Multilevel Group Testing via Sparse-graph Codes

Pedro Abdalla
Department of Electrical and Computer Engineering
University of California, Santa Barbara
pabdallateixeira@hotmail.com

Amirhossein Reisizadeh
Department of Electrical and Computer Engineering
University of California, Santa Barbara
reisizadeh@ucsb.edu

Ramtin Pedarsani
Department of Electrical and Computer Engineering
University of California, Santa Barbara
ramtin@ece.ucsb.edu

***Abstract*—In this paper, we consider the problem of multilevel group testing, where the goal is to recover a set of K defective items in a set of n items by pooling groups of items and observing the result of each test. The main difference of multilevel group testing with the classical non-adaptive group testing problem is that the result of each test is an integer in the set $[L] = \{0, 1, \cdots, L\}$: if there are $i \leq L$ defective items in the pool, the result of the test is i, and if there are more than L items in the pool, the result of the test is L. We develop a multilevel group testing algorithm using sparse-graph codes that has low sample and computational complexity. More precisely, with high probability, our algorithm provably recovers $(1 - \epsilon)$ fraction of the defective items using $C(\epsilon, L)K \log(n)$ tests, where $C(\epsilon, L)$ is a constant that only depends on ϵ and the number of levels L, and it can be precisely characterized for arbitrary L and ϵ. Furthermore, the computational complexity of our algorithm is $\mathcal{O}(K \log(n))$. As an example, our algorithm is able to recover $(1 - 10^{-3})$ fraction of the defective items with only $13.8K \log(n)$ measurements for $L = 2$. We also provide numerical results that show tight agreement with our theoretical results.**

I. Introduction

The classical group testing problem consists of finding a set of K defective items from a population of n items by performing tests on subsets of n items. Each test returns 1 if the subset contains at least one defective item, and 0 otherwise. The main goal of group testing is to design an algorithm that recovers the K defective items with a small number of tests and low decoding complexity.

Group testing arose in statistics presented by Dorfman during the Second World War [1] in order to detect all the soldiers infected with the syphilis virus. There have been many algorithms proposed under different guarantee requirements since then and today, group testing appears in a large variety of fields such as biology [2], machine learning [3], computer science [4], data analysis [5], and signal processing [6], and communications [7].

A. Our Contributions

In this work, we tackle a different group testing problem called multilevel group testing. The goal is to recover a set of K defective items in a pool of n items where the result of each test is an integer $0 \leq i \leq L$ if there are i defective items in the pool, and the result of the test is L, if there are more than L defective items in the pool. Clearly, the classical group testing problem is a special case of the multilevel problem for $L = 1$. Building on the SAFFRON algorithm proposed in [8], we provide an algorithm that recovers $(1 - \epsilon)K$ defective items with $6C(\epsilon, L)K \log(n)$ tests where $C(\epsilon, L)$ is a constant that depends only on the target reliability ϵ and the number of logical levels L.

B. Related Work

Here, we provide a short literature review, and refer the readers to [2], [8], [9] for a more thorough one. The best known lower bound on the number of required tests for the classical zero-error group testing problem with no assumptions on the support of defective items is $\Omega(\frac{K^2 \log n}{\log K})$ [10]. In [11], a scheme that requires $O(K^2 \log n)$ tests is presented with polynomial time decoding algorithm. The group testing problem has also been studied from an information-theoretic viewpoint. The works in [12]–[15] consider a prior distribution on the support of defective items.

Among the algorithms with sublinear decoding complexity, GROTESQUE is proposed in [16] for non-adaptive group testing problem. The scheme requires $\mathcal{O}(K \log K \log n)$ tests and its decoding complexity is $\mathcal{O}(K(\log n + \log K))$, which is sublinear in n. Allowing an arbitrarily small fraction of the non-zero components to be not recovered, in [8], the authors propose SAFFRON, an efficient non-adaptive group testing algorithm based on sparse-graph codes with $O(K \log(n))$ sample and decoding complexity. [17], [18] have further optimized the design of the sparse-graph code in SAFFRON to improve its performance.

The multilevel group testing is considered in [19] as semi-quantitative group testing, where the authors propose SQ-disjuct codes to recover the defective items, but the decoding complexity of their algorithm is polynomial in n.

C. Notation

In this paper, we use bold face small and capital letters for vectors and matrices, respectively. We let $[i]$ to be the set $\{1, 2, \cdots, i\}$ for $i \in \mathbb{N}$. For non-negative functions f and g, we denote $f(n) = \mathcal{O}(g(n))$ if there exist $n_0 \in \mathbb{N}$ and $c > 0$ such that $f(n) \leq cg(n)$ for $n \geq n_0$; $f(n) = \Theta(g(n))$ if $f(n) = \mathcal{O}(g(n))$ and $g(n) = \mathcal{O}(f(n))$. Moreover, we denote $f(n) = \Omega(g(n))$ if there exist $n_0 \in \mathbb{N}$ and $C > 0$ such that $f(n) \geq Cg(n)$ for $n \geq n_0$. For a binary vector $\mathbf{b}$, we denote by $\overline{\mathbf{b}}$ as the bit-wise complement vector of $\mathbf{b}$.

Next section formally defines the multilevel group testing problem.

II. Problem Statement

Multilevel group testing aims to identify K defective items out of total n items given the result of m tests. For each test, the result is the number of defective items in the testing pool, if it is less than or equal to L and it is L otherwise. Formally, let $\mathbf{x} \in \{0,1\}^n$ denote the set of n items in which $x_i = 1$ if the ith item is defective and $x_i = 0$ otherwise, for $i \in [n]$. A multilevel group testing problem can be represented by a measurement matrix $\mathbf{A} \in \{0,1\}^{m \times n}$ where the ith row of the matrix, $\mathbf{a}_i$, denotes the items contributing in the ith test. The results of the m tests can then be collected in vector $\mathbf{y} \in \mathcal{L}^m$ where $\mathcal{L} = \{0, 1, \cdots, L\}$ denotes the $(L+1)$ logical levels, i.e.

$$\mathbf{y} = \mathbf{A} \odot \mathbf{x} = \begin{bmatrix} \langle \mathbf{a}_1, \mathbf{x} \rangle_L \\ \vdots \\ \langle \mathbf{a}_m, \mathbf{x} \rangle_L \end{bmatrix}. \tag{1}$$

In this paper, $\langle \cdot, \cdot \rangle_L$ denotes the $(L+1)$-level boolean OR operation on two binary vectors, i.e. for two length-n vectors $\mathbf{a}_i$ and $\mathbf{x}$, we define

$$\langle \mathbf{a}_i, \mathbf{x} \rangle_L = \min\{\sum_{j=1}^{n} a_{ij} x_j, L\}. \tag{2}$$

Moreover, for two integers i, j, we define $(L+1)$-level boolean OR as $i \vee_L j = \min\{i + j, L\}$.

The goal in multilevel group testing problem is to design an efficient measurement matrix $\mathbf{A}$ that has a small number of tests, m, and recovers $\mathbf{x}$ given the test results $\mathbf{y}$ with low decoding complexity. Multilevel group testing problem reduces to the classical group testing problem when $L = 1$.

In [8], the authors propose Sparse-grAph codes Framework For gROup testiNg (SAFFRON) for classical group testing. In this work, we employ the same framework and similar techniques to leverage the additional information available to tackle the multilevel group testing problem. We provide a brief review of the SAFFRON algorithm in the next section.

III. Review of SAFFRON for group testing

The key idea of the SAFFRON algorithm is to design the measurement matrix based on a sparse bipartite graph, and to utilize a peeling-based iterative algorithm for decoding the set of defective items given the measurement vector.

More precisely, consider a bipartite graph consisting of n left nodes and M right nodes where each left node is associated with an item and each right node corresponds to a bundle of tests. Let $T_{\mathcal{G}} \in \{0,1\}^{M \times n}$ denote the adjacency matrix of the bipartite graph $\mathcal{G}(n, M)$. SAFFRON assigns h tests to every right node based on a signature matrix $\mathbf{U} \in \{0,1\}^{h \times n}$ as follows. Let $\mathbf{t}_i \in \{0,1\}^n$ denote the ith row of $T_{\mathcal{G}}$. The sensing matrix associated to the ith right node is defined as $\mathbf{A}_i = \mathbf{U}\,\mathrm{diag}(\mathbf{t}_i) \in \{0,1\}^{h \times n}$ and the overall sensing matrix is $\mathbf{A} = [\mathbf{A}_1^T, \cdots, \mathbf{A}_M^T]^T$. Therefore, the total number of tests is $m = M \times h$. The bipartite graph $\mathcal{G}(n, M)$ is designed as a left-regular graph, i.e. every left node is connected to d right nodes uniformly at random. The signature matrix in [8] is defined as

$$\mathbf{U} = \begin{bmatrix} \mathbf{U}_1 \\ \overline{\mathbf{U}}_1 \\ \mathbf{U}_2 \\ \overline{\mathbf{U}}_2 \\ \mathbf{U}_3 \\ \overline{\mathbf{U}}_3 \end{bmatrix} = \begin{bmatrix} \mathbf{b}_1 & \mathbf{b}_2 & \mathbf{b}_3 & \ldots & \mathbf{b}_{n-1} & \mathbf{b}_n \\ \overline{\mathbf{b}}_1 & \overline{\mathbf{b}}_2 & \overline{\mathbf{b}}_3 & \ldots & \overline{\mathbf{b}}_{n-1} & \overline{\mathbf{b}}_n \\ \mathbf{b}_{i_1} & \mathbf{b}_{i_2} & \mathbf{b}_{i_3} & \ldots & \mathbf{b}_{i_{n-1}} & \mathbf{b}_{i_n} \\ \overline{\mathbf{b}}_{i_1} & \overline{\mathbf{b}}_{i_2} & \overline{\mathbf{b}}_{i_3} & \ldots & \overline{\mathbf{b}}_{i_{n-1}} & \overline{\mathbf{b}}_{i_n} \\ \mathbf{b}_{j_1} & \mathbf{b}_{j_2} & \mathbf{b}_{j_3} & \ldots & \mathbf{b}_{j_{n-1}} & \mathbf{b}_{j_n} \\ \overline{\mathbf{b}}_{j_1} & \overline{\mathbf{b}}_{j_2} & \overline{\mathbf{b}}_{j_3} & \ldots & \overline{\mathbf{b}}_{j_{n-1}} & \overline{\mathbf{b}}_{j_n} \end{bmatrix}, \tag{3}$$

where $\mathbf{b}_i \in \{0,1\}^{\lceil \log n \rceil}$ is the binary representation of $i - 1$ and $(i_1, \ldots, i_n)$ and $(j_1, \ldots, j_n)$ are independent random variables, drawn uniformly at random from the set $[n]^n$.

A right node is called a *singleton* if it is connected to exactly one defective left node. Similarly, a *doubleton* is defined as a right node that is connected to exactly two defective left nodes. The proposed decoding algorithm in [8] is capable of identifying the identity of the defective items connected to singletons or resolvable doubletons. Moreover, SAFFRON declares a wrong defective item with probability no greater than $\frac{1}{n^2}$. The following theorem provides the main result of SAFFRON.

Theorem 1. *[8] With $m = 6C(\epsilon)K \log n$ tests, SAFFRON recovers at least $(1 - \epsilon)K$ defective items with probability $1 - \mathcal{O}(\frac{K}{n^2})$, where ϵ is an arbitrarily-close-to-zero constant and $C(\epsilon)$ is a constant that only depends on ϵ. Moreover, the decoding complexity of SAFFRON is $\mathcal{O}(K \log n)$ which is order-optimal.*

Since the fraction of non-recovered defective items of the iterative decoding algorithm corresponds to the error floor of the employed sparse-graph code, we call ϵ as the error floor of the algorithm. Table I summarizes the values of the constant $C(\epsilon)$ and optimal left-node degree d^* for different error floors.

ϵ	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}	10^{-8}	10^{-9}	10^{-10}
$C(\epsilon)$	6.13	7.88	9.63	11.36	13.10	14.84	16.57	18.30
d^*	7	9	10	12	14	15	17	19

TABLE I: Constant $C(\epsilon)$ vs. error floor ϵ.

IV. Main Results

In this section, we present our iterative algorithm that is built on SAFFRON along with a simple example to clarify the algorithm description. We further provide the details of the performance analysis of our algorithm.

A. Iterative Algorithm for Multilevel Group Testing

Design of the measurement matrix for multilevel group testing problem is essentially the same as the one in SAFFRON with the signature matrix defined in (3). However, the operations follow the rules described in (1) and (2), and the optimal sparse bipartite graph design depends on the level L. We design a d-left-regular bipartite graph $\mathcal{G}(n, M)$ with n left nodes and M right nodes as follows. For each left node, we connect a set of d right nodes uniformly at random. Now, let us generalize the notion of singletons and doubletons for multilevel group testing as follows.

Definition 1. A right node of the bipartite graph is called an ℓ-ton if it is connected to exactly ℓ defective items.

An ℓ-ton right node is said to be resolvable if $\ell - 1$ of the connected defective items are already identified, and $\ell \leq 2L$ as will be shown in Lemma 1. Our algorithm iterates through the all resolvable right nodes using the signature matrix introduced in (3) and boolean operation defined in (1), i.e. it first detects and resolves all the singletons and continues searching and resolving all the ℓ-tons till it can not recover new right nodes. The following lemma characterizes that which ℓ-tons are resolvable.

Lemma 1. *Our algorithm can recover resolvable ℓ-tons if $\ell \leq 2L$. Moreover, for ℓ-tons where $\ell \geq 2L+1$, the algorithm detects a wrong defective item with probability $\mathcal{O}(\frac{1}{n^2})$.*

Proof: (Sketch) Without loss of generality suppose that the ℓ-ton consists of defective items $k_1, k_2, \ldots, k_\ell$ and only k_ℓ has not been identified. The first part of the measurements corresponding to this right node is then

$$\begin{bmatrix} \mathbf{b}_{k_1} \vee_L \mathbf{b}_{k_2} \cdots \vee_L \mathbf{b}_{k_\ell} \\ \bar{\mathbf{b}}_{k_1} \vee_L \bar{\mathbf{b}}_{k_2} \cdots \vee_L \bar{\mathbf{b}}_{k_\ell} \end{bmatrix}$$

Now we show that one can recover $\mathbf{b}_{k_\ell}$. Consider the first entry of this vector $b_{k_\ell,1}$. Then, if $\ell \leq 2L$, either $\sum_{i=1}^{\ell-1} b_{k_i,1} < L$ or $\sum_{i=1}^{\ell-1} \bar{b}_{k_i,1} < L$, which implies that $b_{k_\ell,1}$ can be recovered just by solving linear equation for one of the mentioned measurements. Note that if the sum of the binary numbers in an entry of the above vector is smaller than or equal to L, then the operator $\vee_L$ is the sames as regular addition operation, which implies that $b_{k_\ell,1}$ can be recovered. Similarly, other entries of $\mathbf{b}_{k_\ell}$ can be found.

Moreover, if $\ell \geq 2L+1$, similar to the argument for the SAFFRON algorithm, since the decoder finds the location of the wrongly detected defective item for two random permutations as well, the chance that these indices match is at most $1/n^2$. More precisely, suppose that the decoder finds k_1 to be the location of the recovered defective item in the ℓ-ton. The decoder repeats this procedure with signature matrices $\mathbf{U}_2$ and $\mathbf{U}_3$, and obtains k_2 and k_3 as the permuted location of defective item. Then, the probability that $k_2 = i_{k_1}$ and $k_3 = j_{k_1}$ is $\mathcal{O}(1/n^2)$. ■

Given the identities of the recovered left nodes in the first iteration of the decoding procedure, in the second iteration, more edges are peeled off the graph so that new left nodes can be recovered. The algorithm terminates when no more left nodes can be recovered.

The following example summarizes the steps in our algorithm, from signature matrix design to peeling decoding procedure.

Example 1. Consider a pool of $n = 8$ items where $K = 3$ of them are defective and $\mathbf{x} = (1,1,0,0,0,0,0,1)$. In other words, the defective items are items 1, 2 and 8. We assume that $L = 2$. Consider a bipartite graph represented by the matrix

$$T_{\mathcal{G}} = \begin{bmatrix} 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 & 1 \end{bmatrix}.$$

Let $\mathbf{s_1} = (i_1, \cdots, i_8) = (4,2,1,3,7,8,6,5)$ and $\mathbf{s_2} = (j_1, \cdots, j_8) = (1,3,4,8,7,6,5,2)$ be two permutations of the integers in $[n] = \{1, \cdots, 8\}$ drawn uniformly at random. Therefore, the signature matrix is given by

$$\mathbf{U} = \begin{bmatrix} \mathbf{U}_1 \\ \overline{\mathbf{U}}_1 \\ \mathbf{U}_2 \\ \overline{\mathbf{U}}_2 \\ \mathbf{U}_3 \\ \overline{\mathbf{U}}_3 \end{bmatrix} = \begin{bmatrix} 0&0&0&0&1&1&1&1 \\ 0&0&1&1&0&0&1&1 \\ 0&1&0&1&0&1&0&1 \\ 1&1&1&1&0&0&0&0 \\ 1&1&0&0&1&1&0&0 \\ 1&0&1&0&1&0&1&0 \\ 0&0&0&0&1&1&1&1 \\ 1&0&0&1&1&1&0&0 \\ 1&1&0&0&0&1&1&0 \\ 1&1&1&1&0&0&0&0 \\ 0&1&1&0&0&0&1&1 \\ 0&0&1&1&1&0&0&1 \\ 0&0&0&1&1&1&1&0 \\ 0&1&1&1&1&0&0&0 \\ 0&0&1&1&0&1&0&1 \\ 1&1&1&0&0&0&0&1 \\ 1&0&0&0&0&1&1&1 \\ 1&1&0&0&1&0&1&0 \end{bmatrix} = [\mathbf{u_1}, \cdots, \mathbf{u_8}].$$

According to the operator described in (1), the measurement vector is

$$\mathbf{y} = \begin{bmatrix} \mathbf{z_1} \\ \mathbf{z_2} \\ \mathbf{z_3} \end{bmatrix} = \begin{bmatrix} \mathbf{u_1} \\ \mathbf{u_1} \vee_2 \mathbf{u_2} \\ \mathbf{u_1} \vee_2 \mathbf{u_2} \vee_2 \mathbf{u_8} \end{bmatrix},$$

where the right node measurements are

$$\begin{aligned} \mathbf{z_1} &= (0,0,0,1,1,1,0,1,1,1,0,0,0,0,0,1,1,1)^T, \\ \mathbf{z_2} &= (0,0,1,2,2,1,0,1,2,2,1,0,0,1,0,2,1,2)^T, \\ \mathbf{z_3} &= (1,1,2,2,2,1,1,1,2,2,2,1,0,1,1,2,2,2)^T. \end{aligned}$$

We first detect and solve all singletons by checking whether the weight of the right node measurement is $3\log n = 9$. Therefore, $\mathbf{z_1}$ is singleton and item 1 is detected as defective, since $w(\mathbf{z_1}) = 9$. In the second iteration, the algorithm searches for resolvable doubletons. Note that both $\mathbf{z_2}$ and $\mathbf{z_3}$ can be doubletons. We hypothesize that $\mathbf{z_2}$ is a doubleton containing defective item 1, i.e. we write $\mathbf{z_2} = \mathbf{u_1} \vee_2 \mathbf{u}_{\ell_1}$ and recover ℓ_1, ℓ_2 and ℓ_3. Then, $\ell_1 = 2$, $\ell_2 = 2$ and $\ell_3 = 3$. Since $i_{\ell_1} = i_2 = 2$ and $j_{\ell_1} = j_2 = 3$, we conclude that $\mathbf{z_2}$ is doubleton and declare item 2 as defective. The classical group testing algorithm terminates after finding the doubletons, however, we continue searching for possible 3-tons and so on. We guess $\mathbf{z_3} = \mathbf{u_1} \vee_2 \mathbf{u_2} \vee_2 \mathbf{u}_{\ell_1}$ and recover $\ell_1 = 8$, $\ell_2 = 5$ and $\ell_3 = 2$. We observe that $\ell_2 = i_{\ell_1}$ and $\ell_3 = j_{\ell_1}$, so we declare the item 8 as defective and finish the algorithm since there are no more defective items to be found.

B. Analysis

The main theorem of our algorithm characterizes the precise number of tests required to guarantee a specific fraction of recovered defective items.

Theorem 2. *With $m = 6C(\epsilon, L)K\log n$ tests, our algorithm recovers at least $(1-\epsilon)K$ defective items with probability $1 - \mathcal{O}(\frac{K}{n^2})$, where ϵ is an arbitrarily-close-to-zero constant and $C(\epsilon, L)$ is a constant that only depends on ϵ and L.*

Proof: Since our algorithm designs the same signature matrix as SAFFRON, the proof will follow the same steps as in [8] except for the constant term $C(\epsilon, L)$ which now depends on both the target reliability ϵ and the number of logical levels L.

We use density evolution technique [20] to analyze the performance of the decoding algorithm. As described in the design approach of the measurement matrix, we design a d-left-regular bipartite graph with n left nodes and M right nodes as follows. For each left node, a set of d right nodes are connected to it uniformly at random. We restrict the graph to the bipartite subgraph constructed by the right nodes connected with the set of defective items, i.e, K left nodes (pruned subgraph). Obviously, the average right degree is $\lambda = \frac{Kd}{M}$. For large enough K, the right node degree distribution approaches the Poisson distribution with parameter λ. Let $\rho(x) = \sum_{i=1}^{\infty} \rho_i x^{i-1}$ be the right edge degree distribution, where ρ_i denotes the probability that a randomly picked edge in graph is connected to a right node of degree i. Consdiering the Poisson distribution, we will have

$$\rho_i = \frac{iM}{Kd} e^{-\lambda} \frac{\lambda^i}{i!} = e^{-\lambda} \frac{\lambda^{i-1}}{(i-1)!}. \quad (4)$$

Therefore,

$$\rho(x) = \sum_{i=1}^{\infty} \rho_i x^{i-1} = e^{-\lambda} \sum_{i=1}^{\infty} \frac{\lambda^{i-1}}{(i-1)!} x^{i-1} = e^{-\lambda(1-x)}. \quad (5)$$

As described in the decoding procedure, the first phase detects and resolves all the singletons. Given the identities of resolved defective left nodes from the first phase, the second phase iteratively looks for new unresolved defective left nodes connected to ℓ-tons for $\ell \leq 2L$.

Next, we use density evolution to analyze the fraction of defective items that cannot be recovered at the end of each iteration of the algorithm. Let p_j be the probability that a random defective item is not recovered at iteration j of the algorithm. At iteration $j+1$, a left node v is not recovered by a connected right node c if every neighbour of v is not a resolvable ℓ-ton for every $\ell \leq 2L$. The probability that a specific node has been solved at the iteration j is $\rho_1 + \rho_2(1-p_j) + \cdots + \rho_{2L}(1-p_j)^{2L-1}$. Therefore, density evolution can be written as

$$\begin{aligned} p_{j+1} &= \Pr(\text{not resolvable from one children right node})^{d-1} \\ &= \Big(1 - \sum_{i=1}^{2L} \rho_i (1-p_j)^{i-1}\Big)^{d-1}, \end{aligned} \quad (6)$$

where $\rho_i = e^{-\lambda} \frac{\lambda^{i-1}}{(i-1)!}$.

By analyzing the recursive equation, we can find the error floor as follows.

$$\epsilon = \Big(1 - \sum_{i=1}^{2L} e^{\lambda} \frac{\lambda^{i-1}}{(i-1)!}\Big)^{d-1}. \quad (7)$$

The density evolution curve is illustrated in Figure 1. Now, we simply design a pair (d, M) in order to minimize the number of right nodes M given a fixed error floor ϵ. More precisely, we can write the following optimization problem over λ and d:

$$\min_{\substack{\lambda>0 \\ d\in\mathbb{N}}} \quad M = \frac{Kd}{\lambda} \quad (8)$$

$$\text{subject to} \quad (d-1)\log\Big(1 - \sum_{i=1}^{2L} e^{\lambda} \frac{\lambda^{i-1}}{(i-1)!}\Big) = \log(\epsilon) \quad (9)$$

$$1 - \sum_{i=1}^{2L} e^{\lambda} \frac{\lambda^{i-1}}{(i-1)!} > \lambda. \quad (10)$$

The last constraint (10) is to guarantee that the fixed point of the density evolution equation is unique. We numerically solve the optimization problem. Let (λ^*, d^*) be the optimal pair for the optimization problem. We define the constant $C(\epsilon, L) = \frac{d^*}{\lambda^*}$. ■

We present tables of constants $C(\epsilon, L)$ given the target error floor ϵ according to the number of logical levels $L \in \{2, 3\}$ in Tables II and III. As expected, the table corresponding to $L = 1$ is identical to Table I.

ϵ	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}	10^{-8}	10^{-9}	10^{-10}
$C(\epsilon, L)$	2.30	3.01	3.84	4.05	4.60	5.50	5.85	6.73
d^*	4	4	4	6	7	6	12	7

TABLE II: Constant $C(\epsilon, L)$ vs. error floor ϵ for $L = 2$.

ϵ	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}	10^{-8}	10^{-9}	10^{-10}
$C(\epsilon, L)$	1.29	1.56	1.86	2.15	2.54	3.1	3.3	3.5
d^*	3	4	5	6	8	4	6	11

TABLE III: Constant $C(\epsilon, L)$ vs. error floor ϵ for $L = 3$.

Notice that our strategy is consistent, i.e. we can increase the cardinality of the set of logical levels $\mathcal{L}$ to obtain an arbitrarily small p_j. Formally, we can say $p_j \to 0$ as $L \to \infty$. Firstly, we can rewrite the expression (6) by substituting $\rho_i = e^{-\lambda} \frac{\lambda^{i-1}}{(i-1)!}$:

$$p_{j+1} = \Big(1 - e^{-\lambda} \sum_{i=1}^{2L} \frac{\lambda^{i-1}}{(i-1)!} (1-p_j)^{i-1}\Big)^{d-1}. \quad (11)$$

Letting $L \to \infty$, we will have

$$p_{j+1} = f(p_j) = \Big(1 - e^{-\lambda p_j}\Big)^{d-1}. \quad (12)$$

We can simply observe that $p_j = 0$ implies $p_{j+1} = f(p_j) = 0$. Therefore, $p = 0$ is the fixed point of function f and it is unique due to the last constraint condition (10).

V. Simulation Results

According to Theorem 2, we can recover at least $(1-\epsilon)K$ defective items with high probability and computational complexity $\mathcal{O}(K \log n)$. We simulate the algorithm for different values of (M, d, L) for $K = 100$ defective items in a pool of $n = 2^{16}$ items, where $L \in \{1, 2\}$, $K \leq M \leq 7K$ and $d = \{3, 5, 7, 9\}$. The result is shown in Figure 2. We observe that for $L = 2$, the algorithm recovers almost all the defective items with only $M = 3K$ right nodes.

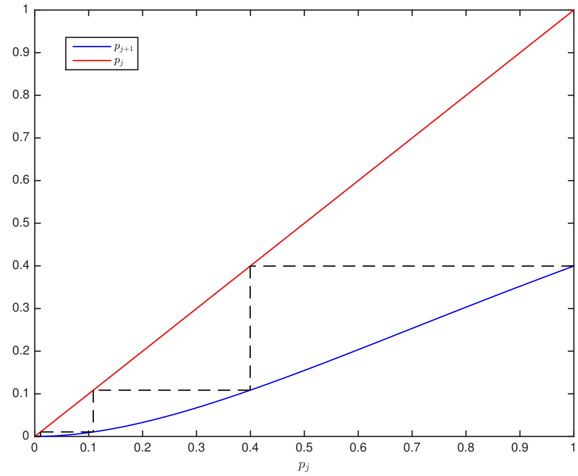

Fig. 1: Density evolution in (11) for $L = 4$ and $\lambda = 1$.

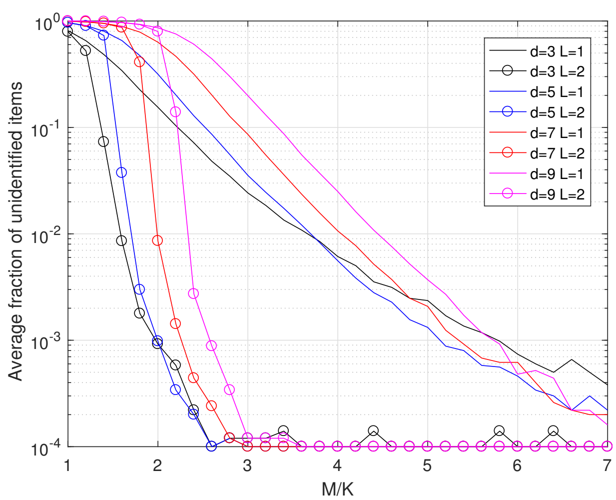

Fig. 2: The average fraction of unidentified defective items obtained via simulations.

VI. Conclusion

In this work, we presented a fast and efficient algorithm for multilevel group testing and provided a recovery algorithm with small number of tests and low computational cost. We also presented the analysis of the algorithm to describe our method and characterized the error floor, along with numerical results to support our work. It is worth mentioning that our algorithm can be robustified to noise using the same strategy as in SAFFRON [8].

References

[1] D. Du, F. K. Hwang, and F. Hwang, *Combinatorial group testing and its applications*. World Scientific, 2000, vol. 12.

[2] H.-B. Chen and F. K. Hwang, "A survey on nonadaptive group testing algorithms through the angle of decoding," *Journal of Combinatorial Optimization*, vol. 15, no. 1, pp. 49–59, 2008.

[3] D. Malioutov and K. Varshney, "Exact rule learning via boolean compressed sensing," in *International Conference on Machine Learning*, 2013, pp. 765–773.

[4] M. T. Goodrich, M. J. Atallah, and R. Tamassia, "Indexing information for data forensics," in *ACNS*, vol. 5. Springer, 2005, pp. 206–221.

[5] A. C. Gilbert, M. A. Iwen, and M. J. Strauss, "Group testing and sparse signal recovery," in *Signals, Systems and Computers, 2008 42nd Asilomar Conference on*. IEEE, 2008, pp. 1059–1063.

[6] A. Emad and O. Milenkovic, "Poisson group testing: A probabilistic model for nonadaptive streaming boolean compressed sensing," in *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*. IEEE, 2014, pp. 3335–3339.

[7] K. Chandrasekher, K. Lee, P. Kairouz, R. Pedarsani, and K. Ramchandran, "Asynchronous and noncoherent neighbor discovery for the iot using sparse-graph codes," in *Communications (ICC), 2017 IEEE International Conference on*. IEEE, 2017, pp. 1–6.

[8] K. Lee, R. Pedarsani, and K. Ramchandran, "Saffron: A fast, efficient, and robust framework for group testing based on sparse-graph codes," in *Information Theory (ISIT), 2016 IEEE International Symposium on*. IEEE, 2016, pp. 2873–2877.

[9] H. Q. Ngo and D.-Z. Du, "A survey on combinatorial group testing algorithms with applications to dna library screening," *Discrete mathematical problems with medical applications*, vol. 55, pp. 171–182, 2000.

[10] A. Dyachkov, V. Rykov, and A. Rashad, "Superimposed distance codes," *Problems of Control and Information Theory-problemy Upravleniya i Teorii Informatsii*, vol. 18, no. 4, pp. 237–250, 1989.

[11] P. Indyk, H. Q. Ngo, and A. Rudra, "Efficiently decodable non-adaptive group testing," in *Proceedings of the twenty-first annual ACM-SIAM symposium on Discrete Algorithms*. SIAM, 2010, pp. 1126–1142.

[12] A. Mazumdar, "Nonadaptive group testing with random set of defectives," *IEEE Transactions on Information Theory*, vol. 62, no. 12, pp. 7522–7531, 2016.

[13] G. K. Atia and V. Saligrama, "Boolean compressed sensing and noisy group testing," *IEEE Transactions on Information Theory*, vol. 58, no. 3, pp. 1880–1901, 2012.

[14] C. L. Chan, S. Jaggi, V. Saligrama, and S. Agnihotri, "Non-adaptive group testing: Explicit bounds and novel algorithms," *IEEE Transactions on Information Theory*, vol. 60, no. 5, pp. 3019–3035, 2014.

[15] T. Wadayama, "An analysis on non-adaptive group testing based on sparse pooling graphs," in *Information Theory Proceedings (ISIT), 2013 IEEE International Symposium on*. IEEE, 2013, pp. 2681–2685.

[16] S. Cai, M. Jahangoshahi, M. Bakshi, and S. Jaggi, "Grotesque: noisy group testing (quick and efficient)," in *Communication, Control, and Computing (Allerton), 2013 51st Annual Allerton Conference on*. IEEE, 2013, pp. 1234–1241.

[17] A. Vem, N. T. Janakiraman, and K. R. Narayanan, "Group testing using left-and-right-regular sparse-graph codes," *arXiv preprint arXiv:1701.07477*, 2017.

[18] T. V. Bui, M. Kuribayashi, and I. Echizen, "Non-adaptive group testing framework based on concatenation code," *arXiv preprint arXiv:1701.06989*, 2017.

[19] A. Emad and O. Milenkovic, "Semi-quantitative group testing," in *Information Theory Proceedings (ISIT), 2012 IEEE International Symposium on*. IEEE, 2012, pp. 1847–1851.

[20] T. Richardson and R. Urbanke, *Modern coding theory*. Cambridge university press, 2008.