Advances in Group Testing

Yongxi Cheng

Contents

Y. Cheng
School of Management, Xi'an Jiaotong University, Xi'an, Shaanxi, People's Republic of China
e-mail: chengyx@mail.xjtu.edu.cn

P.M. Pardalos et al. (eds.), *Handbook of Combinatorial Optimization*,
DOI 10.1007/978-1-4419-7997-1_71,

Abstract

In *combinatorial group testing*, there are n items; each has an unknown binary status, *positive* (i.e., *defective*) or *negative* (i.e., *good*), and the number of positives is upper bounded by an integer d. Suppose there is some method to test whether a subset of items contains at least one positive or not. The test result is said to be positive if it indicates that the subset contains at least one positive item; otherwise, the test result is called negative. The problem is to resolve the status of every item using the minimum number of tests.

Group testing (GT) algorithms can be *adaptive* or *nonadaptive*. An adaptive algorithm conducts the tests one by one and allows to design later tests using the outcome information of all previous tests. A nonadaptive group testing (NGT) algorithm specifies all tests before knowing any test results, and the benefit is that all tests can be performed in parallel. For the above group testing problem, nonadaptive algorithms require inherently more tests than adaptive ones.

Though the research of group testing dates back to Dorfman's 1943 paper, a renewed interest in the subject occurred recently mainly due to the applications of group testing to the area of computational molecular biology. In applications of molecular biology, a group testing algorithm is called a *pooling design*, and the composition of each test is called a *pool*. While it is still important to minimize the number of tests, there are two other goals. First, in the biological setting, screening one pool at a time is far more expensive than screening many pools in parallel; this strongly encourages the use of nonadaptive algorithms. Second, DNA screening is error prone, so it is desirable to design *error-tolerant* algorithms, which can detect or correct some errors in the test results.

In this monograph, some recent algorithmic, complexity, and mathematical results on nonadaptive group testing (and on pooling design) are presented.

1 Introduction

In *combinatorial group testing*, there are n items; each has an unknown binary status, *positive* (i.e., *defective*) or *negative* (i.e., *good*), and the number of positives is upper bounded by an integer d. Suppose there is some method to test whether a subset of items contains at least one positive or not. The test result is said to be positive if it indicates that the subset contains at least one positive item; otherwise, the test result is called negative. The problem is to resolve the status of every item using the minimum number of tests.

Group testing (GT) algorithms can be *adaptive* or *nonadaptive*. An adaptive algorithm conducts the tests one by one and allows to design later tests using the outcome information of all previous tests. A nonadaptive group testing (NGT) algorithm must specify all tests before knowing any test results, and the benefit is that all tests can be performed in parallel. For the above group testing problem, nonadaptive algorithms require inherently more tests than adaptive ones. It is known

that any nonadaptive algorithm must use a number of $\Omega(\frac{d^2 \log n}{\log d})$ tests, and the best known nonadaptive algorithm uses $O(d^2 \log n)$ tests. In contrast, the best adaptive algorithm requires $O(d \log n)$ tests (see, e.g., [17]) in the worst case.

Pooling Design. Though the research of combinatorial group testing dates back to Dorfman's 1943 paper [15], probably the most important modern applications of group testing are in the area of computational molecular biology, in which one important subject is clone library screening [3, 17, 26]. In applications to molecular biology, a group testing procedure is called a *pooling design*, and the composition of each test is called a *pool*.

A DNA library consists of thousands of separate DNA *clones*. The basic task of DNA library screening is, for a collection of *probes*, to determine which clone from the library contains which probe. Given a probe, a clone is said to be positive if it contains the probe; otherwise, it is said to be negative. In practice, to identify all positive clones from a library, clones are often pooled together to be tested against each probe, since checking each clone-probe pair is expensive and usually only a few clones in the library contain a given probe. An example is when sequenced-tagged site markers (also called STS probes) are used [46]. In practice, there are experimental tests, for example, the polymerase chain reaction, which can determine in a given pool whether or not there exists at least one clone containing a given probe.

In applications to molecular biology, while it is still important to minimize the number of tests, there are two other goals. First, in the biological setting, screening one pool at a time is far more expensive than screening many pools in parallel; this strongly encourages the use of nonadaptive algorithms. Second, DNA screening is error prone, so it is desirable to design *error-tolerant* algorithms, which can detect or correct some errors in the test outcomes. The reader is referred to the monograph by Du and Hwang [17] for a comprehensive discussion of this topic.

Between fully adaptive and nonadaptive (one stage) algorithms, the so-called trivial two-stage algorithms [36] are of considerable interest for screening problems. Such an algorithm has two stages. In the first stage, the pools are tested in parallel, and a set CP of *candidate positives* from the items is chosen based on the test results; in the second stage, individual tests are performed on all the items in CP to resolve the status of each item. Previous works on two-stage group testing algorithms are, among others, [4, 14, 24, 36, 41]. The following quotation from Knill [36] well emphasizes the importance of such algorithms: "It is generally feasible to construct a number of pools (much fewer than the number of clones) initially by exploiting parallelism, but adaptive construction of pools with many clones during the testing procedure is discouraged. The technicians who implement the pooling strategies generally dislike even the 3-stage strategies that are often used. Thus the most commonly used strategies for pooling libraries of clones rely on a fixed but reasonably small set of non-singleton pools. The pools are either tested all at once or in a small number of stages (usually at most 2) where the previous stage determines which pools to test in the next stage. The potential positives are then inferred and confirmed by testing of individual clones. In most biological applications each

positive clone must be confirmed even if the pool results unambiguously indicate that it is positive. This is to improve the confidence in the results, given that in practice the tests are prone to errors."

Separating Matrices. A nonadaptive group testing procedure can be represented as a 0-1 matrix $M = (m_{ij})$, in which the columns are associated with the items and the rows are associated with the tests, and $m_{ij} = 1$ indicates that item j is contained in test i. The test outcomes can be represented by a 0-1 vector, the *outcome vector*, where 0 indicates a negative outcome and 1 indicates a positive outcome. It is not hard to verify that if a subset S of columns exactly corresponds to all the positive items, then the outcome vector is equal to vector $U(S)$, the *union* (i.e., the componentwise Boolean sum) of all column vectors in S. Given the matrix representation of an algorithm and the outcome vector, the process of identifying all the positive items is called decoding. For a 0/1 matrix to be a valid nonadaptive group testing algorithm, some separating property is often required. This monograph focuses on two most used and studied separating properties: disjunctness and separability.

In order to identify all positives as long as the number of positives is no more than d, matrix M should satisfy that for any two distinct subsets S_1 and S_2 of columns such that $|S_1| \leq d$ and $|S_2| \leq d$, $U(S_1) \neq U(S_2)$. A matrix satisfying this property is called $\bar{d}$*-separable*. In the definition, if the condition "$|S_1| \leq d$ and $|S_2| \leq d$" is replaced by "$|S_1| = |S_2| = d$," a matrix satisfying this property is called d*-separable*. If the matrix representing a nonadaptive pooling design is $\bar{d}$*-separable* (or d*-separable*), then theoretically based on the test outcomes one can unambiguously identify all the up to d (or exactly d) positives. However, the actual process of determining the positives from the outcome vector, that is, the *decoding* process, could be very time-consuming. In practice, one can adopt matrices with stronger property to make the decoding process more efficient.

For two 0-1 vectors u and v with the same number of components, if for any component of u with value 1, the corresponding component of v is also 1, then u is said to be *covered* by v. A 0-1 matrix is said to be d*-disjunct* if no column is covered by the union of any d other columns. The same structure is also called *cover-free family* in combinatorics [25, 29, 53], and *superimposed code* in information theory [22, 23, 34], and has been extensively studied. Obviously if a matrix is d*-disjunct*, then it is also $\bar{d}$-separable, and thus is d-separable. If the matrix M representing a nonadaptive pooling design is d-disjunct and the number of positives is no more than d, then the following efficient decoding procedure exists with running time linear in the size of M: A column c corresponds to a positive item if and only if c is covered by the outcome vector. d-disjunct matrices are important structures in pooling design, and there have been a lot of works on their constructions [2, 10, 20, 23–25, 28, 32–34, 38, 44, 45, 49, 50].

A 0/1 matrix is said to be $(d; z)$*-disjunct* [22, 39] if for any set D of d columns and any column $c \notin D$, there exist at least z rows such that each of them has value 1 at column c and value 0 at all the d columns of D. Clearly, d-disjunctness

is just $(d;1)$-disjunctness. As mentioned above, if the matrix M representing a nonadaptive group testing algorithm is d-disjunct and the number of positives is no more than d, then there exist efficient decoding procedures with running time linear in the size of M. However, when there are errors in the test outcomes, the above decoding procedure generally does not work, and in this case, the matrix is required to be $(d;z)$-disjunct, which results in a $\lfloor\frac{z-1}{2}\rfloor$-error-correcting NGT algorithm. In this case, linear decoding that successfully identifies all positives still exists, provided that there are no more than d positives and at most $\lfloor\frac{z-1}{2}\rfloor$ errors in the test outcomes. d-disjunct and $(d;z)$-disjunct matrices form the basis of error-free and error-tolerant nonadaptive group testing.

Main Contents. In this monograph, some recent algorithmic, complexity, and mathematical results on nonadaptive group testing (and on pooling design) are presented. The main contents consist of five parts. In the first part, new randomized constructions of one- and two-stage nonadaptive group testing are presented. Comparisons with other known constructions on the number of required tests are also discussed.

In the second part, some complexity results for problems that are basic to nonadaptive group testing are given. The problem to determine whether a given matrix H is $\bar{d}$-separable and minimal, MIN-SEPARABILITY, is showed to be DP-complete. Here the meaning of being minimal is that the removal of any row from H will make it no longer $\bar{d}$-separable. The second problem is, given a binary matrix M and a positive integer d, find a minimum $\bar{d}$-separable submatrix of M with the same number of columns. The complexity of the decision version of this problem, $\bar{d}$-separable submatrix, is conjectured to be Σ_2^P-complete. As an evidence to support this conjecture, the Σ_2^P-completeness of a problem which is a little more general than $\bar{d}$-separable submatrix is established.

In the third part, the parameterized complexities of three basic problems in nonadaptive group testing are studied. They are, given an $m \times n$ binary matrix and a positive integer d, to determine whether the matrix is d-separable ($\bar{d}$-separable, or d-disjunct). Though the three problems are all known to be coNP-complete in the classical complexity theory, the motivation of this study is that in most applications d is very small compared to n; it is interesting to investigate whether there are efficient algorithms solving the above problems when the value of d is small. In this part, the parameterized versions of the three problems, with d as the parameter, are showed to be co-W[2]-complete. The immediate implications of the results are that, given an $m \times n$ binary matrix and a positive integer d, a deterministic algorithm with running time $f(d) \times (mn)^{O(1)}$ (where f is an arbitrary computable function) to determine whether the matrix is d-separable ($\bar{d}$-separable, or d-disjunct) should not be expected.

In the fourth part, upper bounds on the minimum number of rows required by any d-disjunct matrix and any $(d;z)$-disjunct matrix with n columns, $t(d,n)$ and $t(d,n;z)$, respectively, are studied. A very short proof is given for the currently best upper bound on $t(d,n)$; the method is also generalized to obtain a new upper bound

on $t(d,n;z)$. In the final part, a way to transform an error-tolerant separable matrix to an error-tolerant disjunct matrix is given; the optimality of this transformation in some senses is also discussed. If no base is specified, then log is of base 2 throughout.

2 New Constructions of One- and Two-Stage Pooling Design

In [10] new constructions of one- and two-stage pooling design are given. For one-stage pooling design, the focus is on the construction of disjunct matrices, which are widely studied for various applications including the design of nonadaptive group testing algorithms. There have been a lot of works on the construction of disjunct matrices [2, 10, 20, 23–25, 28, 32–34, 38, 44, 45, 49, 50].

For two-stage pooling designs, De Bonis et al. [14] first present an asymptotically optimal two-stage algorithm that requires a number of tests within a constant factor $7.54(1+o(1))$ of the information theoretic lower bound $d\log(n/d)$. Eppstein et al. [24] improve the constant factor to $4(1+o(1))$ by using the concept of (d,k)-*resolvable* matrices (which will be explained later), which is currently the best. There are also probabilistic pooling designs [40, 41, 44] with good performance in practice.

In the sequel of this section, two Las Vegas algorithms for constructing d-disjunct and $(d;z)$-disjunct matrices are presented. For two-stage pooling designs, an algorithm using a number of $C_d(1+o(1))\log n$ tests is presented, where $C_d \le \frac{3}{\log 3}d$ for $d \ge 1$ and $C_d \to d\log e$ as $d \to \infty$. This improves the previously best bound given in [24] by a factor of more than 2. New probabilistic pooling designs are also proposed. Compared to [44], the new probabilistic designs have different type of possible errors and require much fewer tests. All the results presented in this section are from [10].

2.1 Preliminaries

Transversal Design. A pooling design is transversal if the pools can be divided into disjoint families, each of which is a partition of all items. The concept of *q-ary $(d,1)$-disjunct* matrix will be first introduced: A q-ary matrix is $(d,1)$-disjunct if for any column c and any set D of d other columns, there exists at least one element in c such that the element does not appear in any column of D in the same row.

As described in [17, 20], one can transform a q-ary $(d,1)$-disjunct matrix M' into a (binary) d-disjunct matrix M as follows. Replace each row R_i of M' by several rows indexed with entries of R_i; for each entry x of R_i, the row with index x is obtained from R_i by turning all x's into 1's and all others into 0's. Thus, the following theorem holds.

Theorem 1 ((Theorem 3.6.1 in [17])) *A $t_0 \times n$ q-ary $(d,1)$-disjunct matrix M' yields a $t \times n$ d-disjunct matrix M with $t \le t_0 q$.*

Clearly, one can perform the above transformation even when the q-ary matrix M' is not $(d, 1)$-disjunct. Transversal designs are favorable in practice because every column of the resulting matrix M has equal weight, which means that every item is contained in equal number of pools, so that to perform the tests, one needs the same number of copies for each item.

Two Probabilistic Lemmas. The following two lemmas will be useful later. The first is the Markov inequality (see, e.g., Theorem 3.2 in [42]), and the second is commonly known as Chernoff's bounds (Theorems 4.1 and 4.2 in [42]).

Lemma 1 (Markov inequality) *Let Y be a random variable assuming only nonnegative values, then for all $t > 0$,*

$$\Pr[Y \geq t] \leq \frac{E[Y]}{t},$$

where $E[Y]$ is the expectation of Y.

Lemma 2 (Chernoff's bounds) *Let $X_1, X_1, \ldots, X_n$ be independent 0/1 random variables, for $1 \leq i \leq n$, $\Pr[X_i = 1] = p_i$, where $0 < p_i < 1$. Let $X = \sum_{i=1}^{n} X_i$ and $\mu = E[X] = \sum_{i=1}^{n} p_i$. Then, for any $\delta > 0$,*

1. $\Pr[X \geq (1+\delta)\mu] \leq \left(\frac{e^{\delta}}{(1+\delta)^{1+\delta}}\right)^{\mu}$.
2. $\Pr[X \leq (1-\delta)\mu] \leq e^{-\mu\delta^2/2}$.

2.2 One-Stage Pooling Designs

Two efficient randomized constructions will be given for d-disjunct and $(d; z)$-disjunct matrices, respectively. The constructions are based on the transversal design.

2.2.1 A New Construction of d-Disjunct Matrices

A Las Vegas algorithm will be presented next, which for given n, d and $0 < p < 1$, successfully constructs a $t \times n$ d-disjunct matrix with probability at least p, with

$$t \leq cd^2(\log \frac{2}{1-p} + \log n),$$

where $c \approx 4.28$ is constant.

For given n, d and $0 < p < 1$, define $n_0 = 2n$. Let ϵ be the unique positive root of

$$\ln(1+\epsilon) = \frac{2\epsilon}{1+\epsilon}.$$

$\epsilon \approx 3.92$ is chosen to minimize the leading constant of t (see the remarks in later part). Let

Algorithm 1 (constructing d-disjunct matrix $M_{t\times n}$)

1. Construct a random q-ary matrix $M'_{t_0\times n_0}$ with each cell randomly assigned from $\{1, 2, \ldots, q\}$, independently and uniformly.
2. For any $1 \le i < j \le n_0$, let $w_{i,j}$ be the random variable denoting the number of rows r such that the two entries $M'(r, i)$ and $M'(r, j)$ are equal. Then,

$$E[w_{i,j}] = \mu \ (= \frac{t_0}{q}).$$

   Create an edge between columns i and j if $w_{i,j} \ge (1+\epsilon)\mu$.
3. For each edge created in Step 2, remove one of its two columns arbitrarily. Let M'' denote the resulting matrix.
4. If M'' has less than n $(= \frac{n_0}{2})$ columns, exit and the algorithm fail.
5. Using the transformation in Theorem 1, turn the first n columns of M'' into a binary matrix $M_{t\times n}$ with $t \le t_0 q$.

$$q = (1+\epsilon)d, \ \ t_0 = \frac{1+\epsilon}{\epsilon} d \ln \frac{2n-1}{1-p}, \ \ \mu = \frac{t_0}{q}.$$

Please see Algorithm 1 as the algorithm for constructing d-disjunct matrices.

In Algorithm 1 , at Step 3, M'' must be q-ary $(d, 1)$-disjunct since for any column i, the union of any d other columns can only cover less than

$$d \times (1+\epsilon)\mu = d \times \frac{t_0}{d} = t_0$$

entries of column i. Therefore, if the algorithm successfully returns a matrix, it must be d-disjunct. Moreover,

$$\begin{aligned} t &\le t_0 q \\ &= \frac{(1+\epsilon)^2}{\epsilon \log e} d^2 \log \frac{2n-1}{1-p} \\ &< c d^2 (\log \frac{2}{1-p} + \log n), \end{aligned}$$

where $c = \frac{(1+\epsilon)^2}{\epsilon \log e} \approx 4.28$.

2.2.2 Analysis of Algorithm 1

The analysis of the success probability and running time of Algorithm 1 will be presented next.

Success Probability. First, the expectation of the number of edges created at Step 2 is estimated.

Lemma 3 *Let m be the random variable denoting the number of edges created at Step 2 of Algorithm 1 , then $E[m] \le n(1-p)$.*

Proof For $1 \le i < j \le n_0, 1 \le k \le t_0$, at Step 2 of Algorithm 1 , define 0/1 random variable $X_k^{i,j}$ such that

$$X_k^{i,j} = 1 \text{ if and only if } M'(k,i) = M'(k,j).$$

Then,

$$w_{i,j} = X_1^{i,j} + X_2^{i,j} + \cdots + X_{t_0}^{i,j}.$$

Let $X^{i,j}$ be the indicator random variable for the event that there is an edge between column i and column j, that is,

$$X^{i,j} = \begin{cases} 1 \text{ there is an edge between column } i \text{ and column } j, \text{ that is, } w_{i,j} \ge (1+\epsilon)\mu; \\ 0 \text{ otherwise.} \end{cases}$$

Since $w_{i,j}$ is the sum of t_0-independent 0/1 random variables, the Chernoff bound, (1) in Lemma 2, implies that

$$\Pr[X^{i,j} = 1] = \Pr[w_{i,j} \ge (1+\epsilon)\mu] \le \left(\frac{e^\epsilon}{(1+\epsilon)^{1+\epsilon}}\right)^\mu.$$

Notice that $q = (1+\epsilon)d$, and

$$t_0 = \frac{1+\epsilon}{\epsilon} d \ln \frac{2n-1}{1-p}.$$

Then,

$$\mu = E[w_{i,j}] = \frac{t_0}{q} = \frac{1}{\epsilon} \ln \frac{2n-1}{1-p}.$$

From

$$\ln(1+\epsilon) = \frac{2\epsilon}{1+\epsilon},$$

it follows that $(1+\epsilon)^{1+\epsilon} = e^{2\epsilon}$, and so

$$\frac{e^\epsilon}{(1+\epsilon)^{1+\epsilon}} = e^{-\epsilon},$$

which implies that

$$\left(\frac{e^\epsilon}{(1+\epsilon)^{1+\epsilon}}\right)^\mu = (e^{-\epsilon})^{\frac{1}{\epsilon} \ln \frac{2n-1}{1-p}} = \frac{1-p}{2n-1}.$$

Thus,

$$E[X^{i,j}] = \Pr[X^{i,j} = 1] \le \frac{1-p}{2n-1},$$

for $1 \le i < j \le n_0$. Since $m = \sum_{1 \le i < j \le n_0} X^{i,j}$ and all the $X^{i,j}$'s are identically distributed, and $n_0 = 2n$, it follows that

$$E[m] = \binom{n_0}{2} E[X^{1,2}] \leq \binom{n_0}{2} \frac{1-p}{2n-1} = n(1-p). \qquad \square$$

Clearly, m denotes the most number of columns that may be removed at Step 3. Since $E[m] \leq n(1-p)$, by applying the Markov inequality (Lemma 1), the probability that there are less than n columns left in M'' at Step 4 (i.e., the failure probability of Algorithm 1) is at most $\Pr[m > n] \leq \frac{E[m]}{n} \leq 1-p$.

Running Time. The time required by Algorithm 1 is dominated by Step 2, which is

$$\binom{n_0}{2} t_0 = O(dn^2 \ln n),$$

by simply counting, for all pairs of columns, the number of rows at which the two columns have equal entry. In fact, an expected $O(n^2 \ln n)$ running time can be obtained by counting along the rows.

For $1 \leq i < j \leq n_0$, let $n(i,j)$ denote the number of equal entries between column i and column j in the same row. Initially, set $n(i,j) = 0$ for $1 \leq i < j \leq n_0$. For each row r, let $S_{r,1}, S_{r,2}, \ldots, S_{r,q}$ denote the sets of column indices such that

$$S_{r,k} = \{i : M'(r,i) = k\}.$$

Clearly, the sets $S_{r,k}$, $1 \leq k \leq q$, can be constructed in n_0 time. For each k, increase the values of $n(i,j)$ by 1 for all $i < j$ and $i, j \in S_{r,k}$. The expected number of such pairs (i,j) for each $S_{r,k}$ is $E[\binom{|S_{r,k}|}{2}]$. Since $|S_{r,k}|$ are identically distributed for $1 \leq k \leq q$, the expected running time of Step 2 is

$$t_0 \times \left(n_0 + qE\left[\binom{|S_{r,1}|}{2} \right] \right).$$

Notice that $|S_{r,1}|$ has the binomial distribution with parameters n_0 and $1/q$; thus,

$$n_0 + qE\left[\binom{|S_{r,1}|}{2} \right] = n_0 + q\frac{1}{q^2}(n_0^2 - n_0) = O(n^2/d),$$

and so the expected running time of Step 2, which is also the expected running time of Algorithm 1 , is $t_0 \times O(n^2/d) = O(n^2 \ln n)$.

Therefore, the following theorem is established.

Theorem 2 *Given n, d, and $0 < p < 1$, Algorithm 1 successfully constructs a $t \times n$ d-disjunct matrix with probability at least p, with*

$$t < cd^2(\log \frac{2}{1-p} + \log n),$$

where $c \approx 4.28$ is constant. The algorithm runs in expected $O(n^2 \ln n)$ time.

Remarks In Algorithm 1 , ϵ is chosen to minimize the leading constant of t. It is required that

$$(1+\epsilon)\mu \leq \frac{t_0}{d},$$

where $\mu = \frac{t_0}{q}$, that is, $q \geq (1+\epsilon)d$, to guarantee that matrix M'' is $(d,1)$-disjunct. To guarantee that with reasonable probability M'' has at least n columns, it is required that

$$n_0 - E[m] \geq n,$$

where $E[m] = \binom{n_0}{2} E[X^{1,2}]$. This implies that

$$\Pr[X^{1,2} = 1] \leq \frac{n_0 - n}{\binom{n_0}{2}}.$$

Since

$$\max_{n_0} \frac{n_0 - n}{\binom{n_0}{2}} = \frac{1}{2n-1},$$

which can be achieved when $n_0 = 2n-1$ or $n_0 = 2n$, it should have that

$$\Pr[X^{1,2} = 1] \leq \frac{1}{2n-1}.$$

This can be guaranteed by

$$\left(\frac{e^\epsilon}{(1+\epsilon)^{1+\epsilon}}\right)^\mu \leq \frac{1}{2n-1},$$

that is,

$$\mu \ln \frac{(1+\epsilon)^{1+\epsilon}}{e^\epsilon} \geq O(1) + \ln n.$$

By plugging in $\mu = \frac{t_0}{q} = \frac{t}{q^2}$ and $q \geq (1+\epsilon)d$, it follows that

$$t \geq \frac{(1+\epsilon)^2}{\ln \frac{(1+\epsilon)^{1+\epsilon}}{e^\epsilon}} d^2 (O(1) + \ln n).$$

Define

$$f(\epsilon) = \frac{(1+\epsilon)^2}{\ln \frac{(1+\epsilon)^{1+\epsilon}}{e^\epsilon}} = \frac{(1+\epsilon)^2}{(1+\epsilon)\ln(1+\epsilon) - \epsilon}.$$

To minimize $f(\epsilon)$ for $\epsilon > 0$, from basic calculus, $f'(\epsilon) = 0$ implies that

$$\ln(1+\epsilon) = \frac{2\epsilon}{1+\epsilon}.$$

It is easy to verify that this equation has one unique positive root $\epsilon \approx 3.92$. Also, the equation implies that

$$f(\epsilon) = \frac{(1+\epsilon)^2}{(1+\epsilon)\ln(1+\epsilon) - \epsilon} = \frac{(1+\epsilon)^2}{\epsilon}.$$

2.2.3 Error-Tolerance Case

Algorithm 1 is modified next, so that given $n, d, z > 1$ and $0 < p < 1$, the modified algorithm successfully constructs a $t \times n$ $(d; z)$-disjunct matrix with probability at least p, with

$$t \le cd^2\left(\log\frac{2}{1-p} + \log n\right) + 2(1+\epsilon)dz + O\left(\frac{z^2}{\ln n}\right),$$

where $\epsilon \approx 3.92$ and $c \approx 4.28$ are constants, and the $O(\cdot)$ notation hides dependencies on p.

First a generalization of $(d, 1)$-disjunct matrices is given. A q-ary matrix is $(d, 1; z)$-*disjunct* if for any column c and any set D of d other columns, there exist at least z elements in c such that each of these elements does not appear in any column of D in the same row. Clearly, by applying the same transformation in Theorem 1, one can turn a $t_0 \times n$ q-ary $(d, 1; z)$-disjunct matrix into a $(d; z)$-disjunct matrix with n columns and at most $t_0 q$ rows.

For given $n, d, z > 1$ and $0 < p < 1$, let n_0, ϵ be as in Algorithm 1 . Let

$$q = (1+\epsilon)d + \frac{\epsilon z}{\ln\frac{2n-1}{1-p}}, \quad t_0 = z + \frac{1+\epsilon}{\epsilon}d\ln\frac{2n-1}{1-p}, \quad \mu = \frac{t_0}{q}.$$

It can be verified that by this assignment,

$$(1+\epsilon)\mu = \frac{t_0 - z}{d},$$

and

$$\left(\frac{e^\epsilon}{(1+\epsilon)^{1+\epsilon}}\right)^\mu = \frac{1-p}{2n-1}.$$

Please see Algorithm 2 as the algorithm for constructing $(d; z)$-disjunct matrices.

Firstly, at Step 3 of Algorithm 2 , M'' must be q-ary $(d, 1; z)$-disjunct because for any column i, any d other columns can only cover less than $d \times (1+\epsilon)\mu = t_0 - z$ of its entries. Therefore, if the algorithm successfully returns a matrix, it must be $(d; z)$-disjunct. Secondly, when $z = o(d \ln n)$, by similar arguments, Algorithm 2 runs in time $O(dn^2 \ln n)$ in the straightforward manner, and can be improved to expected $O(n^2 \ln n)$ time by counting the pairs of equal entries along the rows. Thirdly,

Algorithm 2 (constructing $(d;z)$-disjunct matrix $M_{t\times n}$)

Algorithm 2 works in the same way as Algorithm 1 , except that with $q = (1+\epsilon)d + \frac{\epsilon z}{\ln \frac{2n-1}{1-p}}$ and $t_0 = z + \frac{1+\epsilon}{\epsilon} d \ln \frac{2n-1}{1-p}$.

$$\begin{aligned} t &\le t_0 q \\ &= \frac{(1+\epsilon)^2}{\epsilon \log e} d^2 \log \frac{2n-1}{1-p} + 2(1+\epsilon)dz + \frac{(\epsilon \log e) z^2}{\log \frac{2n-1}{1-p}} \\ &\le cd^2 \left(\log \frac{2}{1-p} + \log n \right) + 2(1+\epsilon)dz + O\left(\frac{z^2}{\ln n} \right), \end{aligned}$$

where $c = \frac{(1+\epsilon)^2}{\epsilon \log e} \approx 4.28$.

For the success probability, if one let m^* be the random variable denoting the number of edges created at Step 2, since

$$\left(\frac{e^\epsilon}{(1+\epsilon)^{1+\epsilon}} \right)^\mu = \frac{1-p}{2n-1}$$

still holds, the same result in Lemma 3 also holds here, that is,

$$E[m^*] \le n(1-p).$$

Therefore, the probability that there are less than n columns left at Step 4 (i.e., the failure probability of Algorithm 2) is at most $\Pr[m^* > n] \le 1-p$. The following theorem is established.

Theorem 3 *Given n, d, $z > 1$ and $0 < p < 1$, Algorithm 2 successfully constructs a $t \times n$ $(d;z)$-disjunct matrix with probability at least p, with*

$$t \le cd^2 \left(\log \frac{2}{1-p} + \log n \right) + 2(1+\epsilon)dz + O\left(\frac{z^2}{\ln n} \right),$$

where $\epsilon \approx 3.92$ and $c \approx 4.28$ are constants. When $z = o(d \ln n)$, the algorithm runs in expected $O(n^2 \ln n)$ time.

2.3 Two-Stage Pooling Designs

New two-stage pooling designs, which require a number of tests asymptotically no more than a factor of $\frac{3}{\log 3}$ (the factor approaches $\log_2 e \approx 1.44$ as d tends to infinity) of the information-theoretic lower bound $d \log(n/d)$, will be presented next.

This improves the previously best upper bound of $4(1+o(1))$ times the information theoretic bound in [24] by a factor of more than 2.

For a 0/1 matrix M, let C denote the set of columns of M, recall that M is d-disjunct if for any d-sized subset D of C, each column in $C-D$ is not covered by $U(D)$, where $U(D)$ denotes the union of the columns in D. Such matrices form the basis for nonadaptive (one-stage) pooling designs. However, a d-disjunct matrix with n columns requires no less than $\Omega(\frac{d^2 \log n}{\log d})$ rows, which is a factor of $d/\log d$ of the information-theoretic lower bound.

Instead of determining all the positives immediately, in [24] the authors relax the property of M by introducing the concept of (d,k)-*resolvable* matrices, which form a good two-stage group testing regimen. A 0/1 matrix M is called (d,k)-*resolvable* if, for any d-sized subset D of C, there are fewer than k columns in $C-D$ that are covered by $U(D)$. Thus, a matrix is d-disjunct if and only if it is $(d,1)$-resolvable.

For a set of n items in which at most d are positives, one can construct a "trivial two-stage" pooling design based on a $t \times n$ (d,k)-resolvable matrix as follows. Define the first round tests according to the rows of the matrix. By identifying the items in a negative pool (a pool with negative test outcome) as negatives, one can restrict the positives to a set D' of size smaller than $d+k$. Then, perform an additional round of tests on each item in D' individually. Thus, the total number of tests of the two stages is less than $t+d+k$.

2.3.1 Near Optimal Two-Stage Pooling Designs

Let M_1 be a q-ary matrix, and let C denote the set of columns of M_1. Matrix M_1 is said to be $(d,1;k)$-*resolvable* if, for any d-sized subset D of C, there are fewer than k columns in $C-D$ that are *covered* by D. Here by saying a column c is *covered* by D, it means that for each element of c, the element appears at least once in some column of D in the same row. By applying the transformation in Theorem 1, one can turn a $t_0 \times n$ q-ary $(d,1;k)$-resolvable matrix into a $t \times n$ (d,k)-resolvable matrix with $t \le t_0 q$.

Let M' be a random $t_0 \times n$ q-ary (where q will be specified later) matrix with each cell assigned randomly from $\{1,2,\ldots,q\}$, independently and uniformly. For each set D of d columns and a column $c \notin D$, for each element c_i $(i=1,2,\ldots,t_0)$ in c, the probability that c_i appears in some column of D in the same row is $1-\left(1-\frac{1}{q}\right)^d$; thus, the probability that every element in c appears in some column of D in the same row, that is, c is covered by D, is $[1-\left(1-\frac{1}{q}\right)^d]^{t_0}$. Parameter t_0 is chosen such that

$$\left[1-\left(1-\frac{1}{q}\right)^d\right]^{t_0} = \frac{1}{n-d},$$

that is,

$$t_0 = -\frac{\log(n-d)}{\log\left[1-\left(1-\frac{1}{q}\right)^d\right]}.$$

Let C denote the set of columns of M'. For any set D of d columns of M', and for each $c \in C - D$, let X_c be the indicator variable such that $X_c = 1$ if and only if c is covered by D. Then,

$$\Pr[X_c = 1] = \frac{1}{n-d}.$$

Define

$$X_D = \sum_{c \in C-D} X_c.$$

Then, X_D is the random variable denoting the number of columns in $C - D$ that are covered by D. Since X_D is the sum of $(n-d)$ i.i.d. 0/1 random variables and $E[X_D] = 1$, the Chernoff's bound implies that the probability that D covers at least $(1+\delta)$ columns in $C - D$ is

$$\Pr[X_D \geq (1+\delta)] \leq \frac{e^\delta}{(1+\delta)^{1+\delta}}.$$

Therefore, the probability that M' is not $(d, 1; 1+\delta)$-resolvable, that is, there exists some set D of d columns that covers at least $(1+\delta)$ columns in $C - D$, is at most

$$p = \binom{n}{d} \frac{e^\delta}{(1+\delta)^{1+\delta}}.$$

In order to satisfy $p < 1$, it suffices to assign δ such that

$$\left(\frac{1+\delta}{e}\right)^{1+\delta} = n^d,$$

since which implies that

$$\frac{(1+\delta)^{1+\delta}}{e^\delta} > \frac{(1+\delta)^{1+\delta}}{e^{1+\delta}} = n^d > \binom{n}{d}.$$

Notice that $(\frac{1+\delta}{e})^{1+\delta} = n^d$ implies $(\frac{1+\delta}{e})^{\frac{1+\delta}{e}} = n^{\frac{d}{e}}$; thus,

$$1 + \delta = (1 + o(1)) \frac{d \ln n}{\ln(d \ln n)}.$$

Hence, by probabilistic arguments, the existence of a $t_0 \times n$ q-ary $(d, 1; 1+\delta)$-resolvable matrix with t_0 and δ as specified above has been proved.

By applying the transformation in Theorem 1, one can turn the $t_0 \times n$ q-ary $(d, 1; 1+\delta)$-resolvable matrix M' into a (binary) $t \times n$ $(d, 1+\delta)$-resolvable matrix M with

$$t \leq t_0 q < -\frac{q \log n}{\log\left[1-(1-\frac{1}{q})^d\right]}.$$

Define

$$C_d(x) = \frac{x}{-\log\left[1-(1-\frac{1}{x})^d\right]}, \quad \text{for } x > 1.$$

Choosing q to be the positive integer that minimizes $C_d(x)$, and let $C_d = C_d(q)$. Then, $t \leq C_d \log n$. For $d \geq 1$,

$$C_d/d \leq C_d(3d)/d = \frac{3}{-\log\left[1-(1-\frac{1}{3d})^d\right]} \leq \frac{3}{\log 3}.$$

Also it is not hard to see that when $d = 1$, $C_1 = C_1(3) = \frac{3}{\log 3}$ indeed holds. Furthermore, the following lemma estimates that $q = \Theta(d)$ and $C_d \to d \log e$ as $d \to \infty$.

Lemma 4 *For $d \geq 1$, let $q = q(d)$ be the point that minimizes $C_d(x) = \frac{x}{-\log[1-(1-\frac{1}{x})^d]}$ for $x > 1$, and let $C_d = C_d(q)$. Then, $q(d) = \Theta(d)$, and $\lim_{d\to\infty} C_d/d = \log e$.*

To prove Lemma 4, the following useful fact is proved first.

Fact 1 *Let $f(y) = \ln y \ln(1-y)$, $0 < y < 1$. Then $f(y)$ achieves maximum at $y = \frac{1}{2}$.*

Proof of Fact 1: By symmetry, it is sufficient to show that $f'(y) > 0$ for $0 < y < \frac{1}{2}$. Since

$$\begin{aligned} f'(y) &= \frac{1}{y}\ln(1-y) - \frac{1}{1-y}\ln y \\ &= \frac{1}{y(1-y)}\left[(1-y)\ln(1-y) - y\ln y\right], \end{aligned}$$

let

$$g(y) = (1-y)\ln(1-y) - y\ln y,$$

Next it will be showed that $g(y) > 0$ for $0 < y < \frac{1}{2}$.

Rewrite

$$g(y) = \ln(1-y) + y\ln\frac{1}{y(1-y)}.$$

For $0 < y < \frac{1}{2}$, $\ln\frac{1}{y(1-y)} > \ln 4$ since $y(1-y) < \frac{1}{4}$; thus,

$$g(y) > \ln(1-y) + y\ln 4.$$

Let

$$h(y) = \ln(1-y) + y\ln 4.$$

Notice that $h'(y) = \ln 4 - \frac{1}{1-y}$, $h'(y) > 0$ for $0 < y < 1 - \frac{1}{\ln 4}$ and $h'(y) < 0$ for $1 - \frac{1}{\ln 4} < y < \frac{1}{2}$. Thus, $h(y)$ is monotone increasing when $0 < y < 1 - \frac{1}{\ln 4}$ and monotone decreasing when $1 - \frac{1}{\ln 4} < y < \frac{1}{2}$. From $h(0) = h(\frac{1}{2}) = 0$, one can obtain that

$$h(y) > 0 \text{ for } 0 < y < \frac{1}{2}.$$

Therefore, for $0 < y < \frac{1}{2}$, $g(y) > h(y) > 0$, and so $f'(y) = \frac{1}{y(1-y)} g(y) > 0$. □

Proof of Lemma 4: Notice that if q_1 satisfies $(1 - \frac{1}{q_1})^d = \frac{1}{2}$, then $q_1 = \Theta(d)$ since $\frac{q_1}{d} \to \log e$ as $d \to \infty$. Moreover,

$$C_d(q_1) = q_1 = \Theta(d).$$

The lemma is proved by contradiction. First assume that $q(d) = O(d)$ does not hold, that is, for any $c > 0$ and any $d_0 > 0$, there exists $d > d_0$ such that $q(d) > cd$. Then, since $\frac{q}{d} > c$ (for simplicity q is used instead of $q(d)$, if it is clear from the context), as $c \to \infty$,

$$\begin{aligned} C_d(q) &= \frac{q}{-\log\left[1 - \left(1 - \frac{1}{q}\right)^d\right]} \\ &\sim \frac{q}{-\log\left[1 - \left(1 - \frac{d}{q}\right)\right]} \\ &= d\,\frac{\frac{q}{d}}{\log\frac{q}{d}} \\ &= \omega(d); \end{aligned}$$

here, $a \sim b$ means that $\lim_{c\to\infty} \frac{a}{b} = 1$. However, this contradicts since q is the point that minimizes $C_d(q)$ and on the other hand $C_d(q_1) = \Theta(d)$. On the other hand, assume that $q(d) = \Omega(d)$ does not hold, that is, for any $c > 0$ and any $d_0 > 0$, there exists $d > d_0$ such that $q(d) < cd$. Write

$$\begin{aligned} C_d(q) &= \frac{q}{-\log\left[1 - \left(1 - \frac{1}{q}\right)^d\right]} \\ &= \frac{q\ln 2}{-\ln\left\{1 - \left[\left(1 - \frac{1}{q}\right)^q\right]^{\frac{d}{q}}\right\}}. \end{aligned}$$

Since $0 < \left(1-\frac{1}{q}\right)^q < \frac{1}{e}$ for $q > 1$, as $c \to 0$, $\frac{d}{q} > \frac{1}{c} \to \infty$, and $\left[\left(1-\frac{1}{q}\right)^q\right]^{\frac{d}{q}} < e^{-\frac{d}{q}} \to 0$; thus,

$$C_d(q) \sim \frac{q \ln 2}{\left[\left(1-\frac{1}{q}\right)^q\right]^{\frac{d}{q}}} > e^{\frac{d}{q}} q \ln 2 = d \frac{e^{\frac{d}{q}} \ln 2}{\frac{d}{q}} = \omega(d);$$

this also contradicts (here $a \sim b$ means that $\lim_{c\to 0} \frac{a}{b} = 1$). Therefore, $q(d) = \Theta(d)$.

Next C_d as $d \to \infty$ is estimated. Since $\left(1-\frac{1}{q}\right)^q < \frac{1}{e}$ for $q > 1$, thus

$$\left(1-\frac{1}{q}\right)^d < \left(\frac{1}{e}\right)^{\frac{d}{q}} = e^{-\frac{d}{q}},$$

and

$$-\log\left[1-\left(1-\frac{1}{q}\right)^d\right] < -\log\left(1-e^{-\frac{d}{q}}\right),$$

it follows that

$$\begin{aligned} C_d(q) &= \frac{q}{-\log\left[1-\left(1-\frac{1}{q}\right)^d\right]} \\ &> \frac{q}{-\log\left(1-e^{-\frac{d}{q}}\right)} \\ &= \frac{d \ln 2}{\left[-\frac{d}{q}\ln\left(1-e^{-\frac{d}{q}}\right)\right]}. \end{aligned}$$

Let $y = e^{-\frac{d}{q}}$, then $-\frac{d}{q} = \ln y$, and $C_d(q) > \frac{d \ln 2}{\ln y \ln(1-y)}$. Since $\ln y \ln(1-y)$ achieves maximum at $y = \frac{1}{2}$ (Fact 1), $C_d(q) > d \log e$ for $q > 1$, thus $C_d > d \log e$ for $d \geq 1$. On the other hand, as mentioned at the beginning of the proof, as $d \to \infty$, $\frac{q_1}{d} \to \log e$, and $C_d(q_1) = q_1 \to d \log e$. Therefore, as $d \to \infty$, $C_d \to d \log e$. □

The above arguments showed existence of a $t \times n$ $(d, 1+\delta)$-resolvable matrix with $t \leq C_d \log n$ and $1 + \delta = (1 + o(1))\frac{d \ln n}{\ln(d \ln n)}$, which implies the following theorem.

Theorem 4 *Given n and d, there exists a two-stage pooling design for finding up to d positives from n items using no more than $C_d \log n + d + \delta + 1$ tests, where*

$$C_d = \min_{x \in \mathcal{N}} \frac{x}{-\log\left[1 - \left(1 - \frac{1}{x}\right)^d\right]} \leq \frac{3}{\log 3} d,$$

for $d \geq 1$, *and* $\delta = (1 + o(1))\frac{d \ln n}{\ln(d \ln n)}$. *Moreover,*

$$\lim_{d \to \infty} C_d / d = \log e.$$

2.4 Probabilistic Pooling Designs

A probabilistic pooling design identifying up to d positives from n items with high probability will be presented next. In a probabilistic group testing algorithm, one may identify a positive item as negative; such a wrongly identified item is called a *false negative*; a negative item which is wrongly identified as positive is called a *false positive*. Clearly, the algorithm correctly identifies all positives if and only if there are no false positives or false negatives. Previous works on probabilistic nonadaptive group testing algorithms can be found from, among others, [40,41,44].

Algorithm. Given n and d, first construct a $t_0 \times n$ random q-ary matrix M' with each cell randomly assigned from $\{1, 2, \ldots, q\}$ independently and uniformly (where t_0 and q will be specified later). Then, use the transformation in Theorem 1 to obtain a $t \times n$ 0/1 matrix M with $t \leq t_0 q$. Associate the n items with the columns of M, and test the pools indicated by the rows of M. The items not in any negative pool are identified as positives.

Analysis. Let D be the set of columns corresponding to the positives, then $|D| \leq d$. First, it is easy to see that no positive item will be identified as negative if there is no error in the test outcomes. For any negative item, let c denote the column associated with it, then the item is wrongly identified if and only if c is covered by $U(D)$ in M, or equivalently, c is covered by D in M' (here the same notations c and D are used for different matrices M and M', to denote the corresponding columns). The probability that c is covered by D, as analyzed in Sect. 2.3, is

$$\left[1 - \left(1 - \frac{1}{q}\right)^{|D|}\right]^{t_0} \leq \left[1 - \left(1 - \frac{1}{q}\right)^{d}\right]^{t_0}.$$

Choosing q and t_0 such that

$$\left[1 - \left(1 - \frac{1}{q}\right)^{d}\right]^{t_0} = \frac{1 - p}{n},$$

that is,

$$t_0 = -\frac{\log n + \log \frac{1}{1-p}}{\log[1 - (1 - \frac{1}{q})^d]}.$$

Then, the probability that there exists some negative item wrongly identified is no more then

$$(n - |D|)[1 - (1 - \frac{1}{q})^d]^{t_0} \leq 1 - p,$$

which implies that with probability at least p, the above algorithm successfully identifies all the positives. The number of pools required is no more than

$$t \leq t_0 q = -\frac{q}{\log[1 - (1 - \frac{1}{q})^d]}(\log n + \log \frac{1}{1-p}).$$

By choosing q to be the positive integer minimizing

$$C_d(x) = \frac{x}{-\log[1 - (1 - \frac{1}{x})^d]} \text{ for } x > 1,$$

obtaining

$$t \leq C_d(\log n + \log \frac{1}{1-p}).$$

Theorem 5 *The above one-stage algorithm, with probability at least p, correctly identifies up to d positives from n items using no more than $C_d(\log n + \log \frac{1}{1-p})$ tests.*

Remarks

1. The one-stage probabilistic pooling design is also transversal. This design never gets false negatives, while the probabilistic algorithms in [40, 41, 44] never get false positives. The algorithm in [44] identifies up to 9 positives from 18,918,900 items using 5,460 tests, with success probability of 98.5 %. For the method proposed here, $n = 18{,}918{,}900$, $d = 9$, and $p = 0.985$, by choosing $q = 14$, it requires $C_d(q)(\log n + \log \frac{1}{1-p}) < 408$ tests, which is much fewer.
2. In contrast to the two-stage design in Sect. 2.3, this probabilistic algorithm is explicitly given and can be easily implemented in practice. In addition, one can extend this algorithm to two stage, by performing an additional round of individual tests on the candidate positives identified by the first round, so that no item will be wrongly identified. It is easy to verify that, for this extended two-stage probabilistic algorithm, by choosing the same value q, and choosing t_0 such that $[1 - (1 - \frac{1}{q})^d]^{t_0} = \frac{1}{n}$, the expected total number of tests required is no more than $C_d \log n + d + 1$, which is better than the deterministic two-stage design in Sect. 2.3.

2.5 Conclusion and Future Studies

New one- and two-stage pooling designs, together with new probabilistic pooling designs, are presented in this section. The approach presented works for both error-free and error-tolerance scenarios. The following remarks end off this section:

1. The constructions of pooling designs in Sects. 2.3 and 2.4 can also be generalized to error-tolerance case, in a similar manner as in the construction of $(d; z)$-disjunct matrices in Sect. 2.2.3. The details are omitted due to the similarities.

2. Efficient constructions (i.e., in time polynomial in n and d) of the two-stage designs in Sect. 2.3 are not given. Up to now, no efficient construction of two-stage pooling designs using the number of tests within a constant factor of the information theoretic lower bound is known. In [14], the construction requires $\binom{n}{\frac{n}{2d}}$ time, and in [24], the authors gave existence proof as in this section. Although once such a design is found it can be used as many times as wanted, efficient construction is an important issue.
3. The two-stage pooling design presented in Sect. 2.3 uses the number of tests asymptotically within a factor of C_d/d ($\leq \frac{3}{\log 3}$ for general d, and tends to $\log_2 e \approx 1.44$ as $d \to \infty$) of the information theoretic bound $d \log(n/d)$. Can two-stage algorithms do as good as fully adaptive algorithms, that is, achieve a factor of asymptotically 1 of the information theoretic bound? Or, how good could it be?
4. Last but the least, efficient (i.e., polynomial time in n and d) deterministic constructions of d-disjunct matrices with $t = O(d^2 \log n)$ are known [2, 50]. Regarding the leading constant within the big-O notation, the results indicate that they are considerably larger than the result given in this section (where the leading constant is approximately 4.28). An efficient randomized construction of d-disjunct matrices with $t = O(d^2 \log n)$ and efficient decoding (in time polynomial in t) is given in [33], and an efficient deterministic construction with the same properties is obtained recently [45]. Improved constructions of disjunct matrices are interesting to investigate.

3 New Complexity Results on Nonunique Probe Selection

Given a collection of n targets and a sample S containing at most d of these targets, and a collection of m probes each of them hybridizes to a subset of the given targets, the goal is to select a subset of probes, such that all targets in S can be identified by observing the hybridization reactions between the selected probes and S. For each probe p, there is hybridization reaction between p and S if S contains at least one target that hybridizes with p, otherwise there is no hybridization reaction. The above probe selection problem has been extensively studied recently [5, 31, 51, 52, 56] due to its important applications, particularly in molecular biology. For example, one application of this identification problem is to identify viruses (targets) from a blood sample. The presence or absence of the viruses is established by observing the hybridization reactions between the blood sample and some probes; here, each probe is a short oligonucleotide of size 8–25 that can hybridize to one or more of the viruses.

A probe is called *unique* if it hybridizes to only one target; otherwise, it is called *nonunique*. Identifying targets using unique probes is straightforward. However, in situations where the targets have a high degree of similarity, for instance when identifying closely related virus subtypes, finding unique probes for every target is difficult. In [54], Schliep, Torney, and Rahmann proposed a group testing method using nonunique probes to identify targets in a given sample. Since each

nonunique probe can hybridize with more than one target, the identification problem becomes more complicated. One important issue is to select a subset from the given nonunique probes so that the hybridization results can be decoded, that is, determine the presence or absence of targets in sample S. Also, the number of selected probes is exactly the number of hybridization experiments required, and it is desirable to select as few probes as possible to reduce the experimental cost. In [35, 54], two heuristics using greedy and linear programming-based techniques, respectively, are proposed for choosing a suitable subset of nonunique probes. In [11], the computational complexities of some basic problems in nonunique probe selection are studied, in the context of the theory of NP-completeness (see Chap. 10 in [17, 19] and [30]). The complexity results in [11] will be presented in this section.

3.1 Preliminaries

The nonunique probe selection problem can be formulated as follows. Given a collection of n targets $t_1, t_2, \ldots, t_n$, and a collection of m nonunique probes $p_1, p_2, \ldots, p_m$, a sample S is known to contain at most d of the n targets. The probe-target hybridizations can be represented by an $m \times n$ 0-1 matrix M. $M_{i,j} = 1$ indicates that probe p_i hybridizes to target t_j, and $M_{i,j} = 0$ indicates otherwise. The subset of probes selected corresponds to a subset of rows in M, which forms a submatrix H of M with the same number of columns. The hybridization results between the selected probes and S also can be represented as a 0-1 vector V. $V_i = 1$ indicates that there is hybridization reaction between p_i and S, that is, p_i hybridizes to at least one target in S, and $V_i = 0$ indicates otherwise. If there is no error in the hybridization experiments, then V is equal to the union of the columns of H that correspond to the targets in S. Here, the union of a subset of columns is simply the Boolean sum of these column vectors. In order to identify all targets in S, the submatrix H should satisfy that all unions of up to d columns in H are different; in other words, H should be $\bar{d}$-*separable*. Also, as mentioned above, it is desirable to minimize the number of rows in H.

A matrix H is said to be $\bar{d}$-separable if all unions of up to d columns in H are different. However, the following equivalent definition is more useful in the proofs here. Let H be a $t \times n$ Boolean matrix. For each $i \in \{1, 2, \ldots, t\}$, define

$$H_i = \{j \mid 1 \leq j \leq n, H_{i,j} = 1\}.$$

For any subset S of $\{1, 2, \ldots, n\}$ and any $i \in \{1, 2, \ldots, t\}$, write

$$H_i(S) = \begin{cases} 1 \text{ if } H_i \cap S \neq \emptyset; \\ 0 \text{ otherwise.} \end{cases}$$

Two sets $S_1, S_2 \subseteq \{1, 2, \ldots, n\}$ are said to be *separated* by H if there exists an integer i, $1 \leq i \leq t$, such that $H_i(S_1) \neq H_i(S_2)$. Matrix H is said to be $\bar{d}$-separable if for any two different subsets S_1, S_2 of $\{1, 2, \ldots, n\}$, with $|S_1| \leq d$ and $|S_2| \leq d$, S_1 and S_2 can be separated by H.

3.2 Complexity of Minimal $\bar{d}$-Separable Matrix

In nonunique probe selection, one natural problem of interests is to determine whether a submatrix H chosen is $\bar{d}$-separable and minimal. By minimal, it means that the removal of any row from H will make it no longer $\bar{d}$-separable. The problem can be formulated as follows.

MIN-SEPARABILITY (MINIMAL SEPARABILITY): Given a $t \times n$ Boolean matrix H and an integer $d \leq n$, determine whether it is true that (a) H is $\bar{d}$-separable, and (b) for any submatrix Q of H of size $(t-1) \times n$, Q is not $\bar{d}$-separable.

For a given binary matrix H and a positive integer d, the problem to determine whether H is $\bar{d}$-separable is known to be $coNP$-complete ([17], Theorem 10.2.1). Here a DP-completeness proof of problem MIN-SEPARABILITY will be presented.

The class DP is the collection of sets A which are the intersection of a set $X \in NP$ and a set $Y \in coNP$. The notion of DP-completeness has been used to characterize the complexity of the "exact-solution" version of many NP-complete problems. For instance, the exact traveling salesman problem, which asks, for a given edge-weighted complete graph G and a constant K, whether the minimum weight of a traveling salesman tour of the graph G is equal to K, is DP-complete (see [47], Theorem 17.2). In addition, the "critical" version of some NP-complete problems is also known to be DP-complete. For instance, the following problem is the critical version of the 3-satisfiability problem and has been shown to be DP-complete by Papadimitriou and Wolfe [48]:

MIN-3-UNSAT: Given a 3-CNF Boolean formula φ which consists of clauses $C_1, C_2, \ldots, C_m$, determine whether it is true that (a) φ is not satisfiable, and (b) for any j, $1 \leq j \leq m$, the formula φ_j that consists of all clauses C_ℓ, $\ell \in \{1, 2, \ldots, m\} - \{j\}$, is satisfiable.

Although most exact-solution version of NP-complete problems have been shown to be DP-complete, many critical versions are not known to be DP-complete. The problem MIN-SEPARABILITY may be viewed as a critical version of the $\bar{d}$-separability problem. Its DP-completeness will be proved by constructing a reduction from MIN-3-UNSAT.

Theorem 6 MIN-SEPARABILITY *is* DP*-complete.*

Proof Recall that $DP = \{X \cap Y \mid X \in NP,\ Y \in coNP\}$. A problem A is *DP-complete* if $A \in DP$ and, for all $B \in DP$, $B \leq_m^P A$. For convenience, for any $t \times n$ matrix H, $\widetilde{H}_j$ is used to denote the $(t-1) \times n$ submatrix of H with the jth row removed. First, to see that MIN-SEPARABILITY $\in DP$, let

$$X = \left\{(H, d) \mid H \text{ is a } t \times n \text{ 0/1 matrix}, 1 \leq d \leq n, (\forall j, 1 \leq j \leq t)\ \widetilde{H}_j \text{ is not } \bar{d}\text{-separable}\right\},$$

and

$$Y = \{(H, d) \mid H \text{ is a } t \times n \text{ 0/1 matrix}, 1 \leq d \leq n, H \text{ is } \bar{d}\text{-separable}\}.$$

It is clear that MIN-SEPARABILITY $= X \cap Y$. It is also not hard to see that $X \in NP$ and $Y \in coNP$. In particular, to see that $X \in NP$, note that $(H, d) \in X$ if and only if there exist $2t$ subsets $S_{j,1}$, $S_{j,2}$ of $\{1, 2, \ldots, n\}$, for $j \in \{1, 2, \ldots, t\}$, such that, for each j, $H_k(S_{j,1}) = H_k(S_{j,2})$ for all $k \in \{1, 2, \ldots, t\} - \{j\}$.

Next, a reduction from MIN-3-UNSAT to MIN-SEPARABILITY will be described. Let φ be a 3-CNF Boolean formula which consists of m clauses $C_1, C_2, \ldots, C_m$, over n variables $x_1, x_2, \ldots, x_n$. For each $j \in \{1, 2, \ldots, m\}$, let φ_j denote the Boolean formula that consists of all clauses C_ℓ for $\ell \in \{1, 2, \ldots, m\} - \{j\}$. From φ, a $(3n + m + 1) \times (2n + 2)$ Boolean matrix H will be constructed, and define $d = n + 1$. For convenience, the columns of H are denoted by

$$X = \{x_i, \bar{x}_i \mid 1 \le i \le n\} \cup \{y, z\},$$

and denote the rows of H by

$$T = \{x_i, \bar{x}_i, u_i \mid 1 \le i \le n\} \cup \{y\} \cup \{C_j \mid 1 \le j \le m\}.$$

Next H is defined by specifying each row of it:

1. For each $1 \le i \le n$, let $H_{x_i} = \{x_i\}$, $H_{\bar{x}_i} = \{\bar{x}_i\}$, and $H_{u_i} = \{x_i, \bar{x}_i, z\}$.
2. $H_y = \{y\}$.
3. For each $1 \le j \le m$, let $H_{C_j} = \{x_i \mid x_i \in C_j\} \cup \{\bar{x}_i \mid \bar{x}_i \in C_j\} \cup \{y, z\}$ (so that $|H_{C_j}| = 5$).

To prove the correctness of the reduction, first verify that if φ is not satisfiable, then H is $\bar{d}$-separable. To see this, let S_1 and S_2 be two subsets of X, each of size $\le n + 1$.

Case 1. $S_1 - \{z\} \ne S_2 - \{z\}$. Then, there exists $v \in X - \{z\}$ such that $v \in S_1 \Delta S_2$. Then, $H_v(S_1) \ne H_v(S_2)$.

Case 2. $S_1 - \{z\} = S_2 - \{z\}$. Then, it must be true that $S_1 \Delta S_2 = \{z\}$. Without loss of generality, assume $S_2 = S_1 \cup \{z\}$. Note that $|S_2| \le n + 1$ implies $|S_1| \le n$.

Subcase 2.1. There exists an integer i such that $|S_1 \cap \{x_i, \bar{x}_i\}| \ne 1$. First, if $|S_1 \cap \{x_i, \bar{x}_i\}| = 0$ for some i, then $H_{u_i}(S_1) = 0$ and $H_{u_i}(S_2) = 1$ (because $z \in S_2$). Next, if $|S_1 \cap \{x_i, \bar{x}_i\}| = 2$ for some i, then it must have $|S_1 \cap \{x_k, \bar{x}_k\}| = 0$ for some k, because $|S_1| \le n$. Then, again $H_{u_k}(S_1) = 0 \ne 1 = H_{u_k}(S_2)$.

Subcase 2.2. $|S_1 \cap \{x_i, \bar{x}_i\}| = 1$ for all $i \in \{1, 2, \ldots, n\}$. Note that, in this case, $y \notin S_1$. Define a Boolean assignment $\tau : \{x_1, x_2, \ldots, x_n\} \to \{\text{TRUE}, \text{FALSE}\}$ by

$$\tau(x_i) = \text{TRUE} \text{ if and only if } x_i \in S_1.$$

Since φ is not satisfiable, there exists a clause C_j that is not satisfied by τ. It means that $C_j \cap S_1 = \emptyset$, and so $H_{C_j}(S_1) = 0$. However, $H_{C_j}(S_2) = 1$ since $z \in S_2$.

The above completes the proof that H is $\bar{d}$-separable.

Next, it will be showed that if φ_j is satisfiable for all $j = 1, 2, \ldots, m$, then $\widetilde{H}_v$ is not $\bar{d}$-separable for all $v \in T$. First, for $v \in X - \{z\}$, let $S_1 = \{z\}$ and $S_2 = \{v, z\}$.

Then, for all rows $w \in X - \{z, v\}$, $H_w(S_1) = 0 = H_w(S_2)$. Also, for all other rows $w \in T - X$, $H_w(S_1) = H_w(S_2) = 1$ since $z \in H_w$. So, S_1 and S_2 are not separable by $\widetilde{H}_v$.

Next, consider the case $v = u_i$ for some $i \in \{1, 2, \ldots, n\}$. Let

$$S_1 = \{x_k \mid 1 \leq k \leq n, k \neq i\} \cup \{y\},$$

and $S_2 = S_1 \cup \{z\}$. It is clear that $|S_1| = n$ and $|S_2| = n + 1$. Now the following claim is made: S_1 and S_2 are not separable by $\widetilde{H}_{u_i}$.

To prove the claim, note that the rows H_{x_k}, $H_{\bar{x}_k}$, for $1 \leq k \leq n$, and row H_y cannot separate S_1 from S_2, since $S_1 - \{z\} = S_2 - \{z\}$. Also, rows $H_{u_k}(S_1) = H_{u_k}(S_2) = 1$, for all $k \in \{1, 2, \ldots, n\} - \{i\}$, because $|S_1 \cap \{x_k, \bar{x}_k\}| = 1$ if $k \neq i$. In addition, for any $j = 1, 2, \ldots, m$, $H_{C_j}(S_1) = 1 = H_{C_j}(S_2)$, since $y \in S_1$. It follows that $\widetilde{H}_{u_i}$ cannot separate S_1 from S_2.

Finally, consider the case $v = C_j$ for some $j \in \{1, 2, \ldots, m\}$. Note that φ_j is satisfiable. So, there is a Boolean assignment $\tau : \{x_1, x_2, \ldots, x_n\} \to \{\text{TRUE}, \text{FALSE}\}$ satisfying all clauses C_ℓ, except C_j. Define

$$S_1 = \{x_i \mid \tau(x_i) = \text{TRUE}\} \cup \{\bar{x}_i \mid \tau(x_i) = \text{FALSE}\},$$

and $S_2 = S_1 \cup \{z\}$. Then, similar to the argument for the case $v = u_i$, one can verify that $H_w(S_1) = H_w(S_2)$ for $w \in X - \{z\}$, and for $w \in \{u_i \mid 1 \leq i \leq n\}$. In addition, for any clause C_ℓ, with $\ell \neq j$, C_ℓ is satisfied by τ. It follows that $C_\ell \cap S_1 \neq \emptyset$ and $H_{C_\ell}(S_1) = 1 = H_{C_\ell}(S_2)$. This completes the proof that $\widetilde{H}_v$ is not $\bar{d}$-separable, for all $v \in T$.

Conversely, it will be showed that if $\varphi \notin$ MIN-3-UNSAT, then $(H, n+1) \notin$ MIN-SEPARABILITY. First, consider the case that φ is a satisfiable formula. Let $\tau : \{x_1, x_2, \ldots, x_n\} \to \{\text{TRUE}, \text{FALSE}\}$ be a Boolean assignment satisfying φ. Define

$$S_1 = \{x_i \mid \tau(x_i) = \text{TRUE}\} \cup \{\bar{x}_i \mid \tau(x_i) = \text{FALSE}\},$$

and $S_2 = S_1 \cup \{z\}$. Then, similar to the earlier proof, one can verify that H cannot separate S_1 from S_2. In particular, $H_{C_j}(S_1) = 1$ for all $j \in \{1, 2, \ldots, m\}$, because τ satisfies C_j and so $C_j \cap S_1 \neq \emptyset$. Thus, $(H, n+1) \notin$ MIN-SEPARABILITY.

Next, assume that there exists an integer $j \in \{1, 2, \ldots, m\}$ such that φ_j is not satisfiable. The following claim is made: $\widetilde{H}_{C_j}$ is $\bar{d}$-separable. The proof of the claim is similar to the first part of the proof (for the statement that if φ is not satisfiable then H is $\bar{d}$-separable).

Case 1. $S_1 - \{z\} \neq S_2 - \{z\}$. Then, there exists $v \in X - \{z\}$ such that $v \in S_1 \Delta S_2$. So, $H_v(S_1) \neq H_v(S_2)$.

Case 2. $S_1 - \{z\} = S_2 - \{z\}$. Then, it must be true that $S_1 \Delta S_2 = \{z\}$, and one may assume $S_2 = S_1 \cup \{z\}$. It must have $|S_2| \leq n + 1$ and $|S_1| \leq n$.

Subcase 2.1. There exists an integer i such that $|S_1 \cap \{x_i, \bar{x}_i\}| \neq 1$. Similar to the earlier proof, if $|S_1 \cap \{x_i, \bar{x}_i\}| = 0$ for some $i = 1, 2, \ldots, n$, then H_{u_i} can be

used to separate S_1 from S_2. If $|S_1 \cap \{x_i, \bar{x}_i\}| = 2$ for some $i = 1, 2, \ldots, n$, then $|S_1 \cap \{x_k, \bar{x}_k\}| = 0$ for some k, and again H_{u_k} separates S_1 from S_2.

Subcase 2.2. $|S_1 \cap \{x_i, \bar{x}_i\}| = 1$ for all $i \in \{1, 2, \ldots, n\}$. Then, since $|S_1| \leq n$, $y \notin S_1$. Define a Boolean assignment $\tau : \{x_1, x_2, \ldots, x_n\} \rightarrow \{\text{TRUE}, \text{FALSE}\}$ by $\tau(x_i) = \text{TRUE}$ if and only if $x_i \in S_1$. Since φ_j is not satisfiable, there exists a clause C_ℓ, $\ell \neq j$, such that $\tau(C_\ell) = \text{FALSE}$. It means that $C_\ell \cap S_1 = \emptyset$, and so $H_{C_\ell}(S_1) = 0$. However, $H_{C_\ell}(S_2) = 1$ since $z \in S_2$. So, H_{C_ℓ} separates S_1 from S_2. This completes the proof that $\widetilde{H}_{C_j}$ is $\bar{d}$-separable, and hence $(H, n+1) \notin$ MIN-SEPARABILITY. □

3.3 Minimum $\bar{d}$-Separable Submatrix

A more important problem in nonunique probe selection is to find a minimum subset of probes that can identify up to d targets in a given sample. In the matrix representation, the problem can be formulated as the following: Given a binary matrix M and a positive integer d, find a minimum $\bar{d}$-separable submatrix of M with the same number of columns (problem MIN-$\bar{d}$-SS in [17], Chap. 10).

For $d = 1$, MIN-$\bar{d}$-SS has been proved to be NP-hard ([17], Theorem 10.3.2), by modifying a reduction used in the proof of the NP-completeness of the problem MINIMUM-TEST-SETS in [30]. For fixed $d > 1$, MIN-$\bar{d}$-SS is believed to be NP-hard; however, up to now, no formal proof is known. Next the decision version of MIN-$\bar{d}$-SS is considered.

$\bar{d}$-SS ($\bar{d}$-SEPARABLE SUBMATRIX): Given a $t \times n$ Boolean matrix M and two integers $d, k > 0$, determine whether there is a $k \times n$ submatrix H of M that is $\bar{d}$-separable.

Recall that Σ_2^P is the complexity class of problems that are solvable in nondeterministic polynomial time with the help of an NP-complete set as an oracle. For instance, the following problem SAT$_2$ is Σ_2^P-complete ([19], Theorem 3.13): Given a Boolean formula φ over two disjoint sets X and Y of variables, determine whether there exists an assignment to variables in X so that the resulting formula (over variables in Y) is a tautology. It is easy to see that $\bar{d}$-SS is in Σ_2^P. It is conjectured to be Σ_2^P-complete. Here a similar problem that is a little more general than $\bar{d}$-SS will be considered, and its Σ_2^P-completeness will be proved.

$\bar{d}$-SSRR ($\bar{d}$-SEPARABLE SUBMATRIX WITH RESERVED ROWS): Given a $t \times n$ Boolean matrix M and three integers $d > 0$, s, and $k \geq 0$, determine whether there is a $\bar{d}$-separable $(s + k) \times n$ submatrix H of M that contains the first s rows of M and k rows from the remaining $t - s$ bottom rows of M.

Let φ be a Boolean formula, an *implicant* of φ is a conjunction C of literals that implies φ. The following problem is proved to be Σ_2^P-complete by Umans [55].

SHORTEST IMPLICANT CORE: Given a DNF formula $\varphi = T_1 + T_2 + \cdots + T_m$, and an integer p, determine whether φ has an implicant C that consists of p literals from the last term T_m.

By a reduction from SHORTEST IMPLICANT CORE, one can obtain the following result.

Theorem 7 *$\bar{d}$-SSRR is Σ_2^P-complete.*

Proof The problem $\bar{d}$-SSRR can be solved by a nondeterministic machine that guesses a $(s+k)\times n$ submatrix H of M which contains the first s rows of M and then determines whether H is $\bar{d}$-separable. Note that the problem of determining whether a given matrix H is $\bar{d}$-separable is in $coNP$. Thus, $\bar{d}$-SSRR $\in \Sigma_2^P$.

Next, $\bar{d}$-SSRR is proved to be Σ_2^P-complete by constructing a polynomial-time reduction from SHORTEST IMPLICANT CORE to it. To define the reduction, let (φ, p) be an instance of the problem SHORTEST IMPLICANT CORE, that is, let

$$\varphi = T_1 + T_2 + \cdots + T_m$$

be a DNF formula over n variables $x_1, x_2, \ldots, x_n$, and let p be an integer > 0. Note that each term T_j, $1 \le j \le m$, of φ is a conjunction of some literals. Also, T_j is used to denote the set of these literals. Assume that the last term T_m of φ has q literals $\ell_1, \ell_2, \ldots, \ell_q$. Define a $(3n+m+q)\times(2n+1)$ Boolean matrix M as follows:

1. Let the $2n+1$ columns of M be $X = \{x_1, \bar{x}_1, x_2, \bar{x}_2, \ldots, x_n, \bar{x}_n, z\}$ and the $3n+m+q$ rows of M be $T = \{x_i, \bar{x}_i, u_i \mid 1 \le i \le n\} \cup \{t_j \mid 1 \le j \le m\} \cup \{c_j \mid 1 \le j \le q\}$.
2. For $i = 1, 2, \ldots, n$, $M_{x_i} = \{x_i\}$, $M_{\bar{x}_i} = \{\bar{x}_i\}$, and $M_{u_i} = \{x_i, \bar{x}_i, z\}$.
3. For $j = 1, 2, \ldots, m$, $M_{t_j} = \{x_i \mid \bar{x}_i \in T_j\} \cup \{\bar{x}_i \mid x_i \in T_j\} \cup \{z\}$. (Note that $M_{t_j} \cap T_j = \emptyset$).
4. The bottom q rows of M are $M_{c_j} = \{\ell_j, z\}$, for $j = 1, 2, \ldots, q$.

Let $d = n+1$, $s = 3n+m$, and $k = p$, and consider the instance (M, d, s, k) for the problem $\bar{d}$-SSRR.

First assume that φ has an implicant C of size p that is a subset of T_m. Let H be the submatrix of M that consists of the first $s = 3n+m$ rows plus the $k = p$ rows M_{c_j} for which $\ell_j \in C$. The following claim is made: H is $\bar{d}$-separable. That is, for any subsets S_1 and S_2 of $\{x_1, \bar{x}_2, \ldots, x_n, \bar{x}_n, z\}$ of size $\le d$, there exists a row in H that separates them.

Case 1. $S_1 - \{z\} \ne S_2 - \{z\}$. Then, there exists $v \in X - \{z\}$ such that $v \in S_1 \Delta S_2$. Then, $M_v(S_1) \ne M_v(S_2)$, and so H separates S_1 from S_2.

Case 2. $S_1 - \{z\} = S_2 - \{z\}$. Then, it must be true that $S_1 \Delta S_2 = \{z\}$. Without loss of generality, assume $S_2 = S_1 \cup \{z\}$. Note that $|S_2| \le n+1$ implies $|S_1| \le n$.

Subcase 2.1. There exists an integer i such that $|S_1 \cap \{x_i, \bar{x}_i\}| \ne 1$. First, if $|S_1 \cap \{x_i, \bar{x}_i\}| = 0$ for some i, then $M_{u_i}(S_1) = 0$ and $M_{u_i}(S_2) = 1$ (because $z \in S_2$). Next, if $|S_1 \cap \{x_i, \bar{x}_i\}| = 2$ for some i, then it must have $|S_1 \cap \{x_k, \bar{x}_k\}| = 0$ for some k, because $|S_1| \le n$. Then, again $M_{u_k}(S_1) = 0 \ne 1 = M_{u_k}(S_2)$. It follows that H separates S_1 from S_2.

Subcase 2.2. $|S_1 \cap \{x_i, \bar{x}_i\}| = 1$ for all $i \in \{1, 2, \ldots, n\}$. Define a Boolean assignment $\tau : \{x_1, x_2, \ldots, x_n\} \to \{\text{TRUE}, \text{FALSE}\}$ by $\tau(x_i) = \text{TRUE}$ if and only if $x_i \in S_1$. This is further divided into two subcases:

Subcase 2.2.1. τ satisfies the conjunction C. Since C is an implicant of $\varphi = T_1 + T_2 + \cdots + T_m$, τ must satisfy some T_j, $1 \leq j \leq m$. Thus, $T_j \subseteq S_1$: For any $x_i \in T_j$, $\tau(x_i) = \text{TRUE}$, and so $x_i \in S_1$, and for any $\bar{x}_i \in T_j$, $\tau(x_i) = \text{FALSE}$, and so $\bar{x}_i \in S_1$. It follows that $M_{t_j}(S_1) = 0$ since $M_{t_j} \cap T_j = \emptyset$. On the other hand, $M_{t_j}(S_2) = 1$ since $z \in M_{t_j} \cap S_2$. So, M_{t_j}, and hence H, separates S_1 from S_2.

Subcase 2.2.2. τ does not satisfy C. Then, for some literal $\ell_j \in C$, $\tau(\ell_j) = 0$. Thus, $\ell_j \notin S_1$, and $M_{c_j}(S_1) = 0$. On the other hand, $M_{c_j}(S_2) = 1$ since $z \in M_{c_j}$. Thus, M_{c_j}, which is a row in H, separates S_1 from S_2.

Conversely, assume that H is a $(3n + m + k) \times (2n + 1)$ submatrix of M that contains the first $3n + m$ rows of M and is $\bar{d}$-separable. Let C be the conjunction of literals ℓ_j for which M_{c_j} is a row in H. Then, obviously, $|C| = k$. Now the following claim is made: C is an implicant of φ.

Let $\tau : \{x_1, x_2, \ldots, x_n\} \to \{\text{TRUE}, \text{FALSE}\}$ be a Boolean assignment that satisfies C. It will be showed that τ satisfies φ. Let

$$S_1 = \{x_i \mid \tau(x_i) = \text{TRUE}\} \cup \{\bar{x}_i \mid \tau(x_i) = \text{FALSE}\},$$

and $S_2 = S_1 \cup \{z\}$. Then, S_1 and S_2 can be separated by some row in H. Since $S_2 = S_1 \cup \{z\}$, they are not separable by a row M_{x_i} or $M_{\bar{x}_i}$, for any $i = 1, 2, \ldots, n$. In addition, since $|S_1 \cap \{x_i, \bar{x}_i\}| = 1$ for all $i = 1, 2, \ldots, n$, they cannot be separated by row M_{u_i}, for any $i = 1, 2, \ldots, n$. Furthermore, note that for any literal $\ell_j \in C$, $\tau(\ell_j) = 1$ and so $\ell_j \in S_1$ and $M_{c_j}(S_1) = M_{c_j}(S_2) = 1$. Thus, S_1 and S_2 cannot be separated by any row M_{c_j} of H.

Therefore, S_1 and S_2 must be separable by a row M_{t_j}, for some $j = 1, 2, \ldots, m$. That is, $M_{t_j}(S_1) = 0 \neq 1 = M_{t_j}(S_2)$. Since M_{t_j} contains the complements of the literals in T_j, $T_j \subseteq S_1$. It follows that τ satisfies the term T_j, and hence φ. □

3.4 Conclusion

In this section, the computational complexities of problems related to nonunique probe selection are presented. The problem of verifying the minimality of a $\bar{d}$-separable matrix is showed to be DP-complete, and hence is intractable, unless $DP = P$. For the problem of finding a minimum $\bar{d}$-separable submatrix, it is conjectured to be Σ_2^P-complete and, hence, is even more difficult than the minimal $\bar{d}$-separability problem. To support this conjecture, the problem $\bar{d}$-SSRR, which is a little more general than the minimum $\bar{d}$-separable submatrix problem, is shown to be Σ_2^P-complete. The complexity of the original problem MIN-$\bar{d}$-SS remains open.

4 Parameterized Complexity Results on NGT

Given an $m \times n$ binary matrix and a positive integer d, to decide whether the matrix is d-separable ($\bar{d}$-separable, or d-disjunct) is a basic problem in nonadaptive group testing (NGT). They are known to be coNP-complete in classical complexity theory [18]. Thus, one should not expect any polynomial time algorithm to solve any of them. However, since in most applications $d \ll n$, an interesting question is whether there are efficient algorithms solving the above decision problems for small values of d.

In [12] by studying the parameterized complexity of the above three problems with d as the parameter, the authors gave a negative answer to the above question. More formally, they studied the parameterized decision problems p-DISJUNCTNESS-TEST, p-SEPARABILITY-TEST, and p-SEPARABILITY*-TEST defined as follows (where $\mathcal{N}$ denotes the set of positive integers).

p-DISJUNCTNESS-TEST
Instance: A binary matrix $\mathcal{M}$ and $d \in \mathcal{N}$.
Parameter: d.
Problem: Decide whether $\mathcal{M}$ is d-disjunct.
p-SEPARABILITY-TEST
Instance: A binary matrix $\mathcal{M}$ and $d \in \mathcal{N}$.
Parameter: d.
Problem: Decide whether $\mathcal{M}$ is d-separable.
p-SEPARABILITY*-TEST
Instance: A binary matrix $\mathcal{M}$ and $d \in \mathcal{N}$.
Parameter: d.
Problem: Decide whether $\mathcal{M}$ is $\bar{d}$-separable.

The main results obtained in [12] will be presented in this section; they are summarized in the following theorem.

Theorem 8 p-DISJUNCTNESS-TEST, p-SEPARABILITY*-TEST, *and* p-SEPARABILITY-TEST *are all co-W[2]-complete.*

W[2] is the parameterized complexity class at the second level of the W-hierarchy, and co-W[2] is the class of all parameterized problems whose complements are in W[2]. They will be formally introduced in the sequel. Theorem 8 indicates that, given an $m \times n$ binary matrix and a positive integer d, a deterministic algorithm with running time $f(d) \times (mn)^{O(1)}$ (where f is an arbitrary computable function) to decide whether the matrix is d-separable ($\bar{d}$-separable, or d-disjunct) does not exist unless the class W[2] collapses to FPT (the class of all fixed-parameter tractable problems), which is commonly conjectured to be false.

4.1 Preliminaries

Before proving Theorem 8, the notions of fixed-parameter tractability, relational structures, first-order logic, and the W-hierarchy of parameterized complexity classes are formally introduced.

4.1.1 Fixed-Parameter Tractability

The theory of *fixed-parameter tractability* [16, 27] has received considerable attention in recent years, for both theoretical research and practical computation. The notations and conventions in [27] are adopted here. Let Σ denote a fixed finite alphabet. A *parameterization* of Σ^* is a polynomial time computable mapping $\kappa : \Sigma^* \to \mathcal{N}$. A *parameterized problem* (over Σ) is a pair (Q, κ) consisting of a set $Q \subseteq \Sigma^*$ and a parameterization κ of Σ^*.

An algorithm A with input alphabet Σ is an *fpt-algorithm with respect to* κ, if for every $x \in \Sigma^*$ the running time of A on input x is at most $f(\kappa(x))|x|^{O(1)}$, for some computable function f. A parameterized problem (Q, κ) is *fixed-parameter tractable* if there is an fpt-algorithm with respect to κ that decides Q. The key point of the definition of fpt-algorithm is that the superpolynomial growth of running time is confined to the parameter $\kappa(x)$, which is usually known to be comparatively small. The class of all fixed-parameter tractable problems is denoted by FPT.

Many NP-hard problems such as the VERTEX COVER problem [8] and the ML TYPE-CHECKING problem [37] have been shown to be fixed-parameter tractable. On the other hand, there is strong theoretical evidence that certain well-known parameterized problems, for instance the INDEPENDENT SET problem and the DOMINATING SET problem, are not fixed-parameter tractable [16]. This evidence is provided, similar to the theory of NP-completeness, via a completeness theory based on the following notion of reductions: Let (Q, κ) and (Q', κ') be parameterized problems over alphabets Σ and Σ', respectively. An *fpt-reduction* from (Q, κ) to (Q', κ') is a mapping $R : \Sigma^* \to (\Sigma')^*$ such that:

1. For all $x \in \Sigma^*$, $x \in Q$ if and only if $R(x) \in Q'$.
2. R is computable by an fpt-algorithm (with respective to κ). That is, there is a computable function f such that $R(x)$ is computable in time $f(\kappa(x))|x|^{O(1)}$.
3. There is a computable function $g : \mathcal{N} \to \mathcal{N}$ such that $\kappa'(R(x)) \leq g(\kappa(x))$, for all $x \in \Sigma^*$.

In the above definition, the last requirement is to ensure that class FPT is closed under fpt-reductions, that is, if a parameterized problem (Q, κ) is reducible to another parameterized problem (Q', κ') and $(Q', \kappa') \in$ FPT, then $(Q, \kappa) \in$ FPT.

4.1.2 Relational Structures

In later discussions, the conventions in descriptive complexity theory are adopted, in which instances of decision problems are viewed as structures of some vocabulary instead of languages over some finite alphabet.

A *(relational) vocabulary* τ is a set of relation symbols. Each relation symbol $R \in \tau$ has an *arity* $\text{arity}(R) \geq 1$. A *structure* $\mathcal{A}$ of vocabulary τ consists of a set A called the *universe* and an interpretation $R^{\mathcal{A}} \subseteq A^{arity(R)}$ of each relation symbol $R \in \tau$. For a tuple $\bar{a} \in A^{arity(R)}$, write $R^{\mathcal{A}}\bar{a}$ (or $\bar{a} \in R^{\mathcal{A}}$) to denote that $\bar{a}$ belongs to the relation $R^{\mathcal{A}}$. Here only nonempty finite vocabularies and structures with a finite universe are considered.

Recall that a hypergraph is a pair $H = (V, E)$ consisting of a set V of vertices and a set E of hyperedges. Each hyperedge is a subset of V. Graphs are hypergraphs with hyperedges of cardinality two. The following example illustrates how to represent a hypergraph using a relational structure.

Example 1 Let τ_{HG} be the vocabulary consisting of the unary relation symbols $VERT$ and $EDGE$ and the binary relation symbol I. A hypergraph $H = (V, E)$ can be represented by a relational structure $\mathcal{H}$ of vocabulary τ_{HG} as follows:

- The universe of $\mathcal{H}$ is $V \cup E$.
- $VERT^{\mathcal{H}} := V$ and $EDGE^{\mathcal{H}} := E$.
- $I^{\mathcal{H}} := \{(v, e) : v \in V, e \in E, \text{and } v \in e\}$ is the *incidence relation*.

4.1.3 First-Order Logic

First the syntax of first-order logic is briefly recalled. Let τ be a vocabulary. *Atomic* first-order formulas of vocabulary τ are of the form $x = y$ or $Rx_1 \dots x_\ell$, where $R \in \tau$ is ℓ-ary (i.e., has arity ℓ) and $x, y, x_1, \dots, x_\ell$ are variables. First-order formulas of vocabulary τ are built from atomic formulas using Boolean connectives $\wedge$ (and), $\vee$ (or), $\neg$ (negation), together with the existential and universal quantifiers $\exists$ and $\forall$. The connectives $\rightarrow$ (implication) and $\leftrightarrow$ (equivalence) are not part of the language defining first-order formulas, but they are used as abbreviations: $\varphi \rightarrow \psi$ stands for $\neg\varphi \vee \psi$, and $\varphi \leftrightarrow \psi$ stands for $(\varphi \rightarrow \psi) \wedge (\psi \rightarrow \varphi)$.

A variable x is called a free variable of φ if x occurs in φ but is not in the scope of a quantifier binding x. Write $\varphi(x_1, \dots, x_\ell)$ to indicate that all free variables of φ belong to set $\{x_1, \dots, x_\ell\}$. A formula without free variables is called a *sentence*. Let both Σ_0 and Π_0 denote the class of quantifier-free first-order formulas. For $t \geq 0$, let Σ_{t+1} be the class of all formulas $(\exists x_1 \dots \exists x_\ell)\varphi$, where $\varphi \in \Pi_t$, and let Π_{t+1} be the class of all formulas $(\forall x_1 \dots \forall x_\ell)\varphi$, where $\varphi \in \Sigma_t$.

For formulas of second-order logic, in addition to the individual variables, they may also contain *relation variables*; each of the relation variables has a prescribed arity. Here lowercase letters (e.g., x, y, z) are used to denote individual variables, and uppercase letters (e.g., X, Y, Z) are used to denote relation variables. As in [27], for convenience *free* relation variables are allowed to be in first-order formulas, since the crucial difference between first-order and second-order logic is not that second-order formulas can have relation variables, but that second-order formulas can quantify over relations. Therefore, here the syntax of first-order logic is enhanced by

including new atomic formulas of the form $Xx_1 \ldots x_\ell$, where X is an ℓ-ary relation variable. The meaning of formula $Xx_1 \ldots x_\ell$ is: The tuple of elements interpreting $(x_1, \ldots, x_\ell)$ is contained in the relation interpreting the relation variable X. The classes such as Σ_t and Π_t are also extended to include formulas with free relation variables. It is worth emphasizing again that in first-order logic quantification over relation variables is now allowed.

4.1.4 W-Hierarchy

First a brief introduction to the W-hierarchy of parameterized complexity classes is given. Roughly speaking, the W-hierarchy classifies problems according to the syntactic form of their definitions, and the definitions are formalized using languages of mathematical logic. The W-hierarchy can be defined in several different ways; here the following definition based on the *weighted Fagin-defined problems* is adopted.

Let $\varphi(X)$ be a first-order formula with a free relation variable X of arity s. Define p-WD$_\varphi$ to be the following parameterized decision problem.
p-WD$_\varphi$:
Instance: A *structure* $\mathcal{A}$ and $k \in \mathcal{N}$.
Parameter: k.
Problem: Decide whether there is a relation $S \subseteq A^s$ of cardinality k such that $\mathcal{A} \models \varphi(S)$.

Here, $\mathcal{A} \models \varphi(S)$ stands for that structure $\mathcal{A}$ satisfies sentence $\varphi(S)$ (or, $\mathcal{A}$ is a model of $\varphi(S)$), and S is called a *solution* for φ in structure $\mathcal{A}$. The readers are referred to, for example, Sect. 4.2 of [27], for more detailed introduction to the semantics of first-order formulas.

For a class Φ of first-order formulas, let p-WD-Φ be the class of all parameterized problems p-WD$_\varphi$ with $\varphi \in \Phi$. For $t \geq 1$, define W[t]:= $[p\text{-WD-}\Pi_t]^{fpt}$, which is the class of all parameterized problems that are fpt-reducible to some problems in p-WD-Π_t. The classes W[t], for $t \geq 1$, form the W-hierarchy. Thus, the levels of W-hierarchy essentially correspond to the number of alternations between universal and existential quantifiers in the definitions of their complete problems. Problems hard for W[1] or higher class are assumed not to be fixed-parameter tractable. For instance, the INDEPENDENT SET problem is W[1]-complete, and the DOMINATING SET problem is W[2]-complete.

For a parameterized problem (Q, κ) over the alphabet Σ, let $(Q, \kappa)^c$ denote its complement, that is, the parameterized problem $(\Sigma^* \setminus Q, \kappa)$. Let C be a parameterized complexity class. Then co-C is defined to be the class of all parameterized problems (Q, κ) such that $(Q, \kappa)^c \in C$. Clearly, FPT = co-FPT. From the definition of fpt-reductions, it is easy to see that if class C is closed under fpt-reductions, so is co-C. In particular, each class W[t], $t \geq 1$, gives rise to a new parameterized complexity class co-W[t]. Also, it is easy to prove that if (Q, κ) is complete in parameterized complexity class C under fpt-reductions, then $(Q, \kappa)^c$ is complete in class co-C under fpt-reductions.

4.2 Proof of Theorem 8

Now the proof of Theorem 8 is ready to presented. For a binary matrix M, let R_M be the set consisting of all rows in M, and let C_M be the set consisting of all columns in M.

Relational Structure for a Binary Matrix. Let τ_{BM} be the vocabulary consisting of the unary relation symbols *ROW* and *COLUMN* and the binary relation symbol *I*. Then, the binary matrix M can be represented by a structure $\mathcal{M}$ of vocabulary τ_{BM}, where the universe of $\mathcal{M}$ is $R_M \cup C_M$, and the interpretations for the relation symbols in τ_{BM} are as follows:

- $ROW^{\mathcal{M}} := R_M$.
- $COLUMN^{\mathcal{M}} := C_M$.
- $I^{\mathcal{M}} := \{(r, c) : r \in R_M, c \in C_M, \text{ and } M(r, c) = 1\}$, which is the incidence relation.

The proof of Theorem 8 is partitioned into the following six lemmas:

Lemma 5 p-DISJUNCTNESS-TEST $\in$ co-W[2].

Proof Consider the following complement problem of p-DISJUNCTNESS-TEST.

p-NONDISJUNCTNESS-TEST
Instance: A binary matrix $\mathcal{M}$ and $d \in \mathcal{N}$.
Parameter: d.
Problem: Decide whether $\mathcal{M}$ is NOT d-disjunct.

A Π_2 formula *nondisj(X)* with a free relation variable X of arity 2 will be defined, and p-NONDISJUNCTNESS-TEST will be showed to be equal to p-WD$_{nondisj(X)}$ (see Sect. 4.1.4 for the definition of problem p-WD$_\varphi$). This implies that p-NONDISJUNCTNESS-TEST is in p-WD-Π_2, therefore is in W[2].

A binary matrix is not d-disjunct if and only if there is a set D of d columns and another column $c \notin D$ such that $U(D)$ covers c. The idea here is assuming that the solution S to X is of the form $\{(c_i, c) : c_i \in D\}$. Therefore, X should be a binary relation variable, and the solution S to X should have cardinality d.

Define

$$\chi_1 := \forall c_1 \forall c_2 (Xc_1c_2 \rightarrow (COLUMNc_1 \wedge COLUMNc_2 \wedge (c_1 \neq c_2))),$$

and define

$$\chi_2 := \forall c_3 \forall c_4 \forall c_5 \forall c_6 ((Xc_3c_4 \wedge Xc_5c_6) \rightarrow (c_4 = c_6)).$$

Here χ_1 and χ_2 are to guarantee that the solution S to X of cardinality d has the form $\{(c_1, c), \ldots, (c_d, c)\}$, where $c \in C_M$, $c_i \in C_M$, and $c_i \neq c$, for $1 \leq i \leq d$. Define

$$nondisj'(X) := \forall r \exists c_7 \exists c_8 (ROWr \rightarrow (Xc_7c_8 \wedge (Irc_8 \rightarrow Irc_7))).$$

Here $nondisj'(X)$ is to guarantee that the solution S to X satisfies that the union of columns in $\{c_i : (c_i, c) \in S\}$ covers c (so that $\mathcal{M}$ is not d-disjunct). Finally, define

$$nondisj(X) := \chi_1 \wedge \chi_2 \wedge nondisj'(X),$$

which is equivalent to a Π_2 formula.

From the above definition, clearly if there exists a relation $S \subseteq C_M^2$ of cardinality d such that $\mathcal{M} \models nondisj(S)$, then $\mathcal{M}$ is not d-disjunct. On the other hand, if $\mathcal{M}$ is not d-disjunct, then there exist a subset D of d columns $c_1, \ldots, c_d$ and another column $c \notin D$ such that c is covered by $U(D)$. It is not hard to verify that the relation $S = \{(c_1, c), \ldots, (c_d, c)\}$ satisfies $\mathcal{M} \models nondisj(S)$. Therefore, $\mathcal{M}$ is not d-disjunct if and only if there exists a relation S of cardinality d such that $\mathcal{M} \models nondisj(S)$. That is, p-NONDISJUNCTNESS-TEST is p-WD$_{nondisj(X)}$. Thus, p-NONDISJUNCTNESS-TEST $\in$ W[2], and so p-DISJUNCTNESS-TEST $\in$ co-W[2]. □

Lemma 6 p-SEPARABILITY-TEST $\in$ co-W[2].

Proof Consider the following complement problem of p-SEPARABILITY-TEST.

p-NONSEPARABILITY-TEST
Instance: A binary matrix $\mathcal{M}$ and $d \in \mathcal{N}$.
Parameter: d.
Problem: Decide whether $\mathcal{M}$ is NOT d-separable.

A formula $nonsep(Y)$ with a free relation variable Y of arity 2 will be defined, and p-NONSEPARABILITY-TEST will be shown to be equal to p-WD$_{nonsep(Y)}$.

A binary matrix is not d-separable if and only if there exist two distinct subsets D_1 and D_2, each contains d columns such that $U(D_1) = U(D_2)$. Assume that $D_1 = \{c_{11}, c_{12}, \ldots, c_{1d}\}$ and $D_2 = \{c_{21}, c_{22}, \ldots, c_{2d}\}$. The idea is to assume that the solution S to Y is of the form $\{(c_{11}, c_{21}), (c_{12}, c_{22}), \ldots, (c_{1d}, c_{2d})\}$, and so Y should be a binary relation variable, and the solution S to Y should have cardinality d. Define the formula $nonsep(Y)$ such that it satisfies the following: There exists a relation $S \subseteq C_M^2$ of cardinality d such that $\mathcal{M} \models nonsep(S)$ if and only if $\mathcal{M}$ is not d-separable.

Define

$$\chi_3 := \forall c_1 \forall c_2 (Yc_1c_2 \rightarrow (COLUMNc_1 \wedge COLUMNc_2)),$$

$$\chi_4 := \forall c_3 \forall c_4 \forall c_5 \forall c_6 ((Yc_3c_4 \wedge Yc_5c_6) \rightarrow ((c_3 = c_5) \leftrightarrow (c_4 = c_6))),$$

and

$$\chi_5 := \exists c_7 \exists c_8 \forall c_9 ((Yc_7c_8 \wedge \neg Yc_9c_7)).$$

Here χ_3 is to guarantee that the relation variable $Y \subseteq C_M^2$; χ_4 is to build a bijection between the first component of elements in S and the second component of elements in S, which guarantees that the two subsets $\{c_{1i} : \exists c\ s.t.\ (c_{1i}, c) \in S\}$ and $\{c_{2j} : \exists c\ s.t.\ (c, c_{2j}) \in S\}$ (which intend to be D_1 and D_2, respectively) have the same cardinality; χ_5 is to guarantee that the two subsets $\{c_{1i} : \exists c\ s.t.\ (c_{1i}, c) \in S\}$ and $\{c_{2j} : \exists c\ s.t.\ (c, c_{2j}) \in S\}$ are distinct from each other. Define

$$nonsep'(Y) := \forall r(ROWr \rightarrow ((\exists c_{10}\exists c_{11} Y c_{10} c_{11} \wedge Irc_{10})$$

$$\leftrightarrow (\exists c_{12}\exists c_{13} Y c_{12} c_{13} \wedge Irc_{13}))),$$

which is to guarantee that the solution S to Y satisfies that the union of columns in $\{c_{1i} : \exists c\ s.t.\ (c_{1i}, c) \in S\}$ is equal to the union of columns in $\{c_{2j} : \exists c\ s.t.\ (c, c_{2j}) \in S\}$. From basic logic computation, it is not hard to verify that $nonsep'(Y)$ is a Π_2 formula with free relation variable Y. Finally, define

$$nonsep(Y) := \chi_3 \wedge \chi_4 \wedge \chi_5 \wedge nonsep'(Y).$$

From the above definition of $nonsep(X)$, if a relation $S \subseteq C_M^2$ of cardinality d satisfies that $\mathcal{M} \models nonsep(S)$, then the two subsets $\{c_{1i} : \exists c\ s.t.\ (c_{1i}, c) \in S\}$ and $\{c_{2j} : \exists c\ s.t.\ (c, c_{2j}) \in S\}$ both contain d columns of $\mathcal{M}$ and are distinct from each other; moreover, their unions are identical. This implies that $\mathcal{M}$ is not d-separable. On the other hand, if $\mathcal{M}$ is not d-separable, then there exist two distinct subsets D_1 and D_2, each contains d columns such that $U(D_1) = U(D_2)$. Assume that $D_1 = \{c_{11}, \ldots, c_{1d}\}$ and $D_2 = \{c_{21}, \ldots, c_{2d}\}$. It is not hard to verify that the relation

$$S = \{(c_{11}, c_{21}), (c_{12}, c_{22}), \ldots, (c_{1d}, c_{2d})\}$$

satisfies $\mathcal{M} \models nonsep(S)$.

From above, there exists a relation $S \subseteq C_M^2$ of cardinality d such that $\mathcal{M} \models nonsep(S)$ if and only if $\mathcal{M}$ is not d-separable; therefore, p-NONSEPARABILITY-TEST is p-WD$_{nonsep(Y)}$. χ_3 and χ_4 are Π_1 formulas; χ_5 is a Σ_2 formula; $nonsep'(Y)$ is a Π_2 formula; therefore,

$$nonsep(Y) = \chi_3 \wedge \chi_4 \wedge \chi_5 \wedge nonsep'(Y)$$

is equivalent to a Σ_3 formula, which implies that p-NONSEPARABILITY-TEST is in p-WD-Σ_3. Here the following fact is applied: p-WD-$\Sigma_3 \subseteq p$-WD-Π_2 (more generally, p-WD-$\Sigma_{t+1} \subseteq p$-WD-Π_t, for $t \geq 1$. The main idea to prove this conclusion is not complicated; the reader is referred to, e.g., Proposition 5.4 in [27] for the proof). Thus, p-NONSEPARABILITY-TEST $\in$ W[2], and so p-SEPARABILITY-TEST $\in$ co-W[2]. □

Lemma 7 p-SEPARABILITY*-TEST $\in$ co-W[2].

Proof Since W[2] is closed under fpt-reductions, so is co-W[2]. It will be showed next that p-SEPARABILITY*-TEST is fpt-reducible to p-SEPARABILITY-TEST. Since the latter is in co-W[2] (Lemma 6), this implies that p-SEPARABILITY*-TEST $\in$ co-W[2]. The reduction can be obtained immediately from the following fact (Lemma 2.1.6 in [17]): A binary matrix M' containing a zero column is d-separable if and only if the matrix M obtained by removing this zero column from M' is $\bar{d}$-separable.

Let (M, d) be an instance of p-SEPARABILITY*-TEST, where M is a binary matrix and the parameter d is a positive integer. Map (M, d) to (M', d), where M' is obtained by adding a zero column to M. From the above lemma, $(M, d) \in p$-SEPARABILITY*-TEST if and only if $(M', d) \in p$-SEPARABILITY-TEST. It is easy to see that this is an fpt-reduction from p-SEPARABILITY*-TEST to p-SEPARABILITY-TEST.

Lemma 8 p-DISJUNCTNESS-TEST *is* co-W[2]*-complete.*

Proof A *hitting set* in a hypergraph $\mathcal{H} = (V, E)$ is a set T of vertices that intersects each hyperedge, that is, $T \cap e \neq \emptyset$ for all $e \in E$. The classical HITTING-SET problem is to find a hitting set of a given cardinality k in a given hypergraph $\mathcal{H}$, which is known to be NP-complete. The following parameterized hitting set problem is W[2]-complete (see, e.g., Theorem 7.14 in [27]).

p-HITTING-SET
Instance: A hypergraph $\mathcal{H}$ and $k \in \mathcal{N}$.
Parameter: k.
Problem: Decide whether $\mathcal{H}$ has a hitting set of k vertices.

An ftp-reduction from p-HITTING-SET to p-NONDISJUNCTNESS-TEST will be given, based on an idea similar to that in [18]. Let $(\mathcal{H}, k)$ with $\mathcal{H} = (V, E)$ be an instance of p-HITTING-SET, where $V = \{1, \ldots, n\}$, $E = \{e_1, \ldots, e_m\}$, and each e_i, $1 \leq i \leq m$, is a subset of V. Define $d = k$, and define an $(n + m) \times (n + 1)$ binary matrix M with rows R_i as follows (here each row is represented as a subset of the set of all columns $\{1, 2, \ldots, n + 1\}$, in the most natural way):

$$R_i = \{i\}, \qquad i = 1, \ldots, n;$$
$$R_{n+j} = e_j \cup \{n + 1\}, \qquad j = 1, \ldots, m.$$

First, assume that $\mathcal{H}$ has a hitting set $T \subseteq V$ of size k. Consider the subset $S_1 = T$ of columns of M. Since T is a hitting set of $\mathcal{H}$, $U(S_1)$ covers column $n + 1$. Notice that $|S_1| = d$ and column $n + 1$ is not in S_1, M is not d-disjunct.

Conversely, assume that M is not d-disjunct. Then, there exist a subset S_1 of d columns in $\{1, 2, \ldots, n + 1\}$ and another column $c \notin S_1$ such that $U(S_1)$ covers c. From the way the first n rows of matrix M are defined, c can only be column $n + 1$. Thus, column $n + 1$ is not in S_1. Set $T = S_1$, then $|T| = k$, and T is a subset of V. Since $U(S_1)$ covers column $n + 1$, it is easy to see that T is a hitting set of $\mathcal{H}$.

It is not hard to verify that the above is an fpt-reduction. Therefore, p-NONDISJUNCTNESS-TEST is W[2]-complete, and so p-DISJUNCTNESS-TEST is co-W[2]-complete. □

Lemma 9 p-SEPARABILITY*-TEST *is* co-W[2]-*complete.*

Proof To prove the lemma, an ftp-reduction from p-HITTING-SET to p-NONSEPARABILITY*-TEST will be given. For an instance $(\mathcal{H}, k)$ of p-HITTING-SET, define matrix M in the same way as in the proof of Lemma 8, and define $d = k + 1$. Next the correctness of this reduction will be shown.

First, assume that $\mathcal{H}$ has a hitting set $T \subseteq V$ of size k. Consider the following two subsets of columns in M: $S_1 = T$ and $S_2 = T \cup \{n+1\}$. Then, for $1 \leq i \leq n$, it is obvious that $U(S_1)_i = U(S_2)_i$; for $n < i \leq n + m$, since T is a hitting set of $\mathcal{H}$, $U(S_1)_i = 1 = U(S_2)_i$. Notice that $|S_1|, |S_2| \leq d$ and $S_1 \neq S_2$, M is not $\bar{d}$-separable.

Conversely, assume that M is not $\bar{d}$-separable. Then, there exist two subsets S_1 and S_2 of columns in $\{1, 2, \ldots, n+1\}$ such that $|S_1|, |S_2| \leq d$, $S_1 \neq S_2$, and $U(S_1) = U(S_2)$. Since $U(S_1)_i = U(S_2)_i$ for $1 \leq i \leq n$, it follows that

$$S_1 \cap \{1, \ldots, n\} = S_2 \cap \{1, \ldots, n\}.$$

To have $S_1 \neq S_2$, column $n+1$ must belong to exactly one of S_1 and S_2. Without loss of generality, assume that $n + 1 \notin S_1$ and $n + 1 \in S_2$. Set $T = S_1$, then

$$|T| = |S_1| = |S_2| - 1 \leq d - 1 = k,$$

and T is a subset of V. From $U(S_1)_i = U(S_2)_i = 1$ for $n < i \leq n + m$, T is a hitting set of $\mathcal{H}$.

Therefore, p-NONSEPARABILITY*-TEST is W[2]-complete, and so p-SEPARABILITY*-TEST is co-W[2]-complete.

Lemma 10 p-SEPARABILITY-TEST *is* co-W[2]-*complete.*

Proof Since as proved before in Lemma 7 that p-SEPARABILITY*-TEST is fpt-reducible to p-SEPARABILITY-TEST, and in Lemma 9 that p-SEPARABILITY*-TEST is co-W[2]-complete, p-SEPARABILITY-TEST is co-W[2]-hard. Together with Lemma 6 that p-SEPARABILITY-TEST $\in$ co-W[2], it is obtained that p-SEPARABILITY-TEST is co-W[2]-complete.

4.3 Discussion

In this section, the parameterized complexity is established for the following three basic problems in pooling design: Given an $m \times n$ binary matrix and a positive integer d, to decide whether the matrix is d-separable ($\bar{d}$-separable, or

d-disjunct). It is showed that these problems are co-W[2]-complete; thus, do not admit algorithms with running time $f(d) \times (mn)^{O(1)}$ for any computable function f. The best known algorithms for the above general problems are all in a brute-force manner. It is interesting to investigate that whether these problems admit better algorithms. For instance, are there algorithms with running time $n^{o(d)}O(m)$ to solve these problems when d is small compared to n?

5 Upper Bounds on the Minimum Number of Rows of Disjunct Matrices

A 0-1 matrix is *d-disjunct* if no column is covered by the union of any d other columns, where the union means the bitwise Boolean sum of these d column vectors. In other words, a 0-1 matrix is called d-disjunct if for any column C and any d other columns, there exists at least one row such that the row has value 1 at column C and value 0 at all d other columns. The same structure is also called *cover-free family* [25, 29, 53] in combinatorics and *superimposed code* [22, 23, 34] in information theory. It is called a d-disjunct matrix in group testing [17, 32, 39]. A 0-1 matrix is *$(d; z)$-disjunct* [22,39] if for any column C and any d other columns, there exist at least z rows such that each of them has value 1 at column C and value 0 at all the other d columns. Thus, d-disjunct is $(d; 1)$-disjunct. Besides other applications, d-disjunct and $(d; z)$-disjunct matrices form the basis for error-free and error-tolerant nonadaptive group testing algorithms, respectively. These algorithms have applications in many practical areas such as DNA library screening [3, 6, 17, 43] and multi-access communications [57].

Let $t(d, n)$ denote the minimum number of rows required by a d-disjunct matrix with n columns. The bounds on $t(d, n)$ have been extensively studied in the fields of combinatorics, information theory, and group testing, under different terminologies. For lower bounds, it is known that $t(d, n) = \Omega(\frac{d^2 \log n}{\log d})$ [21, 29, 53]. In particular, D'yachkov and Rykov [21] proved that $t(d, n) \geq \frac{d^2}{2 \log d}(1 + o(1)) \log n$, which is the best lower bound so far. For upper bounds on $t(d, n)$, it is known that $t(d, n) = O(d^2 \log n)$ [2, 22, 32, 33, 45, 50]. In [22], Dyachkov, Rykov, and Rashad obtained the following asymptotic upper bound on $t(d, n)$ with a rather involved proof, which is currently the best.

Theorem 9 (Dyachkov, Rykov, and Rashad [22]) *For d constant and $n \to \infty$,*

$$t(d, n) \leq \frac{d}{A_d}[1 + o(1)] \log n,$$

where

$$A_d = \max_{0 \leq p \leq 1} \max_{0 \leq P \leq 1} \left\{ -(1 - P) \log(1 - p^d) + d \left[P \log \frac{p}{P} + (1 - P) \log \frac{1 - p}{1 - P} \right] \right\}.$$

Moreover, $A_d \to \frac{1}{d \log e}$ *as* $d \to \infty$.

For $(d; z)$-disjunct matrices, let $t(d, n; z)$ denote the minimum number of rows required by a $(d; z)$-disjunct matrix with n columns. For given d and z, D'yachkov, Rykov, and Rashad [22] studied the value of $\lim_{n\to\infty} \frac{\log n}{t}$, among others, and they proved that $t(d, n; z) \geq c[\frac{d^2 \log n}{\log d} + (z-1)d]$ where c is a constant.

In [13], by using *q-ary* $(d, 1)$*-disjunct* matrices [17, 20] and the probabilistic method (see, e.g., [1]), the authors gave a very short proof for the currently best upper bound on $t(d, n)$. In contrast to the previous result in [22] (Theorem 9), which is an asymptotic upper bound, the upper bound on $t(d, n)$ in [13] does not contain the asymptotic term $o(1)$. Also, the method in [13] is generalized to obtain a new upper bound on $t(d, n; z)$. These results will be presented in this section.

5.1 Upper Bounds on $t(d, n)$

Theorem 10 *For* $n > d \geq 1$,

$$t(d, n) \leq \frac{d+1}{B_d} \log n,$$

where $B_d = \max_{q>1} \frac{-\log\left[1-\left(1-\frac{1}{q}\right)^d\right]}{q}$. *Moreover,* $B_d \to \frac{1}{d \log e}$ *as* $d \to \infty$.

Before proving the above theorem, the concept of *q-ary* $(d, 1)$*-disjunct* matrix will be first introduced: A matrix is called q-ary $(d, 1)$-disjunct if it is q-ary, and for any column C and any set D of d other columns, there exists an element in C such that the element does not appear in any column of D in the same row.

As described in [17, 20], one can transform a q-ary $(d, 1)$-disjunct matrix M to a (binary) d-disjunct matrix M' as follows. Replace each row R_i of M by several rows indexed with entries of R_i. For each entry x of R_i, the row with index x is obtained from R_i by turning all x's into 1's and all others into 0's. The following fact is useful in later proof, which is the same as Theorem 1 only using different notations:

Fact 2 (Theorem 3.6.1 in [17]) *A* $t \times n$ *q-ary* $(d, 1)$*-disjunct matrix* M *yields a* $t' \times n$ d*-disjunct matrix* M' *with* $t' \leq tq$.

Now it is time to present the proof of Theorem 10.

Proof of Theorem 10: Given $n > d \geq 1$, first construct a random $t \times n$ q-ary $(q > 1)$ matrix M with each entry assigned randomly and uniformly from $\{1, 2, \ldots, q\}$, where q and t will be specified later. For each column C and a set D of d other columns, for each element c_i $(i = 1, 2, \ldots, t)$ of C, the probability that c_i appears in some column of D in the same row is $1 - (1 - \frac{1}{q})^d$. Thus, the probability that every element of C appears in some column of D in the same row is $[1 - (1 - \frac{1}{q})^d]^t$.

M is not $(d,1)$-disjunct if and only if there exist a column C and a set D of d other columns such that the above holds. Therefore, the probability that M is not $(d,1)$-disjunct is no more than

$$(n-d)\binom{n}{d}[1-(1-\frac{1}{q})^d]^t.$$

What follows next is to try to minimize tq, the number of rows of the d-disjunct matrix M' as in Fact 2, under the condition that q and t satisfy

$$n^{d+1}[1-(1-\frac{1}{q})^d]^t \leq 1. \tag{1}$$

Notice that Eq. (1) implies

$$(n-d)\binom{n}{d}[1-(1-\frac{1}{q})^d]^t < 1;$$

thus, the probability that M is $(d,1)$-disjunct is greater than zero. Therefore, by probabilistic argument, Eq. (1) implies the existence of a $t \times n$ q-ary $(d,1)$-disjunct matrix, and so a d-disjunct matrix with n columns and at most tq rows.
To satisfy Eq. (1), let

$$t = \frac{(d+1)\log n}{-\log[1-(1-\frac{1}{q})^d]}.$$

Define

$$B_d(q) = \frac{-\log[1-(1-\frac{1}{q})^d]}{q},$$

then

$$tq = \frac{(d+1)\log n}{B_d(q)}.$$

Let q_0 be the point that maximizes $B_d(q)$, and let $B_d = B_d(q_0)$ (one can estimate that $q_0 = \Theta(d)$ and $B_d = \Theta(\frac{1}{d})$, since the proof here can stand alone without this observation; they are put into Lemma 11) in later part. By assigning $q = q_0$, it follows that

$$t(d,n) \leq (tq)|_{q=q_0} = \frac{(d+1)\log n}{B_d(q_0)} = \frac{(d+1)\log n}{B_d}.$$

Next estimate B_d as $d \to \infty$. Since $(1-\frac{1}{q})^q < \frac{1}{e}$ for $q > 1$,

$$\left(1-\frac{1}{q}\right)^d < \left(\frac{1}{e}\right)^{\frac{d}{q}} = e^{-\frac{d}{q}},$$

and

$$-\log[1-(1-\frac{1}{q})^d] < -\log(1-e^{-\frac{d}{q}}).$$

It follows that

$$\begin{aligned} B_d(q) &= \frac{-\log[1-(1-\frac{1}{q})^d]}{q} \\ &< \frac{-\log(1-e^{-\frac{d}{q}})}{q} \\ &= \frac{1}{d\ln 2}[-\frac{d}{q}\ln(1-e^{-\frac{d}{q}})]. \end{aligned}$$

Let $x = e^{-\frac{d}{q}}$, then $-\frac{d}{q} = \ln x$, and

$$B_d(q) < \frac{1}{d\ln 2}\ln x \ln(1-x).$$

Since $\ln x \ln(1-x)$ achieves its maximum at $x = \frac{1}{2}$, it follows that $B_d(q) < \frac{\ln 2}{d}$ for $q > 1$. Thus, $B_d < \frac{\ln 2}{d}$ for $d \geq 1$. On the other hand, when q satisfies $(1-\frac{1}{q})^d = \frac{1}{2}$, as $d \to \infty$, it is easy to see that $\frac{q}{d} \to \frac{1}{\ln 2}$, and $B_d(q) = \frac{1}{q} \to \frac{\ln 2}{d}$. Therefore, as $d \to \infty$, $B_d \to \frac{\ln 2}{d} = \frac{1}{d\log e}$. □

Lemma 11 *Given $d \geq 1$, let $q_0 = q_0(d)$ be the point that maximizes $B_d(q) = \frac{-\log[1-(1-\frac{1}{q})^d]}{q}$ for $q > 1$. Then, as $d \to \infty$, $q_0(d) = \Theta(d)$, and $B_d = B_d(q_0) = \Theta(\frac{1}{d})$.*

Proof Notice that if q_1 satisfies $(1-\frac{1}{q_1})^d = \frac{1}{2}$, then $q_1 = \Theta(d)$, since $\frac{q_1}{d} \to \frac{1}{\ln 2}$ as $d \to \infty$. Moreover, $B_d(q_1) = \frac{1}{q_1} = \Theta(\frac{1}{d})$. The lemma is proved by contradiction. First assume that $q_0 = O(d)$ does not hold, that is, for any $c > 0$ and any $d_0 > 0$, there exists $d > d_0$ such that $q_0(d) > cd$. Then, since $\frac{q_0}{d} > c$, as $c \to \infty$,

$$\begin{aligned} B_d(q_0)d &= \frac{-\log[1-(1-\frac{1}{q_0})^d]}{q_0}d \\ &\sim \frac{-\log[1-(1-\frac{d}{q_0})]}{q_0}d \\ &= \frac{\log\frac{q_0}{d}}{\frac{q_0}{d}} \\ &= o(1), \end{aligned}$$

here $a \sim b$ means that $\lim_{c\to\infty} \frac{a}{b} = 1$. Thus,

$$B_d(q_0) = \frac{o(1)}{d}.$$

However, this is a contradiction since q_0 is the maximum point of $B_d(q)$ and $B_d(q_1) = \Theta(\frac{1}{d})$ with $(1 - \frac{1}{q_1})^d = \frac{1}{2}$.

On the other hand, assume that $q_0 = \Omega(d)$ does not hold, that is, for any $c > 0$ and any $d_0 > 0$, there exists $d > d_0$ such that $q_0(d) < cd$. Then,

$$\begin{aligned} B_d(q_0)d &= \frac{-\log[1-(1-\frac{1}{q_0})^d]}{q_0} d \\ &= \frac{-\ln\{1-[(1-\frac{1}{q_0})^{q_0}]^{\frac{d}{q_0}}\}}{q_0 \ln 2} d. \end{aligned}$$

Since $0 < (1 - \frac{1}{q_0})^{q_0} < \frac{1}{e}$ for $q_0 > 1$, as $c \to 0$, $\frac{d}{q_0} > \frac{1}{c} \to \infty$, and

$$[(1 - \frac{1}{q_0})^{q_0}]^{\frac{d}{q_0}} < e^{-\frac{d}{q_0}} \to 0.$$

Thus,

$$\begin{aligned} B_d(q_0)d &\sim \frac{[(1-\frac{1}{q_0})^{q_0}]^{\frac{d}{q_0}}}{q_0 \ln 2} d \\ &= \frac{1}{\ln 2} \frac{d}{q_0} [(1-\frac{1}{q_0})^{q_0}]^{\frac{d}{q_0}} \\ &< \frac{1}{\ln 2} \frac{d}{q_0} e^{-\frac{d}{q_0}} \\ &= o(1), \end{aligned}$$

which is also a contradiction (here $a \sim b$ means that $\lim_{c\to 0} \frac{a}{b} = 1$). Therefore, $q_0(d) = \Theta(d)$. Then, $(1 - \frac{1}{q_0})^d < 1$ is $\Theta(1)$, and thus

$$B_d(q_0) = \frac{\Theta(1)}{q_0} = \Theta\left(\frac{1}{d}\right). \qquad \square$$

5.2 New Upper Bounds on $t(d, n; z)$

The above method is now generalized to obtain new upper bounds for $(d; z)$-disjunct matrices, by establishing the following theorem:

Theorem 11 *For d, z constants, and $n \to \infty$,*

$$t(d,n;z) \leq \frac{d+1}{B_d}\log n + \frac{z}{B_d}\log\log n + O(1),$$

where $B_d = \max_{q>1} \frac{-\log[1-(1-\frac{1}{q})^d]}{q}$. *Moreover,* $B_d \to \frac{1}{d\log e}$ *as* $d \to \infty$.

A q-ary matrix is called $(d,1;z)$-*disjunct* if for any column C and any set D of d other columns, there exists at least z elements in C such that each of these elements does not appear in any column of D in the same row. Clearly, by using the same method mentioned above, one can transform a $t \times n$ q-ary $(d,1;z)$-disjunct matrix to a $(d;z)$-disjunct matrix with n columns and at most tq rows.

Proof of Theorem 14: For given n, d, and z, similarly construct a random $t \times n$ q-ary ($q > 1$) matrix M with each entry assigned randomly and uniformly from $\{1,2,\ldots,q\}$; q and t will be specified later. For each column C and a set D of d other columns, for each element c_i of C, the probability that c_i appears in some column of D in the same row is $1-(1-\frac{1}{q})^d$. Thus, the probability that there exist $t-z+1$ elements of C such that each of them appears in some column of D in the same row is at most

$$\binom{t}{t-z+1}[1-(1-\frac{1}{q})^d]^{t-z+1} = \binom{t}{z-1}[1-(1-\frac{1}{q})^d]^{t-z+1}.$$

M is not $(d,1;z)$-disjunct if and only if there exists a column C and a set D of d other columns such that the above holds. Therefore, the probability that M is not $(d,1;z)$-disjunct is no more than

$$(n-d)\binom{n}{d}\binom{t}{z-1}[1-(1-\frac{1}{q})^d]^{t-z+1}.$$

The goal is to minimize tq, the number of rows of the corresponding $(d;z)$-disjunct matrix, under the condition that

$$n^{d+1}t^z[1-(1-\frac{1}{q})^d]^{t-z} \leq 1. \tag{2}$$

Notice that Eq. (2) implies

$$(n-d)\binom{n}{d}\binom{t}{z-1}[1-(1-\frac{1}{q})^d]^{t-z+1} < 1.$$

Thus, the probability that M is $(d,1;z)$-disjunct is greater than zero, which similarly implies the existence of a $t \times n$ q-ary $(d,1;z)$-disjunct matrix, and a $(d;z)$-disjunct matrix with n columns and at most tq rows.

Let q_0 be the point that maximizes

$$B_d(q) = \frac{-\log[1-(1-\frac{1}{q})^d]}{q}.$$

Assign $q = q_0$. To satisfy Eq. (2), which is equivalent to

$$(d+1)\log n + z\log t \leq -(t-z)\log[1-(1-\frac{1}{q_0})^d],$$

let

$$t = \frac{(d+1)\log n}{-\log[1-(1-\frac{1}{q_0})^d]} + z + t_1.$$

Then, t_1 should satisfy

$$z\log\left\{\frac{(d+1)\log n}{-\log[1-(1-\frac{1}{q_0})^d]} + z + t_1\right\} \leq -t_1\log[1-(1-\frac{1}{q_0})^d] \tag{3}$$

Let

$$t_1 = \frac{z\log\log n}{-\log[1-(1-\frac{1}{q_0})^d]} + t_2.$$

From Eq. (3), t_2 should satisfy that

$$\frac{z}{-\log[1-(1-\frac{1}{q_0})^d]}\log\left\{\frac{(d+1)}{-\log[1-(1-\frac{1}{q_0})^d]} + \frac{1}{\log n}\left(\frac{z\log\log n}{-\log[1-(1-\frac{1}{q_0})^d]} + z + t_2\right)\right\} \leq t_2 \tag{4}$$

For d and z constants (thus, q_0 is also constant), as $n \to \infty$, the minimum value of t_2 satisfying Eq. (4) is

$$t_2 = \frac{z}{-\log[1-(1-\frac{1}{q_0})^d]}\log\frac{(d+1)}{-\log[1-(1-\frac{1}{q_0})^d]} = O(1).$$

Thus,

$$t = \frac{(d+1)\log n}{-\log[1-(1-\frac{1}{q_0})^d]} + \frac{z\log\log n}{-\log[1-(1-\frac{1}{q_0})^d]} + O(1)$$

satisfies Eq. (2) (where the constant term z in t is absorbed in O(1)). Therefore, the number of rows of the corresponding $(d;z)$-disjunct matrix is at most

$$tq_0 = \frac{d+1}{B_d}\log n + \frac{z}{B_d}\log\log n + O(1),$$

where

$$B_d = B_d(q_0) = \max_{q>1} \frac{-\log[1-(1-\frac{1}{q})^d]}{q}.$$

Also,

$$B_d \to \frac{1}{d\log e} \text{ as } d \to \infty,$$

as proved in Theorem 10. □

6 Transformation from Error-Tolerant Separable Matrices to Error-Tolerant Disjunct Matrices

Let M be a 0/1 matrix. For any set S of columns of M, $U(S)$ will denote the union of the row indices of 1-entries of all columns in S. When S is the singleton set $\{C\}$, by abusing the notation, $U(S)$ is simply written as C. Matrix M is called d-*separable* if for any two distinct d-sets S and S' of columns, $U(S) \neq U(S')$. M is called $\bar{d}$-*separable* if the restrictions $|S| = d$ and $|S'| = d$ above are changed to $|S| \leq d$ and $|S'| \leq d$, respectively. Finally, M is called d-*disjunct* if for any d-set S of columns and any column C not in S, C is not contained in $U(S)$. These three properties of 0/1 matrices have been widely studied in the literature of nonadaptive group testing designs (pooling designs), which have applications in DNA screening [17, 25, 32, 38, 39].

It has long been known that d-disjunctness implies $\bar{d}$-separability which in turn implies d-separability [17, Chap. 2]. Recently, Chen and Hwang [7] found a way to construct a disjunct matrix from a separable matrix to complete the cycle of implications.

Theorem 12 (Chen and Hwang [7]) *Suppose M is a $2d$-separable matrix. Then one can construct a d-disjunct matrix by adding at most one row to M.*

The notion of d-separability, $\bar{d}$-separability, and d-disjunctness has their error-tolerant versions. A 0/1 matrix M is called $(d; z)$-*separable* if $|U(S) \triangle U(S')| \geq z$ for any two d-sets of columns of M. It is $(\bar{d}; z)$-*separable* if the restriction of d-sets is changed to two sets each with at most d elements. Finally, M is $(d; z)$-*disjunct* if for any d-set S of columns and any column C not in S, $|C \setminus U(S)| \geq z$. Note that the variable z represents some redundancy to tolerate errors [17]. For $z = 1$, the error-tolerant version is reduced to the original version.

In [17], Du and Hwang attempted to extend Theorem 12 to its error-tolerant version, as stated in the following theorem:

Theorem 13 ([17], Theorem 2.7.6) *Suppose M is a $(2d; z)$-separable matrix. Then one can obtain a $(d; z)$-disjunct matrix by adding at most z rows to M.*

By Theorem 13, Du and Hwang obtained the following corollary:

Corollary 1 ([17], Theorem 2.7.7) *A $(d; 2z)$-separable matrix can be obtained from a $(2d; z)$-separable matrix by adding at most z rows.*

Unfortunately, Theorem 13 is incorrect; thus, Corollary 1 is also incorrect, as seen by the following counterexample. Let

$$M_1 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

It is easily verified that M_1 is $(2; 2)$-separable. It will be showed next adding two rows to M_1 cannot produce a $(1; 2)$-disjunct matrix.

Let C_1, C_2, C_3, and C_4 denote the four columns of M_1. Suppose setting $C = C_i$ and $S = \{C_j\}$, $i \neq j$. Then two rows are needed such that each containing C_i but not C_j. One such row is already provided by M_1. So one $(1, 0)$-pair is needed in a new row. Since this is required for each pair of (i, j) with $i \neq j$, there are $4 \times 3 = 12$ choices of (i, j) pair, and each such pair needs a $(1, 0)$-pair in a new row, or equivalently, the new rows should provide 12 such $(1, 0)$-pairs. However, one new row can provide at most four $(1, 0)$-pairs (achieved by a row with two 1-entries and two 0-entries). So two new rows are not sufficient to provide the 12 $(1, 0)$-pairs required by the $(1; 2)$-disjunctness property.

In [9], the authors gave a correct version of Theorem 13 and obtained a more rigorous statement of Theorem 12. Their results will be presented in this section.

6.1 Main Results

Lemma 12 ([17], Lemma 2.1.1) *Suppose M is a d-separable matrix with n columns where $d < n$, then it is k-separable for every positive integer $k \leq d$.*

Note that the condition $d < n$ in Lemma 12 is necessary as seen by the following example: Let

$$M_2 = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix}.$$

M_2 is trivially 3-separable. However, it is not 2-separable, as the union of any pair of its columns is identical. Now Lemma 12 is generalized to an error-tolerant version.

Lemma 13 *If a matrix M with n columns is $(d; z)$-separable for $d < n$, then it is $(k; z)$-separable for every positive integer $k \leq d$.*

Proof It suffices to prove M is $(d-1;z)$-separable. Assume that M is not $(d-1;z)$-separable. Then there exist two distinct sets S and S' each consisting of $d-1$ columns of M such that $|U(S)\triangle U(S')| < z$.

If $|S \setminus S'| = |S' \setminus S| \geq 2$, then there must exist a pair of columns (C_x, C_y) such that $C_x \in S \setminus S'$ and $C_y \in S' \setminus S$. It is easy to see that

$$|U(S \cup \{C_y\})\triangle U(S' \cup \{C_x\})| \leq |U(S \cup \{C_y\})\triangle U(S')| \leq |U(S)\triangle U(S')|.$$

This violates the $(d;z)$-separability of M, as desired.

Now consider the case of $|S \setminus S'| = |S' \setminus S| = 1$. It is obvious that $|S \cup S'| = d$. Thanks to $d < n$, one can take a column C of M which is in neither S nor S'. It is easily seen that

$$|U(S \cup \{C\})\triangle U(S' \cup \{C\})| \leq |U(S)\triangle U(S')| < z.$$

This contradicts the $(d;z)$-separability of M, completing the proof. □

Now it is ready to give a correct version of Theorem 13.

Theorem 14 *Suppose M is a $(2d;z)$-separable matrix with n columns where $n \geq 2d+1$. Then one can obtain a $(d;\lceil z/2\rceil)$-disjunct matrix by adding at most $\lceil z/2\rceil$ rows to M.*

Proof Suppose M is not $(d;\lceil z/2\rceil)$-disjunct. Then there exist a column C and a set S of d other columns such that $|C \setminus U(S)| < \lceil z/2\rceil$. By adding at most $\lceil z/2\rceil$ rows to M such that each row has a 1-entry at column C and 0-entries at all columns in S, one can obtain $|C \setminus U(S)| \geq \lceil z/2\rceil$. Of course, there may exist another pair (C', S') where C' is a column and S' is a set of d columns other than C', such that $|C' \setminus U(S')| < \lceil z/2\rceil$ in M. Then break it up by using those $\lceil z/2\rceil$ rows in the same fashion. Here what one needs to show is that this procedure is not self-conflicting, that is, there does not exist two pairs (C, S) and (C', S') such that $|C \setminus U(S)| < \lceil z/2\rceil$, yet on the other hand $C \in S'$ while $|C' \setminus U(S')| < \lceil z/2\rceil$.

Suppose to the contrary that there exist two pairs (C, S) and (C', S') in M as described above with $|S| = |S'| = d$. Define

$$S_0 = \{C'\} \cup S \cup S', \quad S_1 = S_0 \setminus \{C\}, \quad \text{and } S_2 = S_0 \setminus \{C'\}.$$

Let $s = |S_0|$, then $s \leq 2d+1$ and $|S_1| = |S_2| = s-1 \leq 2d$.

Note that $S_1 \neq S_2$, but they have the same cardinality which is less than $2d+1$. Next it will be showed that the symmetric difference of $U(S_1)$ and $U(S_2)$ is less than z, thus violating the assumption of $(2d;z)$-separability.

Since the only column in S_1 but not in S_2 is C' and $|C' \setminus U(S')| < \lceil z/2\rceil$, it follows that

$$|U(S_2) \setminus U(S_1)| < \lceil z/2\rceil. \tag{5}$$

Similarly, one can obtain

$$|U(S_1) \setminus U(S_2)| < \lceil z/2 \rceil. \tag{6}$$

Equation (5) along with Eq. (6) gives $|U(S) \triangle U(S')| < z$, implying that M is not $(s-1; z)$-separable. This contradicts with Lemma 13, and so the theorem is proved. □

Corollary 2 *Suppose M is a $2d$-separable matrix with n columns where $n \geq 2d+1$. Then one can obtain a d-disjunct matrix by adding at most one row to M.*

Proof It follows from Theorem 14 by setting $z = 1$.

Corollary 2 is a more rigorous version of Theorem 12. The following example shows the necessity of the extra condition $n \geq 2d + 1$ in Corollary 2. Let

$$M_3 = \begin{pmatrix} 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

Then M_3 is trivially 4-separable, but it can be easily verified that no row can be added to M_3 to make it 2-disjunct. Similarly, any matrix with $2d$ columns is trivially $(2d; z)$-separable, and one does not expect that adding $\lceil z/2 \rceil$ rows to an arbitrary matrix with $2d$ columns would make it $(d; \lceil z/2 \rceil)$-disjunct. To see a specific counterexample, note that M_1 is trivially a $(4; 4)$-separable matrix, but adding two rows does not make it a $(2; 2)$-disjunct matrix – It is even not $(1; 2)$-disjunct as indicated at the end of Sect. 1.

Corollary 3 *Suppose M is a $(2d; z)$-separable matrix with n columns where $n \geq 2d + 1$. Then, for any positive integer $k \leq \lceil z/2 \rceil$, one can obtain a $(d; k)$-disjunct matrix by adding at most k rows to M.*

Proof The proof of Theorem 14 shows that there does not exist two pairs (C, S) and (C', S') such that $|C \setminus U(S)| < \lceil z/2 \rceil$, yet on the other hand, $C \in S'$ while $|C' \setminus U(S')| < \lceil z/2 \rceil$. In fact, the term $\lceil z/2 \rceil$ can be replaced by any positive integer k which satisfies the symmetric difference of $U(S_1)$ and $U(S_2)$ is less than z. Therefore, for any $k \leq \lceil z/2 \rceil$, one can obtain a $(d; k)$-disjunct matrix by adding at most k rows to M in the same fashion. □

The following equivalence relation is given in [17] without giving a proof. Now a proof is given, and a stronger result is obtained by using the equivalence relation.

Lemma 14 ([17], Lemma 2.7.5) *A matrix M is $(\bar{d}; z)$-separable if and only if it is $(d; z)$-separable and $(d-1; z)$-disjunct.*

Proof Suppose M is $(\bar{d}; z)$-separable but not $(d-1; z)$-disjunct; in other words, there exists a set S of $d-1$ columns other than a column C such that $|C \setminus U(S)| \leq z$. Then it is easy to see that

$$|U(S \cup \{C\}) \triangle U(S)| = |U(S \cup \{C\}) \setminus U(S)| \leq z,$$

a contradiction to $(\bar{d}; z)$-separability. Thus, M is $(d-1; z)$-disjunct and $(d; z)$-separable trivially.

Let M be $(d; z)$-separable and $(d-1; z)$-disjunct. It suffices to show that

$$|U(X) \triangle U(Y)| \geq z$$

for any two sets X, Y of at most d columns. If $|X| = |Y| \leq d$, then $|U(X) \triangle U(Y)| \geq z$ by $(d; z)$-separability and Lemma 13. Assume $|X| < |Y| \leq d$, then there exists a column $C_y \in Y$ but not in X. By $(d-1; z)$-disjunctness, it must have $|C_y \setminus U(X)| \geq z$; hence, $|U(X) \triangle U(Y)| \geq z$. It concludes the proof. □

By Lemmas 14 and 13, Corollary 3 is extended to a stronger version.

Corollary 4 *Suppose M is a $(2d; z)$-separable matrix with n columns where $n \geq 2d+1$. Then, for any positive integer $k \leq \lceil z/2 \rceil$, one can obtain a $(\bar{d+1}; k)$-separable matrix by adding at most k rows to M.*

6.2 Concluding Remarks

The following remarks demonstrate the optimality of the results presented in this section.

1. The constraint $k \leq \lceil z/2 \rceil$ in Corollary 3 is necessary to make the number of rows added to be independent of n and d. To see a specific example, consider that M is an $(n\lceil z/2 \rceil) \times n$ matrix such that each column has $\lceil z/2 \rceil$ 1-entries and any 2 columns have no intersection. Then, M is $(2d; z)$-separable. Since every column has only $\lceil z/2 \rceil$ 1-entries, to make M $(d; k)$-disjunct by adding rows, the rows added must form a $(d; k - \lceil z/2 \rceil)$-disjunct submatrix when $k > \lceil z/2 \rceil$. In this case, the minimum number of rows required would depend on n, d, and $k - \lceil z/2 \rceil$.
2. Let N be a 0/1 matrix of constant row sum 1 and constant column sum z, and let M be obtained from N by adding one zero column. It is easy to verify that M is $(2d; z)$-separable. Since there is a zero column in M, one cannot obtain from M a $(d; k)$-disjunct matrix by adding less than k rows. This shows that the bound on the number of additional rows given in Corollary 3 is optimal in this sense.

7 Conclusion

Some recent algorithmic, complexity, and mathematical results on nonadaptive group testing (and on pooling design) are presented in this monograph. New construction of disjunct matrices with even reduced number of rows remains interesting to investigate. The complexity of the problem MIN-$\bar{d}$-SS introduced in Sect. 3.3 remains unsolved. On the bounds of the minimum number $t(d,n)$ of rows of d-disjunct matrices with n columns, closing the gap between $O(d^2 \log n)$ and $\Omega(\frac{d^2 \log n}{\log d})$ remains as a major open problem in extremal combinatorics.

Cross-References

▸Combinatorial Optimization Algorithms

Recommended Reading

1. N. Alon, J.H. Spencer, *The Probabilistic Method* (Wiley, New York, 1992). (Second Edition 2000)
2. N. Alon, D. Moshkovitz, S. Safra, Algorithmic construction of sets for k-restrictions. ACM Trans. Algorithms **2**(2), 153–177 (2006)
3. D.J. Balding, W.J. Bruno, E. Knill, D.C. Torney, A comparative survey of non-adaptive pooling designs, in *Genetic Mapping and DNA Sequencing* (Springer, New York, 1996), pp. 133–154
4. T. Berger, J.W. Mandell, P. Subrahmanya, Maximally efficient two-stage group testing. Biometrics **56**, 833–840 (2000)
5. J. Borneman, M. Chrobak, G. Della Vedova, A. Figueroa, T. Jiang, Probe selection algorithms with applications in the analysis of microbial communities. Bioinformatics **17**(Suppl.), S39–S48 (2001)
6. W.J. Bruno, D.J. Balding, E. Knill, D.C. Bruce et al., Efficient pooling designs for library screening. Genomics, **26**, 21–30 (1995)
7. H.B. Chen, F.K. Hwang, Exploring the missing link among d-separable, $\bar{d}$-separable and d-disjunct matrices. Discret. Appl. Math. **133**, 662–664 (2007)
8. J. Chen, I.A. Kanj, W. Jia, Vertex cover: further observations and further improvements. J. Algorithm **41**(2), 280–301 (2001)
9. H.B. Chen, Y. Cheng, Q. He, C. Zhong, Transforming an error-tolerant separable matrix to an error-tolerant disjunct matrix. Discret. Appl. Math. **157**(2), 387–390 (2009)
10. Y. Cheng, D.Z. Du, New constructions of one- and two-stage pooling designs. J. Comput. Biol. **15**, 195–205 (2008)
11. Y. Cheng, K.-I Ko, W. Wu, On the complexity of non-unique probe selection. Theor. Comput. Sci. **390**(1), 120–125 (2008)
12. Y. Cheng, D.Z. Du, K.-I. Ko, G. Lin, On the parameterized complexity of pooling design. J. Comput. Biol. **16**, 1529–1537 (2009)
13. Y. Cheng, D.Z. Du, G. Lin, On the upper bounds of the minimum number of rows of disjunct matrices. Optim. Lett. **3**(2), 297–302 (2009)
14. A. De Bonis, L. Gasieniec, U. Vaccaro, Optimal two-stage algorithms for group testing problems. SIAM J. Comput. **34**, 1253–1270 (2005)
15. R. Dorfman, The detection of defective members of large populations. Ann. Math. Stat. **14**, 436–440 (1943)

16. R.G. Downey, M.R. Fellows, *Parameterized Complexity* (Springer, New York, 1999)
17. D.-Z. Du, F.K. Hwang, *Pooling Designs and Nonadaptive Group Testing: Important Tools for DNA Sequencing* (World Scientific, New Jersey, 2006)
18. D.-Z. Du, K.-I. Ko, Some completeness results on decision trees and group testing. SIAM J. Algebra. Discret. **8**(4), 762–777 (1987)
19. D.-Z. Du, K.-I. Ko, *Theory of Computational Complexity* (Wiley, New York, 2000)
20. D.Z. Du, F.K. Hwang, W. Wu, T. Znati, New construction for transversal design. J. Comput. Biol. **13**, 990–995 (2006)
21. A.G. D'yachkov, V.V. Rykov, Bounds of the length of disjunct codes. Probl. Control Inf. Theory **11**, 7–13 (1982)
22. A.G. D'yachkov, V.V. Rykov, A.M. Rashad, Superimposed distance codes. Probl. Control Inf. Theory **18**, 237–250 (1989)
23. A.G. D'yachkov, A.J. Macula, V.V. Rykov, New constructions of superimposed codes. IEEE Trans. Inf. Theory **46**, 284–290 (2000)
24. D. Eppstein, M.T. Goodrich, D.S. Hirschberg, Improved combinatorial group testing algorithms for real-world problem sizes. SIAM J. Comput. **36**, 1360–1375 (2007)
25. P. Erdős, P. Frankl, Z. Füredi, Families of finite sets in which no set is covered by the union of r others. Isr. J. Math. **51**, 79–89 (1985)
26. M. Farach, S. Kannan, E. Knill, S. Muthukrishnan, Group testing problems with sequences in experimental molecular biology, in *Proceedings of the Compression and Complexity of Sequences*, ed. by B. Carpentieri et al. (IEEE Press, Los Alamitos, 1997), pp. 357–367
27. J. Flum, M. Grohe, *Parameterized Complexity Theory*. Texts in Theoretical Computer Science, an EATCS Series, vol. XIV (Springer, Berlin, 2006)
28. H.L. Fu, F.K. Hwang, A novel use of t-packings to construct d-disjunct matrices. Discret. Appl. Math. **154**, 1759–1762 (2006)
29. Z. Füredi, On r-cover-free families. J. Comb. Theory Ser. A **73**, 172–173 (1996)
30. M.R. Garey, D.S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness* (Freeman, San Francisco, 1979)
31. R. Herwig, A.O. Schmitt, M. Steinfath, J. O' Brien et al., Information theoretical probe selection for hybridisation experiments. Bioinformatics **16**, 890–898 (2000)
32. F.K. Hwang, V.T. Sós, Non-adaptive hypergeometric group testing. Studia Sci. Math. Hung. **22**, 257–263 (1987)
33. P. Indyk, H.Q. Ngo, A. Rudra, Efficiently decodable non-adaptive group testing, in *Proceedings of 21st Annual ACM-SIAM Symposium on Discrete Algorithms*, Austin, 2010, pp. 1126–1142
34. W.H. Kautz, R.C. Singleton, Nonrandom binary superimposed codes. IEEE Trans. Inf. Theory **10**, 363–377 (1964)
35. G.W. Klau, S. Rahmann, A. Schliep, M. Vingron, K. Reinert, Optimal robust non-unique probe selection using integer linear programming. Bioinformatics **20**, i186–i193 (2004)
36. E. Knill, Lower bounds for identifying subset members with subset queries, in *Proceedings of 6th ACM-SIAM Symposium on Discrete Algorithms*, San Francisco, CA, USA, 1995, pp. 369–377
37. O. Lichtenstein, A. Pnueli, Checking that finite state concurrent programs satisfy their linear specification, in *Proceedings of 12th ACM Symposium on Principles of Programming Languages (POPL' 85)*, 107, New York, 1985, pp. 97–107
38. A.J. Macula, A simple construction of d-disjunct matrices with certain constant weights. Discret. Math. **162**, 311–312 (1996)
39. A.J. Macula, Error-correcting nonadaptive group testing with d^e-disjunct matrices. Discret. Appl. Math. **80**, 217–222 (1997)
40. A.J. Macula, Probabilistic nonadaptive group testing in the presence of errors and DNA library screening. Ann. Comb. **3**, 61–69 (1999)
41. A.J. Macula, Probabilistic nonadaptive and two-stage group testing with relatively small pools and DNA library screening. J. Comb. Optim. **2**, 385–397 (1999)
42. R. Motwani, P. Raghavan, *Randomized Algorithms* (Cambridge University Press, New York, 1995)

43. H.Q. Ngo, D.Z. Du, A survey on combinatorial group testing algorithms with applications to DNA library screening, in *DIMACS: Series in Discrete Mathematics and Theoretical Computer Science*, vol. 55 (American Mathematical Society, Providence, 2000), pp. 171–182
44. H.Q. Ngo, D.Z. Du, New constructions of non-adaptive and error-tolerance pooling designs. Discret. Math. **243**, 161–170 (2002)
45. H.Q. Ngo, E. Porat, A. Rudra, Efficiently decodable error-correcting list disjunct matrices and applications, in *Proceedings of the 38th International Colloquium on Automata, Languages and Programming*, Zurich, Switzerland, 2011, pp. 557–568
46. M. Olson, L. Hood, C. Cantor, D. Botstein, A common language for physical mapping of the human genome. Science **245**, 1434–1435 (1989)
47. C.H. Papadimitriou, *Computational Complexity* (Addison-Wesley, New York, 1994)
48. C.H. Papadimitriou, D. Wolfe, The complexity of facets resolved, J. Comput. Syst. Sci. **37**, 2–13 (1988)
49. H. Park, W. Wu, Z. Liu, X. Wu, H.G. Zhao, DNA screening, pooling design and simplicial complex. J. Comb. Optim. **7**, 389–394 (2003)
50. E. Porat, A. Rothschild, Explicit non-adaptive combinatorial group testing schemes, in *Proceedings of the 35th International Colloquium on Automata, Languages and Programming*, Reykjavik, Iceland, 2008, pp. 748–759
51. S. Rahmann, Rapid large-scale oligonucleotide selection for microarrays, in *Proceedings of the 1st IEEE Computer Society Conference on Bioinformatics (CSB' 02)*, Stanford, CA, USA, 2002, pp. 54–63
52. S. Rahmann, Fast and sensitive probe selection for DNA chips using jumps in matching statistics, in *Proceedings of the 2nd IEEE Computer Society Bioinformatics Conference (CSB' 03)*, Stanford, CA, USA, 2003, pp. 57–64
53. M. Ruszinkó, On the upper bound of the size of the r-cover-free families. J. Comb. Theory Ser. A **66**, 302–310 (1994)
54. A. Schliep, D.C. Torney, S. Rahmann, Group testing with DNA chips: generating designs and decoding experiments, in *Proceedings of the 2nd IEEE Computer Society Bioinformatics Conference (CSB' 03)*, Stanford, CA, USA, 2003, pp. 84–93
55. C. Umans, The minimum equivalent DNF problem and shortest implicants, in *Proceedings of 39th IEEE Symposium on Foundation of Computer Science*, Palo Alto, CA, USA, 1998, pp. 556–563
56. X. Wang, B. Seed, Selection of oligonucleotide probes for protein coding sequences. Bioinformatics **19**, 796–802 (2003)
57. J.K. Wolf, Born again group testing: multiaccess communications. IEEE Trans. Inf. Theory **IT-31**, 185–191 (1985)