Improved algorithms for non-adaptive group testing with consecutive positives

Thach V. Bui*†, Mahdi Cheraghchi‡, and Thuc D. Nguyen*†

*Faculty of Information Technology, University of Science, Ho Chi Minh city, Vietnam
†Vietnam National University, Ho Chi Minh city, Vietnam
{bvthach,ndthuc}@fit.hcmus.edu.vn

‡University of Michigan, Ann Arbor MI, USA
mahdich@umich.edu

***Abstract*—The goal of group testing is to efficiently identify a few specific items, called positives, in a large population of items via tests. A test is an action on a subset of items that returns positive if the subset contains at least one positive and negative otherwise. In non-adaptive group testing, all tests are independent, can be performed in parallel, and represented as a measurement matrix. In this work, we consider non-adaptive group testing with consecutive positives in which the items are linearly ordered and the positives are consecutive in that order.**

We present two algorithms for efficiently identifying consecutive positives. In particular, without storing measurement matrices, we can identify up to d consecutive positives with $2\log_2\frac{n}{d}+2d$ ($4\log_2\frac{n}{d}+2d$, resp.) tests in $O\left(\log_2^2\frac{n}{d}+d\right)$ ($O\left(\log_2\frac{n}{d}+d\right)$, resp.) time. These results significantly improve the state-of-the-art scheme in which it takes $5\log_2\frac{n}{d}+2d+21$ tests to identify the positives in $O\left(\frac{n}{d}\log_2\frac{n}{d}+d^2\right)$ time with the measurement matrices associated with the scheme stored somewhere.

I. Introduction

A. Group testing

The goal of group testing (GT) is to efficiently identify up to d positive items in a large population of n items. Positive items satisfy some specific properties while negative items do not. Emerged by the seminal work of Dorfman [2] in WW2, GT was considered as an efficient way to save time and money in identifying syphilitic draftees among a large population of draftees. With the ongoing Covid-19 pandemic since 2020, GT has been found to be an efficient tool for mass testing to identify infected persons [3], [4]. Instead of testing each item one by one to verify whether it is positive or negative, a group of items is pooled then tested. In the noiseless setting, the outcome of a test on a group of items is positive if the group has at least one positive and negative otherwise.

There are two basic approaches to designing tests. The first is *adaptive* group testing in which the design of a test depends on the designs of the previous tests. The second approach, which is *non-adaptive* group testing (NAGT), is to design all tests independently such that they can be performed simultaneously. Because of time saving, NAGT has been widely applied in various fields such as biology [5] and networking [6].

NAGT can be represented by a measurement matrix in which an entry at row i and column j equals to 1 indicates that the jth item in the input set belongs to test i and that item does not belong to test i otherwise. The procedure to get the matrix is called *construction*, the procedure to get the outcomes of all tests using the matrix is called *encoding*, and the procedure to get positives from the outcomes is called *decoding*.

A measurement matrix is random if it satisfies the preconditions after the construction procedure with high probability. Meanwhile, a measurement matrix is explicit if it can be constructed in $\mathsf{poly}(d,n)$ time. However, random and explicit matrices are not ideal in practice because they are usually to be saved somewhere before use. A matrix being a good fit in practice is strongly explicit in which every entry in the matrix can be generated in $\mathsf{poly}(d,\log n)$ time. This implies that it is unnecessary to store the matrix.

There are two main requirements to tackle group testing: minimize the number of tests and efficiently identify the set of positive items. To combinatorial GT, several schemes have been proposed to achieve either a small number of tests, says $O(d^{1+o(1)}\ln^{1+o(1)} n)$, or low decoding complexity, says $\mathsf{poly}(d,\ln n)$, such as [5], [7]–[15].

B. Group testing with consecutive positives

Inspired by genetic mapping and sequencing for linear DNA [16], [17], Colbourn [18] firstly considered a specific case of group testing called *group testing with consecutive positives* in which the input items are linearly ordered and the positives are consecutive in that order. In this setting, the number of tests required can be reduced to $O(\log(dn))$ and $O\left(\log\frac{n}{d-1}+d\right)$ for adaptive and non-adaptive designs, respectively, which is much smaller than the bounds $O(d\log n)$ and $\Omega(d^2\log\frac{n}{d})$ in combinatorial group testing. Juan and Chang [19] could make the number of tests fall in a very tight interval $[\lceil\log_2(dn)\rceil-1,\lceil\log_2(dn)\rceil+1]$ in adaptive design.

With non-adaptive approach, Muller and Jimbo [20] considered the case $d=2$ and could construct an explicit measurement matrix with $\lceil\log_2\lceil\frac{n}{d-1}\rceil\rceil+2d+1$ rows and n columns. Unfortunately, neither Colbourn nor Muller and Jimbo showed how to efficiently identify positives. Chang et al. [21] later used random measurement matrices with $5\log_2\frac{n}{d}+2d+21$ rows to identify all positives in $O\left(\frac{n}{d}\log_2\frac{n}{d}+d^2\right)$ time. The focus of this work is on NAGT with consecutive positives.

C. Contributions

We have reduced the number of tests and the decoding complexity without storing measurement matrices for effi-

The full version of this paper is available at [1].

ciently identifying up to d consecutive positives. In particular, without storing measurement matrices, we can identify up to d consecutive positives with $2\log_2(n/d)+2d$ ($4\log_2(n/d)+2d$, resp.) tests in $O\left(\log_2^2(n/d)+d\right)$ ($O\left(\log_2(n/d)+d\right)$, resp.) time. These results significantly improve the state-of-the-art scheme [21] in which it takes $5\log_2(n/d)+2d+21$ tests to identify the positives in $O\left(n/d \cdot \log_2(n/d)+d^2\right)$ time with the measurement matrices associated with the scheme stored somewhere. A summary of our comparison is shown in Table I.

D. General idea of improved algorithms

Although our improved algorithms reflect Colbourn's strategy [18], we refine every technical detail to attain efficient encoding and decoding procedures. More importantly, Colbourn only designed measurement matrices but not decoding procedures. Here, we have presented two improved algorithms to identify up to d consecutive positives. Colbourn's strategy consists of two simultaneous phases. In the first phase, he partitioned the n (linearly ordered) items into subpools in which we here call them *super items* such that there are up to two super positive items and if two super items are positive, they are consecutive. Hence, the objective of this phase is to locate the super positives among the super items. In the second phase, with the careful design of measurement matrices, the true positives can be identified based on the location of the super positives. Briefly speaking, for the first phase, Colbourn's original strategy used Gray code while ours uses strongly explicit matrices. For the second phase, Colbourn used $2d-2$ tests while we used $2d$ tests.

Our improved algorithms are based on two inseparable compartments: the linear order of the input items which contain consecutive positives and strongly explicit matrices designed based on that linear order. We first create *super items* with linear order in which each super item contains exactly d consecutive items. Naturally, a super item is positive if it contains at least one positive item and negative otherwise. For the first phase, from the original set of items $N=[n]=\{1,2,\ldots,n\}$, we generate a subset (or two subsets) of super items with linear order and their corresponding measurement matrix (matrices). For the second phase, we simply generate a $2d\times n$ measurement matrix by horizontally placing $2d\times 2d$ identity matrices in a series.

The decoding procedure is as follows. By using the input set(s) of super items with their corresponding measurement matrix (matrices), given an outcome vector(s) generated from them, we can recover up to two super positive items, i.e., there are up to $2d$ potential positives after using super items. Because the measurement matrix associated with the set of n items is composed by a series of $2d\times 2d$ identity matrices, we finally can verify which potential positive is truly positive.

II. Preliminaries

For simplicity, we assume n is divisible by d (in case n is not divisible by d, we can add $d\lceil n/d\rceil - n$ dummy negative items into the set of items such that the total number of items is $d\lceil n/d\rceil$). Any set $C=\{c_1,\ldots,c_k\}$ used in this work is equipped with the linear order $c_i \prec c_{i+1}$ for $1\le i<k$, where $\prec$ is the linear order notation. There are n items indexed from 1 to n. Two sets of items are considered throughout this paper, which are $N=[n]=\{1,2,\ldots,n\}$ and $P=\{2,3,\ldots,n\}$. We should keep in mind that the index of an item may be *different* to its position in a set of items containing it, i.e., the jth item in a set may be not item j. Precisely, the position of an item in N is identical to its index. However, the position of an item in P is one unit smaller than to its index. For example, let us consider two sets $N=\{1,2,3,4\}$ and $P=\{2,3,4\}$ for $n=4$. The position of item 2 in set N is 2, which is identical to the index of item 2, but its position in set P is 1.

A. Notations

We use capital calligraphic letters for matrices, bold letters for vectors, and capital letters for sets. All matrix and vector entries are binary. Parameters n and d are the number of items and the maximum number of positives. Row i of matrix $\mathcal{T}$ is denoted to as $\mathcal{T}_{i,*}$. The ith entry in the vector $\mathbf{v}$ is $\mathbf{v}(i)$. Calculating an index from a vector means we convert that vector from the binary representation to the decimal one. For example, if the input is 011 then the index is $0\times 2^0+1\times 2^1+1\times 2^2=6$. Finally, $\log$ is the logarithm to base 2.

B. Problem definition

We index the population of n items from 1 to n. Let P be the positive set, where $|P|\le d$. The outcome of a test on a subset of items is positive if the subset contains at least one positive, is negative otherwise. We can model NAGT with consecutive positives as follows. A $t\times n$ binary matrix $\mathcal{T}=(t_{ij})$ is defined as a measurement matrix, where n is the number of items and t is the number of tests. Vector $\mathbf{x}=(x_1,\ldots,x_n)^T$ is the binary representation vector of n items, where there exists two indexes j_1 and l such that $x_{j_1+1}=x_{j_1+2}=\ldots=x_{j_1+l}=1$ and $x_j=0$ when $j\in N\setminus\{j_1+1,\ldots,j_1+l\}$ for $l\le d$. An entry $x_j=1$ indicates that item j is positive, and $x_j=0$ indicates otherwise. The jth item corresponds to the jth column of the matrix. An entry $t_{ij}=1$ naturally means that item j belongs to test i, and $t_{ij}=0$ means otherwise. The outcome of all tests is $\mathbf{y}=(y_1,\ldots,y_t)^T$, where $y_i=1$ if test i is positive and $y_i=0$ otherwise. Outcome vector $\mathbf{y}$ is given by $\mathbf{y}=\mathcal{T}\odot\mathbf{x}=[\mathcal{T}_{1,*}\odot\mathbf{x},\ldots,\mathcal{T}_{t,*}\odot\mathbf{x}]^T=[y_1,\ldots,y_t]^T$, where $\odot$ is a notation for the test operation in non-adaptive group testing, $y_i=\mathcal{T}_{i,*}\odot\mathbf{x}=1$ if $\sum_{j=1}^n x_j t_{ij}\ge 1$, and $y_i=\mathcal{T}_{i,*}\odot\mathbf{x}=0$ if $\sum_{j=1}^n x_j t_{ij}=0$ for $i=1,\ldots,t$.

Our objective is to find an efficient encoding and decoding scheme to identify up to d consecutive positives in NAGT by minimizing t and the time for recovering $\mathbf{x}$ from $\mathbf{y}$.

III. Identification of two consecutive positives

A. Overview

Our objective is to identify positives in a set of items which contains exactly two consecutive positives or up to one positive. This will serve as a building block for the general case in Section IV. The basic idea of our proposed scheme is to exploit the structure of a measurement matrix and the linear

Scheme	No. of positives	Design approach	Construction type of measurement matrices	Number of tests t	Decoding time (Decoding complexity)
Colbourn [18]	$\leq d$	Adaptive	Not available	$\lceil \log_2(dn) \rceil + c$	t stages
Juan and Chang [19]	$\leq d$	Adaptive	Not available	$\lceil \log_2(dn) \rceil - 1 \leq t \leq \lceil \log_2(dn) \rceil + 1$	t stages
Colbourn [18]	$\leq d$	Non-adaptive	Explicit	$\lceil \log_2 \lceil \frac{n}{d-1} \rceil \rceil + 2d + 1$	Not available
Muller and Jimbo [20]	$d = 2$	Non-adaptive	Explicit	$\lceil \log_2 \lceil \frac{n}{d-1} \rceil \rceil + 2d - 1$	Not available
Chang et al. [21]	$\leq d$	Non-adaptive	Random	$5\lceil \log_2 \frac{n}{d} \rceil + 2d + 21$	$O\left(\frac{n}{d} \log_2 \frac{n}{d} + d^2 \right)$
First improved algorithm (Theorem 1) **Second improved algorithm (Theorem 2)**	$\leq d$	**Non-adaptive**	**Strongly explicit**	$2\lceil \log_2 \frac{n}{d} \rceil + 2d$ $4\lceil \log_2 \frac{n}{d} \rceil + 2d$	$O\left(\log_2^2 \frac{n}{d} + d \right)$ $O\left(\log_2 \frac{n}{d} + d \right)$

TABLE I

COMPARISON OF IMPROVED ALGORITHMS WITH PREVIOUS ONES. "NOT AVAILABLE" MEANS THAT THE CRITERION DOES NOT HOLD OR IS NOT CONSIDERED FOR THAT SCHEME. PARAMETER c IS SOME CONSTANT.

order of n items. We create a strongly explicit measurement matrix such that the union of any two consecutive columns in it is different from the union of other two consecutive columns. Based on this property and the measurement matrix structure, we carefully develop a decoding scheme accordingly whose decoding time is up to the square of the number of measurements. The first encoding and decoding procedures are described in Algorithm 1 and summarized in Lemma 1.

Lemma 1: Let n be a positive integer and N be the set of linearly ordered items. Then there exists a strongly explicit $2\lceil \log n \rceil \times n$ measurement matrix such that: if N has exactly two positives which are consecutive and the index of the first positive is $1 \leq a \leq n-1$, the two positives can be identified with $s = 2\lceil \log n \rceil$ tests in s time if a is odd and in $s^2 = O(\log^2 n)$ time if a is even; and if N has up to one positive, the decoding complexity is $s = 2\lceil \log n \rceil$.

Since the decoding complexity in Lemma 1 is $O(\log^2 n)$ which is larger $O(\log n)$, our next objective is to design an encoding procedure such that its decoding complexity is just $O(\log n)$ by exploiting properties of consecutive positives. Lemma 1 tells us that if the index of the first positive in a measurement matrix, which is also its position in set N, is odd, it can be identified in time $O(\log n)$. Therefore, thanks to the linear order of the input set, we can remove the first item in N to create P and assure that the position of the first positive in P is odd in case its position in N is even. In particular, it is possible to construct two measurement matrices of size $s \times n$ and $s \times (n-1)$ such that item $j \geq 2$ is represented by column j and column $j-1$ in the first and second matrices, respectively, where $s = 2\lceil \log n \rceil$. As a result, we only need $2s$ tests to recover the two consecutive positives in $2s$ time. This idea is summarized as follows.

Lemma 2: Let n be a positive integer and $N = \{1, 2, \ldots, n\}$ be the set of linearly ordered items. There exist two strongly explicit measurement matrices with size of $2\lceil \log n \rceil \times n$ and $2\lceil \log n \rceil \times (n-1)$ such that if N has exactly two positives which are consecutive or has up to one positive, they can be identified with $s = 4\lceil \log n \rceil$ tests in s time.

B. First encoding procedure

Let $\mathcal{S}$ be an $s \times n$ measurement matrix associated with the input set of items $N = \{1, 2, \ldots, n\}$:

$$\mathcal{S} = \begin{bmatrix} \mathbf{b}_1 & \mathbf{b}_2 & \ldots & \mathbf{b}_n \\ \overline{\mathbf{b}}_1 & \overline{\mathbf{b}}_2 & \ldots & \overline{\mathbf{b}}_n \end{bmatrix} = \begin{bmatrix} \mathcal{S}_1 & \ldots & \mathcal{S}_n \end{bmatrix}, \qquad (1)$$

where $s = 2\lceil \log n \rceil$, $\mathbf{b}_j$ is the $\lceil \log n \rceil$-bit binary representation of integer $j-1$, $\overline{\mathbf{b}}_j$ is the complement of $\mathbf{b}_j$, and $\mathcal{S}_j := \begin{bmatrix} \mathbf{b}_j \\ \overline{\mathbf{b}}_j \end{bmatrix}$ for $j = 1, 2, \ldots, n$. Column $\mathcal{S}_j$ represents for the jth item of N and that the weight of every column in $\mathcal{S}$ is $s/2 = \lceil \log n \rceil$. Furthermore, the jth item of N, which is also item j, is uniquely identified by $\mathbf{b}_j$.

Let $\mathbf{s} = (s_1, \ldots, s_n)^T$ be a binary representation vector of set N in which an entry $s_j = 1$ indicates that the jth item in the set N is positive and $s_j = 0$ indicates otherwise. The outcome vector by performing tests on the input set of items N and its measurement matrix $\mathcal{S}$ is $\mathbf{y} = \mathcal{S} \odot \mathbf{s}$.

C. First decoding procedure

The decoding procedure is summarized in Algorithm 1. Step 1 is to verify whether there are no positives in the input set. If there exists at least one positive, we will proceed to Step 2. Step 3 is to verify whether there is only one positive in the input set. From Step 4, it suffices to say that the input set has exactly two consecutive positives. Step 4 is to initialize vector $\mathbf{z}$ which is presumed to be $\mathbf{b}_a \vee \mathbf{b}_{a+1}$ for some integer $1 \leq a \leq n-1$ by using the outcome vector $\mathbf{y}$ and Step 5 is to calculate that a. Because $\mathbf{b}_a$ is the $\lceil \log n \rceil$-bit binary representation of integer $j-1$, we shift a one unit to $a+1$ for simple representation in later steps. Once a is odd, Steps 6 to 7 are to recover a. However, if a is even, we have to scan every possibility of odd numbers generated from $\mathbf{y}$ by altering one bit in the first half of $\mathbf{y}$. This procedure is done by Steps 9 to 12. Finally, Steps 14 to 16 are simply to verify whether the value a obtained from Step 13 for an alteration is genuinely the index of the first positive.

To reduce the decoding complexity in Algorithm 1, we have to use alternative measurement matrices and decoding procedure. The details are presented below.

D. Second encoding procedure

Let $P = \{2, \ldots, n\}$ be a set of items. Let $\mathcal{P}$ be an $s \times (n-1)$ measurement matrix created by removing the last column of $\mathcal{S}$ in (1). Column $\mathcal{S}_j$ represents for the $(j+1)$th item in the set P which is uniquely identified by $\mathbf{b}_j$.

The outcome vector is obtained by performing tests on two distinct pairs of inputs set of items and their corresponding measurement matrices as $\mathbf{z} = \begin{bmatrix} \mathcal{S} \odot \mathbf{s} \\ \mathcal{P} \odot \mathbf{p} \end{bmatrix} = \begin{bmatrix} \mathbf{y} \\ \mathbf{w} \end{bmatrix}$, where $\mathbf{y} =$

Algorithm 1 DecConsecutivePositives$(\mathbf{y}, \mathcal{S})$: Decoding procedure for up to two consecutive positives.

Input: Outcome vector $\mathbf{y}$, matrix $\mathcal{S}$ of size $s \times n$ defined in (1).
Output: Set of two consecutive positives.

if $\mathbf{y} \equiv \mathbf{0}$ **then** Return $P = \emptyset$ **end if**
Calculate an index a with the input as the first half of $\mathbf{y}$. Set $a := a + 1$.
if $\mathcal{S}_a \equiv \mathbf{y}$ **then** Return $P = \{a\}$ **end if**
Initialize a $1 \times s/2$ vector $\mathbf{z}$ by setting $\mathbf{z}(1) = 0$ and $\mathbf{z}(i) = \mathbf{y}(i)$ for $i = 2, \ldots, s/2$.
Calculate an index a with the input $\mathbf{z}$. Set $a := a + 1$.
if $\mathcal{S}_a \vee \mathcal{S}_{a+1} \equiv \mathbf{y}$ **then**
  Return the set $P = \{a, a+1\}$.
else
  for $i = 1$ to $s/2$ **do**
    Set $\mathbf{z} = (\mathbf{y}(1), \ldots, \mathbf{y}(s/2))^T$.
    if $\mathbf{y}(i) = 1$ **then**
      Set $\mathbf{z}(i) = 0$.
      Calculate index a with the input $\mathbf{z}$. Set $a := a + 1$.
      if $\mathcal{S}_a \vee \mathcal{S}_{a+1} \equiv \mathbf{y}$ **then**
        Return the set $P = \{a, a+1\}$.
      end if
    end if
  end for
end if

$\mathcal{S} \odot \mathbf{s}$, $\mathbf{w} = \mathcal{P} \odot \mathbf{p}$, $\mathbf{s} = (s_1, \ldots, s_n)^T$ and $\mathbf{p} = (p_1, \ldots, p_{n-1})^T$ are the binary representation vectors of items in sets N and P, respectively. An entry $s_j = 1$ ($p_j = 1$, resp.) indicates that the jth item in the set N (P, resp.) is positive, and $s_j = 0$ ($p_j = 0$, resp.) indicates otherwise.

E. Second decoding procedure

The decoding procedure is summarized in Algorithm 2. Step 1 is to identify whether there are no positives in the input set. If there exists at least one positive, we will proceed to Step 2. Since there are two measurement matrices associated with two input sets of items, we need two vectors to recover the index(es) of the positive(s) in the two input sets from two outcome vectors. Steps 2 and 6 are to initiate those vectors. Since the first set of items is N, if there is only one positive present, the condition in Step 4 holds and it returns that index.

If the input set has exactly two consecutive positives, the algorithm will proceed to Step 4. If the index of the first positive is odd, Step 5 is to recover it, and hence the index of the second positive is also obtained. However, if the index of the first positive is even, the condition in Step 5 does not hold but the one in Step 8 does. It then returns that index.

IV. Improved algorithms

As described in Section I-D, we first create super items. For the first phase, our improved algorithms are based on two inseparable compartments: the linear order of the input

Algorithm 2 DecConsecutivePositives$(\mathbf{y}, \mathcal{S}, \mathbf{w}, \mathcal{P})$: Decoding procedure for up to two consecutive positives.

Input: Outcome vectors $\mathbf{y}$ and $\mathbf{w}$, matrices $\mathcal{S}$ and $\mathcal{P}$ defined in (1) and Section III-D.
Output: Set of up to two consecutive positives.

1: **if** $\mathbf{y} \equiv \mathbf{w} \equiv \mathbf{0}$ **then** Return $P = \emptyset$ **end if**
2: Initialize a $1 \times s/2$ vector $\mathbf{y}'$ by setting $\mathbf{y}'(1) = 0$ and $\mathbf{y}'(i) = \mathbf{y}(i)$ for $i = 2, \ldots, s/2$.
3: Calculate index a with the input $\mathbf{y}'$. Set $a = a + 1$.
4: **if** $\mathcal{S}_a \equiv \mathbf{y}$ **then** Return $P = \{a\}$ **end if**
5: **if** $\mathcal{S}_a \vee \mathcal{S}_{a+1} \equiv \mathbf{y}$ **then** Return $P = \{a, a+1\}$ **end if**
6: Initialize a $1 \times s/2$ vector $\mathbf{w}'$ by setting $\mathbf{w}'(1) = 0$ and $\mathbf{w}'(i) = \mathbf{w}(i)$ for $i = 2, \ldots, s/2$.
7: Calculate index b with the input $\mathbf{w}'$. Set $b = b + 1$.
8: **if** $\mathcal{S}_b \vee \mathcal{S}_{b+1} \equiv \mathbf{w}$ **then** Return $P = \{b+1, b+2\}$ **end if**

items which contain consecutive positives and strongly explicit matrices designed based on that linear order. For the second phase, we simply generate a measurement matrix by horizontally placing a series of $2d \times 2d$ identity matrices. The details of our proposed algorithms are described below.

A. Super items

We first create *super items* with linear order in which each super item contains exactly d items, except for the last super item which may contain less than d items. In particular, the n items are distributed into n/d subsets (for simplicity, we assume n is divisible by d) in which the jth subset contains items indexed from $(j-1)d + 1$ to jd. The jth super item is the jth subset. A super item is positive if it contains at least one positive item and negative otherwise. There are up to two super positive items which are consecutive because the input items are linearly ordered, the number of positive items is up to d, the positive items are consecutive and each super item contains up to d items. This procedure is illustrated in Fig. 1.

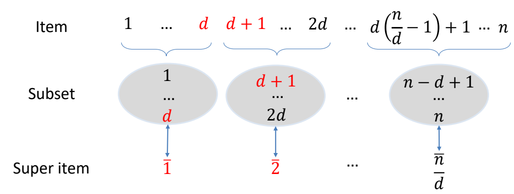

Fig. 1. Creating super items. A super item is a subset of items.

B. Encoding procedure

The encoding procedure includes the first and seconds phases as described in Section I-D. Regarding the first phase, there are two designs for measurement matrices corresponding to two improved algorithms.

1) First phase in the first improved algorithm: Matrix $\mathcal{S}$ used here is the same as the one in Section III-B by replacing items with super items and n with n/d.

2) First phase in the second improved algorithm: The measurement matrices used here, namely $\mathcal{S}$ and $\mathcal{P}$, are the same as the one in Section III-D by replacing items with super items and n with n/d.

3) Second phase: A $2d \times n$ measurement matrix $\mathcal{H}$, called a verification matrix, is created as $\mathcal{H} = [\mathcal{I}_{2d} \ \ldots \mathcal{I}_{2d} \ \mathcal{I}_{:,n-2d\lfloor \frac{n}{2d} \rfloor}]$, where $\mathcal{I}_{2d}$ is a $2d \times 2d$ identity matrix and matrix $\mathcal{I}\left(:, 1:n-2d\lfloor \frac{n}{2d} \rfloor\right)$ contains the first $n-2d\lfloor \frac{n}{2d} \rfloor$ columns of $\mathcal{I}_{2d}$. There are $\lfloor \frac{n}{2d} \rfloor$ such $\mathcal{I}_{2d}$ matrices.

The outcome vector by using $\mathcal{H}$ is $\mathbf{h} = \mathcal{H} \odot \mathbf{x}$, where $\mathbf{x} = (x_1, \ldots, x_n)^T$ is a binary representation vector of set N in which an entry $x_j = 1$ indicates that the jth item in the set N is positive and $x_j = 0$ indicates otherwise.

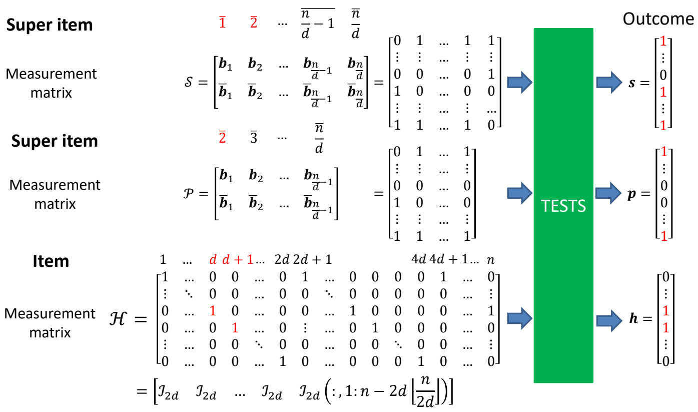

Fig. 2. Encoding procedure. Each measurement matrix is associated with a set of items or super items.

C. Decoding procedure

The flow of the decoding procedure is illustrated in Fig. 3. We first identify up to two consecutive super positives to get a range of items which contains all positives. The true positives are then identified by using the verification matrix $\mathcal{H}$ and the outcome vector $\mathbf{h}$.

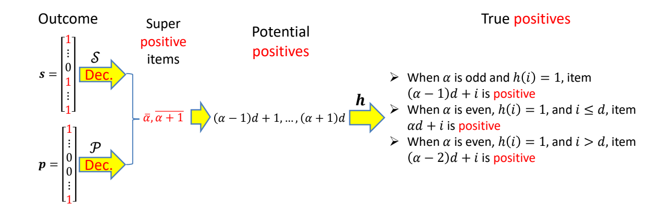

Fig. 3. Decoding procedure. From the outcome vector(s) generated by super items, we can identify up to two super positive items. Then there are up to $2d$ potential consecutive positives. Every entry in the outcome vector $\mathbf{h}$ is then scanned to identify its corresponding positive by using some specific rules.

The details of the decoding procedure, which merges the first and second improved algorithms, in Algorithm 3. With the input in the first (second, resp.) improved algorithm, Algorithm 3 skips Step 2 (1, resp.). Step 3 returns an empty set of positives because there are no super positives in the input set of items when $T = \emptyset$. Steps 4 and 5 are to get the first super positive and initialize an empty positive set, respectively. The usage of the first phase in the encoding procedure ends here. We now proceed to identify the true positives. Because all positives lie in the index range from $(\alpha-1)d+1, \ldots, (\alpha+1)d$, we scan every entry in the outcome vector $\mathbf{h}$ in Step 6 then its corresponding positive is identified by using the rules in Steps 6 to 14. Step 15 simply returns the positive set.

Algorithm 3 Decoding procedure for up to d consecutive positives.

Input in the first improved algorithm: Outcome vector $\mathbf{y}$, matrix $\mathcal{S}$ of size $s \times n$ defined in (1).
Input in the second improved algorithm: Outcome vectors $\mathbf{y}$ and $\mathbf{w}$, matrices $\mathcal{S}$ and $\mathcal{P}$ defined in (1) and Section III-D.
Output: Set of consecutive positives.

1: $T = \text{DecConsecutivePositives}(\mathbf{y}, \mathcal{S})$.
2: $T = \text{DecConsecutivePositives}(\mathbf{y}, \mathcal{S}, \mathbf{w}, \mathcal{P})$.
3: Return $P = \emptyset$ if $T = \emptyset$.
4: Let α be the first item in T.
5: Initialize the positive set $P = \emptyset$.
6: **for** $i = 1$ to $2d$ **do**
7:     **if** α is odd and $h(i) = 1$ **then**
8:         $P = P \cup \{(\alpha-1)d + i\}$.
9:     **end if**
10:     **if** α is even and $h(i) = 1$ **then**
11:         **if** $i \leq d$ **then** $P = P \cup \{\alpha d + i\}$
12:         **else** $P = P \cup \{(\alpha-2)d + i\}$ **end if**
13:     **end if**
14: **end for**
15: Return P.

D. The decoding complexity

It is easy to confirm that the complexity of Steps 3 to 14 is $O(d)$. Therefore, the decoding complexities of the first and second improved algorithms vary with the complexities of Steps 2 and 1, respectively. The complexities of Steps 2 and 1 are summarized in Lemmas 1 and 2, which are $O\left(\log^2 \frac{n}{d}\right)$ and $O\left(\log \frac{n}{d}\right)$, respectively. We summarize the results of our two improved algorithms in the two following theorems.

Theorem 1: (The first improved algorithm) Let n be a positive integer and $N = \{1, 2, \ldots, n\}$ be the set of linearly ordered items. Then there exist strongly explicit measurement matrices such that up to d consecutive positives can be identified with $2\lceil \log_2 \frac{n}{d} \rceil + 2d$ tests in $O\left(\log^2 \frac{n}{d} + d\right)$ time.

Theorem 2: (The second improved algorithm) Let n be a positive integer and $N = \{1, 2, \ldots, n\}$ be the set of linearly ordered items. There exist strongly explicit measurement matrices such that up to d consecutive positives can be identified with $4\lceil \log_2 \frac{n}{d} \rceil + 2d$ tests in $O\left(\log \frac{n}{d} + d\right)$ time.

V. Acknowledgements

Thach V. Bui and Thuc D. Nguyen were supported in part by Vietnam National University Ho Chi Minh City (VNU-HCM) under grant number NCM2019-18-01. Mahdi Cheraghchi's research was partially supported by the National Science Foundation under Grant No. CCF-2006455.

References

[1] T. V. Bui, M. Cheraghchi, and T. D. Nguyen, “Improved algorithms for non-adaptive group testing with consecutive positives,” *arXiv preprint arXiv:2101.11294*, 2021.

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

[3] N. Shental, S. Levy, V. Wuvshet, S. Skorniakov, B. Shalem, A. Ottolenghi, Y. Greenshpan, R. Steinberg, A. Edri, R. Gillis, *et al.*, “Efficient high-throughput sars-cov-2 testing to detect asymptomatic carriers,” *Science advances*, vol. 6, no. 37, p. eabc5961, 2020.

[4] R. Gabrys, S. Pattabiraman, V. Rana, J. Ribeiro, M. Cheraghchi, V. Guruswami, and O. Milenkovic, “Ac-dc: Amplification curve diagnostics for covid-19 group testing,” *arXiv preprint arXiv:2011.05223*, 2020.

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

[6] A. D'yachkov, N. Polyanskii, V. Shchukin, and I. Vorobyev, “Separable codes for the symmetric multiple-access channel,” in *2018 IEEE ISIT*, pp. 291–295, IEEE, 2018.

[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, SIAM, 2010.

[8] E. Porat and A. Rothschild, “Explicit nonadaptive combinatorial group testing schemes,” *IEEE Trans. Inf. Theory*, vol. 57, no. 12, pp. –, 2011.

[9] H. Q. Ngo, E. Porat, and A. Rudra, “Efficiently decodable error-correcting list disjunct matrices and applications,” in *ICALP*, pp. 557–568, Springer, 2011.

[10] M. Cheraghchi, “Noise-resilient group testing: Limitations and constructions,” *Discrete Applied Mathematics*, vol. 161, no. 1-2, pp. 81–95, 2013.

[11] S. Cai, M. Jahangoshahi, M. Bakshi, and S. Jaggi, “Efficient algorithms for noisy group testing,” *IEEE Transactions on Information Theory*, vol. 63, no. 4, pp. 2113–2136, 2017.

[12] S. Bondorf, B. Chen, J. Scarlett, H. Yu, and Y. Zhao, “Sublinear-time non-adaptive group testing with $o(k \log n)$ tests via bit-mixing coding,” *arXiv preprint arXiv:1904.10102*, 2019.

[13] T. V. Bui, M. Kuribayashi, T. Kojima, R. Haghvirdinezhad, and I. Echizen, “Efficient (nonrandom) construction and decoding for non-adaptive group testing,” *Journal of Information Processing*, vol. 27, pp. 245–256, 2019.

[14] M. Cheraghchi and V. Nakos, “Combinatorial group testing and sparse recovery schemes with near-optimal decoding time,” in *61st Annual IEEE Symposium on Foundations of Computer Science*, IEEE, 2020.

[15] E. Price and J. Scarlett, “A fast binary splitting approach to non-adaptive group testing,” in *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2020)*, Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2020.

[16] W. J. Bruno, E. Knill, D. J. Balding, D. C. Bruce, N. A. Doggett, W. W. Sawhill, R. L. Stallings, C. C. Whittaker, and D. C. Torney, “Efficient pooling designs for library screening,” *Genomics*, vol. 26, no. 1, pp. 21–30, 1995.

[17] D. J. Balding and D. C. Torney, “The design of pooling experiments for screening a clone map,” *Fungal Genetics and Biology*, vol. 21, no. 3, pp. 302–307, 1997.

[18] C. J. Colbourn, “Group testing for consecutive positives,” *Annals of Combinatorics*, vol. 3, no. 1, pp. 37–41, 1999.

[19] J. S.-T. Juan and G. J. Chang, “Adaptive group testing for consecutive positives,” *Discrete mathematics*, vol. 308, no. 7, pp. 1124–1129, 2008.

[20] M. Müller and M. Jimbo, “Consecutive positive detectable matrices and group testing for consecutive positives,” *Discrete mathematics*, vol. 279, no. 1-3, pp. 369–381, 2004.

[21] H. Chang, Y.-C. Chiu, and Y.-L. Tsai, “A variation of cover-free families and its applications,” *Journal of Computational Biology*, vol. 22, no. 7, pp. 677–686, 2015.