SAFFRON: A Fast, Efficient, and Robust Framework for Group Testing based on Sparse-Graph Codes

Kangwook Lee, Ramtin Pedarsani, and Kannan Ramchandran
Dept. of Electrical Engineering and Computer Sciences, University of California, Berkeley
{kw1jjang, ramtin, kannanr}@eecs.berkeley.edu

***Abstract*—The group testing problem is to identify a population of K defective items from a set of n items by *pooling* groups of items. The result of a test for a group of items is *positive* if any of the items in the group is defective and *negative* otherwise. The goal is to judiciously group subsets of items such that defective items can be reliably recovered using the minimum number of tests, while also having a low-complexity decoder.**

We describe SAFFRON (Sparse-grAph codes Framework For gROup testiNg), a *non-adaptive* group testing scheme that recovers at least a $(1-\epsilon)$-fraction (for any arbitrarily small $\epsilon > 0$) of K defective items with high probability with $m = 6C(\epsilon)K\log_2 n$ tests, where $C(\epsilon)$ is a precisely characterized constant that depends only on ϵ. For instance, it can provably recover at least $(1-10^{-6})K$ defective items with $m \simeq 68K\log_2 n$ tests. The computational complexity of the decoding algorithm is $\mathcal{O}(K\log n)$, which is order-optimal. Further, we describe a systematic methodology to robustify SAFFRON such that it can reliably recover the set of K defective items even in the presence of erroneous or *noisy* test results. We also propose Singleton-Only-SAFFRON, a variant of SAFFRON, that recovers *all* the K defective items with $m = 2e(1+\alpha)K\log K\log_2 n$ tests with probability $1-\mathcal{O}\big(\frac{1}{K^\alpha}\big)$, where $\alpha > 0$ is a constant. Our key intellectual contribution involves the pioneering use of powerful density-evolution methods of modern coding theory (e.g. sparse-graph codes) for efficient group testing design and performance analysis.

I. Introduction

The group testing problem is to identify a population of K defective items from a set of n items by *pooling* groups of items efficiently in order to cut down the number of tests needed. Group testing was developed during the Second World War in order to detect the soldiers infected with syphilis virus without needing to test them individually [1]. Since then, varied theoretical aspects of group testing have been studied, and more applications of group testing have been discovered in a variety of fields spanning across biology [2], machine learning [3], medicine [4], computer science [5], data analysis [6], and signal processing [7].

Main contributions. Despite the long history of group testing, our paper has some novel intellectual and practical contributions to the field of large-scale probabilistic group testing. More precisely, our goal is to design a non-adaptive group testing which allows a *computationally efficient decoding algorithm, i.e. sublinear in n, when K is sublinear in n*, and a targeted arbitrarily-small fraction of defective items can be missed. In this work, we introduce SAFFRON (Sparse-grAph codes Framework For gROup testiNg), a powerful framework for non-adaptive group testing based on sparse-graph codes [8]. Our key intellectual contribution involves the pioneering use of powerful density-evolution methods of modern coding theory (e.g. sparse-graph codes) for efficient group testing design and performance analysis. Sparse-graph codes (e.g. Low-Density-Parity-Check (LDPC) codes [8]) form the backbone of reliable modern communication systems. However, pooling test design for our targeted group testing problem seems quite different from code design for the noisy communication problem since group testing deals with the boolean OR operator, and its non-linearity complicates the use of classical coding theory. In this paper, we show how to overcome this challenge with the use of elegant density evolution methods and simple randomized sparse-graph coding designs. Further, we are able to specify *precise constants* in the number of tests needed while simultaneously having provable performance guarantees and order-optimal decoding complexity in the large-scale probabilistic group testing setting. Finally, we show how one can systematically robustify SAFFRON by making use of efficient error-correcting codes.

Problem definition. We formally define the (non-adaptive) group testing problem as follows. There are n items, and exactly K items among them are defective. We define the support vector $\mathbf{x} \in \{0,1\}^n$, of which the i^{th} component is 1 if and only if item i is defective, and define $\text{supp}(\cdot)$ as the set of indices of non-zero elements. A subset of items can be pooled and tested, and the test result is either 1 (positive) if any of the items in the subset is defective, or 0 (negative) otherwise. For notational simplicity, we denote a subset by a binary row vector $\mathbf{a}$, of which the i^{th} component is 1 if and only if item i belongs to the subset. Then, a group testing result y can be expressed as $y = \langle \mathbf{a}, \mathbf{x} \rangle \stackrel{\text{def}}{=} \bigvee_{i=1}^{n} a_i x_i$, where $\vee$ is a boolean OR operator. Let m be the number of pools. Denoting the i^{th} pool by $\mathbf{a}_i$, we define *the group testing matrix* as $A \stackrel{\text{def}}{=} (\mathbf{a}_1^\intercal, \mathbf{a}_2^\intercal, \ldots, \mathbf{a}_m^\intercal)^\intercal \in \{0,1\}^{m\times n}$. The group testing results from m pools can also be represented as a column vector $\mathbf{y} = (y_1, y_2, \ldots, y_m)^\intercal \in \{0,1\}^m$, where y_i is the outcome of the i^{th} test. Then,

$$\mathbf{y} = A \odot \mathbf{x} \stackrel{\text{def}}{=} (\langle \mathbf{a}_1, \mathbf{x}\rangle, \langle \mathbf{a}_2, \mathbf{x}\rangle, \ldots, \langle \mathbf{a}_m, \mathbf{x}\rangle)^\intercal . \quad (1)$$

The goal is to design the group testing matrix A and efficiently recover the set of defective items using the m (potentially noisy) test results. Denoting the decoding function by $g_A \colon \{0,1\}^m \to \{0,1\}^n$, we want the decoding result $\hat{\mathbf{x}} = g_A(y)$ to satisfy that $\text{supp}(\hat{\mathbf{x}}) \subseteq \text{supp}(\mathbf{x})$ and $|\text{supp}(\hat{\mathbf{x}})| \geq (1-\epsilon)K$ with high probability.

Related Works. We provide a brief survey of the existing results in the literature. We refer the readers to [2],

[9]–[11] for a detailed survey. The minimum number of tests required to solve a group testing problem has been extensively studied in the literature. For the zero-error case, the best known lower bound on the number of required tests is $\Omega(\frac{K^2}{\log K}\log n)$ [12]. The best known group testing scheme under this setup requires $\mathcal{O}(K^2 \log n)$ tests [1]. A recent work [13] presents the first scheme that requires $\mathcal{O}(K^2 \log n)$ tests, while having a computationally efficient decoding algorithm.

Another line of work is based on an information-theoretic formulation of the group testing problem. That is, one assumes a prior distribution on the set of defective items, and searches for a group testing scheme with vanishing error probability [11], [14]–[16]. The proposed group testing schemes in [15] achieve an order-optimal number of tests with a decoding complexity of $\mathcal{O}(nK \log n)$, which is superlinear in n. In [16], the author considers a group testing matrix constructed based on a left-regular and right-regular bipartite graph. Using the analysis of LDPC codes, the author presents information-theoretic lower bounds and analyzes performance of a typicality-based decoding algorithm. Though theoretically interesting, the complexity of the proposed decoding algorithm is prohibitive in practice. One of the most notable exceptions is [17]. In this work, Cai et al. propose GROTESQUE for non-adaptive group testing. The scheme requires $\mathcal{O}(K \log K \log n)$ tests, and its decoding complexity is $\mathcal{O}(K(\log n + \log^2 K))$, which is sublinear in n. Moreover, the authors propose the idea of using expander codes for noisy group testing. Our Singleton-Only-SAFFRON can be viewed as a strict improvement over the non-adaptive GROTESQUE as Singleton-Only-SAFFRON requires a significantly smaller number of tests due to its efficient deterministic signature matrix. For noisy group testing, we use capacity-achieving codes to further reduce the number of tests.

II. The SAFFRON Scheme

The key idea of SAFFRON is the adoption of a design principle called '*sparse signal recovery via sparse-graph codes*' that is applicable to a varied class of problems such as computing a sparse Fast Fourier Transform, a sparse Walsh Hadamard Transform, and the design of measurement systems for compressive sensing and compressive phase-retrieval [18]–[20]. In all these problems, one designs an efficient way of sensing or *measuring* an unknown sparse signal such that the decoder can estimate the unknown signal with low decoding complexity. The overarching design principle is 1) to design a sensing matrix based on a sparse bipartite graph and 2) to decode the observed measurements using a simple peeling-like iterative algorithm. In this paper, we show how this same design principle allows us to tackle the group testing problem.

Consider a bipartite graph with n left nodes and M right nodes. Here, the n left nodes correspond to the n items, and the M right nodes correspond to the M bundles of test results. We design a bipartite graph based on left-regular construction. That is, each left node is connected to constant number d of right nodes uniformly at random. We denote the incidence matrix of a bipartite graph $\mathcal{G}$ by $T_{\mathcal{G}} \in \{0,1\}^{M \times n}$, or simply T if $\mathcal{G}$ is clear from the context. Let $\mathbf{t}_i$ be the ith row of $T_{\mathcal{G}}$. We associate each left node with a carefully designed *signature* (column) vector $\mathbf{u}$ of length h, i.e., $\mathbf{u} \in \{0,1\}^h$. Let us denote the signature vector of item i by $\mathbf{u}_i$. We define the signature matrix $U \stackrel{\text{def}}{=} [\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_{n-1}, \mathbf{u}_n] \in \{0,1\}^{h \times n}$.

Given a graph $\mathcal{G}$ and a signature matrix U, we design our group testing matrix A to be a *row tensor product* of $T_{\mathcal{G}}$ and U, which is defined as $A = T_{\mathcal{G}} \otimes U \stackrel{\text{def}}{=} [A_1^\intercal, A_2^\intercal, \ldots, A_M^\intercal]^\intercal \in \{0,1\}^{hM \times n}$, where $A_i = U\text{diag}(\mathbf{t}_i) \in \{0,1\}^{h \times n}$, and $\text{diag}(\cdot)$ is the diagonal matrix constructed by the input vector. For example,

$$T = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 1 & 0 \end{bmatrix}, U = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}, T \otimes U = \left[\begin{array}{ccc} 0 & \mathbf{0} & \mathbf{1} \\ 0 & \mathbf{1} & \mathbf{1} \\ \hline \mathbf{1} & \mathbf{0} & 0 \\ \mathbf{0} & \mathbf{1} & 0 \end{array}\right].$$

For notational simplicity, we define the observation vector corresponding to right node i as $\mathbf{z}_i \stackrel{\text{def}}{=} \mathbf{y}_{(i-1)h+1:ih}$ or $\mathbf{z}_i = A_i \odot \mathbf{x}$ for $1 \leq i \leq M$. In other words, $\mathbf{z}_i$ is the bitwise logical ORing of all the signature vectors of the active left nodes that are connected to right node i.

Our decoding algorithm simply iterates through all the right node measurement vectors $\{\mathbf{z}_i\}_{i=1}^M$, and checks whether a right node is *resolvable* or not. A right node is resolvable if exactly one new defective item can be detected by processing the right node, i.e., the location index of the defective item is found. The decoding algorithm is terminated when there are no more resolvable right nodes.

We now present the following terminologies. A right node that is connected to one defective item is called a *singleton*. A right node that is connected to two defective items is called a *doubleton*. We now show that with the aid of our signature matrix, 1) a singleton is always resolvable, and 2) a doubleton is resolvable if one of the two defective items is already identified (in the previous iterations of the algorithm).

Detecting and Resolving a Singleton. Consider the following signature matrix where the i^{th} column is a vertical concatenation of $\mathbf{b}_i$ and $\overline{\mathbf{b}_i}$ (bit-wise complement of $\mathbf{b}_i$), where $\mathbf{b}_i$ is the L-bits binary representation of an integer $i-1$, for $i \in [n]$, so $L = \lceil \log_2 n \rceil$ [1], where $[n] \stackrel{\text{def}}{=} \{1, 2, \ldots, n\}$.

$$\begin{bmatrix} U_1 \\ \overline{U_1} \end{bmatrix} = \begin{bmatrix} \mathbf{b}_1 & \mathbf{b}_2 & \ldots & \mathbf{b}_{n-1} & \mathbf{b}_n \\ \overline{\mathbf{b}_1} & \overline{\mathbf{b}_2} & \ldots & \overline{\mathbf{b}_{n-1}} & \overline{\mathbf{b}_n} \end{bmatrix} \tag{2}$$

We now show that a singleton can be detected and resolved with the aid of this signature matrix. First, note that the sum of the weight of any binary vector and the weight of its complement is always the length of the vector, L. Thus, given a singleton, the weight of the measurement vector is L. Furthermore, if the right node is connected to zero or more than one defective items, the weight of the measurement vector *will not be* L. Therefore, by just checking the weight of the right-node measurement vector, one can simply detect whether the right node is a singleton or not. Further, one can also read the first half of the measurement of the detected singleton in order to find the index location of the defective item.

Detecting and Resolving a Doubleton. We now design the full signature matrix U that allows us to detect and resolve

[1]For simplicity, the rest of the paper will assume that n is a power of 2, and hence $L = \log_2 n$.

doubletons as well:

$$U = \begin{bmatrix} U_1 \\ \overline{U}_1 \\ U_2 \\ \overline{U}_2 \\ U_3 \\ \overline{U}_3 \end{bmatrix} = \begin{bmatrix} \mathbf{b}_1 & \mathbf{b}_2 & \dots & \mathbf{b}_{n-1} & \mathbf{b}_n \\ \overline{\mathbf{b}_1} & \overline{\mathbf{b}_2} & \dots & \overline{\mathbf{b}_{n-1}} & \overline{\mathbf{b}_n} \\ \mathbf{b}_{i_1} & \mathbf{b}_{i_2} & \dots & \mathbf{b}_{i_{n-1}} & \mathbf{b}_{i_n} \\ \overline{\mathbf{b}_{i_1}} & \overline{\mathbf{b}_{i_2}} & \dots & \overline{\mathbf{b}_{i_{n-1}}} & \overline{\mathbf{b}_{i_n}} \\ \mathbf{b}_{j_1} & \mathbf{b}_{j_2} & \dots & \mathbf{b}_{j_{n-1}} & \mathbf{b}_{j_n} \\ \overline{\mathbf{b}_{j_1}} & \overline{\mathbf{b}_{j_2}} & \dots & \overline{\mathbf{b}_{j_{n-1}}} & \overline{\mathbf{b}_{j_n}} \end{bmatrix}, \quad (3)$$

where $\mathbf{s}_1 = (i_1, i_2, \dots, i_n)$ and $\mathbf{s}_2 = (j_1, j_2, \dots, j_n)$ are drawn uniformly at random from the set $[n]^n$. We define $\mathbf{z}_k^w$ to be the w^{th} section of the k^{th} right-node measurement vector or $\mathbf{z}_k^w \stackrel{\text{def}}{=} \mathbf{z}_{k,(w-1)L+1:wL}$ for $w \in \{1, 2, \dots, 6\}$. That is, each right node's measurement vector has 6 sections of length L. We now show that a doubleton can be detected and resolved as follows. Assume that right node k is connected to exactly one identified defective item, say ℓ_0. The decoder first *guesses* that the right node is a resolvable doubleton. That is, the right node is connected to exactly two defective items: one of them is the identified defective item ℓ_0, and the other is the unidentified defective item ℓ_1 (to be found). Then,

$$\begin{bmatrix} \mathbf{z}_k^1 \\ \mathbf{z}_k^2 \end{bmatrix} = \mathbf{u}_{\ell_0} \vee \mathbf{u}_{\ell_1} = \begin{bmatrix} \mathbf{b}_{\ell_0} \vee \mathbf{b}_{\ell_1} \\ \overline{\mathbf{b}_{\ell_0}} \vee \overline{\mathbf{b}_{\ell_1}} \end{bmatrix}. \quad (4)$$

Assuming the above structure, one can always recover $\mathbf{b}_{\ell_1}$ as follows. Consider the first bit of $\mathbf{b}_{\ell_1}$. If $\mathbf{b}_{\ell_0,1} = 0$, then $\mathbf{b}_{\ell_1,1} = \mathbf{z}_{k,1}^1$. If not, one can read the first bit from the second section and invert it, i.e., $\mathbf{b}_{\ell_1,1} = \overline{\mathbf{z}_{k,1}^2}$. Hence, the decoder is able to recover the other defective item's index, ℓ_1. Similarly, the decoder applies this recovery procedure to the other four sections from $\mathbf{z}_k^3$ to $\mathbf{z}_k^6$, and obtain two other indices, ℓ_2 and ℓ_3. Finally, the decoder *checks* whether $\ell_2 = i_{\ell_1}$ *and* $\ell_3 = j_{\ell_1}$. If the above condition does not hold, the decoder concludes that the guess is wrong. If it holds, the decoder affirms the hypothesis, declares the right node to be resolved, and declares a new defective item of index ℓ_1. Indeed, the probability of this simple *guess-and-check* procedure declaring a false doubleton is no greater than $\frac{1}{n^2}$, and the proof can be found in [9].

Example. In this section, we provide an illustrative example of the decoding algorithm of SAFFRON. Consider a group testing problem with $n = 8$ items and $K = 3$ defective items. Let $\mathbf{x} = (1, 0, 1, 0, 0, 0, 0, 1)$, i.e., item 1, item 3 and item 8 are defective items. We show how SAFFRON can find the set of defective items. Recall that we design our group testing matrix A to be a row tensor product of $T_{\mathcal{G}}$ and U. Assume that the bipartite graph $\mathcal{G}$ is designed as follows [2].

$$T_{\mathcal{G}} = \begin{bmatrix} 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 \end{bmatrix} \in \{0,1\}^{M \times n}$$

We have $M = 4$ right nodes, and $n = 8$ items are connected to them according to $T_{\mathcal{G}}$. Assume that the randomly drawn sequences $\mathbf{s}_1$ and $\mathbf{s}_2$ are as follows: $\mathbf{s}_1 = (5, 2, 4, 8, 7, 1, 3, 6)$, $\mathbf{s}_2 = (3, 1, 5, 6, 3, 8, 2, 7)$. Thus, the measurement matrix of SAFFRON is as follows.

$$U = [\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3, \mathbf{u}_4, \mathbf{u}_5, \mathbf{u}_6, \mathbf{u}_7, \mathbf{u}_8] \quad (5)$$

$$= \begin{bmatrix} \mathbf{b}_1 & \mathbf{b}_2 & \mathbf{b}_3 & \mathbf{b}_4 & \mathbf{b}_5 & \mathbf{b}_6 & \mathbf{b}_7 & \mathbf{b}_8 \\ \overline{\mathbf{b}_1} & \overline{\mathbf{b}_2} & \overline{\mathbf{b}_3} & \overline{\mathbf{b}_4} & \overline{\mathbf{b}_5} & \overline{\mathbf{b}_6} & \overline{\mathbf{b}_7} & \overline{\mathbf{b}_8} \\ \mathbf{b}_5 & \mathbf{b}_2 & \mathbf{b}_4 & \mathbf{b}_8 & \mathbf{b}_7 & \mathbf{b}_1 & \mathbf{b}_3 & \mathbf{b}_6 \\ \overline{\mathbf{b}_5} & \overline{\mathbf{b}_2} & \overline{\mathbf{b}_4} & \overline{\mathbf{b}_8} & \overline{\mathbf{b}_7} & \overline{\mathbf{b}_1} & \overline{\mathbf{b}_3} & \overline{\mathbf{b}_6} \\ \mathbf{b}_3 & \mathbf{b}_1 & \mathbf{b}_5 & \mathbf{b}_6 & \mathbf{b}_3 & \mathbf{b}_8 & \mathbf{b}_2 & \mathbf{b}_7 \\ \overline{\mathbf{b}_3} & \overline{\mathbf{b}_1} & \overline{\mathbf{b}_5} & \overline{\mathbf{b}_6} & \overline{\mathbf{b}_3} & \overline{\mathbf{b}_8} & \overline{\mathbf{b}_2} & \overline{\mathbf{b}_7} \end{bmatrix}. \quad (6)$$

Further, we have the following equations for the four right-node measurement vectors: $\mathbf{z}_1 = \mathbf{u}_3$, $\mathbf{z}_2 = \mathbf{u}_1 \vee \mathbf{u}_3 \vee \mathbf{u}_8$, $\mathbf{z}_3 = \mathbf{u}_1 \vee \mathbf{u}_8$, $\mathbf{z}_4 = \mathbf{u}_3 \vee \mathbf{u}_8$. Thus,

$$\mathbf{z}_1 = (0,1,0|1,0,1|0,1,1|1,0,0|1,0,0|0,1,1)^\mathsf{T}, \quad (7)$$

$$\mathbf{z}_2 = (1,1,1|1,1,1|1,1,1|1,1,1|1,1,0|1,1,1)^\mathsf{T}, \quad (8)$$

$$\mathbf{z}_3 = (1,1,1|1,1,1|1,0,1|0,1,1|1,1,0|1,0,1)^\mathsf{T}, \quad (9)$$

$$\mathbf{z}_4 = (1,1,1|1,0,1|1,1,1|1,1,0|1,1,0|0,1,1)^\mathsf{T}, \quad (10)$$

where the 6 sections of each right node's measurement vector are separated by vertical bars. The decoding algorithm first finds all the singletons by checking whether a right-node measurement's weight is $3L = 3\log_2 n$. Since the weight of $\mathbf{z}_1$ is $3\log_2 n$, the decoder declares that right node 1 is a singleton and reads off the first 3 bits of $\mathbf{z}_1$. As $\mathbf{z}_1^1 = (0, 1, 0)$, the decoder concludes that item 3 is defective.

In the second iteration, the algorithm inspects right nodes that are potentially resolvable doubletons including defective item 3. Since $T_{2,3} = T_{4,3} = 1$, right nodes 2 and 4 are inspected. Consider right node 2. We hypothesize that the right node is a doubleton consisting of defective item 3 and exactly one other unknown defective item, i.e., $\mathbf{z}_2 = \mathbf{u}_3 \vee \mathbf{u}_{\ell_1}$. Using the described guess-and-check procedure, we have $\ell_1 = 6$, $\ell_2 = 8$, $\ell_3 = 3$. Since $i_{\ell_1} = i_6 = 1 \neq \ell_2$, $j_{\ell_1} = j_6 = 8 \neq \ell_3$, the decoder declares that the right node is *not* a resolvable doubleton. Consider right node 4. The decoder again makes a guess that $\mathbf{z}_4 = \mathbf{u}_3 \vee \mathbf{u}_{\ell_1}$. Then, it obtains three indices as follows: $\ell_1 = 8$, $\ell_2 = 6$, $\ell_3 = 7$. By noticing that $i_{\ell_1} = i_8 = 6 = \ell_2$, $j_{\ell_1} = j_8 = 7 = \ell_3$, the decoder declares that right node 4 is a resolvable doubleton including item 3 and finds that the other defective item's index is $\ell_1 = 8$.

In the third iteration, right node 3 is inspected since item 8 is identified, and $T_{3,8} = 1$. The decoder hypothesizes that $\mathbf{z}_3 = \mathbf{u}_8 \vee \mathbf{u}_{\ell_1}$ and reads three indices: $\ell_1 = 1$, $\ell_2 = 5$, $\ell_3 = 3$. Because $i_{\ell_1} = i_1 = 5 = \ell_2$, $j_{\ell_1} = j_1 = 3 = \ell_3$, the decoder declares that right node 3 is a doubleton and item 1 is a defective item. The algorithm is terminated as there are no more right nodes to be resolved, concluding that items 1, 3 and 8 are defective items.

III. Main Results

We present the main theoretical result of this paper.

Theorem 1. *With $m = 6C(\epsilon)K\log_2 n$ 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 depends only on ϵ. Table I shows some pairs of ϵ and $C(\epsilon)$. The computational complexity of the decoding algorithm is $\mathcal{O}(K\log n)$, that is optimal.*

Here, we provide an overview of the proof, referring the readers to [9] for the full proof.

[2] In the interest of conceptual clarity of the toy example, here we present a bipartite graph that is *not* left-regular.

Error floor, ϵ	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}	10^{-8}
$C(\epsilon)=\frac{d^\star}{\lambda^\star}$	6.13	7.88	9.63	11.36	13.10	14.84
Left-deg, $d^\star$	7	9	10	12	14	15

TABLE I: Pairs of ϵ and $C(\epsilon)$

Proof: Since each right node is associated with $6\log_2 n$ tests, we only need to show that the number of required right nodes to guarantee successful completion of the algorithm is $C(\epsilon)K$.

Recall that random d-left-regular bipartite graphs with n left nodes and M right nodes are used for $\mathcal{G}$. For the analysis, we focus on the *pruned* graph constructed by the K defective left nodes and the right nodes. Then, the average right degree is $\lambda=\frac{Kd}{M}$. Further, as K gets large, the degree distribution of right nodes approaches a Poisson distribution with parameter λ. We define the right edge-degree distribution $\rho(x)=\sum_{i=1}^{\infty}\rho_i x^{i-1}$, where ρ_i is the probability that a randomly selected edge in the graph is connected to a right node of degree i. Then, with this design of random bipartite graphs, $\rho_i=\frac{iM}{Kd}\Pr(\text{degree of a random right node}=i)$. With the Poisson approximation, $\rho_i=e^{-\lambda}\frac{\lambda^{i-1}}{(i-1)!}$, and $\rho(x)=e^{-\lambda(1-x)}$.

SAFFRON performs an iterative decoding procedure, and the fraction of defective items that cannot be identified at the end of the decoding algorithm can be analyzed by density evolution, a tool to analyze message-passing algorithms [8], [21]. At iteration j of the algorithm, an unidentified defective item passes a message to its neighbor right nodes that it has not been recovered. Let p_j be the probability that a random defective item is not identified at iteration j. The density evolution relates p_j to p_{j+1} as follows: $p_{j+1}=[1-(\rho_1+\rho_2(1-p_j))]^{d-1}$, where $\rho_1=e^{-\lambda}$, and $\rho_2=\lambda e^{-\lambda}$. To prove this equation, consider a tree-like neighborhood of an edge between left node v and right node c. At iteration $j+1$, left node v passes a 'not-recovered' message to right node c if none of its $d-1$ other neighbor right nodes has been identified as either a singleton or a resolvable doubleton at iteration j. The probability that a particular neighbor right node has been resolved at iteration j is $\rho_1+\rho_2(1-p_j)$. Given a tree-like neighborhood of v, the above equation follows since all the messages are independent.

To characterize the fraction of defective items that will not be recovered by the time the algorithm terminates, we find the limit of the sequence $\{p_j\}$ as $j\to\infty$. Let $\epsilon=\lim_{j\to\infty}p_j$. One can approximately calculate $\epsilon=\left[1-e^{-\lambda}-\lambda e^{-\lambda}\right]^{d-1}$. We now find a pair of design parameters (d,M) that minimize the number of right nodes M (thus the number of tests) given a targeted reliability ϵ. This can be done by minimizing $M=\frac{Kd}{\lambda}$ subject to $(d-1)\log(1-e^{-\lambda}-\lambda e^{-\lambda})=\log\epsilon$. We numerically solve the optimization problem and attain the optimal $\lambda^\star$ and $d^\star$ as a function of ϵ. Defining $C(\epsilon)=\frac{M}{K}=\frac{d^\star}{\lambda^\star}$, we present some of the optimal design parameters for different reliability levels in Table I.

Up to now, we have analyzed the average fraction of unidentified defective items over a randomly constructed bipartite graph. To complete the proof, we need to show that with high probability 1) a constant-depth neighborhood of a random left node is a tree, 2) p_j gets arbitrarily close to ϵ after a fixed number of iterations, and 3) the actual fraction of unidentified defective items is highly concentrated around p_j. We refer the readers to [9] for the proof of these statements.

Finally, the computational complexity of the decoding algorithm is $\mathcal{O}(K\log n)$ because 1) the algorithm is terminated after a finite number of iterations, and 2) the computational complexity of each iteration is $\mathcal{O}(K\log n)$. ■

Singleton-Only-SAFFRON. We now present *Singleton-Only-SAFFRON*, a variant of the SAFFRON scheme that only detects and resolves singletons. The measurement matrix of Singleton-Only-SAFFRON is similar to the one of SAFFRON with the difference that the signature matrix only consists of U_1 and $\overline{U}_1$ as in (2). With this signature matrix, the Singleton-Only-SAFFRON detects and resolves all singletons in a single-stage procedure. We present the following theorem on the Singleton-Only-SAFFRON.

Theorem 2. *With* $m=2e(1+\alpha)K\log K\log_2 n\simeq 5.437(1+\alpha)K\log K\log_2 n$ *tests, Singleton-Only-SAFFRON finds* all *the* K *defective items with probability* $1-\mathcal{O}(\frac{1}{K^\alpha})$*, where* e *is the base of the natural logarithm, and* $\alpha>0$*. The computational complexity of the decoding algorithm is* $\mathcal{O}(K\log K\log n)$*.*

We refer the readers to [9] for the proof.

Robustified SAFFRON for noisy group testing. In this section, we robustify SAFFRON such that it can recover the set of K defective items with erroneous or *noisy* test results. We assume that each test result is 'wrong' with probability q, i.e., $\mathbf{y}=A\odot\mathbf{x}+\mathbf{w}$, where the addition is over binary field, and $\mathbf{w}$ is an i.i.d. noise vector whose components are 1 with probability $0<q<\frac{1}{2}$ and 0 otherwise [3].

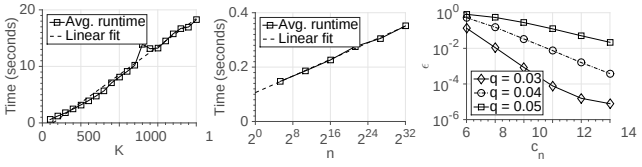
Our approach is simple: we design the robust signature matrix U' consisting of *encoded* columns of U_j for $1\le j\le 3$ and their complements. Intuitively, we treat each column of U_j as a message that needs to be transmitted over a noisy memoryless communication channel. An efficient modern error-correcting code guarantees reliable decoding of the signature. Specifically, we use the following signature matrix $U'\in\{0,1\}^{\frac{6\log_2 n}{R}\times n}$

$$U'=\begin{bmatrix} f(\mathbf{b}_1) & f(\mathbf{b}_2) & \dots & f(\mathbf{b}_{n-2}) & f(\mathbf{b}_{n-1}) \\ \overline{f(\mathbf{b}_1)} & \overline{f(\mathbf{b}_2)} & \dots & \overline{f(\mathbf{b}_{n-2})} & \overline{f(\mathbf{b}_{n-1})} \\ f(\mathbf{b}_{i_1}) & f(\mathbf{b}_{i_2}) & \dots & f(\mathbf{b}_{i_{n-1}}) & f(\mathbf{b}_{i_n}) \\ \overline{f(\mathbf{b}_{i_1})} & \overline{f(\mathbf{b}_{i_2})} & \dots & \overline{f(\mathbf{b}_{i_{n-1}})} & \overline{f(\mathbf{b}_{i_n})} \\ f(\mathbf{b}_{j_1}) & f(\mathbf{b}_{j_2}) & \dots & f(\mathbf{b}_{j_{n-1}}) & f(\mathbf{b}_{j_n}) \\ \overline{f(\mathbf{b}_{j_1})} & \overline{f(\mathbf{b}_{j_2})} & \dots & \overline{f(\mathbf{b}_{j_{n-1}})} & \overline{f(\mathbf{b}_{j_n})} \end{bmatrix},$$

where $f(\cdot):\{0,1\}^N\to\{0,1\}^{N/R}$ is the encoding function of a spatially-coupled LDPC codes [22]. We now present the following main theorem for the robust SAFFRON scheme.

Theorem 3. *With* $m=6\beta(q)C(\epsilon)K\log_2 n$ *tests, the Robustified-SAFFRON can recover at least* $(1-\epsilon)K$ *defective items with probability* $1-\mathcal{O}(\frac{K}{n^{2+\zeta}})$*, where* ϵ *is an arbitrarily-close-to-zero constant,* $C(\epsilon)$ *is a constant that depends only on* ϵ*, and* $\beta(q)=\frac{1}{R}>\frac{1}{1-H(q)-\delta}$ *for an arbitrarily small constant* $\delta>0$*. Table I shows some*

[3] If $q>\frac{1}{2}$, one can always take the complement of all the test results, and treat the channel as if each test result is wrong with crossover probability $0<\tilde{q}=1-q<\frac{1}{2}$.



(a) Noiseless, $n = 2^{32}$ (b) Noiseless, $K = 2^5$ (c) Noisy group testing

Fig. 1: **Simulation results of SAFFRON.** (a), (b): We measure run-time of SAFFRON for varying values for n and K. (c): We measure the average fraction of missed defective items with noisy test results for varying value for c_n. The y-axis is the average fraction of unidentified defective items.

pairs of ϵ and $C(\epsilon)$. The computational complexity of the decoding algorithm is $\mathcal{O}(K \log n)$.

We refer the readers to [9] for the description of the decoding algorithm of the robustified-SAFFRON, the proof of the theorem, and the corresponding theorem for the robustified-Singleton-Only-SAFFRON scheme.

IV. Simulation results

We implement the SAFFRON simulator in Python and simulate SAFFRON on a laptop with 2GHz Intel Core i7 and 8GB memory. We first simulate how computationally-efficient SAFFRON's decoding algorithm is. In Fig. 1a and Fig. 1b, we plot the average runtime of the decoding algorithm, which clearly demonstrate that the time-complexity of the decoding algorithm is $\mathcal{O}(K \log n)$. The robustified SAFFRON scheme is also simulated: we choose $n = 2^{32} \simeq 4.3 \times 10^9$ and $K = 2^7 = 128$. For random bipartite graphs, we use $d = 12$ and $M = 11.36K$. We vary the probability of error q from 0.03 to 0.05. While we make use of capacity-achieving codes in Theorem 3, we use Reed-Solomon (RS) codes for simulations for simplicity [23]. A Reed-Solomon code takes a message of c_k symbols from a finite field of size $c_q \geq c_n$, for a prime power c_q, and then encodes the message into c_n symbols. This code can correct upto any $\lfloor \frac{c_n - c_k}{2} \rfloor$ symbol errors. By using a field of size $c_q = 2^8$, a binary representation of length L $(= \log_2 n)$ can be viewed as a 4-symbol message, i.e., $c_k = 4$. Thus, the overall number of tests is as follows:

$$m = \underbrace{11.36K}_{\text{Number of right nodes}} \times \underbrace{c_n/c_k}_{\text{RS code expansion}} \times \underbrace{6 \log_2 n}_{\text{Number of message bits}}.$$

By having $c_n = c_k + 2t$, the robustified SAFFRON scheme can correct upto t symbol errors within each section of the right-node measurement vector. Thus, we evaluate the performance of the robustified SAFFRON scheme with $c_n \in \{6, 8, \ldots, 16\}$ for various noise levels. We measure the average fraction of unidentified defective items over 1000 runs for each setup. Fig. 1c shows the simulation results; the x-axis is the block-length of the used code, and the logarithmic y-axis is the average fraction of unidentified defective items. Further, the decoding time takes only about 3.8 seconds on average.

V. Conclusion

In this paper, we have proposed SAFFRON (Sparse-grAph codes Framework For gROup testiNg), which recovers an arbitrarily-close-to-one $(1 - \epsilon)$-fraction of K defective items with high probability with $6C(\epsilon)K \log_2 n$ tests, where $C(\epsilon)$ is a relatively small constant that depends only on ϵ. Also, the computational complexity of the decoding algorithm of SAFFRON is order-optimal. We have described the design and analysis of SAFFRON based on the powerful modern coding-theoretic tools of sparse-graph coding and density evolution. We have also proposed Singleton-Only-SAFFRON, which recovers all defective items with $2e(1+\alpha)K \log K \log_2 n$ tests, and the robustified SAFFRON that can recover the set of defective items with noisy test results using modern error-correcting codes. We have provided extensive simulation results to support our theoretical results.

References

[1] F. K. Hwang *et al.*, *Combinatorial group testing and its applications*. World Scientific, 2000.

[2] H.-B. Chen and F. 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 *Proceedings of The 30th ICML*, 2013.

[4] A. Ganesan, S. Jaggi, and V. Saligrama, "Learning immune-defectives graph through group tests," in *IEEE ISIT*, June 2015.

[5] M. Goodrich, M. Atallah, and R. Tamassia, "Indexing information for data forensics," in *Applied Cryptography and Network Security*. Springer Berlin Heidelberg, 2005, vol. 3531, pp. 206–221.

[6] A. Gilbert, M. Iwen, and M. Strauss, "Group testing and sparse signal recovery," in *The 42nd Asilomar Conference on Signals, Systems and Computers*, Oct 2008.

[7] A. Emad and O. Milenkovic, "Poisson group testing: A probabilistic model for nonadaptive streaming boolean compressed sensing," in *IEEE ICASSP*, May 2014.

[8] T. Richardson and R. Urbanke, *Modern Coding Theory*. Cambridge University Press, 2008.

[9] K. Lee, R. Pedarsani, and K. Ramchandran, "SAFFRON: A fast, efficient, and robust framework for group testing based on sparse-graph codes," *CoRR*, vol. abs/1508.04485, 2015.

[10] 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.

[11] A. Mazumdar, "Nonadaptive group testing with random set of defectives via constant-weight codes," *CoRR*, vol. abs/1503.03597, 2015.

[12] A. G. D'yachkov and V. V. Rykov, "Superimposed distance codes," *Problems Control Inform. Theory/Problemy Upravlen. Teor. Inform.*, vol. 18, no. 4, pp. 273–250, 1989.

[13] 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*, 2010.

[14] 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.

[15] C. L. Chan, S. Jaggi, V. Saligrama, and S. Agnihotri, "Non-adaptive group testing: Explicit bounds and novel algorithms," in *IEEE ISIT*, July 2012.

[16] T. Wadayama, "An analysis on non-adaptive group testing based on sparse pooling graphs," in *IEEE ISIT*, July 2013.

[17] S. Cai, M. Jahangoshahi, M. Bakshi, and S. Jaggi, "GROTESQUE: Noisy group testing (quick and efficient)," in *The 51st Annual Allerton Conference on Communication, Control, and Computing*, Oct 2013.

[18] S. Pawar and K. Ramchandran, "Computing a k-sparse n-length Discrete Fourier Transform using at most $4k$ samples and $\mathcal{O}(k \log k)$ complexity," in *IEEE ISIT*, July 2013.

[19] X. Li, J. Bradley, S. Pawar, and K. Ramchandran, "The SPRIGHT algorithm for robust sparse Hadamard transforms," in *IEEE ISIT*, June 2014.

[20] R. Pedarsani, K. Lee, and K. Ramchandran, "Phasecode: Fast and efficient compressive phase retrieval based on sparse-graph codes," in *The 52nd Annual Allerton Conference on Communication, Control, and Computing*, Sept 2014.

[21] A. Shokrollahi, "LDPC codes: An introduction," in *Coding, cryptography and combinatorics*. Springer, 2004, pp. 85–110.

[22] S. Kudekar, T. Richardson, and R. Urbanke, "Threshold saturation via spatial coupling: Why convolutional LDPC ensembles perform so well over the BEC," *IEEE Transactions on Information Theory*, vol. 57, no. 2, pp. 803–834, Feb 2011.

[23] S. Lin and D. J. Costello, *Error control coding*, 2nd ed. Pearson, 2004.