations to boost the error probability, as mentioned above. Within each iteration, we make use of a random matrix, which we write as $\mathsf{X}_l = \big[x_{ti}^{(l)}\big] \in \{0,1\}^{\#\text{tests}\times\#\text{nodes}}$ (the dependence on the iteration number is left implicit), where $\#\text{tests} = n/\rho$ and $\#\text{nodes} = \frac{n}{\rho^{1-l/C}}$. We pick X_l by sampling uniformly from all $\frac{n}{\rho} \times \frac{n}{\rho^{1-l/C}}$ matrices with exactly $\rho^{l/C}$ nodes per test (i.e., a row weight of $\rho^{l/C}$), and each node sampled exactly once (i.e., a column weight of one). These choices ensure that each test contains at most ρ items, as required. The column weight of one is not strictly imposed by the testing constraints, but helps in avoiding "bad" events where some nodes are not tested.

With this notation in place, the testing procedure is formally described in Algorithm 5, and the decoding procedure is described in Algorithm 6.

4.2 Algorithmic Guarantees

We are now ready to state our main result for the case of size-constrained tests. In this case, we slightly strengthen the assumption $k = o(n)$ to $k = n^{1-\Omega(1)}$, and we slightly strengthen the assumption $\rho = o\big(\frac{n}{k}\big)$ (see the discussion following (1.2)) to $\rho = (n/k)^{1-\Omega(1)}$. These additional

[1]For notational convenience, we assume that $\rho^{1/C}$ is an integer. Since we already assumed that ρ is a power of two, if $\rho = O(1)$, then it will suffice to let C be that power (see Lemma 4.3.2, in which we handle the case $\rho = O(1)$ separately). Otherwise, if $\rho = \omega(1)$, then the effect of rounding is insignificant since $C = O(1)$.

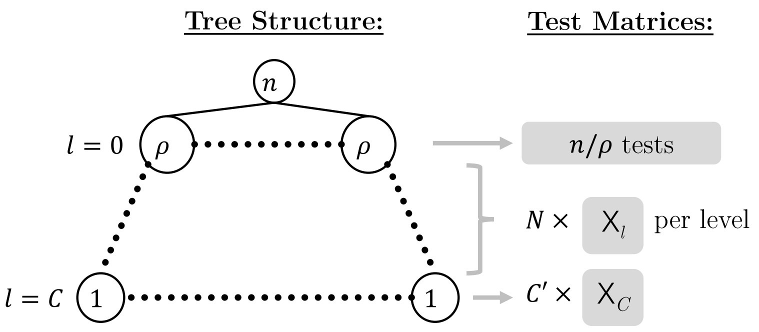

Figure 4.1: Tree structure of our algorithm. From the second level onwards, the branching factor is $\rho^{1/C}$.

restrictions only rule out scaling regimes that are very close to linear (e.g., $k = \frac{n}{\log n}$), and were similarly imposed in [2].

Theorem 4.2.1. (Algorithmic guarantees) *Let $\mathcal{S}$ be a (defective) subset of $\{1, \dots, n\}$ of cardinality $k = O\big(n^{1-\epsilon_1}\big)$ for some $\epsilon_1 \in (0, 1]$ and the test size constraint be $\rho = O\big((n/k)^{1-\epsilon_2}\big)$ for some $\epsilon_2 \in (0, 1]$. For any $\zeta > 0$, there exist choices of $C, C', N = O(1)$ such that with $O\big(n/\rho\big)$ tests, the preceding algorithm satisfies the following with probability at least $1 - O\big(n^{-\zeta}\big)$:*

- *The returned estimate $\widehat{\mathcal{S}}$ equals $\mathcal{S}$;*
- *The decoding time is $O\big(n/\rho\big)$;*

As summarized in Table 1.2, this is the first algorithm to attain $O\big(n/\rho\big)$ scaling in both the number of tests and the decoding time.

4.3 Analysis

We start at level $l = 0$ (see Figure 4.1), where we note that the probability of a non-defective node being placed in a positive test is zero because each node is placed in its own individual test. For subsequent levels, we proceed with the following lemma.

Lemma 4.3.1. (Probabilities of Non-Defectives Being in Positive Tests) *Under the above test design, for any given level $l = 1, \dots, C$ and any iteration indexed by $\{1, \dots, N\}$, each non-defective node has probability at most $k\rho/n$ of being placed in a positive test.*

Algorithm 5 Testing procedure for ρ-sized tests

```
Require: Number of items $n$, number of defective items $k$, and maximal test size $\rho$
 1: At level $l = 0$ (see Figure 4.1), perform an individual test for each node.
 2: for each $l = 1, \dots, C-1$ (for some constant $C$ chosen later) do
 3:     for each iteration in $\{1, \dots, N\}$ (for some constant $N \geq 1$) do
 4:         Pick a new $\mathsf{X}_l$ of size $\frac{n}{\rho} \times \frac{n}{\rho^{1-l/C}}$, with column weight 1 and row weight $\rho^{l/C}$.
 5:         for each row $t$ in $\mathsf{X}_l$ do
 6:             Conduct a single test for the mixture of nodes $\mathcal{G}_j^{(l)}$ with $x_{tj}^{(l)} = 1$.
 7: Set $l = C$ (final level)
 8: for each iteration in $\{1, \dots, C'\}$ (for some constant $C'$ chosen later) do
 9:     Pick a new $\mathsf{X}_C$ of size $\frac{n}{\rho} \times n$, with column weight 1 and row weight $\rho$.
10:     for each row $t$ in $\mathsf{X}_C$ do
11:         Conduct a single test for the mixture of (singleton) nodes $\mathcal{G}_j^{(l)}$ with $x_{tj}^{(l)} = 1$.
```

Proof. At any given level $l = 1, \dots, C$, the probability that a non-defective node u collides (i.e., in the same test) with a given defective node v is

$$\frac{\#\text{matrices with } u \,\&\, v \text{ in test } 1}{\#\text{matrices with } v \text{ in test } 1} \overset{(a)}{=} \frac{\binom{\frac{n}{\rho^{1-l/C}}-2}{\rho^{l/C}-2} \prod_{i=1}^{n/\rho-1} \binom{\frac{n}{\rho^{1-l/C}}-i\rho^{l/C}}{\rho^{l/C}}}{\binom{\frac{n}{\rho^{1-l/C}}-1}{\rho^{l/C}-1} \prod_{i=1}^{n/\rho-1} \binom{\frac{n}{\rho^{1-l/C}}-i\rho^{l/C}}{\rho^{l/C}}} \tag{4.1}$$

$$= \frac{\binom{\frac{n}{\rho^{1-l/C}}-2}{\rho^{l/C}-2}}{\binom{\frac{n}{\rho^{1-l/C}}-1}{\rho^{l/C}-1}} \overset{(b)}{=} \frac{\rho^{l/C}-1}{\frac{n}{\rho^{1-l/C}}-1} \tag{4.2}$$

$$= \frac{\rho}{n}\left(\frac{\rho^{l/C}-1}{\rho^{l/C}-\rho/n}\right) \overset{(c)}{\leq} \frac{\rho}{n}, \tag{4.3}$$

where:

- (a) follows by considering the rows of the matrix X_l (of size $\frac{n}{\rho} \times \frac{n}{\rho^{1-l/C}}$, column weight one, and row weight $\rho^{l/C}$) sequentially to count the number of possible matrices. For the numerator, we start with the first row, where u and v collide. The number of ways to fill this row (i.e., assigning items to this test) is the first term in the numerator. For the remaining $n/\rho - 1$ rows, in any particular order, the number of ways to fill those rows (while maintaining column weights of one) is represented by the second product term.

Algorithm 6 Decoding procedure for ρ-sized tests

```
Require: Outcomes of $T$ non-adaptive tests, number of items $n$, number of defective items $k$,
    and maximal test size $\rho$
1: Initialize $\mathcal{PD}^{(0)} = \{\mathcal{G}_j^{(0)}\}_{j=1}^{n/\rho}$
2: for each group $\mathcal{G} \in \mathcal{PD}^{(0)}$ do
3:     if the single test of $\mathcal{G}$ is positive then add all children of $\mathcal{G}$ to $\mathcal{PD}^{(1)}$
4: for $l = 1, \dots, C-1$ do
5:     for each group $\mathcal{G} \in \mathcal{PD}^{(l)}$ do
6:         if all $N$ tests of $\mathcal{G}$ are positive then add all children of $\mathcal{G}$ to $\mathcal{PD}^{(l+1)}$
7: Let the estimate $\widehat{\mathcal{S}}$ be the set of elements in $\mathcal{PD}^{(C)}$ that are not included in any negative
    test at the final level.
8: Return $\widehat{\mathcal{S}}$
```

The same analysis is then repeated for the denominator.

- (b) follows by expanding the binomial coefficient in terms of factorials, and then simplifying.
- (c) follows from the fact that $\rho/n \le 1$.

Since there are at most k defective nodes, by the union bound, we get the probability that a non-defective node collides with any defective node is at most $k\rho/n$. $\square$

The following technical lemma will also be used on several occasions.

Lemma 4.3.2. *For any k and ρ satisfying $k = O\big(n^{1-\epsilon_1}\big)$ for some $\epsilon_1 \in (0,1]$ and $\rho = O\big((n/k)^{1-\epsilon_2}\big)$ for some $\epsilon_2 \in (0,1]$, we have the following:*

- *For sufficiently large C, we have $\frac{k\rho^{1/C}}{n/\rho} = n^{-\Omega(1)}$;*
- *For any $\zeta_1 > 0$, we have for sufficiently large C and N that $\rho^{1/C}\big(\frac{k\rho}{n}\big)^N = O(n^{-\zeta_1})$.*

In addition, if $\rho = O(1)$, then the same holds true for any fixed $C > 0$, only requiring N to be sufficiently large in the second part.

Proof. For the first part, we write

$$\frac{n/\rho}{k\rho^{1/C}} = \frac{n/k}{\rho^{1+1/C}} \overset{(a)}{=} \Omega\bigg(\Big(\frac{n}{k}\Big)^{\epsilon_2 - \frac{1-\epsilon_2}{C}}\bigg) \overset{(b)}{=} \Omega\big(n^{\epsilon_1(\epsilon_2 - \frac{1-\epsilon_2}{C})}\big), \tag{4.4}$$

where (a) is by substituting $\rho = O\big((n/k)^{1-\epsilon_2}\big)$ and simplifying, and (b) is by substituting $k = O\big(n^{1-\epsilon_1}\big)$ and simplifying. Note that the power is positive for sufficiently large C.

For the second part, we write

$$\rho^{1/C}\Big(\frac{k\rho}{n}\Big)^N \overset{(a)}{=} O\bigg(\Big(\frac{n}{k}\Big)^{\frac{1-\epsilon_2}{C} - \epsilon_2 N}\bigg) \overset{(b)}{=} O\big(n^{\epsilon_1(\frac{1-\epsilon_2}{C} - \epsilon_2 N)}\big), \tag{4.5}$$

where (a) is by substituting $\rho = O\big((n/k)^{1-\epsilon_2}\big)$ and simplifying, and (b) is by substituting $k = O\big(n^{1-\epsilon_1}\big)$ and simplifying. Note that the power can be made arbitrarily negative by choosing N and C sufficiently large.

For the final part regarding $\rho = O(1)$, we simply note that the two claims reduce to (i) $\frac{k}{n} = n^{-\Omega(1)}$, and (ii) $\big(\frac{k}{n}\big)^N = O(n^{-\zeta_1})$ for sufficiently large N. Both of these are true since $k = O(n^{1-\epsilon_1})$. $\square$

We will show that throughout the course of the algorithm, for levels $l = 1, \dots, C$, the size of the possibly defective set $\mathcal{PD}^{(l)}$ remains at $O\big(k\rho^{1/C}\big)$ with high probability. We show this using an induction argument.

4.3.1 Analysis of Levels $l = 1, \dots, C-1$

For the base case $l = 1$, we start by looking at the preceding level $l = 0$. Each node at level $l = 0$ is allocated to an individual test, which implies that all nodes in $l = 0$ are identified correctly. Hence, only the children of the defective nodes in $l = 0$ are "explored" further in $l = 1$. Since the number of defective nodes in $l = 0$ is at most k and each node has $\rho^{1/C}$ children, we have $|\mathcal{PD}^{(1)}| \le k\rho^{1/C}$.

Consider a non-defective node indexed by i at a given level $l > 1$ having $k' \le k$ defective nodes, and let A_i be the indicator random variable of that non-defective node colliding with at least one defective node in all of its N repetitions. The dependence of these quantities on l is left implicit. We condition on all of the test placements performed at the earlier levels, writing $\mathbb{E}_l[\cdot]$ for the conditional expectation. By the inductive hypothesis, we have $|\mathcal{PD}^{(l)}| = O\big(k\rho^{1/C}\big)$.

Lemma 4.3.3. *Under the preceding setup and definitions, if* $|\mathcal{PD}^{(l)}| = O\big(k\rho^{1/C}\big)$*, then we have*

$$\mathbb{E}_l\Big[\sum_i A_i\Big] = O\bigg(k\rho^{1/C} \cdot \Big(\frac{k\rho}{n}\Big)^N\bigg). \tag{4.6}$$

Proof. From Lemma 4.3.1, we know that a given non-defective item i has a probability at most $k\rho/n$ of being placed in a positive test. Since we used N independent test design matrices X_l to assign i to N tests, we have $\mathbb{P}_l[A_i] \le (k\rho/n)^N$. Hence, we have

$$\mathbb{E}_l\Big[\sum_i A_i\Big] = \sum_i \mathbb{E}_l[A_i] = \sum_i \mathbb{P}_l[A_i = 1] \le \sum_i \Big(\frac{k\rho}{n}\Big)^N = O\bigg(k\rho^{1/C} \cdot \Big(\frac{k\rho}{n}\Big)^N\bigg), \tag{4.7}$$

where we used the linearity of expectation and the fact that $|\mathcal{PD}^{(l)}| = O\big(k\rho^{1/C}\big)$. □

Lemma 4.3.4. *For any constant* $\zeta_1 > 0$*, there exist choices of* C *and* N *such that the following holds: Conditioned on the* l*-th level having* $|\mathcal{PD}^{(l)}| = O\big(k\rho^{1/C}\big)$*, the same is true at the* $(l+1)$*-th level with probability* $1 - O\big(n^{-\zeta_1}\big)$*.*

Proof. Among the possibly defective nodes at the l-th level, at most k are defective, amounting to at most $k\rho^{1/C}$ children at the next level. Furthermore, by Lemma 4.3.3 and Markov's inequality, at most k non-defective nodes are marked as possibly defective, with probability at least

$$1 - O\bigg(\rho^{1/C}\Big(\frac{k\rho}{n}\Big)^N\bigg) = 1 - O(n^{-\zeta_1}), \tag{4.8}$$

where the equality holds for any $\zeta_1 > 0$ by suitable choices of C and N (see Lemma 4.3.2). Thus, this also amounts to at most $k\rho^{1/C}$ additional children at the next level. Summing these together, we have $|\mathcal{PD}^{(l+1)}| \le 2k\rho^{1/C}$, with probability at least $1 - O\big(n^{-\zeta_1}\big)$. □

By induction, for any given level l, we have $|\mathcal{PD}^{(l)}| = O\big(k\rho^{1/C}\big)$ with conditional probability at least $1 - O\big(n^{-\zeta_1}\big)$. Taking a union bound over all C levels (with $C = O(1)$), the same follows for all levels simultaneously with probability at least $1 - O\big(n^{-\zeta_1}\big)$.

4.3.2 Analysis of the Final Level

Recall that at the final level, we perform $C'n/\rho$ tests. We study the error probability conditioned on the high-probability event $|\mathcal{PD}^{(C)}| = O\big(k\rho^{1/C}\big)$.

For a given non-defective item in a single iteration of the C' independent iterations of tests, by Lemma 4.3.1, the probability of appearing in a positive test is at most $k\rho/n$. Since the non-defective item participates in C' independent tests, the probability of it appearing only in positive tests is $(k\rho/n)^{C'}$. By a union bound over the $O\big(k\rho^{1/C}\big)$ non-defective singletons at the final level, the error probability is upper bounded by

$$O\bigg(k\rho^{1/C}\Big(\frac{k\rho}{n}\Big)^{C'}\bigg) = O(n^{-\zeta_2}), \tag{4.9}$$

where the equality holds for any $\zeta_2 > 0$ and suitably-chosen C and C' due to Lemma 4.3.2 (with C' replacing N).

4.3.3 Number of Tests, Error Probability, and Decoding Time

- **Number of tests:** We used CNn/ρ tests in the first C levels and $C'n/\rho$ tests in the final level, which sums up to $CNn/\rho + C'n/\rho = O(n/\rho)$.

- **Error probability:** For each level l, we have $|\mathcal{PD}^{(l)}| = O\big(k\rho^{1/C}\big)$ with probability $1 - O\big(n^{-\zeta_1}\big)$. Furthermore, the final level incurs $O\big(n^{-\zeta_2}\big)$ error probability. This gives us a total error probability of $O\big(n^{-\zeta_1} + n^{-\zeta_2}\big) = O\big(n^{-\zeta}\big)$, where $\zeta = \min\{\zeta_1, \zeta_2\}$. Since we allowed ζ_1 and ζ_2 to be arbitrarily large, the same holds for ζ.

- **Decoding time:** The decoding time is dominated by the test outcome checks in our decoding procedure. For the first level $l = 0$, we have $|\mathcal{PD}^{(0)}| = n/\rho$, which is equivalent to the total number of test outcome checks. For the remaining $C-1$ levels $l \in \{1, \dots, C-1\}$, we considered a total of $O\big(k\rho^{1/C}\big)$ possibly defective nodes w.h.p., and for each possibly defective item, we conducted N test outcome checks. This gives us total number of $O\big(k\rho^{1/C}\big)$ test outcome checks. At the final level, for each of the $O\big(k\rho^{1/C}\big)$ relevant leaf nodes, we perform C' test outcome checks for a total time of $O\big(k\rho^{1/C}\big)$. Summing these gives $O(n/\rho)$, since $O\big(k\rho^{1/C}\big) = o(n/\rho)$ for a sufficiently large C (refer to (4.4)). Since it takes $O(1)$ time to check whether each node is in a positive or negative test, we get a total decoding time of $O(n/\rho)$.

Chapter 5

Storage Reductions via Hashing

In order to reduce the storage, we make one modification to each algorithm. Instead of storing the test outcomes of every node, we interpret the node-to-test mappings at each level (except for one-to-one mappings) as hash functions. Since the high storage comes from explicitly storing the corresponding test outcomes of nodes, the key to reducing the overall storage is to use lower storage hash families. The reduced storage comes at the expense of reduced independence between different hash values. This drawback has a negligible effect on the guarantees of our algorithm under the noisy setting and size-constrained tests constraint, as the proofs of Theorem 2.2.1 and Theorem 4.2.1 require only pairwise independence or weaker. However, the drawback is significant for our algorithm under the finitely divisible items constraint, as our proof of Theorem 3.2.1 uses full independence.

5.1 Noisy Setting

For levels $l = \log_2 k, \dots, \log_2 n$, we interpret the node-to-test mappings $\{1, \dots, 2^l\} \to \{1, \dots, Ck\}$ at each level as hash functions.

5.1.1 Algorithmic Guarantees

With the hashing modification to the algorithm, we have the following counterpart to Theorem 2.2.1:

Theorem 5.1.1. (Algorithmic guarantees) *Let $\mathcal{S}$ be a (defective) subset of $\{1,\dots,n\}$ of cardinality $k = o(n)$. For any constants $\epsilon > 0$ and $t > 0$ satisfying $\epsilon t > 1$, there exists choices of $C, C', N = O(1)$ and $r = O(\log k + \log\log n)$ such that with $O(k \log n)$ tests, the algorithm adapted from Section 2.1, with a $O(1)$-wise independent hash family satisfies the following with probability at least $1 - O\big((k \log \frac{n}{k})^{1-\epsilon t}\big)$:*

- *The returned estimate $\widehat{\mathcal{S}}$ equals $\mathcal{S}$;*
- *The decoding time is $O\big(\mathsf{T}_{\text{hash}}(k \log \frac{n}{k})^{1+\epsilon}\big)$, where T_{hash} is the evaluation time for one hash value;*
- *The storage required is $O(\mathsf{S}_{\text{hash}} \log n + k \log^2 n)$ bits, where S_{hash} is the number of bits of storage required for one hash function.*

We briefly discuss some explicit values that can be attained for T_{hash} and S_{hash}. Suppose that all variables are powers of two, we can adopt the classical approach of Wegman and Carter [33], and consider a random polynomial over the Galois field $\mathrm{GF}(2^m)$, where $m \in \{\log_2 k, \dots, \log_2 n\}$ (depending on the level). In this case, one attains d-wise independence by storing d elements of $\mathrm{GF}(2^m)$ (or $O(d \log n)$ bits), and performing $O(d)$ additions and multiplications in $\mathrm{GF}(2^m)$ to evaluate the hash. As a result, with $d = 2$, we get

$$\mathsf{T}_{\text{hash}} = O(1) \text{ and } \mathsf{S}_{\text{hash}} = O(\log n), \tag{5.1}$$

under the assumption that operations in $\mathrm{GF}(2^m)$ can be performed in constant time. Hence, Theorem 5.1.1 gives $O\big((k \log n)^{1+\epsilon}\big)$ decoding time and $O(k \log^2 n)$ storage.

5.1.2 Outline of Analysis

The lengths of the test sequences remain the same, which implies that there are no changes to the number of tests required. Since the hash family is (at least) pairwise independent, our analysis of the error probability for Theorem 2.2.1 still holds. This implies that there is no change to the scaling of the error probability either. The analysis of the decoding time is also similar to that of Theorem 2.2.1, except that each test outcome check (i.e., a hash) takes T_{hash} time to compute.

Hence, the decoding time is $O\big(\mathsf{T}_{\text{hash}}\big(k\log\frac{n}{k}\big)^{1+\epsilon}\big)$. The main change is in the storage. Recalling that we use $N = O(1)$ hashes at each level except the last, and $C'N\log_2 n = O(\log n)$ hashes at the final level, for a total of $O\big(\log_2\frac{n}{k}\big) + O(\log n) = O(\log n)$ hashes, requiring $O(\mathsf{S}_{\text{hash}}\log n)$ bits of storage. In addition, for any level l, we know that $|\mathcal{PD}^{(l)}| = O(k\log n)$ w.h.p. Hence, the storage required for the possibly defective set is $O(k\log n)$ integers, or $O(k\log^2 n)$ bits. The total storage required is $O(\mathsf{S}_{\text{hash}}\log n + k\log^2 n)$ bits.

5.2 Finitely Divisible Items

We make a slight modification to the algorithm: At the second last level $l = \gamma' - 1$, we set the length of our test sequence to $Ck(n/k)^{1/\gamma'}$ (i.e., $T'_{\text{len}} = T_{\text{len}}$). With this modification, for levels $l = 2,\dots,\gamma'-1$, we interpret the node-to-test mappings $\big\{1,\dots,k\big(\frac{n}{k}\big)^{l/\gamma'}\big\} \to \big\{1,\dots,Ck\big(\frac{n}{k}\big)^{1/\gamma'}\big\}$ at each level as hash functions. Similarly, for the final level, we interpret the node-to-test mappings $\{1,\dots,n\} \to \{1,\dots,T''_{\text{len}}\}$ as hash functions

5.2.1 Algorithmic Guarantees

With the hashing modification to the algorithm, we have the following counterpart to Theorem 3.2.1:

Theorem 5.2.1. (Algorithmic guarantees) *Let $\mathcal{S}$ be a fixed (defective) subset of $\{1,\dots,n\}$ of cardinality k, let $\gamma = o(\log n)$ (with $\gamma \ge 3$) be the maximum number of times each item can be tested, fix $\gamma' \in \{3,\dots,\gamma\}$, and any function β_n decaying as n increases. There exists a choice of T_{len}, T'_{len}, and T''_{len} such that with*

$$T = O\left(\gamma k \max\left\{\left(\frac{n}{k}\right)^{\frac{1}{\gamma'}}, \left(\frac{k}{\beta_n}\right)^{\frac{1}{\gamma-\gamma'+1}}\left(\frac{n}{k}\right)^{\frac{1}{\gamma'(\gamma-\gamma'+1)}}\right\}\right), \tag{5.2}$$

the algorithm adapted from Section 3.1, with a $O(\gamma)$-wise independent hash family, yields the following with probability at least $1 - O(\gamma/k + \beta_n)$:

- *The returned estimate $\widehat{\mathcal{S}}$ equals $\mathcal{S}$;*
- *The decoding time is $O\big(\mathsf{T}_{\text{hash}}\gamma k(n/k)^{1/\gamma'}\big)$, where T_{hash} is the evaluation time for one hash value;*

- *The storage required is* $O\big(k(n/k)^{1/\gamma'}\log n + \mathsf{S}_{\text{hash}}\gamma\big)$ *bits, where* S_{hash} *is the number of bits of storage required for one hash function.*

We conduct a one-to-one map for $l = 1$ and start the hashing from the second level onwards. For the other levels, we consider a random polynomial over the Galois field GF(2^m) (see paragraph above (5.1) for more details), where $m \in \left\{\log_2\left(k\left(\frac{n}{k}\right)^{2/\gamma'}\right), \dots, \log_2 n\right\}$ (depending on the level). By storing $O(\gamma)$ elements of GF(2^m) (or $O(\gamma \log n)$ bits), we attain $O(\gamma)$-wise independence, and get

$$\mathsf{T}_{\text{hash}} = O(\gamma) \text{ and } \mathsf{S}_{\text{hash}} = O(\gamma \log n), \tag{5.3}$$

under the assumption that operations in GF(2^m) can be performed in constant time. Hence, Theorem 5.2.1 gives $O\big(\gamma^2 k(n/k)^{1/\gamma'}\big)$ decoding time and $O\big(\big(k(n/k)^{1/\gamma'} + \gamma^2\big)\log n\big)$ bits of storage. Minor improvements could be made to the scaling of decoding time and storage by considering the multi-point evaluation of polynomials as in [7].

5.2.2 Analysis

At $l = 1$, the probability of a non-defective node being placed in a positive test is zero because each node is placed in its own individual test. We will show that throughout the course of the algorithm for levels $l = 2, \dots, \gamma'$, the size of our $\mathcal{PD}^{(l)}$ set remains at $O\big(k(n/k)^{1/\gamma'}\big)$ with high probability. We will show this using an induction argument.

For the base case $l = 2$, we start by looking at the preceding level. Each node in the previous level $l = 1$ is allocated to an individual test, which implies that all nodes in $l = 1$ are identified correctly. Hence, only the children of the defective nodes in $l = 1$ are "explored" further in $l = 2$. Since the number of defective nodes in $l = 1$ is at most k and each node has $M^{1/(\gamma'-1)} = (n/k)^{1/\gamma'}$ children, we have $|\mathcal{PD}^{(2)}| \le k(n/k)^{1/\gamma'}$.

Consider two non-defective nodes indexed by u and v at a given level l having $k' \le k$ defective nodes. Let A_u and A_v be the indicator random variables of u and v colliding with any defective node respectively. The dependence of these quantities on l is left implicit. We condition on all of the test placements performed at the earlier levels, accordingly $\mathbb{E}_l[\cdot]$ and

$\mathrm{Var}_l[\cdot]$ for the conditional expectation and conditional variance. By the inductive hypothesis, we have $|\mathcal{PD}^{(l)}| = O\big(k(n/k)^{1/\gamma'}\big)$.

Lemma 5.2.2. *Under the preceding setup and definitions, if $|\mathcal{PD}^{(l)}| = O\big(k(n/k)^{1/\gamma'}\big)$, then we have*

$$\mathbb{E}_l\Big[\sum_u A_u\Big] = O(k) \text{ and } \mathrm{Var}_l\Big[\sum_u A_u\Big] = O(k), \tag{5.4}$$

where the sums are over all non-defective nodes in $\mathcal{PD}^{(l)}$.

Proof. For ease of notation, we leave the subscripts $(\cdot)_l$ implicit throughout the proof, but the associated conditioning is understood to apply to all probabilities, expectations, variance terms, and so on.

We first prove $\mathbb{E}\big[\sum_u A_u\big] = O(k)$. The event A_u occurs if u is hashed into the same test as any of the $k' \le k$ defective nodes. Since we are hashing into $\big\{1,\dots,Ck(n/k)^{1/\gamma'}\big\}$ and the hash family is (at least) pairwise independent, each collision occurs with probability $(1/(Ck))(n/k)^{-1/\gamma'}$. Hence, by the union bound over all k' defective nodes, u is in a positive test with probability at most $(1/C)(n/k)^{-1/\gamma'}$. We then have

$$\mathbb{E}_l\Big[\sum_u A_u\Big] = \sum_u \mathbb{E}[A_u] = \sum_u \mathbb{P}[A_u] \le \sum_u \frac{1}{C}\Big(\frac{n}{k}\Big)^{-\frac{1}{\gamma'}} = O\bigg(k\Big(\frac{n}{k}\Big)^{\frac{1}{\gamma'}}\bigg)\frac{1}{C}\Big(\frac{n}{k}\Big)^{-\frac{1}{\gamma'}} = O(k). \tag{5.5}$$

As for $\mathrm{Var}_l\big[\sum_u\big]$, we first characterize $\mathrm{Cov}[A_u, A_v]$, writing

$$\mathrm{Cov}[A_u, A_v] = \mathbb{E}[A_u, A_v] - \mathbb{E}[A_u]\mathbb{E}[A_v] \tag{5.6}$$

$$= \mathbb{P}[A_u \cap A_v] - \mathbb{P}[A_u]\mathbb{P}[A_v] \tag{5.7}$$

$$\overset{(a)}{=} \mathbb{P}[A_u] + \mathbb{P}[A_v] - \mathbb{P}[A_u \cup A_v] - \mathbb{P}[A_u]\mathbb{P}[A_v], \tag{5.8}$$

where (a) is by the inclusion-exclusion principle. We proceed by bounding $\mathbb{P}[A_u]$ (the same bound holds for $\mathbb{P}[A_v]$) and $\mathbb{P}[A_u \cup A_v]$ separately.

Probability of the individual event: Fix a non-defective node u. Let $h(\cdot)$ denote the random hash function with output values in $\big\{1,\dots,Ck(n/k)^{1/\gamma'}\big\}$, and for each defective node indexed by $i \in \{1, \dots, k'\}$, let B_i be the "bad" event that $h(i) = h(u)$. We apply the inclusion-exclusion principle, which is written in terms of the following quantities for $j = 1, \dots, k'$:

$$T_j = \sum_{1 \le i_1 < \cdots < i_j \le k'} \mathbb{P}[B_{i_1} \cap \cdots \cap B_{i_j}]. \tag{5.9}$$

If the hash function is $(j+1)$-wise independent, this simplifies to

$$T_j = \binom{k'}{j} \left(\frac{1}{Ck} \left(\frac{n}{k} \right)^{-\frac{1}{\gamma'}} \right)^j. \tag{5.10}$$

Hence, if the hash function is $(j_{\max}+1)$-wise independent for some $j_{\max}$, then the inclusion-exclusion principle gives

$$\mathbb{P}[A_u] = \mathbb{P}\Big[\bigcup_{i=1,\dots,k'} B_i \Big] \le \sum_{j=1}^{j_{\max}} (-1)^{j+1} T_j \tag{5.11}$$

$$= \sum_{j=1}^{j_{\max}} (-1)^{j+1} \binom{k'}{j} \left(\frac{1}{Ck} \left(\frac{n}{k} \right)^{-\frac{1}{\gamma'}} \right)^j, \tag{5.12}$$

for odd-valued $j_{\max}$, and the reverse inequality for the even-valued $j_{\max}$. Using the fact that $\sum_{j=1}^{k'} (-1)^{j+1} \binom{k'}{j} \big(\frac{1}{Ck} \big(\frac{n}{k} \big)^{1/\gamma'} \big)^j$
$= 1 - \big(1 - \big(\frac{1}{Ck} \big(\frac{n}{k} \big)^{1/\gamma'} \big)^{k'}$, we can write (5.12) as

$$\mathbb{P}[A_u] \le 1 - \left(1 - \frac{1}{Ck} \left(\frac{n}{k} \right)^{-\frac{1}{\gamma'}} \right)^{k'} - \sum_{j=j_{\max}+1}^{k'} (-1)^{j+1} \binom{k'}{j} \left(\frac{1}{Ck} \left(\frac{n}{k} \right) \right)^j. \tag{5.13}$$

The final term can then be bounded as follows in absolute value:

$$\left| \sum_{j=j_{\max}+1}^{k'} (-1)^{j+1} \binom{k'}{j} \left(\frac{1}{Ck} \left(\frac{n}{k} \right) \right)^j \right| \le \sum_{j=j_{\max}+1}^{k'} \binom{k'}{j} \left(\frac{1}{Cd} \left(\frac{n}{k} \right) \right)^j \tag{5.14}$$

$$\overset{(a)}{\le} \sum_{j=j_{\max}+1}^{\infty} \left(\frac{1}{C} \left(\frac{n}{k} \right) \right)^j \tag{5.15}$$

$$\overset{(b)}{=} \frac{\big(\frac{1}{C} \big(\frac{n}{k} \big)^{-1/\gamma'} \big)^{j_{\max}+1}}{1 - \frac{1}{C} \big(\frac{n}{k} \big)^{-1/\gamma'}} \tag{5.16}$$

$$\overset{(c)}{\le} \left(\frac{1}{C} \left(\frac{n}{k} \right)^{-\frac{1}{\gamma'}} \right)^{j_{\max}}, \tag{5.17}$$

where (a) uses $\binom{k'}{j} \le (k')^j$ and $k' \le k$, (b) applies the geometric series formula, and (c) holds for $C \ge 2(n/k)^{-1/\gamma'}$ (i.e., any positive integer C). We can upper bound (5.17) by any target value δ provided that $j_{\max} \ge (\log \delta) / \big(\frac{1}{C} \big(\frac{n}{k} \big)^{-1/\gamma'} \big)$. Recall also that (5.13) is reversed for even-valued

$j_{\max}$, so loosening the preceding requirement to $j_{\max} \geq \lceil (\log \delta)/\log\left(\frac{1}{C}\left(\frac{n}{k}\right)^{-1/\gamma'}\right)\rceil + 1$ gives

$$1 - \left(1 - \frac{1}{Ck}\left(\frac{n}{k}\right)^{-\frac{1}{\gamma'}}\right)^{k'} - \delta \leq \mathbb{P}[A_u] \leq 1 - \left(1 - \frac{1}{Ck}\left(\frac{n}{k}\right)^{-\frac{1}{\gamma'}}\right)^{k'} + \delta. \tag{5.18}$$

Since u is arbitrary, the same bound also holds for $\mathbb{P}[A_v]$.

Probability of the union of two events: We can decompose $\mathbb{P}[A_u \cup A_v]$ as follows:

$$\begin{aligned}\mathbb{P}[A_u \cup A_v] = &\left(1 - \frac{1}{Ck}\left(\frac{n}{k}\right)^{-\frac{1}{\gamma'}}\right)\mathbb{P}[A_u \cup A_v | h(u) \neq h(v)] \\ &+ \frac{1}{Ck}\left(\frac{n}{k}\right)^{-\frac{1}{\gamma'}}\mathbb{P}[A_u \cup A_v | h(u) = h(v)] \qquad (5.19)\\ = \mathbb{P}[A_u \cup A_v | h(u) \neq h(v)] &+ \frac{1}{Ck}\left(\frac{n}{k}\right)^{-\frac{1}{\gamma'}}\big(\mathbb{P}[A_u \cup A_v | h(u) = h(v)] \\ &- \mathbb{P}[A_u \cup A_v | h(u) \neq h(v)]\big). \qquad (5.20)\end{aligned}$$

To simplify (5.20), we look at the probability terms separately. We first focus on the case $h(u) = h(v)$. Similar to the above, let B'_i be the "bad" event that $h(i) = h(u)$ (and hence also equals $h(v)$), and define

$$T'_j = \sum_{1 \leq i_1 < \cdots < i_j \leq k'} \mathbb{P}[B'_{i_1} \cap \cdots \cap B'_{i_j} | h(u) = h(v)]. \tag{5.21}$$

If the hash function is $(j+2)$-wise independent, this simplifies to

$$T'_j = \binom{k'}{j}\left(\frac{1}{Ck}\left(\frac{n}{k}\right)^{-\frac{1}{\gamma'}}\right), \tag{5.22}$$

which is the same as (5.10). Following a similar argument from (5.11) to (5.18), we find that if (i) $C \geq 2(n/k)^{-1/\gamma'}$, (ii) $j_{\max} \geq \lceil (\log \delta)/\log\left(\frac{1}{C}\left(\frac{n}{k}\right)^{-1/\gamma'}\right)\rceil + 1$, and (iii) the hash function is $(j_{\max} + 2)$-wise independent, then the following holds:

$$1 - \left(1 - \frac{1}{Ck}\left(\frac{n}{k}\right)^{-\frac{1}{\gamma'}}\right)^{k'} - \delta \leq \mathbb{P}[A_u \cup A_v | h(u) = h(v)] \leq 1 - \left(1 - \frac{1}{Ck}\left(\frac{n}{k}\right)^{-\frac{1}{\gamma'}}\right)^{k'} + \delta \tag{5.23}$$

Next, we focus on the case $h(u) \neq h(v)$. Similar to the above, let B''_i be the "bad" event that $h(i) \in \{h(u), h(v)\}$, and define

$$T'_j = \sum_{1 \leq i_1 < \cdots < i_j \leq k'} \mathbb{P}[B''_{i_1} \cap \cdots \cap B''_{i_j} | h(u) \neq h(v)]. \tag{5.24}$$

If the hash function is $(j+2)$-wise independent, this simplifies to

$$T'_j = \binom{k'}{j}\left(\frac{2}{Ck}\left(\frac{n}{k}\right)^{-\frac{1}{\gamma'}}\right), \tag{5.25}$$

where the factor of two comes from the possibility of colliding with either u or v. Following the same argument from (5.11) to (5.18), we find that if (i) $C \geq 4(n/k)^{-1/\gamma'}$, (ii) $j_{\max} \geq \lceil (\log \delta)/\log\left(\frac{2}{C}\left(\frac{n}{k}\right)^{-1/\gamma'}\right)\rceil + 1$, and (iii) the hash function is $(j_{\max}+2)$-wise independent, then the following analog of (5.18) holds:

$$1-\left(1-\frac{2}{Ck}\left(\frac{n}{k}\right)^{-\frac{1}{\gamma'}}\right)^{k'} - \delta \leq \mathbb{P}[A_u \cup A_v | h(u) \neq h(v)] \leq 1-\left(1-\frac{2}{Ck}\left(\frac{n}{k}\right)^{-\frac{1}{\gamma'}}\right)^{k'} + \delta \quad (5.26)$$

With the simplified probabilities, we get the following:

$$\begin{aligned}
&\mathbb{P}[A_u \cup A_v | h(u) = h(v)] - \mathbb{P}[A_u \cup A_v | h(u) \neq h(v)] \\
&\quad = 1-\left(1-\frac{1}{Ck}\left(\frac{n}{k}\right)^{-\frac{1}{\gamma'}}\right)^{k'} + O(\delta) - \left(1-\left(1-\frac{2}{Ck}\left(\frac{n}{k}\right)^{-\frac{1}{\gamma'}}\right)^{k'} + O(\delta)\right) && (5.27)\\
&\quad = \left(1-\frac{2}{Ck}\left(\frac{n}{k}\right)^{-\frac{1}{\gamma'}}\right)^{k'} - \left(1-\frac{1}{Ck}\left(\frac{n}{k}\right)^{-\frac{1}{\gamma'}}\right)^{k'} + O(\delta) && (5.28)\\
&\quad \overset{(a)}{=} \left(1-O\left(\left(\frac{n}{k}\right)^{-\frac{1}{\gamma'}}\right)\right) - \left(1-O\left(\left(\frac{n}{k}\right)^{-\frac{1}{\gamma'}}\right)\right) + O(\delta) && (5.29)\\
&\quad \overset{(b)}{=} O\left(\left(\frac{n}{k}\right)^{-\frac{1}{\gamma'}}\right), && (5.30)
\end{aligned}$$

where (a) applies a first order Taylor expansion, and (b) assumes that $\delta = o((n/k)^{-1/\gamma'})$. Using (5.26), (5.30), and (5.20), we deduce

$$\begin{aligned}
\mathbb{P}[A_u \cup A_v] &= 1-\left(1-\frac{2}{Ck}\left(\frac{n}{k}\right)^{-\frac{1}{\gamma'}}\right)^{k'} + O(\delta) - \frac{1}{Ck}\left(\frac{n}{k}\right)^{-\frac{1}{\gamma'}} O\left(\left(\frac{n}{k}\right)^{-\frac{1}{\gamma'}}\right) && (5.31)\\
&= 1-\left(1-\frac{2}{Ck}\left(\frac{n}{k}\right)^{-\frac{1}{\gamma'}}\right)^{k'} + O(\delta) - O\left(\frac{1}{k}\left(\frac{n}{k}\right)^{-\frac{2}{\gamma'}}\right). && (5.32)
\end{aligned}$$

Combining and simplifying: Setting $\delta = (1/k)(n/k)^{-2/\gamma'}$ and

$$\begin{aligned}
j_{\max} &= \left\lceil \frac{\log \delta}{\log\left(\frac{1}{C}\left(\frac{n}{k}\right)^{-1/\gamma'}\right)} \right\rceil + 1 = O\left(\frac{\log\left(\frac{1}{k}\left(\frac{n}{k}\right)^{-2/\gamma'}\right)}{\log\left(\frac{1}{C}\left(\frac{n}{k}\right)^{-1/\gamma'}\right)}\right) && (5.33)\\
&= O\left(\frac{\log k + \frac{2}{\gamma'}\log\left(\frac{n}{k}\right)}{\log C + \frac{1}{\gamma'}\log\left(\frac{n}{k}\right)}\right) = O\left(\frac{\gamma' \log k}{\log n - \log k}\right) && (5.34)\\
&\overset{(a)}{=} O(\gamma') = O(\gamma), && (5.35)
\end{aligned}$$

where (a) where $k = o(n)$ implies that we can only have either $\log k = O(\log n)$ or $\log k = o(\log n)$. In both cases, we have $\frac{\log k}{\log n - \log k} = O(1)$. Combining the this result with the above findings from (5.13) and (5.32), we deduce that for a $O(\gamma)$-wise independent hash,

$$\mathbb{P}[A_u] = 1 - \left(1 - \frac{1}{Ck}\left(\frac{n}{k}\right)^{-\frac{1}{\gamma'}}\right)^{k'} + O\left(\frac{1}{k}\left(\frac{n}{k}\right)^{-\frac{2}{\gamma'}}\right) \tag{5.36}$$

$$\mathbb{P}[A_u \cup A_v] = 1 - \left(1 - \frac{2}{Ck}\left(\frac{n}{k}\right)^{-\frac{1}{\gamma'}}\right)^{k'} + O\left(\frac{1}{k}\left(\frac{n}{k}\right)^{-\frac{2}{\gamma'}}\right) \tag{5.37}$$

The idea in the following is to approximate $1 - \frac{w}{Ck(n/k)^{1/\gamma'}} \approx \exp\big(-\frac{w}{Ck(n/k)^{1/\gamma'}}\big)$ for $w = 1, 2$, and substitute into (5.8). To make this more precise, we use the fact that $k' \le k$ to write the following for $w = 1$:

$$\begin{aligned}
\left(1 - \frac{1}{Ck}\left(\frac{n}{k}\right)^{-\frac{1}{\gamma'}}\right)^{k'} &= \left(\exp\left(-\frac{1}{Ck}\left(\frac{n}{k}\right)^{-\frac{1}{\gamma'}} + O\left(\frac{1}{k^2}\left(\frac{n}{k}\right)^{-\frac{2}{\gamma'}}\right)\right)\right)^{k'} & (5.38)\\
&= \exp\left(-\frac{k'}{Ck}\left(\frac{n}{k}\right)^{-\frac{1}{\gamma'}} + O\left(\frac{1}{k}\left(\frac{n}{k}\right)^{-\frac{2}{\gamma'}}\right)\right) & (5.39)\\
&\overset{(a)}{=} \exp\left(-\frac{k'}{Ck}\left(\frac{n}{k}\right)^{-\frac{1}{\gamma'}}\right)\left(1 + O\left(\frac{1}{k}\left(\frac{n}{k}\right)^{-\frac{2}{\gamma'}}\right)\right)(1 + o(1)) & (5.40)\\
&= \exp\left(-\frac{k'}{Ck}\left(\frac{n}{k}\right)^{-\frac{1}{\gamma'}}\right) + O\left(\frac{1}{k}\left(\frac{n}{k}\right)^{-\frac{2}{\gamma'}}\right), & (5.41)
\end{aligned}$$

where (a) uses $e^x = (1 + x)(1 + o(1))$ when $x = o(1)$. Applying a similar argument to $1 - \frac{2}{Ck(n/k)^{1/\gamma'}}$ and substituting into (5.8), we obtain

$$\begin{aligned}
\mathrm{Cov}[A_u, A_v] &= 2\left(1 - \exp\left(-\frac{k'}{Ck}\left(\frac{n}{k}\right)^{-\frac{1}{\gamma'}}\right)\right) - \left(1 - \exp\left(-\frac{2k'}{Ck}\left(\frac{n}{k}\right)^{-\frac{1}{\gamma'}}\right)\right)\\
&\quad - \left(1 - \exp\left(-\frac{k'}{Ck}\left(\frac{n}{k}\right)^{-\frac{1}{\gamma'}}\right)\right)^2 + O\left(\frac{1}{k}\left(\frac{n}{k}\right)^{-\frac{2}{\gamma'}}\right) & (5.42)\\
&= O\left(\frac{1}{k}\left(\frac{n}{k}\right)^{-\frac{2}{\gamma'}}\right), & (5.43)
\end{aligned}$$

since the first three terms cancel upon expanding the square. The proof is concluded by writing

$$\begin{aligned}
\mathrm{Var}\Big[\sum_u A_u\Big] &= \sum_u \mathrm{Var}[A_u] + \sum_{u \ne v} \mathrm{Cov}[A_u, A_v] & (5.44)\\
&\overset{(a)}{=} O\left(k\left(\frac{n}{k}\right)^{\frac{1}{\gamma'}} \cdot \left(\frac{n}{k}\right)^{-\frac{1}{\gamma'}}\right) + 2\binom{k(n/k)^{1/\gamma'}}{2} O\left(\frac{1}{k}\left(\frac{n}{k}\right)^{-\frac{2}{\gamma'}}\right) & (5.45)\\
&= O(k), & (5.46)
\end{aligned}$$

where (a) is because there are $O\big(k(n/k)^{1/\gamma'}\big)$ values of u by assumption, and $\mathrm{Var}[A_u] \le \mathbb{P}[A_u] = O\big((n/k)^{-1/\gamma'}\big)$. □

Given Lemma 5.2.2, we easily deduce the following.

Lemma 5.2.3. *For a sufficiently large C, conditioned on $|\mathcal{PD}^{(l)}| = O\big(k(n/k)^{1/\gamma'}\big)$, the same is true for $|\mathcal{PD}^{(l+1)}|$ with probability $1 - O(k^{-1})$.*

Proof. Among the possibly defective nodes at the l-th level, at most k are defective, amounting to at most $k(n/k)^{1/\gamma'}$ children at the next level. By Lemma 5.2.2 and Chebyshev's inequality, with probability $1 - O(k^{-1})$, $O(k)$ non-defective nodes are marked as possibly defective, thus also amount to $O\big(k(n/k)^{1/\gamma'}\big)$ at the next level, for a total of $O\big(k(n/k)^{1/\gamma'}\big)$. □

Error Probability: By induction, for all levels l, we have $|\mathcal{PD}^{(l)}| = O\big(k(n/k)^{1/\gamma'}\big)$ with probability at least $1 - O(k^{-1})$. Taking union bound over all $O(\gamma)$ levels, the same follows for all levels simultaneously with probability $1 - O(\gamma/k)$.

Recall that by choosing $T''_{\text{len}} = k(k/\beta_n)^{\frac{1}{\gamma-\gamma'+1}}(n/k)^{\frac{1}{\gamma'(\gamma-\gamma'+1)}}$, we attain β_n error probability at the final level. The analysis of the final level in Section 3.3.2 did not rely on $h(\cdot)$ being a fully independent hash function, but rather, only relied on the condition of collision probability k/T''_{len} between any given two nodes. Since this condition still holds for any pairwise (or higher) independent hash family, we immediately deduce the same conclusion: Conditioned on the final level having $O\big(k(n/k)^{1/\gamma'}\big)$ nodes marked as possibly defective, we attain β_n error probability.

Combining the above, we get a total error probability of $O(\gamma/k + \beta_n)$, which is in $o(1)$ if $\gamma/k = o(1)$.

Number of Tests, Decoding Time, and Storage:

- **Number of tests:** Despite decreasing T'_{len}, the scaling of tests remains the same as the fully independent case, which is

$$T = O\left(\gamma k \max\left\{\left(\frac{n}{k}\right)^{\frac{1}{\gamma'}}, \left(\frac{k}{\beta_n}\right)^{\frac{1}{\gamma-\gamma'+1}}\left(\frac{n}{k}\right)^{\frac{1}{\gamma'(\gamma-\gamma'+1)}}\right\}\right). \tag{5.47}$$

- **Decoding time:** Since we reached $O\big(\gamma' k(n/k)^{1/\gamma'}\big)$ nodes in the first $\gamma' - 1$ levels and $O\big(k(n/k)^{1/\gamma'}\big)$ nodes in the final level, we get the same scaling as the fully independent case on the number of times we check the test outcome of each node. In this case, each check (hash) takes T_{hash} time to compute. Hence, the total decoding time is $O\big(\mathsf{T}_{\text{hash}}\gamma k(n/k)^{1/\gamma'}\big)$. Note that we assume that the one-to-one node-to-test mapping function at $l = 1$ takes $O(1)$ time per map.

- **Storage:** We use one hash at each level except the last, and $O(\gamma)$ hashes at the final level, for a total of $O(\gamma)$ hashes, requiring $(\mathsf{S}_{\text{hash}}\gamma)$. In addition, under the high probability event that there are $O\big(k(n/k)^{1/\gamma'}\big)$ possibly defective nodes at each level. Hence, the storage of the possibly defective set requires $O\big(k(n/k)^{1/\gamma'}\big)$ integers in $\{1,\dots,n\}$, or $O\big(k(n/k)^{1/\gamma'}\log n\big)$ bits. Summing the storage up, we attain $O\big(k(n/k)^{1/\gamma'}\log n + \mathsf{S}_{\text{hash}}\gamma\big)$ bits of total storage. Note that we assume that the one-to-one node-to-test mapping function at $l = 1$ takes $O(1)$ storage.

5.3 Size-Constrained Tests

For levels $l = 1,\dots,C$, we interpret the node-to-test mappings $\big\{1,\dots,n/\rho^{1-l/C}\big\} \to \{1,\dots,n/\rho\}$ at each level as hash functions. Since nodes are allocated to their tests using a matrix with fixed row and column weights, we take extra care to ensure that these constraints are satisfied by our hash function. To achieve this we first define the following:

Definition 5.3.1. (Approximately pairwise-independent permutation) Fix m, and let $\pi : \{1,\dots,m\} \to \{1,\dots,m\}$ be a random permutation. We say that π is *approximately pairwise-independent* if, for any $i, i' \in \{1,\dots,m\}$ and any integer t, we have $\mathbb{P}[|\pi(i) - \pi(i')| \le t] \le 4t/m$.

It is well known that such permutations exist in the form of a simple modulo-m multiplication; we will specifically use the following lemma from [34].

Lemma 5.3.1. (Choice of permutation [34, Lemma 3.2]) *Let m be a power of two, and define $\pi(i) = \sigma \cdot i$, where σ is chosen uniformly at random from the odd numbers in $\{1,\dots,m\}$. Then π is an approximately pairwise-independent random permutation.*

5.3.1 Algorithmic Guarantees

With the hashing modification to the algorithm, we have the following counterpart to Theorem 4.2.1:

Theorem 5.3.2. (Algorithmic guarantees) *Let $\mathcal{S}$ be a (defective) subset of $\{1,\dots,n\}$ of cardinality $k = O\big(n^{1-\epsilon_1}\big)$ for some $\epsilon_1 \in (0,1]$ and the test size constraint be $\rho = O\big((n/k)^{1-\epsilon_2}\big)$ for* *some $\epsilon_2 \in (0,1]$. For any $\zeta > 0$, there exists a choice of $C, C', N = O(1)$ such that with $O\big(n/\rho\big)$ tests, the algorithm adapted from Section 4.1, with an approximately pairwise-independent permutation, satisfies the following with probability at least $1 - O\big(n^{-\zeta}\big)$:*

- *The returned estimate $\widehat{\mathcal{S}}$ equals $\mathcal{S}$;*
- *The decoding time is $O\big(\mathsf{T}_{\text{hash}} n/\rho\big)$, where T_{hash} is the evaluation time for one hash value;*
- *The storage required is $O(\mathsf{S}_{\text{hash}} + n/\rho)$ bits, where S_{hash} is the number of bits of storage required for one hash function.*

Using an approximately pairwise-independent permutation, we discuss some explicit values that can be attained for T_{hash} and S_{hash}. For $l = 0$, assuming that nodes are placed into tests in an ordered manner (i.e., first node goes to the first test and so on), we can get the test outcome of a given i-th node in $O(1)$ time by checking the i-th test. At each level $l \in \{1, \dots, C\}$, we desire a hash function $h_l : \big\{1, \dots, n/\rho^{1-l/C}\big\} \to \{1, \dots, n/\rho\}$ such at each bucket (test) has a load (number of nodes) of exactly $\rho^{l/C}$. This can be achieved by first applying the permutation in Lemma 5.3.1 and then truncating the last $(l/C)\log_2 \rho$ bits of the permutation value. As a result, we get $\mathsf{T}_{\text{hash}} = O(1)$ and $\mathsf{S}_{\text{hash}} = O(1)$, which together with Theorem 5.3.2 gives us $O(n/\rho)$ decoding time and $O(n/\rho)$ storage.

5.3.2 Outline of Analysis

We start by noting that the length of test sequences remain the same, which implies that there are no changes to the number of tests required. We introduce a corollary below before studying the changes with the error probability, decoding time, and storage.

Corollary 5.3.2.1. *For each level $l \in \{1, \dots, C\}$, for any $i, i' \in \{1, \dots, n/\rho^{1-l/C}\}$, we have $\mathbb{P}[h_l(i) = h_l(i')] \leq 4\rho/n$.*

Proof. Recall that $h_l : \big\{1, \dots, n/\rho^{1-l/C}\big\} \to \{1, \dots, n/\rho\}$. For any $i, i' \in \big\{1, \dots, n/\rho^{1-l/C}\big\}$, we have

$$\mathbb{P}[h_l(i) = h_l(i')] \overset{(a)}{=} \mathbb{P}[|\pi(i) - \pi(i')| \leq \rho^{l/C}] \overset{(b)}{\leq} \frac{4\rho}{n}, \tag{5.48}$$

where (a) is because if i and i' are in the same bucket, then all their bits except the last $(l/C)\log_2 \rho$ bits are the same. Hence, $\pi(i)$ and $\pi(i')$ can be at most $\rho^{l/C}$ (bucket size) apart. (b) is by applying Definition 5.3.1. □

For $l = 1, \ldots, C$, we use the hash function introduced above, where we interpret the node-to-test mappings $\{1, \ldots, n/\rho^{1-l/C}\} \to \{1, \ldots, n/\rho\}$ as hash functions. By Corollary 5.3.2.1, we see that the probability of a non-defective node colliding with a defective node is still in $O(\rho/n)$. By union bound over all defective nodes (k is the worst case), the probability of a non-defective node being placed in a positive test is $O(k\rho/n)$, which has the same scaling as the probability presented in Lemma 4.3.1. Hence, the scaling of the error probability remains unchanged.

The analysis of the decoding time is similar to that of Theorem 4.2.1, but each hash takes T_{hash} time to compute. Hence, the decoding time is $O\big(\mathsf{T}_{\text{hash}} n/\rho\big)$.

Regarding the storage, we use $N = O(1)$ hashes at each level (except $l = 0$ and $l = C$), and $C' = O(1)$ hashes at the final level, for a total of $O(1)$ hashes, requiring $O(\mathsf{S}_{\text{hash}})$ storage. In addition, under the high probability event that there are $O(k\rho^{1/C})$ possibly defective nodes at each level, their storage requires $O(k\rho^{1/C})$ integers, or $O(k\rho^{1/C}\log n) = O(n/\rho)$ bits (refer to (4.4) to see why $O(k\rho^{1/C}) = O(n/\rho)$). Note that we can ignore the $\log n$ factor because it is negligible in the power of n as $n \to \infty$, which is used for comparison in (4.4). Hence, the total storage requires $O(\mathsf{S}_{\text{hash}} + n/\rho)$ bits.

Chapter 6

Conclusion

We have provided fast splitting algorithms for noisy and sparsity-constrained group testing, significantly improving the decoding times of previous works while maintaining optimality or near-optimality in the scaling of the number of tests. Two potentially interesting directions for further work include:

- Develop a sublinear-time algorithm for the finitely-divisible setting that attains the stronger bound of the DD algorithm [29, 1], rather than only that of the COMP algorithm (see Table 1.2 and Figure 3.2).
- Handle the case that tests are simultaneously noisy and size-constrained. This appears to be non-trivial, as our variations for the noisy and size-constrained settings currently do not "fit together" nicely; the former looks $\omega(1)$ levels down the tree, while the latter uses a tree of depth $O(1)$.

References

O. Gebhard, M. Hahn-Klimroth, O. Parczyk, M. Penschuck, M. Rolvien, J. Scarlett, and N. Tan. Near optimal sparsity-constrained group testing: improved bounds and algorithms. https://arxiv.org/abs/2004.11860, 2020.

V. Gandikota, E. Grigorescu, S. Jaggi, and S. Zhou. Nearly optimal sparse group testing. *IEEE Trans. Inf. Theory*, 65(5):2760 – 2773, 2019. doi: 10.1109/TIT.2019.2891651.

M. Aldridge, O. Johnson, and J. Scarlett. Group testing: An information theory perspective. *Found. Trend. Comms. Inf. Theory*, 15(3–4):196–392, 2019.

C. A. Hogan, M. K. Sahoo, and B. A. Pinsky. Sample Pooling as a Strategy to Detect Community Transmission of SARS-CoV-2. *JAMA*, 323(19):1967–1969, 05 2020. doi: 10.1001/jama.2020.5445. URL https://doi.org/10.1001/jama.2020.5445.

C. M. Verdun, T. Fuchs, P. Harar, D. Elbrächter, D. S. Fischer, J. Berner, P. Grohs, F. J. Theis, and F. Krahmer. Group testing for SARS-CoV-2 allows for up to 10-fold efficiency increase across realistic scenarios and testing strategies. *medRxiv*, 2020. doi: 10.1101/2020.04.30.20085290.

I. Yelin, N. Aharony, E. Shaer-Tamar, A. Argoetti, E. Messer, D. Berenbaum, E. Shafran, A. Kuzli, N. Gandali, T. Hashimshony, Y. Mandel-Gutfreund, M. Halberthal, Y. Geffen, M. Szwarcwort-Cohen, and R. Kishony. Evaluation of COVID-19 RT-qPCR test in multi-sample pools. *medRxiv*, 2020. doi: 10.1101/2020.03.26.20039438.

M. Cheraghchi and V. Nakos. Combinatorial group testing and sparse recovery schemes with near-optimal decoding time. https://arxiv.org/abs/2006.08420, 2020.

E. Price and J. Scarlett. A fast binary splitting approach to non-adaptive group testing. In *RANDOM*, 2020.

R. Dorfman. The detection of defective members of large populations. *Ann. Math. Stats.*, 14 (4):436–440, 1943.

C. L. Chan, S. Jaggi, V. Saligrama, and S. Agnihotri. Non-adaptive group testing: Explicit bounds and novel algorithms. *IEEE Trans. Inf. Theory*, 60(5):3019–3035, May 2014. ISSN 0018-9448.

M. Aldridge, L. Baldassini, and O. Johnson. Group testing algorithms: Bounds and simulations. *IEEE Trans. Inf. Theory*, 60(6):3671–3687, June 2014. ISSN 0018-9448. doi: 10.1109/TIT.2014.2314472.

J. Scarlett and V. Cevher. Phase transitions in group testing. In *Proc. ACM-SIAM Symp. Disc. Alg. (SODA)*, 2016.

O. Johnson, M. Aldridge, and J. Scarlett. Performance of group testing algorithms with near-constant tests-per-item. *IEEE Trans. Inf. Theory*, 65(2):707–723, Feb. 2019.

W. H. Bay, E. Price, and J. Scarlett. Optimal non-adaptive probabilistic group testing requires $\Theta(\min\{k \log n, n\})$ tests. https://arxiv.org/abs/2006.01325, 2020.

M. Malyutov. The separating property of random matrices. *Math. Notes Acad. Sci. USSR*, 23 (1):84–91, 1978. ISSN 0001-4346. doi: 10.1007/BF01104893.

J. Scarlett and V. Cevher. Near-optimal noisy group testing via separate decoding of items. *IEEE Trans. Sel. Topics Sig. Proc.*, 2(4):625–638, 2018.

O. Gebhard, O. Johnson, P. Loick, and M. Rolvien. Improved bounds for noisy group testing with constant tests per item. https://arxiv.org/abs/2007.01376, 2020.

H. A. Inan, P. Kairouz, M. Wootters, and A. Özgür. On the optimality of the Kautz-Singleton construction in probabilistic group testing. *IEEE Trans. Inf. Theory*, 65(9):5592–5603, Sept. 2019.

H. A. Inan and A. Ozgur. Strongly explicit and efficiently decodable probabilistic group testing. In *IEEE Int. Symp. Inf. Theory*, 2020.

J. Scarlett and O. Johnson. Noisy non-adaptive group testing: A (near-)definite defectives approach. *IEEE Trans. Inf. Theory*, PP:1–1, 01 2020. doi: 10.1109/TIT.2020.2970184.

S. Cai, M. Jahangoshahi, M. Bakshi, and S. Jaggi. Efficient algorithms for noisy group testing. *IEEE Trans. Inf. Theory*, 63(4):2113–2136, 2017. doi: 10.1109/TIT.2017.2659619.

K. Lee, R. Pedarsani, and K. Ramchandran. SAFFRON: A fast, efficient, and robust framework for group testing based on sparse-graph codes. In *IEEE Int. Symp. Inf. Theory*, 2016.

S. Bondorf, B. Chen, J. Scarlett, H. Yu, and Y. Zhao. Sublinear-time non-adaptive group testing with $O(k \log n)$ tests via bit-mixing coding. *IEEE Trans. Inf. Theory*, pages 1–1, 2020.

M. Cheraghchi. Noise-resilient group testing: Limitations and constructions. In *Int. Symp. Found. Comp. Theory*, pages 62–73, 2009.

P. Indyk, H. Q. Ngo, and A. Rudra. Efficiently decodable non-adaptive group testing. In *ACM-SIAM Symp. Disc. Alg. (SODA)*, 2010.

H. Q. Ngo, E. Porat, and A. Rudra. Efficiently decodable error-correcting list disjunct matrices and applications. In *Int. Colloq. Automata, Lang., and Prog.*, 2011.

H. A. Inan, P. Kairouz, and A. Ozgur. Sparse combinatorial group testing. *IEEE Trans. Inf. Theory*, 66(5):2729–2742, 2020.

N. Tan and J. Scarlett. Near-optimal sparse adaptive group testing. In *IEEE Int. Symp. Inf. Theory*, 2020.

N. Tan and J. Scarlett. Improved bounds and algorithms for sparsity-constrained group testing. https://arxiv.org/abs/2004.03119, 2020.

F. K. Hwang. A method for detecting all defective members in a population by group testing. *J. Amer. Stats. Assoc.*, 67(339):605–608, 1972.

J.-C. Aval. Multivariate fuss–catalan numbers. *Discrete Mathematics*, 308(20):4660 – 4669, 2008.

R. Vershynin. Introduction to the non-asymptotic analysis of random matrices. *Compressed Sensing: Theory and Applications*, page 210–268, 2010. doi: 10.1017/CBO9780511794308.006.

M. N. Wegman and J. Carter. New hash functions and their use in authentication and set equality. *Journal of Computer and System Sciences*, 22(3):265 – 279, 1981. ISSN 0022-0000. doi: https://doi.org/10.1016/0022-0000(81)90033-7.

P. Indyk and M. Kapralov. Sample-optimal fourier sampling in any fixed dimension. In *IEEE Symp. Found. Comp. Sci. (FOCS)*, 2014.