Non-Adaptive Group Testing Framework based on Concatenation Code

Thach V. Bui*, Minoru Kuribayashi‡, and Isao Echizen*†

*SOKENDAI (The Graduate University for Advanced Studies), Hayama, Kanagawa, Japan
bvthach@nii.ac.jp

‡Graduate School of Natural Science and Technology, Okayama University, Okayama, Japan
kminoru@okayama-u.ac.jp

†National Institute of Informatics, Tokyo, Japan
iechizen@nii.ac.jp

Abstract

We consider an efficiently decodable non-adaptive group testing (NAGT) problem that meets theoretical bounds. The problem is to find a few specific items (at most d) satisfying certain characteristics in a colossal number of N items as quickly as possible. Those d specific items are called *defective items*. The idea of NAGT is to pool a group of items, which is called *a test*, then run a test on them. If the test outcome is *positive*, there exists at least one defective item in the test, and if it is *negative*, there exists no defective items. Formally, a binary $t \times N$ measurement matrix $\mathcal{M} = (m_{ij})$ is the representation for t tests where row i stands for test i and $m_{ij} = 1$ if and only if item j belongs to test i.

There are three main objectives in NAGT: minimize the number of tests t, construct matrix $\mathcal{M}$, and identify defective items as quickly as possible. In this paper, we present a strongly explicit construction of $\mathcal{M}$ for when the number of defective items is at most 2, with the number of tests $t \simeq 16 \log N = O(\log N)$. In particular, we need only $K \simeq N \times 16 \log N = O(N \log N)$ bits to construct such matrices, which is optimal. Furthermore, given these K bits, any entry in the matrix can be constructed in time $O(\ln N / \ln \ln N)$. Moreover, $\mathcal{M}$ can be decoded with high probability in time $O\left(\frac{\ln^2 N}{\ln^2 \ln N}\right)$. When the number of defective items is greater than 2, we present a scheme that can identify at least $(1-\epsilon)d$ defective items with $t \simeq 32C(\epsilon)d \log N = O(d \log N)$ for any close-to-zero ϵ, where $C(\epsilon)$ is a constant that depends only on ϵ.

I. Introduction

Group Testing dates back to World War II, when a statistician, Robert Dorfman, solved the problem of identifying which draftees had syphilis [1]. It turned out to a problem of finding very characteristic items in a huge number of items. Nowadays, the problem is called *group testing* and has attracted researchers in various fields. There are two main approaches to group testing. In *adaptive group testing*, tests are performed in many stages and the later tests depend on the earlier tests. With this approach, the number of test can be theoretically optimized [2]. However, it takes much time due to many stages. In Non-Adaptive Group Testing (NAGT), all tests are designed in advance and are performed at the same time, i.e. simultaneously. This approach is most useful for parallel architecture such as multiple access communication [3], biology [4] because it saves time. Here we focus on NAGT. Additional information on group testing can be found [5], [2].

We can present N items as a vector $\mathbf{x} = (x_1, x_2, \ldots, x_N) \in \{0,1\}^N$ and $|\mathbf{x}| = \sum_{j=1}^{N} x_j \leq d$ where x_j stands for item j and $x_j = 1$ if and only if item j is defective. A binary $t \times N$ measurement matrix $\mathcal{M} = (m_{ij})$ is the representation for t tests where row i stands for test i and $m_{ij} = 1$ iff item j belongs to test i. If the test outcome is *positive*, there exists at least one defective item in the test and *negative* otherwise. For error-free NAGT, the normal decoding complexity is $O(tN)$. Cheraghchi [6], Indyk et al. [7], and Ngo et al. [8] made a breakthrough on this issue by developing sub-linear decoding algorithm (say $ploy(d, \log N)$). However, the decoding time is quite big. Lee et al. [9] proposed SAFFRON scheme that the number of test and the decoding complexity are optimal based on probabilistic approaches.

TABLE I
COMPARISON WITH EXISTING SCHEMES

		Construction type	Tests t	Decoding complexity	Decoding type	False positive
$d=2$	Ngo et al. [10]	Strongly explicit	$19200 \log N$	$poly(d) \cdot t \log^2 t + O(t^2)$	Det., All	No
	SAFFRON [9]	Explicit	$10.8731(1+\log \frac{1}{\epsilon}) \log N$	$O(\log N)$	Rand., All	No
	Proposed scheme	**Strongly explicit**	$\mathbf{16} \log \mathbf{N}$	$\mathbf{O}\left(\frac{\ln^2 \mathbf{N}}{\ln^2 \ln \mathbf{N}}\right)$	**Rand., All**	**No**
$d \geq 3$	Ngo et al. [10]	Strongly explicit	$4800d^2 \log N$	$poly(d) \cdot t \log^2 t + O(t^2)$	Det., All	No
	SAFFRON [9]	Explicit	$6C(\epsilon)d \log N$	$O(d \log N)$	Rand., Partially	Yes
	Proposed scheme	**Explicit**	$\mathbf{32C}(\epsilon)\mathbf{d} \log \mathbf{N}$	$\mathbf{O}(\mathbf{d} \log \mathbf{N})$	**Det., Partially**	**No**

A. Contributions

We classify d-disjunct matrices in two categories: $d = 2$ and $d \geq 3$. We do not consider the case $d = 1$ because it is trivial and can be easily solved using the 1-disjunct matrix in Eqn. (9). Our goal is to design a strongly explicit construction in which each column (entry) of the matrix can be constructed in polynomial time of t, i.e. $poly(t)$ when $d = 2$ and an explicit construction in which the matrix can be constructed in polynomial time of t and N, i.e. $poly(t, N)$, such that $d \geq 3$ defective items can be found efficiently.

We use concatenation codes, which concatenate an outer code and an inner code, to construct 2-disjunct matrices. In coding theory, a message is encoded into a large message to be transmitted over noisy channels such that, if there are a certain number of errors in the received message, the receiver can recover the original message. Concatenated codes are a rich family of codes that can handle a large fraction of errors with high reliability. To efficiently identify defective items, we first use Reed-Solomon codes over a "large" field as an outer code and a very small inner code whose members can be efficiently exhaustively searched. Second, we use statistics to minimize decoding failure by repeating our decoding algorithm. When $d \geq 3$, we mainly rely on SAFFRON scheme. SAFFRON scheme use an incidence matrix of a sparse graph and a signature matrix to construct and decode (nearly) d-disjunct matrix with high probability. In our scheme, we use 2-disjunct matrices in the previous result as signature matrices to prevent SAFFRON scheme from accusing wrong defective items. Our scheme is compared with existing schemes in Table I. This paper makes two contributions:

- It presents a strongly explicit construction of 2-disjunct matrices with $t \simeq 16 \log N = O(\log N)$ tests. In particular, only $K \simeq N \times 16 \log N = O(N \log N)$ bits are needed to construct such matrices, which is optimal. Furthermore, if these K bits are given, any entry in the matrix can be constructed in time $O(\ln N / \ln \ln N)$. Moreover, $\mathcal{M}$ can be decoded in time $O\left(\frac{\ln^2 N}{\ln^2 \ln N}\right)$ with high probability when $N \leq 2^{56}$.
- When the number of defective items is greater than 2, we present a scheme that can identify at least $(1-\epsilon)d$ defective items with $t \simeq 32C(\epsilon)d \log N = O(d \log N)$ for any close-to-zero ϵ, where $C(\epsilon)$ is a constant that depends only on ϵ, as shown in Table III.

B. Related work

When generating $t \times N$ d-disjunct matrices, we classify into three categories: 1) random construction, in which all columns of a matrix are generated randomly; 2) explicit construction, in which the matrix can be constructed in polynomial time of t and N, i.e., $poly(t, N)$; and 3) strongly explicit construction, in which each column (entry) of a matrix can be constructed in polynomial time of t, i.e., $poly(t)$. Lee et al. [9] proposed the *SAFFRON scheme*, which uses explicit construction. Although the resulting matrix is highly like d-disjunctive, its construction is not strongly explicit. Therefore, the whole matrix, which is very large when d and N are large, must be stored then reused when necessary. This is not suitable

for some applications, such as data stream ones [11], in which routers have limited resources and need to access the column assigned to an internet protocol (IP) address as quickly as possible to perform their functions.

There are two approaches for identifying defective items: deterministic and randomized algorithms. Deterministic algorithms means they run deterministically without any randomness included and get their goals with the accuracy of 100%. Randomized algorithms means some parts of the algorithms run randomly and the whole algorithm will be successful with probability at least $1-\epsilon$ for any $\epsilon > 0$. Probabilistic algorithms sacrifice accuracy but usually reduce the number of tests and decoding complexity. Although there are two different approaches when thinking about decoding algorithms, the decoding algorithms share same four sub-approaches. First, the decoding algorithms identify all d defective items. Second, the decoding algorithms identify a fraction of d defective items with no wrong defective item accusation, say $(1-\delta)d$ items for any $\delta > 0$. Third, the decoding algorithms identify a fraction of d defective items with some wrong defective item accusation, say $(1-\delta)d$ items for any $\delta > 0$. Fourth, the decoding algorithms identify all d defective items and some wrong defective items, say $(1+\delta)d$ items for any $\delta > 0$. If an algorithm identifies all defective items (no matter false positives accused), we call it an *all identifying* algorithm and denote *(Det./Rand., All)*. If an algorithm recovers a fraction of defective items (no matter false positives accused), we call it a *partially recovering* algorithm and denote *(Det./Rand., Partially)*.

Lee et al. [9] proposed a *SAFFRON* scheme for NAGT based on sparse-graph coding theory. The scheme identifies a close-to-one fraction of the defective items. It requires $t = 6C(\epsilon)d\log N$ tests, where $C(\epsilon)$ is a constant that depends on ϵ. The decoding complexity is $O(d\log N)$. They also proposed Singleton-Only-SAFFRON scheme, which identifies all defective items with high probability $(1-\epsilon)$ using $t = 2e(1+\log\frac{1}{\epsilon})d\log d\log N$ tests, for any $\epsilon > 0$ is a constant. The decoding complexity is $O(\log\frac{1}{\epsilon}d\log d\log N)$. Previous SAFFRON schemes are based on the *GROSTEQUE* scheme. Cai et al. [12] propose Grotesque for non-adaptive group testing and adaptive group testing. The GROTESQUE scheme requires $O(d\log d\log N)$ tests, which is nearly order-optimal, decoding complexity of $O(d(\log N+\log^2 d))$.

Ngo et al. [7] proposed a strongly explicit construction of d-disjunct matrices with $t \simeq 4800d^2\log N$ that enables defective items to be identified in $poly(d)\cdot t\log^2 t + O(t^2)$ time. When $d=2$, the number of tests is substantially greater than $16\log N$ and the decoding time is longer than the $O\left(\frac{\ln^2 N}{\ln^2\ln N}\right)$ of our scheme. For $d=2$, number of tests with the SAFFRON scheme is about $227\log N$ (for $\epsilon = 10^{-6}, N = 2^{32}$), which is more than with our scheme, and the decoding time is $\simeq 227\log N$ which is slightly less than with our scheme. However, there is no explicit construction, some defective items may not be identified, and there may be some false positives. The advantage of our algorithm is that it always identifies 2 defective items without accusing false positive items with high probability.

C. Paper Organization

The rest of the paper is organized as follows. In Section II, we present preliminaries on group testing and concatenated codes. In Section III, we review SAFFRON schemes with algorithms and analyze their drawbacks. Then, we present our main results in Section IV for when the number of defective items is at most 2 and for when the number of defective items is greater than 2. We conclude with a brief summary of the key points in Section V.

II. Preliminaries

Notations are defined here for consistency. We use a capitalized mathcal letter for a matrix, a non-capitalized subscripted letter with subscripts ij for denoting an entry at row i and column j, capitalized matcal letters with subscript i, and $,j$ denote for row i and column j, respectively. For example, entry m_{23} of matrix $\mathcal{M}$ is the entry at row 2 $\mathcal{M}_{2,}$ and column 3 $\mathcal{M}_{,3}$. Furthermore, $\log x$ is $\log_2 x$ and $\ln x$ is the natural logarithm of x. We also denote $[n] = \{0, 1, \ldots, n-1\}$. Vector $\mathbf{x}$ is denoted as a bold letter

TABLE II
SUMMARY OF OUR NOTATION

Notation	Definition
N	Number of items
d	Number of defective items
t	Number of tests
h	Number of right nodes (bundles of tests)
$\mathcal{M}$	d-disjunct matrix
$\mathcal{U}$	Signature matrix
$\mathbf{x}$	Binary representation of set of defective items
$\mathbf{y}$	Binary representation of test outcomes
$wt(.)$	Hamming weight of number of ones in a vector
G	Set of indexes of defective items
y_i^l	Measurement l from the ith right node
$base10(.)$	A function converts a binary vector into a based 10 number

of x. base10(.) is a function coverts a binary vector to a number based 10. For example, base10$(1001) = 1 \cdot 2^3 + 0 \cdot 2^2 + 0 \cdot 2^1 + 1 = 9$. Notations used frequently here are defined in Table II.

A. q-ary d-disjunct matrices and d-disjunct matrix

We define q-ary d-disjunct matrices here.

Definition 1. *A matrix $\mathcal{A}$ is q-ary d-disjunctive if and only if all its entries belong to $\mathbb{F}_q$ and we pick any column, say j_0, and d other columns, say $j_1, \ldots, j_d$, of $\mathcal{A}$, there exists a row, say i_0 such that $a_{i_0 j_0} \neq a_{i_0 j_l}$ for $l = 1, 2, \ldots, d$.*

For example, if we choose $q = 8$, the following matrix is an 8-ary 2-disjunct matrix:

$$\mathcal{A} = \begin{pmatrix} 1 & 1 & 1 & 2 & 2 & 2 & 4 & 4 & 4 & 7 & 0 & 0 \\ 1 & 2 & 4 & 1 & 2 & 4 & 1 & 2 & 4 & 0 & 7 & 0 \\ 1 & 4 & 2 & 4 & 2 & 1 & 2 & 1 & 4 & 0 & 0 & 7 \end{pmatrix} \tag{1}$$

When $q = 2$, we define a d-disjunct matrix as follow:

Definition 2. *A binary matrix $\mathcal{M}$ is d-disjunctive if and only if we pick any column, say j_0, and d other columns, say $j_1, \ldots, j_d$, of $\mathcal{A}$, there exists a row, say i_0 such that $m_{i_0 j_0} = 1$ and $m_{i_0 j_l} = 0$ for $l = 1, 2, \ldots, d$.*

We can model the NAGT problem as follow. Given a Boolean sparse vector $\mathbf{x} = (x_1, x_2, \ldots, x_N) \in \{0, 1\}^N$ represented for N items, where $x_j = 1$ iff item j is defective and $|\mathbf{x}| \leq d$, our aim is to design $t \ll N$ tests such that $\mathbf{x}$ can be reconstructed with the low cost. Suppose that $G = \{j_1, \ldots, j_d\}$ is the set of defective items. Each test combines a subset of N items. Hence, a test can be considered as a binary vector $\{0, 1\}^N$ that is associated with the indexes of items belonging to that test. More generally, a set of t tests can be seen as a measurement matrix $\mathcal{M}$ in which the rows are separate tests and $m_{ij} = 1$ iff item j belongs to test i. The outcome of a test is positive (denoted 1) or negative (denoted 0). Since there are t tests, we can represent their outcome as a binary vector $\mathbf{y}^T = (y_1, \ldots, y_t) \in \{0, 1\}^t$. If $\mathcal{M}$ can identify at most d defective items, we call $\mathcal{M}$ a $t \times N$ d-disjunct matrix.

If we define the boolean sum of two vectors $\mathbf{x} = (x_j)$ and $\mathbf{y} = (y_j)$ as $\mathbf{z} = \mathbf{x} \vee \mathbf{y} = (x_j \vee y_j)$ for $j = 1, 2, \ldots, n$. Vector $\mathbf{x}$ is said to belong to $\mathbf{z}$ or $\mathbf{z}$ contains $\mathbf{x}$ if and only if $\mathbf{z} \vee \mathbf{x} = \mathbf{z}$. Then, the definition 2 is equivalent to the following: $\mathcal{M}$ is a d-disjunct matrix if the boolean sum of any d columns does not contain another column. For example, a 2×2 identity matrix is 2-disjunct matrix. Then, we can

model $\mathbf{y}$ as

$$\mathbf{y} = \bigvee_{j=1}^{N} x_j \mathcal{M}_{,j} = \bigvee_{j \in G} \mathcal{M}_{,j} \tag{2}$$

B. Reed-Solomon codes and concatenated codes

Reed-Solomon (RS) codes [13] are widely used in many fields [14]. They are constructed by using polynomial method over a finite field $\mathbb{F}_q$. A $[n, k, d]_q$-code C, $1 \leq k \leq n \leq q$, is a subset $C \subseteq [q]^n$ of size q^k. Let $\alpha_1, \alpha_2, \ldots, \alpha_n$ be n distinct elements in $\mathbb{F}_q$ and $\mathcal{L} = \{\alpha_1, \alpha_2, \ldots, \alpha_n\}$ be the evaluation set. For a polynomial of degree at most $k-1$, $f(x) \in \mathbb{F}_q^k[x]$, we denote $f(\mathcal{L}) = (f(\alpha_1), \ldots, f(\alpha_n))$ is a codeword of C. Because of algebraic properties, each $f(\alpha_i)$ can be calculated in time $O(k \log n)$, for $i = 1, \ldots, n$. Given a message $\mathbf{m} = (m_0, m_1, \ldots, m_{k-1})$, we construct a corresponding polynomial $f_{\mathbf{m}}(x) = \sum_{i=0}^{k-1} m_i x^i$. Since each codeword of C is generated by the polynomial $f(x) \in \mathbb{F}_q^k[x]$, C has a minimum distance of $\Delta = n - k + 1$, which means any two codewords of C does not agree at least Δ positions. Then, given a received codeword $\mathbf{r}$, if it differs at most $(\Delta - 1)/2$ positions with $f_{\mathbf{m}}(\mathcal{L})$, $\mathbf{m}$ is always recoverable. In the other words, given a received codeword $\mathbf{r}$ with at most $(\Delta - 1)/2$ errors, the exists at most 1 message $\mathbf{m}$ such that $\Delta(f_{\mathbf{m}}(\mathcal{L}), \mathbf{r}) \leq (\Delta - 1)/2$, where $\Delta(\mathbf{x}, \mathbf{y})$ is the number of positions that $\mathbf{x}$ and $\mathbf{y}$ does not agree. Lin et al. [15] have just proposed a very efficiently decoding algorithm for a special class of Reed-Solomon codes as follow:

Theorem 1. *[15] Any Reed-Solomon code* $[n = 2^m, k = 2^{m-1}]_{q=n=2^m}$ *over* $\mathbb{F}^q$ *can be decoded in time* $2n \log n + (n-k) \log^2(n-k) \simeq 1.5n \log n = O(n \log n)$.

Then, we introduce an elegant technique to construct d-disjunct matrix called *concatenation technique*. Forney [16] described the basic idea of *concatenated codes*. Concatenated codes are constructed by using an $[n_1, k_1, \Delta_1]_q$ *outer code* $C_{out} : [q]^{k_1} \to [q]^{n_1}$, where $q = 2^{k_2}$ (in general, $q = p^{k_2}$ where p is a prime), and a $[n_2, k_2, \Delta_2]_2$ binary *inner code* $C_{in} : \{0,1\}^{k_2} \to \{0,1\}^{n_2}$. Given a message $\mathbf{m} \in [q]^{k_1}$ ($m \in [q]$), let denote the encoding codeword of $\mathbf{m}$ of C_{out} (C_{in}) be $C_{out}(\mathbf{m})$ ($C_{in}(m)$, resp.). Given a received codeword $\mathbf{c}$, $\mathbf{c} \in [q]^{n_1}$ ($\mathbf{c} \in [2]^{n_2}$), let denote the decoding codeword of $\mathbf{c}$ of C_{out} (C_{in}) be $D_{C_{out}}(\mathbf{c})$ ($D_{C_{in}}(\mathbf{c})$, resp.).

The concatenated codes $C = C_{out} \circ C_{in}$ is defined as follows. Consider a message $\mathbf{m} \in ([q])^{k_1}$. Let $C_{out}(\mathbf{m}) = (x_1, \ldots, x_{n_1})$. Then, $C_{out} \circ C_{in}(\mathbf{m}) = (C_{in}(x_1), C_{in}(x_2), \ldots, C_{in}(x_{n_1}))$. C has length $n = n_1 n_2$ with message length $k = k_1 k_2$ and a minimum distance at least $\Delta = \Delta_1 \Delta_2$. C has $2^k = 2^{k_1 k_2}$ codewords of length $n = n_1 n_2$.

If we receive a vector $\mathbf{y} = (\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_{n_1}) \in (\{0,1\}^{n_2})^{n_1}$, we want to find a message $\mathbf{m}'$ such that $C_{out} \circ C_{in}(\mathbf{m}')$ is sufficiently 'close' to $\mathbf{y}$, i.e. $\Delta(\mathbf{y}, C_{out} \circ C_{in}(\mathbf{m}')) \leq \Delta(\mathbf{y}, C_{out} \circ C_{in}(\mathbf{m}''))$ for all $\mathbf{m}' \neq \mathbf{m}'' \in \mathbb{F}_q^k$, is the best way to decode $\mathbf{y}$. Forney proposed an efficient algorithm that can find a codeword of C_{out} that is sufficiently close to $\mathbf{y}$ using a probabilistic method. The decoding algorithm for concatenated codes proposed by Guruswami-Rudra-Sudan [17] is described in algorithm 1.

Using a suitable outer code and a suitable inner code, we can generate an d-disjunct matrix. For example, if we concatenate each element of $\mathcal{A}$ in Eqn. (1) with its 3-bit binary representation such as a matrix $\mathcal{B}$, we get a 2-disjunct matrix:

$$\mathcal{B}_{3\times 8} = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix}, \mathcal{M}_{9\times 12} = \mathcal{A} \circ \mathcal{B} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Algorithm 1 Concatenated code decoding algorithm – DecConcate($\mathbf{y}$)

Input: The $t \times 1$ outcome vector $\mathbf{y} = (\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_{n_1}) \in (\{0,1\}^{n_2})^{n_1}$
Output: $\mathbf{m} \in [q]^{k_1}$

1: Pick $\theta \in [0,1]$ uniformly at random.
2: **for** $1 \leq i \leq n_1$ **do**
3: $\quad \mathbf{y}'_i \leftarrow D_{C_{in}}(\mathbf{y}_i)$.
4: $\quad w_i \leftarrow \min\left(\Delta(\mathbf{y}'_i, \mathbf{y}_i), \frac{\Delta_2}{2}\right)$.
5: $\quad$ If $\theta < \frac{2w_i}{\Delta_2}$, set $y''_i =?$, otherwise set $y''_i \leftarrow x$, where $\mathbf{y}'_i = C_{in}(x)$.
6: **end for**
7: $\mathbf{m} \leftarrow D_{C_{out}}(\mathbf{y}'')$, where $\mathbf{y}'' = (y''_1, \cdots, y''_{n_1})$
8: RETURN $\mathbf{m}$

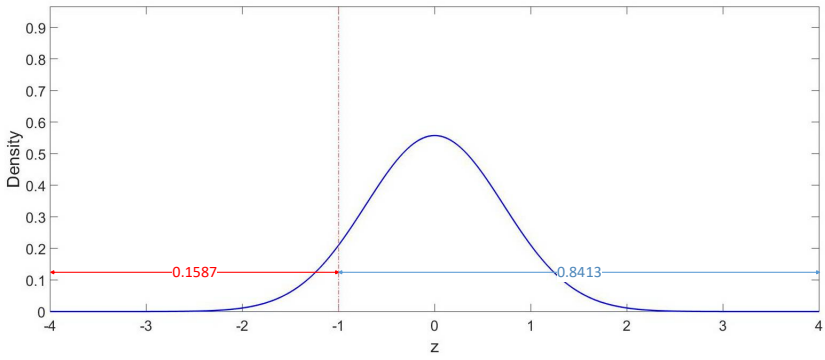

Fig. 1. z-score example for $z \leq -1$

C. z-score

Since z-score will be used in our decoding algorithm, we introduce this concept here. Given a normal distribution $\mathtt{X} \sim \mathbb{N}(\mu, \sigma^2)$ with the mean μ and the standard deviation σ, a z-score is a measurement to indicate how many standard deviations a value of $\mathbb{N}(\mu, \sigma^2)$ is from the mean. For any value $\mathtt{X}$, its corresponding z-score is:

$$z_{\mathtt{X}} = \frac{\mathtt{X} - \mu}{\sigma} \tag{3}$$

z-score also tells us the probability when an event happens. Additional information can be found in any statistical textbook such as [18]. For example, in the figure 1, the probability that an event whose corresponding z-score is not greater than -1 is 0.1587.

III. Review of original SAFFRON scheme

The original SAFFRON scheme [9] is usually simply described in text. To facilitate understanding, here we present it as an algorithm 2 and 3. The key elements in this scheme are the use of a tensor product between the outer binary matrix, which is sparse enough to have a (nearly) d-disjunct property, and the use of a signature matrix, which improves the effectiveness of identifying defective items and reducing the number of false positives.

The outer binary matrix is an incidence matrix $\mathcal{H}$ of a bipartite with N left nodes and h right nodes. The left nodes represent for the N items, and the right nodes represent the h bundles of test outcomes. An m left-regular bipartite graph, in which is each left node is uniformly and randomly connected to

Error floor 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

TABLE III
NUMERICAL PAIRS OF ϵ AND $C(\epsilon)$

exactly m right nodes, is used to construct $\mathcal{H}$. We state the main result of SAFFRON scheme for $d = 2$ and $d \geq 3$ before going to analyze it:

Theorem 2. *[9] For $t = 2e(1 + \log \frac{1}{\epsilon}) \cdot 2 \log 2 \log N \simeq 10.8731(1 + \log \frac{1}{\epsilon}) \log N$ tests, the SAFFRON scheme can identify 2 defective items with probability at least $1 - \epsilon$, in $10.8731(1 + \log \frac{1}{\epsilon}) \log N$ time, where $\epsilon > 0$ and e is the natural base logarithm.*

Theorem 3. *[9] For $t = 6C(\epsilon)d \log N$ tests, the SAFFRON scheme identifies at least $(1 - \epsilon)d$ defective items with probability $1 - O\left(\frac{d}{N^2}\right)$ where $\epsilon > 0$ and $C(\epsilon)$ is a constant that depends only on ϵ as described in Table III. The decoding complexity is $O(d \log N)$.*

A. Generating d-disjunct matrix based on tensor product

First, we describe the *row tensor product* between a matrix with a matrix. Suppose $\mathcal{H} \in \{0,1\}^{h \times N}$ and $\mathcal{U} \in \{0,1\}^{k \times N}$ where $k = a \times L$ (a and L are precisely defined in subsection III-B and III-C), the tensor product of $\mathcal{H}$ and $\mathcal{U}$ is defined as $\mathcal{M} = \mathcal{H} \otimes \mathcal{U} \stackrel{\text{def}}{=} [\mathcal{M}_{1,}^T, \mathcal{M}_{1,}^T, \ldots, \mathcal{M}_{h,}^T]^T \in \{0,1\}^{hk \times N}$, where $\mathcal{M}_{i,} = \mathcal{U}\text{diag}(\mathcal{H}_{i,}) \in \{0,1\}^{h \times N}$, and diag(.) is the diagonal matrix constructed by the input vector. Matrix $\mathcal{U}$ is 'divided' into a blocks, and each block has L rows. For example, the row tensor product of the matrices $\mathcal{H}$ and $\mathcal{U}$ with $a = 1$ and $L = 2$ is $\mathcal{M}$:

$$\mathcal{H} = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix} \text{ and } \mathcal{U} = \begin{pmatrix} 0 & 1 & 1 \\ 1 & 1 & 0 \end{pmatrix}; \qquad \mathcal{M} = \mathcal{H} \otimes \mathcal{U} = \left(\begin{array}{ccc} 0 & 0 & 1 \\ 1 & 0 & 0 \\ \hline 0 & 1 & 0 \\ 0 & 1 & 0 \\ \hline 0 & 1 & 1 \\ 0 & 1 & 0 \end{array}\right) \tag{4}$$

Because each row of $\mathcal{H}$ is critical to identifying defective items, we define the test outcomes corresponding to right node i as

$$\begin{aligned} \mathbf{z}_i^T &= (\mathbf{y}_i^1, \mathbf{y}_i^2, \ldots, \mathbf{y}_i^a) && (5) \\ &= (y_{(i-1)k+1}, \ldots, y_{(i-1)k+L}, y_{(i-1)k+L+1}, \ldots, y_{(i-1)k+2L}, \ldots, y_{(i-1)k+(a-1)L+1}, \ldots, y_{(i-1)k+aL}) && (6) \\ &= (y_{(i-1)k+1}, \ldots, y_{(i-1)k+L}, y_{(i-1)k+L+1}, \ldots, y_{(i-1)k+2L}, \ldots, y_{(i-1)k+k-L+1}, \ldots, y_{ik}) && (7) \end{aligned}$$

Then,

$$\mathbf{z}_i = \bigvee_{h_{ij}=1, x_j=1} \mathcal{U}_{,j}, \quad 1 \leq i \leq h, 1 \leq j \leq N \tag{8}$$

$\mathbf{z}_i$ can be interpreted as the Boolean sum of all signature vectors of the active left nodes connected to right node i.

To get Theorem 2, Lee et al. [9] showed that $h \simeq 2e(1 + \log \frac{1}{\epsilon})d \log d = O(d \log d)$ is enough. To get Theorem 3, Lee et al. [9] showed that $h \simeq 6C(\epsilon)d = O(d \log d)$ is enough.

The SAFFRON scheme searches all right nodes and identifies any node that is resolvable; i.e., a defective item can be found on the basis of its corresponding measurement outcome. There are two types of resolvable right nodes: i) *singletons*, where there is only one defective item connected to a right node,

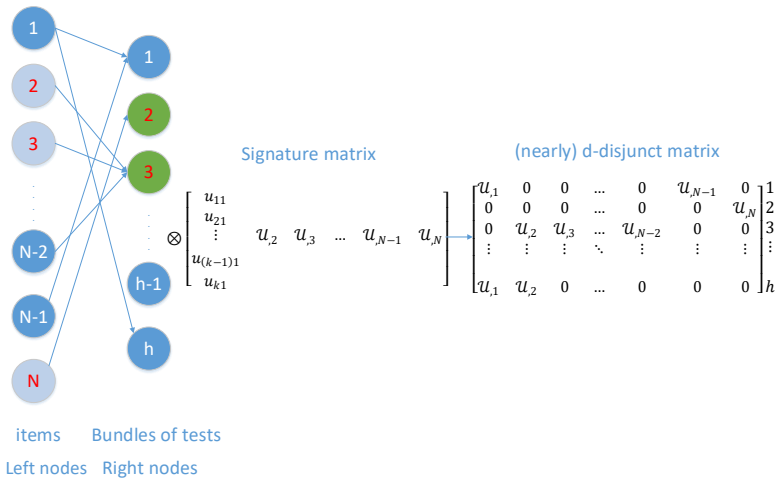

Fig. 2. Illustration of generating (nearly) d-disjunct matrices using row tensor product. In this figure, there are N left nodes representing N items and h right nodes representing for h bundles of tests. The defective items are 2, 3, and N. From the mapping, right nodes 2 and 3 are called a *singleton* and a *doubleton*, respectively. The first column of the signature matrix represents by each entry, which has k entries in total. There are h bundles of tests in the resulting matrix, and each bundle contains k tests. Therefore, there are $t = hk$ tests in total.

and ii) *resolvable doubletons*, where there are two defective items connected to a right node but one of them has been identified in the previous iterations. If a right node is connected to only two defective items and it is not known whether the node is resolvable, we call it a *doubleton*. Figure 2 illustrates how the generation of a (nearly) d-disjunct matrix using the SAFFRON scheme and an example of a singleton and a doubleton.

Finally, we summarize how to generate a d-disjunct matrix using algorithm 2.

Algorithm 2 Generating d-disjunct matrix

Input: An $h \times N$ incidence matrix $\mathcal{H} \in \{0,1\}^{h \times N}$, and a $k \times N$ signature matrix $\mathcal{U}$.
Output: A $t \times N$ d-disjunct matrix $\mathcal{M}$, where $t = hk$.

Initialization $\mathcal{M} \leftarrow \mathcal{H}$.
1: **for** $j = 1$ to N **do**
2: **for** $i = 1$ to h **do**
3: Replace entry m_{ij} with a column $m_{ij} \times \mathcal{U}_{,j}$
4: **end for**
5: **end for**
6: **return** $\mathcal{M}$

B. Detecting and solving a singleton

There are two main steps in SAFFRON scheme: (i) generating a $k \times N$ 1-disjunct signature matrix $\mathcal{U}$ in which all columns have a Hamming weight of $L = k/2$ and can be efficiently decoded in time $O(k)$ and (ii) running the algorithm iteratively to find all singletons and resolvable doubletons. First, an $L \times N$ matrix $\mathcal{U}_1$ is chosen in which the ith column is a vertical representation of $\mathbf{b}_i$, where $\mathbf{b}_i$ is the L-bit binary representation of integer $i-1$ for $i \in [n]$ and $L = \log N$. Then, a complementary matrix $\overline{\mathcal{U}}_1$ of $\mathcal{U}_1$, where the ith column of $\overline{\mathcal{U}}_1$ is the complement of $\mathbf{b}_i$, i.e. $\overline{\mathbf{b}}_i$, is stacked. From this construction,

$a = 2$, $L = \log N$, and $k = 2L = 2\log N$. We can describe $\mathcal{U}$ as

$$\mathcal{U} = \begin{pmatrix} \mathcal{U}_1 \\ \overline{\mathcal{U}}_1 \end{pmatrix} = \begin{pmatrix} \mathbf{b}_1 & \mathbf{b}_2 & \dots & \mathbf{b}_{N-1} & \mathbf{b}_N \\ \overline{\mathbf{b}}_1 & \overline{\mathbf{b}}_2 & \dots & \overline{\mathbf{b}}_{N-1} & \overline{\mathbf{b}}_N \end{pmatrix} = \left(\begin{array}{ccccc} 0 & 0 & \dots & 1 & 1 \\ 0 & 0 & \dots & 1 & 1 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & 1 & 1 \\ 0 & 1 & \dots & 0 & 1 \\ \hline 1 & 1 & \dots & 0 & 0 \\ 1 & 1 & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 1 & 1 & \dots & 0 & 0 \\ 1 & 0 & \dots & 1 & 0 \end{array}\right) \tag{9}$$

We observe that the Hamming weight of each column of $\mathcal{U}$ is $L = \log N$ and that $k = 2L = 2\log N$. Therefore, if any right node is a singleton, the Hamming weight of the corresponding measurement outcome is L, and the decoding time to find that defective item is $2\log N = O(\log N)$. Moreover, if any right node is not connected to a defective item or connected to more than one defective item, the Hamming weight of its corresponding measurement outcome does not equal to L. After detecting which nodes are singletons, the SAFFRON scheme checks the first half of the measurement outcome to find the indexes of the defective items. These findings are used in algorithm 3 from lines 1 to 9.

C. Resolvable doubletons

In the SAFFRON scheme, a doubleton is called *resolvable* iff one of the two defective items was identified in the previous iterations. A signature matrix like that in Eqn. (9) is sufficient for detecting and solving singletons. However, to detect and solve doubletons, the signature matrix must be extended.

$$\mathcal{U} = \begin{pmatrix} \mathcal{U}_1 \\ \overline{\mathcal{U}}_1 \\ \mathcal{U}_2 \\ \overline{\mathcal{U}}_2 \\ \mathcal{U}_3 \\ \overline{\mathcal{U}}_3 \end{pmatrix} = \begin{pmatrix} \mathbf{b}_1 & \mathbf{b}_2 & \dots & \mathbf{b}_{N-1} & \mathbf{b}_N \\ \overline{\mathbf{b}}_1 & \overline{\mathbf{b}}_2 & \dots & \overline{\mathbf{b}}_{N-1} & \overline{\mathbf{b}}_N \\ \mathbf{b}_{i_1} & \mathbf{b}_{i_2} & \dots & \mathbf{b}_{i_{N-1}} & \mathbf{b}_{i_N} \\ \overline{\mathbf{b}}_{i_1} & \overline{\mathbf{b}}_{i_2} & \dots & \overline{\mathbf{b}}_{i_{N-1}} & \overline{\mathbf{b}}_{i_N} \\ \mathbf{b}_{j_1} & \mathbf{b}_{j_2} & \dots & \mathbf{b}_{j_{N-1}} & \mathbf{b}_{j_N} \\ \overline{\mathbf{b}}_{j_1} & \overline{\mathbf{b}}_{j_2} & \dots & \overline{\mathbf{b}}_{j_{N-1}} & \overline{\mathbf{b}}_{j_N} \end{pmatrix} \tag{10}$$

where $\mathsf{l}_1 = (i_1, i_2, \dots, i_N) = \pi_3(1, 2, \dots, N)$ and $\mathsf{l}_2 = (j_1, j_2, \dots, j_N) = \pi_5(1, 2, \dots, N)$ with $\pi_3(.)$ and $\pi_5(.)$ as permutation functions. From this construction ($a = 6$ and $L = \log N$), $k = 6L = 6\log N$.

After all singletons are eliminated, all remaining right nodes are considered to be doubletons. Given doubleton i, $\mathbf{z}_i$ are its outcomes. From Eqn. (8), suppose that f_1 and f_2 are the two defective items in $\mathbf{z}_i$:

$$\mathbf{z}_i = \mathcal{U}_{,f_1} \vee \mathcal{U}_{,f_2} = \begin{pmatrix} \mathbf{b}_{f_1} \\ \overline{\mathbf{b}}_{f_1} \\ \mathbf{b}_{i_{f_1}} \\ \overline{\mathbf{b}}_{i_{f_1}} \\ \mathbf{b}_{j_{f_1}} \\ \overline{\mathbf{b}}_{j_{f_1}} \end{pmatrix} \vee \begin{pmatrix} \mathbf{b}_{f_2} \\ \overline{\mathbf{b}}_{f_2} \\ \mathbf{b}_{i_{f_2}} \\ \overline{\mathbf{b}}_{i_{f_2}} \\ \mathbf{b}_{j_{f_2}} \\ \overline{\mathbf{b}}_{j_{f_2}} \end{pmatrix} \tag{11}$$

Two operations are defined here: $\boxplus$ for *removing a vector from a Boolean vector* and $\overline{\boxplus}$ for *recovering a vector from its complement.* Operation $\boxplus$, which is aimed at removing a known defective item in a doubleton, is defined for two Boolean variables a and b as

$$c = b \boxplus a = \begin{cases} b \text{ if } a = 0 \\ ? \text{ if } a = 1 \end{cases}$$

For two vectors $\mathbf{x} = (x_1, \ldots, x_n) \in \{0,1\}^n$ and $\mathbf{y} = (y_1, \ldots, y_n) \in \{0,1\}^n$, we define

$$\mathbf{x} \boxplus \mathbf{y} = (x_1 \boxplus y_1, \ldots, x_n \boxplus y_n) \tag{12}$$

For example, suppose that $\mathbf{x} = (1,0,0,1,1,0)$ and $\mathbf{y} = (0,0,0,1,0,1)$; then $\mathbf{x} \boxplus \mathbf{y} = (1,0,0,?,1,?)$.

Operation $\overline{\boxplus}$ which is aimed at 'filling up' an incomplete vector, is defined for two Boolean variables a and b as

$$c = b \overline{\boxplus} a = \begin{cases} b \text{ if } b \neq ? \\ \overline{a} \text{ if } b = ? \end{cases}$$

For two vectors, $\mathbf{x} = (x_1, \ldots, x_n) \in \{0,1,?\}^n$ and $\mathbf{y} = (y_1, \ldots, y_n) \in \{0,1,?\}^n$, we define

$$\mathbf{x} \overline{\boxplus} \mathbf{y} = (x_1 \overline{\boxplus} y_1, \ldots, x_n \overline{\boxplus} y_n) \tag{13}$$

For example, suppose that $\mathbf{x} = (1,0,0,?,1,?)$ and $\mathbf{y} = (0,1,1,0,1,1)$; then $\mathbf{x} \overline{\boxplus} \mathbf{y} = (1,0,0,1,1,0)$.

Then, if any defective item of f_1 and f_2, say f_1, is already identified, we can identify the signature vector of f_2. First, we compute $\mathbf{r} = \mathbf{z}_i \boxplus \mathcal{U}_{,f_1} = (\mathbf{r}_1, \mathbf{r}_2, \mathbf{r}_3, \mathbf{r}_4, \mathbf{r}_5, \mathbf{r}_6)$ and get signature vector $\mathbf{r}' = \mathbf{r} \overline{\boxplus} (\mathbf{r}_2, \mathbf{r}_1, \mathbf{r}_4, \mathbf{r}_3, \mathbf{r}_6, \mathbf{r}_5) = (\mathbf{r}'_1, \mathbf{r}'_2, \mathbf{r}'_3, \mathbf{r}'_4, \mathbf{r}'_5, \mathbf{r}'_6)$. Next, we compute $\text{ind} = base10(\mathbf{r}'_1), \text{ind}_3 = base10(\mathbf{r}'_3)$, and $\text{ind}_5 = base10(\mathbf{r}'_5)$. If a right node is a doubleton and composed of defective item of f_1 and f_2, we must have $f_2 = \text{ind} = \pi_3^{-1}(\text{ind}_3) = \pi_5^{-1}(\text{ind}_5)$, where $\pi_3^{-1}()$ and $\pi_5^{-1}()$ are the inverse functions of $\pi_3()$ and $\pi_5()$, respectively. Decoding resolvable doubletons is described from lines 11 to 25.

D. Drawback analysis

There are two main drawbacks of the SAFFRON scheme:

1) There is no strongly explicit construction for incidence matrices $\mathcal{H}$. As a consequence, there is no strongly explicit construction for generating d-disjunct matrices.
2) False positives occur with probability $O\left(\frac{C(\epsilon)d}{N^2}\right)$. Moreover, the SAFFRON scheme identifies at least $(1-\epsilon)d$ defective items with probability $1 - O\left(\frac{C(\epsilon)d}{N^2}\right)$.

The first drawback is due to incidence matrix $\mathcal{H}$, which is generated uniformly and randomly. This is a characteristic of the sparse graphs the authors used.

The second is due to a signature matrix $\mathcal{U}$. Since $\mathcal{U}$ is a 1-disjunct matrix, the Boolean sum of two signature vectors can be equal to the Boolean of two or more than two other signature vectors. For example, consider the case of $N = 8$ for the signature matrix in Eqn. (9). In this case, $\mathcal{U}_{,1} = (0,0,0,1,1,1)^T$, $\mathcal{U}_{,2} = (0,0,1,1,1,0)^T$, $\mathcal{U}_{,3} = (0,1,1,1,0,0)^T$, $\mathcal{U}_{,5} = (1,1,0,0,0,1)^T$, and $\mathcal{U}_{,8} = (1,1,1,0,0,1)^T$. However, $\mathcal{U}_{,1} \vee \mathcal{U}_{,8} = \mathcal{U}_{,2} \vee \mathcal{U}_{,5} = \mathcal{U}_{,2} \vee \mathcal{U}_{,3} \vee \mathcal{U}_{,5}$. To remedy this problem, the authors propose a 'repetitive' signature matrix created by adding two more signature matrices generated by permuting $\mathcal{U}$ in Eqn. (9). However, the resulting matrices are still 1-disjunctive. The new signature matrix in Eqn. (10) may result om false positive (but the probability is low). Precisely, suppose f_0 is the identified defective item and $\mathbf{z}_i$ is the right node that we assume as a doubleton, let denote $\mathbf{r} = (\mathbf{r}_{f_0}, \overline{\mathbf{r}}_{f_0}, \mathbf{r}_{i_{f_1}}, \overline{\mathbf{r}}_{i_{f_1}}, \mathbf{r}_{i_{f_2}}, \overline{\mathbf{r}}_{i_{f_2}})$ as a vector after using algorithm in subsection III-C. Then we compute $\text{ind} = base10(\mathbf{r}_1), \text{ind}_3 = base10(\mathbf{r}_3), \text{ind}_5 = base10(\mathbf{r}_5)$. Let assume that doubleton $\mathbf{z}_i$ **does not** contain a singleton f_0. $\mathbf{r}$ is claimed as a signature vector of a defective item $f_2 = \text{ind}$ if and only if $\text{ind}_3 = \text{ind}$ and $\text{ind}_5 = \text{ind}$. The probability that even occurs is at most $\frac{1}{N^2}$ because there are N signature vectors in total. Since there are $C(\epsilon)d$ right nodes, the number of wrong defective items is at most $O\left(\frac{C(\epsilon)d}{N^2}\right)$.

We overcome the first drawback by using a strongly explicit construction for 2-disjunct matrices. We then use these matrices as a signature matrix to overcome the second drawback for when the maximum number of defective items is greater than 2 in the next section (although there is no strongly explicit construction here).

Algorithm 3 SAFFRON scheme decoding algorithm

Input: $\mathcal{M}, \mathcal{U}, \mathbf{y} = \mathcal{M}\mathbf{x} = (\mathbf{y}_1^1, \mathbf{y}_1^2, \ldots, \mathbf{y}_1^6, \mathbf{y}_2^1, \ldots, \mathbf{y}_2^6, \ldots, \mathbf{y}_h^1, \ldots, \mathbf{y}_h^6)^T \in (\{0,1\}^L)^h$ where $|\mathbf{x}| \leq d$, and $G = \oslash$.

Output: $G = \{j | x_j = 1\}$ or $\mathbf{x}$

Initialisation: count $= 0$;

Find all singletons.

for $i = 1$ to h {*check all right nodes*} **do**
  if $(wt(\mathbf{y}_i^1) + wt(\mathbf{y}_i^2) \equiv L)$ {*satisfy singleton criteria*} **then**
    $g_{count} = \text{base10}(\mathbf{y}_i^1)$ {*find a defective item*}
    if $g_{count} \notin G$ {*check whether this infected item is already in* G} **then**
      $G = G + \{g_{count}\}$
    end if
    $\mathbf{y}' = (\mathbf{y}_1^1, \ldots, \mathbf{y}_1^6, \ldots, \mathbf{y}_{i-1}^1, \ldots, \mathbf{y}_{i-1}^6, \mathbf{y}_{i+1}^1, \ldots, \mathbf{y}_{i+1}^6, \ldots, \mathbf{y}_h^1, \ldots, \mathbf{y}_h^6)$ {*Remove singleton*}
    count $=$ count $+ 1$ {*the size of* $\mathbf{y}'$}
  end if
end for

Find all remaining defective items using doubletons

for $l = 1$ to $|G|$ **do**
  sig $= (\mathcal{U}_{g_l}^T, \ldots, \mathcal{U}_{g_l}^T) = (\mathbf{u}_l^1, \ldots, \mathbf{u}_l^6, \ldots, \mathbf{u}_l^1, \ldots, \mathbf{u}_l^6) \in (\{0,1\}^L)^{6(h-\text{count})}$
  VTest $= \mathbf{y}' \boxplus$ sig $= (\mathbf{vt}_1^1, \mathbf{vt}_1^2, \ldots, \mathbf{vt}_1^6, \mathbf{vt}_2^1, \ldots, \mathbf{vt}_2^6, \ldots, \mathbf{vt}_{h-\text{count}}^1, \ldots, \mathbf{vt}_{h-\text{count}}^6)$
  mask $= (\mathbf{vt}_1^2, \mathbf{vt}_1^1, \mathbf{vt}_1^4, \mathbf{vt}_1^3, \mathbf{vt}_1^6, \mathbf{vt}_1^5, \mathbf{vt}_2^2, \mathbf{vt}_2^1, \ldots, \mathbf{vt}_2^6, \mathbf{vt}_2^5, \ldots, \mathbf{vt}_{h-\text{count}}^2, \ldots, \mathbf{vt}_{h-\text{count}}^5)$
  VTest $=$ VTest $\overline{\boxplus}$ mask.

Find all singletons in VTest

  for $i = 1$ to $h -$ count **do**
    if $(wt(\mathbf{vt}_i^1) + wt(\mathbf{vt}_i^2) \equiv L)$ and $(wt(\mathbf{vt}_i^3) + wt(\mathbf{vt}_i^4) \equiv L)$ and $(wt(\mathbf{vt}_i^5) + wt(\mathbf{vt}_i^6) \equiv L)$ **then**
      $\text{ind}_1 = \text{base10}(\mathbf{vt}_i^1); \text{ind}_3 = \text{base10}(\mathbf{vt}_i^3); \text{ind}_5 = \text{base10}(\mathbf{vt}_i^5)$
      if $\pi_3^{-1}(\text{ind}_3) = \text{ind}_1$ and $\pi_5^{-1}(\text{ind}_5) = \text{ind}_1$ and $\text{ind}_1 \notin G$ **then**
        $G = G + \{\text{ind}_1\}$
      end if
    end if
  end for
end for
return G

IV. Main results

A. Efficiently decodable 2-disjunct matrix ($d = 2$)

Before going to the main result, we state the following lemma which is helpful to achieve a strongly explicit construction of 2-disjunct matrices.

Lemma 1. *If C_{out} is an q-ary d-dsjunct matrix and C_{in} is a binary d-dsjunct matrix with q columns, the matrix generated by taking all codewords of the concatenation $C = C_{out} \circ C_{in}$ as its columns is d-dsjunctive.*

Proof:

Let C_{out} be a $t_1 \times N$ q-ary d-dsjunct matrix $\mathcal{A}$, C_{in} be a $t_2 \times q$ d-dsjunct matrix $\mathcal{B}$, and $\mathcal{M}$ be the $t \times N$ matrix generated by C. If we pick an arbitrary column of $\mathcal{A}$, say $\mathcal{A}_{,j_0}$, and d other columns $\mathcal{A}_{,j_0}, \mathcal{A}_{,j_1}, \ldots, \mathcal{A}_{,j_d}$, there exists a row, say i_0 such that $a_{i_0 j_0} \neq a_{i_0 j_l}$ for $l = 1, 2, \ldots, d$ (the definition of a

q-ary d-dsjunct matrix is given in subsection II-A). We then have $\mathcal{M}_{,j_0} = C_{in}(a_{i_0 j_0}) \neq \mathcal{M}_{,j_l} = C_{in}(a_{i_0 j_l})$ for $l = 1, 2, \ldots, d$. Since C_{in} is d-disjunctive, there exists a row i'_0 such that $m_{i'_0 j_0} = 1$ and $m_{i'_0 j_l} = 0$ for $l = 1, 2, \ldots, d$. Thus, M is d-disjunctive. ■

Theorem 4. *For any $\epsilon > 0$, there exists a strongly explicit construction of $t \times N$ 2-dsjunct matrices such that $t = 16 \log N$, each entry can be generated in $O(\ln N / \ln \ln N)$, and it can be decoded in time $O\left(\left(\frac{\ln N}{\ln \ln N} + \alpha\right) \frac{\ln N}{\ln \ln N}\right)$ with probability at least $1 - \epsilon$ where $\alpha = \frac{\ln \epsilon}{\ln 0.9854}$.*

Proof: We have to prove the conditions:

- There exists a strongly explicit construction of $t \times N$ 2-dsjunct matrices such that $t = 16 \log N$ and each entry can be generated in time $O(\ln N / \ln \ln N)$.
- Defective items can be identified in time $O\left(\frac{\ln^2 N}{\ln^2 \ln N}\right)$.

i) We start proving the first condition. Since we want to utilize Theorem 1, choose a Reed-Solomon code $[n_2 = 2^m = 2k_2, k_2 = 2^{m-1}, k_2 + 1]_{q=2^m}$ as C_{out} and an $n_3 \times q$ 2-disjunct binary matrix $\mathcal{A}$ as C_{in}, where $W(.)$ is Lambert W function such that $W(z)e^{W(z)} = z$. Then we carefully pick $k_2 = \frac{\ln N}{W(2 \ln N)}$. For example, if $N = 2^{32}$, we set $q = n_2 = 2^4 = 16, m = 4$, and $k_2 = 2^4/2 = 2^3 = 8$. Notice that $q^{k_2} = 2^{32} = N$.

The number of codewords of $C = C_{out} \circ C_{in}$ is given by

$$
\begin{aligned}
|C| &= q^{k_2} = \left(\frac{2 \ln N}{W(2 \ln N)}\right)^{\frac{\ln N}{W(2 \ln N)}} && (14)\\
&= \left(\left(e^{W(2 \ln N)}\right)^{e^{W(2 \ln N)}}\right)^{\frac{1}{2}} && (15)\\
&= \left(e^{W(2 \ln N) e^{W(2 \ln N)}}\right)^{\frac{1}{2}} = \left(e^{2 \ln N}\right)^{\frac{1}{2}} && (16)\\
&= N && (17)
\end{aligned}
$$

Eqn. (15) is derived from property of the Lambert function because $\frac{z}{W(z)} = e^{W(z)}$.

Because C_{out} is a $[n_2 = 2k_2, k_2]_{2k_2}$ Reed-Solomon code, if we pick a codeword, say $\mathbf{c}_0$ and another codeword of C_{out}, say $\mathbf{c}_1$, the maximum number of positions $\mathbf{c}_0$ agrees with $\mathbf{c}_1$ is $k_2 - 1$. Therefore, if we pick arbitrarily a codeword, say $\mathbf{c}_0$ and two other codewords of C_{out}, say $\mathbf{c}_1, \mathbf{c}_2$, the maximum positions $\mathbf{c}_0$ agrees with $\mathbf{c}_1$ or $\mathbf{c}_2$ are $2(k_2 - 1)$. So, there exist a row in which a position of $\mathbf{c}_0$ does not agree with $\mathbf{c}_1$ or $\mathbf{c}_2$. From the definition of q-ary d-disjunct in subsection II-A, C_{out} is a $2k_2$-ary 2-disjunct matrix. Porate et al. [19] gave an explicit construction of $\mathcal{A}$ such that $n_3 = \Theta(2^2 \log q) \simeq 8 \log q$. Since C_{in} is a 2-disjunct matrix, $C = C_{out} \circ C_{in}$ is a 2-disjunct matrix because of Lemma 1. We denote the 2-disjunct matrix obtained from C as $\mathcal{M}$.

Since $q = n_2$ and $n_3 = 8 \log q$, the number of rows (tests) of $\mathcal{M}$ or the length of a codeword of $C = C_{out} \circ C_{in}$ is:

$$t = n_2 \times n_3 = 8q \log q = 8 \times 2 \log N = 16 \log N \tag{18}$$

We prove that each entry of $\mathcal{M}$ can be generated in time $O(\ln N / \ln \ln N)$. Indeed, each entry in a codeword of C_{out} can be generated in time $(k_2 - 1) \log n_2$ because a Reed-Solomon code is used. Since each entry of a codeword is concatenated with the corresponding column of C_{in}, the time complexity to

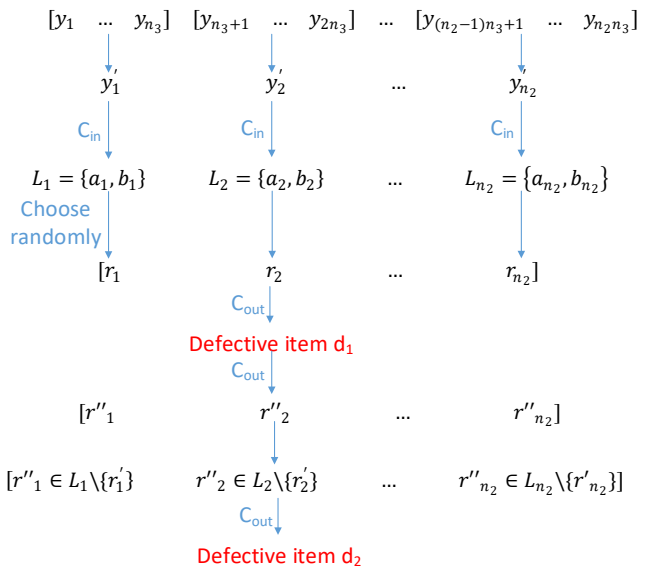

Fig. 3. Decoding procedure of 2-disjunct matrices

generate an entry of $\mathcal{M}$ is

$$(k_2-1)\log n_2 + (q \times n_3) < \frac{1}{2} q \log q + (q \times 8 \log q) \quad (19)$$

$$< \frac{9}{2}\left(\frac{2\ln N}{W(2\ln N)} \times 8 \log\left(\frac{2\ln N}{W(2\ln N)}\right)\right) \quad (20)$$

$$= O\left(\ln N / \ln\ln N\right) \quad (21)$$

where $W(x) \simeq \ln x$, and $\log\left(\frac{2\ln N}{W(2\ln N)}\right)$ is negligible.

Because each entry can be generated in time $O\left(\ln N / \ln\ln N\right)$ or each column of $\mathcal{M}$ can be generated in time $n_2 \times \ln N / \ln\ln N = \frac{2\ln^2 N}{\ln\ln N \cdot W(2\ln N)} = poly(t)$, $\mathcal{M}$ can be constructed explicitly.

ii) Next, we prove the second condition, i.e., that, given an outcome vector $\mathbf{y}$, defective items can be identified in time $O\left(\frac{\ln^2 N}{\ln^2 \ln N}\right)$. Of course $\mathbf{y} \neq 0$, otherwise there is no defective item in N items. First, $\mathbf{y}$ is divided into n_2 blocks, with each block having size n_3. We set $\mathbf{y}^T = (\mathbf{y}'_1, \mathbf{y}'_2, \dots, \mathbf{y}'_{n_2})$ and then find which column of C_{in} belongs to $\mathbf{y}'_i$, for $i = 1, 2, \dots, n_2$. This step takes time $q \times n_3$. Since $\mathbf{y}$ is the Boolean sum of at most two columns of $\mathcal{M}$, there are at most two columns of C_{in} belonging to $\mathbf{y}'_i$. We set $L_i = \{a_i, b_i\}$, where a_i and b_i are the indexes of the columns belonging to $\mathbf{y}'_i$, with $i = 1, 2, \dots, n_2$. This step takes time $n_2 \times qn_3$. If there is only a column belonging to $\mathbf{y}'_i$, we choose $b_i = a_i$. We depict this procedure in Figure 3.

If $a_i = b_i$ for all $i = 1, 2, \dots, n_2$, there is only one defective item. We just decode the vector $\mathbf{r} = (a_1, a_2, \dots, a_{n_2})$ to get the defective item. If $a_i \neq b_i$ for some $i \in \{1, 2, \dots, n_2\}$, there are two defective items, say d_1 and d_2. Let denote $\mathbf{m}_1$ and $\mathbf{m}_2$ be the corresponding messages of d_1 and d_2 in C_{out}, respectively. We take the vector $\mathbf{r} = (r_1, r_2, \dots, r_{n_2})$ where r_i is chosen randomly from L_i for $i = 1, 2, \dots, n_2$. We want to identify defective item d_1 or d_2 from $\mathbf{r}$. We consider $\mathbf{r}$ is a distorted codeword of $C_{out}(\mathbf{m}_1)$ or $C_{out}(\mathbf{m}_2)$. We can identify d_1 from $\mathbf{r}$ if (a) $\Delta\left(\mathbf{r}, C_{out}(\mathbf{m}_1)\right) \leq \frac{n_2-k_2}{2} = \frac{k_2}{2}$ or $\Delta\left(\mathbf{r}, C_{out}(\mathbf{m}_2)\right) \geq \frac{n_2+k_2}{2} = \frac{3}{2}k_2$. We can identify d_2 from $\mathbf{r}$ if (b) $\Delta\left(\mathbf{r}, C_{out}(\mathbf{m}_2)\right) \leq \frac{n_2-k_2}{2} = \frac{k_2}{2}$ or $\Delta\left(\mathbf{r}, C_{out}(\mathbf{m}_1)\right) \geq \frac{n_2+k_2}{2} = \frac{3}{2}k_2$. Because of the property of the decoding algorithm, we just input $\mathbf{r}$ then get d_1 if (a) is satisfied or d_2 if (b) is satisfied as Figure 4.

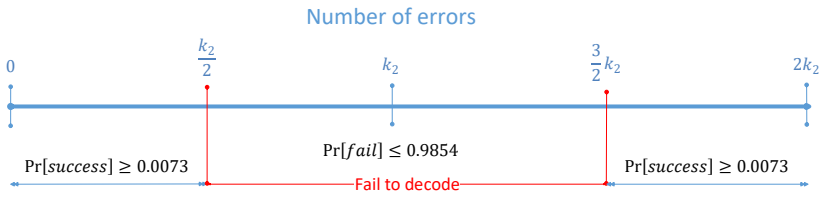

Fig. 4. Probability distribution of errors

Without loss of generality, we want to identify d_1 from $\mathbf{r}$. Because $|L_i| = 2$, for $i = 1, \ldots, n_2$, the probability of choosing a symbol belongs to $C_{out}(d_1)$ is $p = \frac{1}{2}$. We could model this scenario as a *binomial distribution* where choosing a symbol that belongs to $C_{out}(d_1)$ is a Yes/No experiment in which there are n_2 trials in total and the probability of a experiment is successful is $p = \frac{1}{2}$ (because $|L_i| = 2$). Let denote $\mathtt{X}$ be the total number of symbols in $\mathbf{r}$ agreeing with $C_{out}(d_1)$. Therefore, $\mathtt{X} \sim \mathbb{B}(n_2, p)$, where $\mathbb{B}(n_2, p)$ is the binomial distribution with the number of experiments as n_2 and the probability of successful experiments as p. Since $n_2 p > 5$ and $n_2(1-p) > 5$ for any $k_2 \geq 6$, from the central limit theorem, we can estimate $\mathbb{B}(n_2, p) \sim \mathbb{N}(n_2 p, n_2 p(1-p))$ where $\mathbb{N}(n_2 p, n_2 p(1-p))$ is the normal distribution with the mean as $\mu = n_2 p$ and the standard deviation as $\sigma = \sqrt{n_2 p(1-p)}$.

Since C_{out} is $[n_2 = 2^m = 2k_2, k_2 = 2^{m-1}, k_2 + 1]_{q=2^m}$, if the number of errors is equal or smaller than $\frac{k_2}{2}$, d_1 could be identified in time $1.5 \times q \log q$ (Theorem 1). We insist that if the number of errors exceeds $\frac{3}{2}k_2$, d_2 can be also identified in time $1.5 \times q \log q$ (Theorem 1). After identifying the defective item d_1 (d_2), we calculate the corresponding codeword of d_1 (d_2) in C_{out}. Let denote it by $\mathbf{r}' = (r'_1, r'_2, \ldots, r'_{n_2})$. We create a new codeword $\mathbf{r}'' = (r''_1, r''_2, \ldots, r''_{n_2})$ where $r''_i \in L_i \setminus \{r'_i\}$ for $i = 1, 2, \ldots, n_2$. Finally, we get d_2 (d_1) by decoding $\mathbf{r}''$. This step takes $O(n_2 \log n_2)$ because of Theorem 1.

Our task now is to estimate $\Pr[\text{fail}] = \Pr[\frac{k_2}{2} < \mathtt{X} < \frac{3}{2}k_2]$, i.e. the probability that $\mathbf{r}$ does not satisfy (a). In order to do that, we will estimate $\Pr[\text{success}] = \Pr[\mathtt{X} \leq \frac{k_2}{2}] = \Pr[\mathtt{X} \geq \frac{3}{2}k_2]$. Since $\mathtt{X} \sim \mathbb{N}(n_2 p, n_2 p(1-p))$, we will estimate z-score of the point $\frac{k_2}{2}$.

Now, we are going to calculate z-score of the point $\frac{k_2}{2}$ in $\mathbb{N}(n_2 p, n_2 p(1-p))$. When $p = \frac{1}{2}$ and $n_2 = 2k_2$, we have $\mu = \frac{2k_2}{2} = k_2$ and $\sigma = \frac{\sqrt{2k_2}}{2}$. Therefore, z-score of $\frac{k_2}{2}$ is:

$$z^* = \frac{\frac{k_2}{2} - k_2}{\frac{\sqrt{2k_2}}{2}} = -\frac{1}{\sqrt{2}}\sqrt{k_2} \tag{22}$$

We can see that z^* is a decreasing function of k_2 for any $k_2 > 0$. Therefore, for any $k_2 \geq 4$, z^* is decreased then $\Pr[\mathtt{X} \leq z^*\sigma + \mu]$ is increasing. If $k_2 \leq 12$ (which means[1] $N \leq 2^{55}$), $z^* \geq -2.4495$. After using z-score table, we could reference that for any $N \leq 2^{55}$, $\Pr[\text{success}] = \Pr[\mathtt{X} \leq \frac{k_2}{2}] = \Pr[\mathtt{X} \geq \frac{3}{2}k_2] \geq 0.0073$. Therefore, $\Pr[\text{fail}] = 1 - 2 \times \Pr[\text{success}] \leq 0.9854$.

For any $\epsilon > 0$, we repeat the decoding algorithm α times where $\alpha > \frac{\ln \epsilon}{\ln 0.9854}$, the probability that the bad even fails for α consecutive times is less than $0.9854^\alpha < \epsilon$. Finally, the probability that our decoding algorithm works is successful is at least $1 - \epsilon$.

[1]We notice that N can be greater than 2^{55}. In that case, we have to increase k_2, look up to z-table again to find a suitable z^*.

Therefore, the decoding complexity is:

$$
\begin{aligned}
& n_2 \times qn_3 + \alpha \times (n_2 + 1.5n_2 \log n_2) + 1.5n_2 \log n_2 && (23)\\
= \;& 8q^2 \log q + \alpha \times (q + 1.5q \log q) + 1.5q \log q && (24)\\
< \;& (8q + 2.5\alpha + 1.5)q \log q = O\left((q+\alpha)q \log q\right) && (25)\\
= \;& O\left(\left(\frac{\ln N}{\ln \ln N} + \alpha\right)\frac{\ln N}{\ln \ln N}\right) && (26)\\
= \;& O\left(\frac{\ln^2 N}{\ln^2 \ln N}\right) && (27)
\end{aligned}
$$

We get Eqn. (23) because it takes $n_2 \times qn_3$ to get L_i for $i = 1, \ldots, n_2$. Then we run our decoding algorithm for α times. Each time will create a new vector $\mathbf{r}$, i.e. takes time n_2, then decoding $\mathbf{r}$ takes time $1.5n_2 \log n_2$. Therefore, it takes $\alpha \times (n_2 + 1.5n_2 \log n_2)$ time to run α times. Finally, after a defective item d_1 or d_2 is identified, the remaining defective item can be identified in time $1.5n_2 \log n_2$. Because $n_2 = q$ and $n_3 = 8 \log q$, Eqn. (24) is achieved.

We get Eqn. (26) because $n_2 = q = \frac{2 \ln N}{W(2 \ln N)}$, $n_3 = 8 \log q$, $W(x) \simeq \ln x$, and $\log \frac{\ln N}{\ln \ln N}$ is negligible. Eqn. (27) is derived if we consider α as a constant that depends only on ϵ. ■

B. Proposed scheme for $d \geq 3$

If the number of defective items is up to 1, we use the signature matrix in Eqn. (9) as a 1-disjunct matrix. If the number of defective items is up to 2, we use the 2-disjunct matrices given in Theorem 4. If the number of defective items is greater than 2, we use the SAFFRON scheme with a signature matrix as a 2-disjunct matrix as given in Theorem 4. In our proposed scheme, the incidence matrix $\mathcal{H}$ is as a black box. We focus only on signature matrix $\mathcal{U}$. We use the signature matrix as described in Eqn. (9). Since $\mathcal{U}_1$ is a 2-disjunct matrix from Theorem 4, $L = n_2 = q$, $a = 2$ and the signature matrix size is $32 \log N \times N$. We present our scheme in Algorithm 4. This proposed scheme is stated in Theorem 5.

Theorem 5. *The proposed scheme identifies at least $(1-\epsilon)d$ defective items using $t = 32C(\epsilon)d \log N$ tests, where ϵ is an arbitrary constant close to 0, and $C(\epsilon)$ is a constant that depends only on ϵ.*

Proof: Since the failure probability in Theorem 3 occurs because the SAFFRON scheme could accuse false positives. We will prove that our proposed scheme never produce a false positive, meaning that the probability of failure vanishes. Indeed, we assert that using 2-disjunct matrices as signature matrices eliminates false positives. Let us consider right node i for $i = 1, 2 \ldots, h$. The test outcome of right node i is $\mathbf{y}_i^T = (\mathbf{y}_i^1, \mathbf{y}_i^2)$. We consider three case of $wt(\mathbf{y}_i)$. If $wt(\mathbf{y}_i) = 0$, there is no information here. If $wt(\mathbf{y}_i) = L$, $\mathbf{y}_i$ is singleton. If $wt(\mathbf{y}_i) > L$, $\mathbf{y}_i$ is a doubleton or composed of at least three signature vectors. First we consider $\mathbf{y}_i$ is a doubleton. Because U is a 2-disjunct matrix, for any pair distinguish a_1, a_2, a_3, a_4, $a_1 \vee a_2 \neq a_3 \vee a_4$. If $\mathbf{y}$ is a doubleton, we assume $\mathbf{y}_i = a \vee b$, where $a, b \in \mathcal{U}$, $a \neq b$, and a and b are unique. For any $c_1, c_2, \cdots, c_l \in \mathcal{U}$, where $l \geq 3$, it does not satisfy $c_1 \vee c_2 \vee \cdots \vee c_l = \mathbf{y}_i = a \vee b$ if $l \geq 3$ because there exist one row for which the union of $c_1, c_2, \ldots, c_l$ is 1 and the union of a and b is 0. If there exists c_1 and c_2 such that $c_1 \vee c_2 = \mathbf{y}_i = a \vee b$, $(c_1, c_2) = (a, b)$ or $(c_1, c_2) = (b, a)$ because of the definition of 2-disjunct matrix. Therefore, Algorithm 4 never produces a false positive.

Because the signature matrix size is $32 \log N \times N$ and the incidence matrix size is $C(\epsilon)d \times N$ as described in subsection III-A, the number of tests in our proposed scheme is $t = 32C(\epsilon)d \log N$. ■

V. Conclusion

We have presented a strongly explicit construction for 2-disjunct matrices and an explicit construction for d-disjunct matrices when $d \geq 3$. When $d = 2$, our scheme produces better results than state-of-the-art schemes in terms of the number of tests, construction, and decoding complexity. When $d \geq 3$, our scheme is better in term of false positives as well as in terms of the number of tests and decoding complexity.

Algorithm 4 Proposed scheme decoding algorithm

Input: $\mathcal{M}$,$\mathcal{U}$, $\mathbf{y}^T = \mathcal{M}\mathbf{x} = (\mathbf{y}_1^1, \mathbf{y}_1^2, \mathbf{y}_2^1, \mathbf{y}_2^2, \ldots, \mathbf{y}_h^1, \mathbf{y}_h^2) \in (\{0,1\}^L)^{2h}$ where $|\mathbf{x}| \leq d$, and $G = \oslash$.
Output: $G = \{j | x_j = 1\}$ or $\mathbf{x}$

Initialisation: count $= 0$.
Find all singletons.
for $i = 1$ to h **do**
  if $(wt(\mathbf{y}_i^1) + wt(\mathbf{y}_i^2) \equiv L)$ **then**
    $g_{count} = \text{base10}(\text{DecConcate}(\mathbf{y}_i^1))$
    if $g_{count} \notin G$ **then**
      $G = G + \{g_{count}\}$
    end if
    $\mathbf{y}' = (\mathbf{y}_1^1, \mathbf{y}_1^2, \ldots, \mathbf{y}_{i-1}^1, \mathbf{y}_{i-1}^2, \mathbf{y}_{i+1}^1, \mathbf{y}_{i+1}^2, \ldots, \mathbf{y}_h^1, \mathbf{y}_h^2) = (\mathrm{y}_1^1, \mathrm{y}_1^2, \ldots, \mathrm{y}_{h-\text{count}}^1, \mathbf{y}_{h-\text{count}}^2)$
    count $=$ count $+ 1$
  end if
end for
Find all remaining defective items using doubletons
for $l = 1$ to $|G|$ **do**
  sig $= (\mathcal{U}_{g_l}^T, \ldots, \mathcal{U}_{g_l}^T) = (\mathbf{u}_l^1, \mathbf{u}_l^2, \ldots, \mathbf{u}_l^1, \mathbf{u}_l^2) \in (\{0,1\}^L)^{2(h-\text{count})}$
  VTest $= \mathbf{y}' \boxplus$ sig $= (\mathbf{vt}_1^1, \mathbf{vt}_1^2, \mathbf{vt}_2^1, \mathbf{vt}_2^2, \ldots, \mathbf{vt}_{h-\text{count}}^1, \mathbf{vt}_{h-\text{count}}^2)$
  mask $= (\mathbf{vt}_1^2, \mathbf{vt}_1^1, \mathbf{vt}_2^2, \mathbf{vt}_2^1, \ldots, \mathbf{vt}_{h-\text{count}}^2, \mathbf{vt}_{h-\text{count}}^1)$
  VTest $=$ VTest $\overline{\boxplus}$ mask.
  Find all singletons in VTest
  for $i = 1$ to $h -$ count **do**
    if $(wt(\mathbf{vt}_i^1) + wt(\mathbf{vt}_i^2) \equiv L)$ **then**
      ind $=$ base10(DecConcate(vt_i^1));
      if $(\mathbf{vt}_i^1, \mathbf{vt}_i^2) \vee (\mathbf{u}_l^1, \mathbf{u}_l^2) \equiv (\mathrm{y}_i^1, \mathrm{y}_i^2)$ and $\text{ind}_1 \notin G$ **then**
        $G = G + \{\text{ind}\}$
      end if
    end if
  end for
end for
return G

REFERENCES

[1] R. Dorfman, "The detection of defective members of large populations," *The Annals of Mathematical Statistics*, vol. 14, no. 4, pp. 436–440, 1943.

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

[3] J. K. Wolf, "Born again group testing: Multiaccess communications," *Information Theory, IEEE Transactions on*, vol. 31, no. 2, pp. 185–191, 1985.

[4] Y. Erlich et al., "Biological screens from linear codes: theory and tools," *bioRxiv*, p. 035352, 2015.

[5] 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.

[6] M. Cheraghchi, "Noise-resilient group testing: Limitations and constructions," *Discrete Applied Mathematics*, vol. 161, no. 1, pp. 81–95, 2013.

[7] 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*, pp. 1126–1142, Society for Industrial and Applied Mathematics, 2010.

[8] H. Q. Ngo, E. Porat, and A. Rudra, "Efficiently decodable error-correcting list disjunct matrices and applications," in *International Colloquium on Automata, Languages, and Programming*, pp. 557–568, Springer, 2011.

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

[10] H. Q. Ngo, E. Porat, and A. Rudra, "Efficiently decodable error-correcting list disjunct matrices and applications," in *Automata, Languages and Programming*, pp. 557–568, Springer, 2011.
[11] G. Cormode and S. Muthukrishnan, "What's hot and what's not: tracking most frequent items dynamically," *ACM Transactions on Database Systems (TODS)*, vol. 30, no. 1, pp. 249–278, 2005.
[12] 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*, pp. 1234–1241, IEEE, 2013.
[13] I. S. Reed and G. Solomon, "Polynomial codes over certain finite fields," *Journal of the society for industrial and applied mathematics*, vol. 8, no. 2, pp. 300–304, 1960.
[14] S. B. Wicker and V. K. Bhargava, *Reed-Solomon codes and their applications*. John Wiley & Sons, 1999.
[15] S.-J. Lin, T. Y. Al-Naffouri, and Y. S. Han, "Fft algorithm for binary extension finite fields and its application to reed–solomon codes," *IEEE Transactions on Information Theory*, vol. 62, no. 10, pp. 5343–5358, 2016.
[16] G. D. Forney, *Concatenated codes*, vol. 11. Citeseer, 1966.
[17] V. Guruswami, R. Atri, and S. Madhu, *Essential Coding Theory*. 2014.
[18] R. N. Gould and C. N. Ryan, *Introductory statistics: Exploring the world through data*. Pearson Higher Ed, 2012.
[19] E. Porat and A. Rothschild, "Explicit non-adaptive combinatorial group testing schemes," in *International Colloquium on Automata, Languages, and Programming*, pp. 748–759, Springer, 2008.