JOURNAL OF COMPUTATIONAL BIOLOGY
Volume 15, Number 2, 2008
© Mary Ann Liebert, Inc.
Pp. 195–205
DOI: 10.1089/cmb.2007.0195

New Constructions of One- and Two-Stage Pooling Designs

YONGXI CHENG[1] and DING-ZHU DU[2]

ABSTRACT

The study of gene functions requires a DNA library of high quality, such a library is obtained from a large mount of testing and screening. Pooling design is a very helpful tool for reducing the number of tests for DNA library screening. In this paper, we present new one- and two-stage pooling designs, together with new probabilistic pooling designs. The approach in this paper works for both error-free and error-tolerance scenarios.

Key words: DNA library screening, nonadaptive group testing, pooling designs.

1. INTRODUCTION

A DNA LIBRARY CONSISTS OF THOUSANDS of separate recombinant DNA *clones*, each of which represents some contiguous piece of a contiguous superpiece of DNA. The basic task of DNA library screening is, for a collection of *probes* (a probe is a DNA subsequence), to determine which clone from the library contains which probe. For a given 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 pooled together to be tested against each probe, because checking each clone-probe pair is expensive and usually for a given probe only a few clones in the library contain it. There are experimental tests (e.g., the polymerase chain reaction), which can determine whether or not there exists at least one clone from a pool containing a given probe.

The above problem is an instance of the *combinatorial group testing* problem, in which we have n items each has an unknown binary status, *positive* (used to be called *defective*) or *negative* (used to be called *good*), and the number of positives is upper bounded by an integer d (generally we assume $d \ll n$). Suppose there is some method to test whether or not a subset of items contains at least one positive. We say that the test outcome is positive if the test result indicates that the subset contains at least one positive item, otherwise we say that the test outcome is negative. The problem is to resolve the status of every item using the minimum number of tests.

Although the research of combinatorial group testing dates back to Dorfman's 1943 paper (Dorfman, 1943), a renewed interest in the subject has occurred largely due to the applications of group testing to the area of computational molecular biology, in which one important application is clone library screening (Balding et al., 1995; Farach et al., 1997). In the application to 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

[1]Institute for Theoretical Computer Science, Tsinghua University, Beijing, China.
[2]Department of Computer Science, University of Texas at Dallas, Richardson, Texas.

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 (2006) for a comprehensive discussion of this topic.

Group testing 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 algorithm must specify all tests before knowing any test outcomes, the benefit is that all tests can be performed in parallel. For group testing problems, 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,[1] where n is the number of items and d is the maximum number of positives, and the best known nonadaptive algorithm uses $O(d^2 \log n)$ tests. In contrast, there are adaptive algorithms using $O(d \log n)$ tests (Du and Hwang, 2006), which match the information theoretic lower bound.

A nonadaptive group testing algorithm can be represented as a 0/1 matrix $M = (m_{ij})$, where 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. Given the matrix representation of an algorithm and the test outcomes, the process of identifying all the positive items is called *decoding*.

A 0/1 matrix is said to be *d-disjunct* if no column is covered by the union of d other columns, by union we mean bitwise boolean sum. A 0/1 matrix is said to be *$(d; z)$-disjunct* (D'yachkov et al., 1989; Macula, 1997) 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-disjunct is just $(d; 1)$-disjunct. If the matrix M representing a nonadaptive algorithm is d-disjunct and the number of positives is no more than d, then we have efficient decoding procedure with running time linear in the size of M. When there are errors in the test outcomes, we require the matrix to be $(d; z)$-disjunct, which forms a $\lfloor \frac{z-1}{2} \rfloor$-error-correcting algorithm. In this case we still have linear decoding that successfully identifies all positives, provided 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 for error-free and error-tolerant nonadaptive group testing algorithms.

Between fully adaptive and nonadaptive (one-stage) algorithms, the so called *trivial two-stage algorithms* (Knill, 1995) 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 outcomes; in the second stage, individual tests are performed on all the items in CP. Previous works on two-stage group testing algorithms are, among others, Knill (1995), Macula (1999), Berger et al. (2000), De Bonis et al. (2005), and Eppstein et al. (2007). The following quotation from Knill (1995) 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."

1.1. Related work

On constructions of disjunct matrices, Kautz and Singleton (1964) introduced the construction from set packing designs, in the context of superimposed codes. Hwang and Sós (1987) (also cited in the book by Du and Hwang [2006]) give a greedy type construction which results in $t \times n$ d-disjunct matrices with $t \leq 16d^2(1 - \log_3 2 + (\log_3 2) \log n) \approx 5.91d^2 + 10.09d^2 \log n$. In Cheng and Du (2007), they propose a

[1]Throughout the paper, log is of base 2 if no base is specified.

Las Vegas construction, which compared to Hwang and Sós (1987) reduces t by more than half for $d \ln n$ reasonably large, also the time required by the construction is reduced to polynomial in n and d. Other works known on constructing disjunct matrices are those of, among others, Erdös et al. (1985), Macula (1996), D'yachkov et al. (2000), Ngo and Du (2002), Park et al. (2003), Du et al. (2006), Fu and Hwang (2006), and Eppstein et al. (2007).

For two-stage pooling designs, De Bonis et al. (2005) 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. (2007) improve the constant factor to $4(1+o(1))$ by using the concept of (d, k)-*resolvable* matrices (which we will explain later), which is currently the best. There are also probabilistic pooling designs (Macula, 1999a, 1999b; Ngo and Du, 2002) with good performance in practice.

1.2. Main results

We first give two improved Las Vegas algorithms from those in Cheng and Du (2007), for constructing d-disjunct and $(d; z)$-disjunct matrices respectively. Compared to Cheng and Du (2007), our new constructions are transversal designs. They reduce t noticeably in practice and are much more simplified. For two-stage pooling designs, we present an algorithm using a number of $C_d(1+o(1))\log n$ tests, 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 of Eppstein et al. (2007) by a factor of more than 2. We also propose new probabilistic pooling designs. Compared to Ngo and Du (2002), our probabilistic designs have different type of possible errors and require much fewer tests.

2. PRELIMINARIES

2.1. 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. We first introduce the concept of *q-ary $(d, 1)$-disjunct* matrix. 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 Du and Hwang (2006) and Du et al. (2006), 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 Du and Hwang [2006]). *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 we need the same number of copies for each item.

2.2. Two probabilistic lemmas

We present two inequalities that will be useful later. The first is the Markov inequality (see, for example, Theorem 3.2 in Motwani and Raghavan [1995]), and the second is commonly known as Chernoff's bounds (see Theorem 4.1 and 4.2 in Motwani and Raghavan [1995]).

Lemma 2 (Markov Inequality). *Let Y be a random variable assuming only non-negative values, then for all $t > 0$,*

$$\Pr[Y \ge t] \le \frac{E[Y]}{t},$$

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

Lemma 3 (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 $\delta > 0$,*

$$\Pr[X \geq (1+\delta)\mu] \leq \left(\frac{e^{\delta}}{(1+\delta)^{1+\delta}}\right)^{\mu}, \tag{1}$$

$$\Pr[X \leq (1-\delta)\mu] \leq e^{-\mu\delta^2/2}. \tag{2}$$

The Chernoff's bounds in Motwani and Raghavan (1995) are for strict inequalities, but the same bounds also hold for nonstrict inequalities.

3. ONE-STAGE POOLING DESIGNS

We give two efficient constructions of d-disjunct and $(d; z)$-disjunct matrices, respectively. The constructions are transversal, and are based on randomized approach.

3.1. A new construction of d-disjunct matrices

In this section, we present a Las Vegas algorithm, for given n, d and $0 < p < 1$, the algorithm 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 at the end of this subsection). Assign $q = (1+\epsilon)d$, $t_0 = \frac{1+\epsilon}{\epsilon} d \ln \frac{2n-1}{1-p}$, and $\mu = \frac{t_0}{q}$. Please see Algorithm 1 as our algorithm for constructing d-disjunct matrices.

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 $1 \leq i < j \leq n_0$, 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 $w_{i,j} \geq (1+\epsilon)\mu$.
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
   $t \leq t_0 q$.
```

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, we have $t \leq t_0 q = \frac{(1+\epsilon)^2}{\epsilon \log e} d^2 \log \frac{2n-1}{1-p} < cd^2(\log \frac{2}{1-p} + \log n)$, where $c = \frac{(1+\epsilon)^2}{\epsilon \log e} \approx 4.28$.

3.1.1. Analysis of Algorithm 1. In this section, we analyze the success probability and running time of Algorithm 1.

Success probability. First, we estimate the expectation of the number of edges created at Step 2.

Lemma 4. *Let m be the random variable denoting the number of edges created at Step 2 of Algorithm 1, then $E[m] \leq n(1-p)$.*

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 $X_k^{i,j} = 1$ 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

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

Since $w_{i,j}$ is the sum of t_0 independent 0/1 random variables, the Chernoff bound, Equation (1) in Lemma 3, 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, $\mu = E[w_{i,j}] = \frac{t_0}{q} = \frac{1}{\epsilon} \ln \frac{2n-1}{1-p}$, and from $\ln(1+\epsilon) = \frac{2\epsilon}{1+\epsilon}$ we have $(1+\epsilon)^{1+\epsilon} = e^{2\epsilon}$, and so $\frac{e^\epsilon}{(1+\epsilon)^{1+\epsilon}} = e^{-\epsilon}$, which implies that $(\frac{e^\epsilon}{(1+\epsilon)^{1+\epsilon}})^\mu = (e^{-\epsilon})^{\frac{1}{\epsilon}\ln\frac{2n-1}{1-p}} = \frac{1-p}{2n-1}$. 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 $n_0 = 2n$, $E[m] = \binom{n_0}{2} E[X^{1,2}] \le \binom{n_0}{2}\frac{1-p}{2n-1} = n(1-p)$. ■

Clearly, m denotes the most number of columns that may be removed at Step 3. Since $E[m] \le n(1-p)$, by applying the Markov inequality (Lemma 2), 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 $\Pr[m > n] \le \frac{E[m]}{n} \le 1-p$.

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, we can obtain an expected $O(n^2 \ln n)$ running time by counting along the rows.

For $1 \le i < j \le n_0$, 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 $1 \le i < j \le n_0$. 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 $S_{r,k}$, $1 \le k \le q$, can be constructed in n_0 time. For each k, we 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 $1 \le k \le q$, the expected running time of Step 2 is $t_0 \times (n_0 + qE[\binom{|S_{r,1}|}{2}])$. Notice that $|S_{r,1}|$ has the binomial distribution with parameters n_0 and $1/q$, thus $n_0 + qE[\binom{|S_{r,1}|}{2}] = n_0 + q\frac{1}{q^2}(n_0^2 - n_0) = O(n^2/d)$, 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, we have established the following theorem.

Theorem 5. *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, we choose ϵ to minimize the leading constant of t. We require $(1+\epsilon)\mu \le \frac{t_0}{d}$ (where $\mu = \frac{t_0}{q}$), i.e., $q \ge (1+\epsilon)d$, to guarantee that matrix M'' is $(d,1)$-disjunct. To guarantee that with reasonable probability M'' has no less than n columns, we require (at least) $n_0 - E[m] \ge n$ (where $E[m] = \binom{n_0}{2}E[X^{1,2}]$), which implies that $\Pr[X^{1,2} = 1] \le \frac{n_0 - n}{\binom{n_0}{2}}$. 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 $n_0 = 2n$, we should have $\Pr[X^{1,2} = 1] \le \frac{1}{2n-1}$. This can be guaranteed by $(\frac{e^\epsilon}{(1+\epsilon)^{1+\epsilon}})^\mu \le \frac{1}{2n-1}$, that is, $\mu \ln \frac{(1+\epsilon)^{1+\epsilon}}{e^\epsilon} \ge O(1) + \ln n$. By plugging in $\mu = \frac{t_0}{q} = \frac{t}{q^2}$ and $q \ge (1+\epsilon)d$, we get $t \ge \frac{(1+\epsilon)^2}{\ln\frac{(1+\epsilon)^{1+\epsilon}}{e^\epsilon}} d^2(O(1) + \ln n)$. 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, from the equation we have $f(\epsilon) = \frac{(1+\epsilon)^2}{(1+\epsilon)\ln(1+\epsilon)-\epsilon} = \frac{(1+\epsilon)^2}{\epsilon}$.

3.2. *Error-tolerance case*

In this section, we modify Algorithm 1 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 $t \leq cd^2(\log\frac{2}{1-p} + \log n) + 2(1+\epsilon)dz + O(\frac{z^2}{\ln n})$, where $\epsilon \approx 3.92$ and $c \approx 4.28$ are constants, and the $O(\cdot)$ notation hides dependencies on p.

We first give a generalization of $(d, 1)$-disjunct matrices. 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. Assign $q = (1+\epsilon)d + \frac{\epsilon z}{\ln\frac{2n-1}{1-p}}$, $t_0 = z + \frac{1+\epsilon}{\epsilon} d \ln\frac{2n-1}{1-p}$, and $\mu = \frac{t_0}{q}$. It can be verified that by this assignment, $(1+\epsilon)\mu = \frac{t_0 - z}{d}$ and $(\frac{e^\epsilon}{(1+\epsilon)^{1+\epsilon}})^\mu = \frac{1-p}{2n-1}$. Please see Algorithm 2 as our algorithm for constructing $(d; z)$-disjunct matrices.

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

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, $t \leq t_0 q = \frac{(1+\epsilon)^2}{\epsilon \log e} d^2 \log\frac{2n-1}{1-p} + 2(1+\epsilon)dz + \frac{(\epsilon \log e)z^2}{\log\frac{2n-1}{1-p}} \leq cd^2(\log\frac{2}{1-p} + \log n) + 2(1+\epsilon)dz + O(\frac{z^2}{\ln n})$, where $c = \frac{(1+\epsilon)^2}{\epsilon \log e} \approx 4.28$.

For the success probability, if we let m^* be the random variable denoting the number of edges created at Step 2, since $(\frac{e^\epsilon}{(1+\epsilon)^{1+\epsilon}})^\mu = \frac{1-p}{2n-1}$ still holds, the same result in lemma 4 also holds here, that is $E[m^*] \leq n(1-p)$. 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 $\Pr[m^* > n] \leq 1 - p$. We have established the following theorem.

Theorem 6. *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 $t \leq cd^2(\log\frac{2}{1-p} + \log n) + 2(1+\epsilon)dz + O(\frac{z^2}{\ln n})$, 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.*

Remarks. The two constructions in this section are improvements from the previous constructions proposed in Cheng and Du (2007), by incorporating the concepts of transversal designs and *q-ary $(d, 1)$-disjunct* (*$(d, 1; z)$-disjunct*) matrices. Compared to Cheng and Du (2007), the new constructions have several advantages. Firstly, as mentioned before the new constructions are transversal designs requiring equal number of copies for each item, which is practically favorable. Secondly, they both reduce t by a factor of $\frac{1}{(1-\delta)^2}$, which is approximately $(1 + \frac{1}{\sqrt{d \ln n}})$ for a reasonably good choice of parameters c_1 and c_2 in the previous constructions. For instance, for $n = 10{,}000{,}000$ and $d = 5$, we have $1 + \frac{1}{\sqrt{d \ln n}} \approx 1.11$, which means that for this n and d we can reduce the number of tests by approximately 10%. Thirdly, the new constructions are much simplified while require the same running time.

4. TWO-STAGE POOLING DESIGNS

In this section, we present 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 $d\log(n/d)$. This improves the previously best upper bound of $4(1+o(1))$ times the information theoretic bound of Eppstein et al. (2007) 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 Eppstein et al. (2007), the authors relax the property of M by introducing the concept of (d,k)-*resolvable* matrices, which forms 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, we 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 $t+d+k$.

4.1. *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. We say that M_1 is $(d,1;k)$-*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 we mean 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 $t\le t_0q$.

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

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 $\Pr[X_D\ge(1+\delta)]\le\frac{e^\delta}{(1+\delta)^{1+\delta}}$. Therefore, the probability that M' is not $(d,1;1+\delta)$-resolvable, i.e., 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 $(\frac{1+\delta}{e})^{1+\delta}=n^d$, 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 we have proved the existence of a $t_0\times n$ q-ary $(d,1;1+\delta)$-resolvable matrix with t_0 and δ as specified above.

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 $t\le t_0q<-\frac{q\log n}{\log[1-(1-\frac{1}{q})^d]}$. Define

$C_d(x) = \frac{x}{-\log[1-(1-\frac{1}{x})^d]}$ for $x > 1$. We choose q to be the positive integer that minimizes $C_d(x)$, and let $C_d = C_d(q)$. Then, $t \le C_d \log n$. For $d \ge 1$, $C_d/d \le C_d(3d)/d = \frac{3}{-\log[1-(1-\frac{1}{3d})^d]} \le \frac{3}{\log 3}$ (also it is not hard to see that when $d = 1$, $C_1 = C_1(3) = \frac{3}{\log 3}$ indeed holds). Further more, the following lemma estimates that $q = \Theta(d)$ and $C_d \to d \log e$ as $d \to \infty$.

Lemma 7. *For $d \ge 1$, 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 7, we first prove a useful fact.

Fact 8. *Let $f(y) = \ln y \ln(1-y)$, $0 < y < 1$. Then $f(y)$ achieves maximum at $y = \frac{1}{2}$.*

Proof of Fact 8. By symmetry, it is sufficient to show that $f'(y) > 0$ for $0 < y < \frac{1}{2}$. Since $f'(y) = \frac{1}{y}\ln(1-y) - \frac{1}{1-y}\ln y = \frac{1}{y(1-y)}[(1-y)\ln(1-y) - y\ln y]$, let $g(y) = (1-y)\ln(1-y) - y\ln y$, we will show that $g(y) > 0$ for $0 < y < \frac{1}{2}$.

We write $g(y) = \ln(1-y) + y\ln\frac{1}{y(1-y)}$. 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 $h(0) = h(\frac{1}{2}) = 0$, we can obtain $h(y) > 0$ 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 7. 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, $C_d(q_1) = q_1 = \Theta(d)$. We prove the lemma 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 we write q instead of $q(d)$, if it is clear from the context), as $c \to \infty$, $C_d(q) = \frac{q}{-\log[1-(1-\frac{1}{q})^d]} \sim \frac{q}{-\log[1-(1-\frac{d}{q})]} = d\frac{\frac{q}{d}}{\log\frac{q}{d}} = \omega(d)$ (here by $a \sim b$ we mean that $\lim_{c\to\infty}\frac{a}{b} = 1$). However, this contradicts since q is point that minimizes $C_d(q)$ and we already have $C_d(q_1) = \Theta(d)$. 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 $C_d(q) = \frac{q}{-\log[1-(1-\frac{1}{q})^d]} = \frac{q\ln 2}{-\ln\{1-[(1-\frac{1}{q})^q]^{\frac{d}{q}}\}}$. Since $0 < (1-\frac{1}{q})^q < \frac{1}{e}$ for $q > 1$, as $c \to 0$, $\frac{d}{q} > \frac{1}{c} \to \infty$, and $[(1-\frac{1}{q})^q]^{\frac{d}{q}} < e^{-\frac{d}{q}} \to 0$, thus $C_d(q) \sim \frac{q\ln 2}{[(1-\frac{1}{q})^q]^{\frac{d}{q}}} > e^{\frac{d}{q}}q\ln 2 = d\frac{e^{\frac{d}{q}}\ln 2}{\frac{d}{q}} = \omega(d)$, this also contradicts (here by $a \sim b$ we mean that $\lim_{c\to 0}\frac{a}{b} = 1$). Therefore, $q(d) = \Theta(d)$.

Next, we estimate C_d as $d \to \infty$. Since $(1-\frac{1}{q})^q < \frac{1}{e}$ for $q > 1$, thus $(1-\frac{1}{q})^d < (\frac{1}{e})^{\frac{d}{q}} = e^{-\frac{d}{q}}$, and $-\log[1-(1-\frac{1}{q})^d] < -\log(1-e^{-\frac{d}{q}})$, it follows that $C_d(q) = \frac{q}{-\log[1-(1-\frac{1}{q})^d]} > \frac{q}{-\log(1-e^{-\frac{d}{q}})} = \frac{d\ln 2}{[-\frac{d}{q}\ln(1-e^{-\frac{d}{q}})]}$. 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 8), we obtain $C_d(q) > d\log e$ for $q > 1$, thus $C_d > d\log e$ for $d \ge 1$. 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 $d \to \infty$, $C_d \to d\log e$. ■

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

Theorem 9. *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 $C_d = \min_{x \in \mathcal{N}} \frac{x}{-\log[1-(1-\frac{1}{x})^d]} \leq \frac{3}{\log 3} d$ for $d \geq 1$, and $\delta = (1+o(1))\frac{d \ln n}{\ln(d \ln n)}$. Moreover, $\lim_{d\to\infty} C_d/d = \log e$.*

5. PROBABILISTIC POOLING DESIGNS

We present a probabilistic pooling design identifying up to d positives from n items with high probability. In a probabilistic group testing algorithm, one may identify a positive item as negative, we call such a wrongly identified item 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 are, among others, Macula (1999a, 1999b), Ngo and Du (2002).

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 with 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. We identify the items not in any negative pool 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' (we use the same notations c and D for different matrices M and M', to denote the corresponding columns). The probability that c is covered by D, as analyzed in Section 4, is $[1-(1-\frac{1}{q})^{|D|}]^{t_0} \leq [1-(1-\frac{1}{q})^d]^{t_0}$. We choose q and t_0 such that $[1-(1-\frac{1}{q})^d]^{t_0} = \frac{1-p}{n}$, 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 $C_d(x) = \frac{x}{-\log[1-(1-\frac{1}{x})^d]}$ for $x > 1$, we obtain $t \leq C_d(\log n + \log \frac{1}{1-p})$.

Theorem 10. *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.*

Remark 1. Our one-stage probabilistic pooling design is also transversal. This design never gets false negatives, while the probabilistic algorithms in Macula (1999a, 1999b) and Ngo and Du (2002) never get false positives. In Ngo and Du (2002), the algorithm identifies up to 9 positives from 18,918,900 items using 5460 tests, with success probability 98.5%. In our setting, $n = 18{,}918{,}900$, $d = 9$, and $p = 0.985$, by choosing $q = 14$ we require $C_d(q)(\log n + \log \frac{1}{1-p}) < 408$ tests, which is much fewer.

Remark 2. In contrast to the two-stage design in Section 4, 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 "positives" (actually 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 Section 4.

Remark 3. It is not hard to verify that, compared to the simplest design of assigning each entry of (binary) matrix M to be one independently with some constant probability, to achieve the same success probability our design requires noticeably fewer number of tests in general.

6. CONCLUSION

In this paper, we present new one- and two-stage pooling designs, together with new probabilistic pooling designs. The approach in this paper provides better designs then previous ones, and works for both error-free and error-tolerance scenarios. We end the paper off with the following remarks:

1. The constructions of pooling designs in Sections 4 and 5 can also be generalized to error-tolerance case, in a similar manner as in the construction of $(d; z)$-disjunct matrices in Section 3.2. We omit them here due to the similarities.
2. We do not give efficient constructions (i.e., in time polynomial in n and d) of the two-stage designs in Section 4. To the best of our knowledge, 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 De Bonis et al. (2005), the construction requires $\binom{n}{\frac{n}{2d}}$ time, and in Eppstein et al. (2007), the authors give existence proof as in the current paper. Although once such a design is found it can be used as many times as we want, efficient construction is an important issue.
3. The two-stage pooling design we present in Section 4 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 $d \rightarrow \infty$) of the information theoretic bound $d \log(n/d)$. Can two-stage algorithms do as good as fully adaptive algorithms, i.e., achieve a factor of asymptotically 1 of the information theoretic bound? Or, how good could it be?
4. Finally, finding more constructions of disjunct matrices with good properties (e.g., with even fewer number of rows for fixed n and d) is interesting, both theoretically and practically.

ACKNOWLEDGMENTS

Y. Cheng was supported in part by National Science Foundation of China under grant No. 60553001 and National Basic Research program of China under grant Nos. 2007CB807900 and 2007CB807901. D.-Z. Du was supported in part by National Science Foundation under grant No. CCF0621829.

REFERENCES

Balding, D.J., and Torney, D.C. 1996. Optimal pooling designs with error detection. *J. Combin. Theory Ser. A* 74, 131–140.

Berger, T., Mandell, J. W., and Subrahmanya, P. 2000. Maximally efficient two-stage group testing. *Biometrics* 56, 833–840.

Cheng, Y.X., and Du, D.Z. 2007. Efficient constructions of disjunct matrices with applications to DNA library screening. *J. Comput. Biol.* 14, 1208–1216.

De Bonis, A., Gasieniec, L., and Vaccaro, U. 2005. Optimal two-stage algorithms for group testing problems. *SIAM J. Comput.* 34, 1253–1270.

Dorfman, R. 1943. The detection of defective members of large populations. *Ann. Math. Stat.* 14, 436–440.

Du, D.Z., and Hwang, F.K. 2006. *Pooling Designs and Nonadaptive Group Testing: Important Tools for DNA Sequencing*. World Scientific, New York.

Du, D.Z., Hwang, F.K., Wu, W., and Znati T. 2006. New construction for transversal design. *J. Comput. Biol.* 13, 990–995.

D'yachkov, A.G., Macula, A.J., and Rykov, V.V. 2000. New constructions of superimposed codes. *IEEE Trans. Inform. Theory* 46, 284–290.

D'yachkov, A.G., Rykov, V.V., and Rashad A.M. 1989. Superimposed distance codes. *Probl. Contr. Inform. Theory* 18, 237–250.

Eppstein, D., Goodrich, M.T., and Hirschberg, D.S. 2007. Improved combinatorial group testing algorithms for real-world problem sizes. *SIAM J. Comput.* 36, 1360–1375.

Erdös, P., Frankl, P., and Füredi, Z. 1985. Families of finite sets in which no set is covered by the union of r others. *Israel J. Math.* 51, 79–89.

Farach, M., Kannan, S., Knill, E., et al. 1997. Group testing problems with sequences in experimental molecular biology. *Proc. Compression Complexity Seq.* 357–367.

Fu, H.L., and Hwang F.K. 2006. A novel use of t-packings to construct d-disjunct matrices. *Discrete Appl. Math.* 154, 1759–1762.

Hwang, F.K., and Sós, V.T. 1987. Non-adaptive hypergeometric group testing. *Studia Sci. Math. Hungar.* 22, 257–263.

Kautz, W.H., and Singleton, R.C. 1964. Nonrandom binary superimposed codes. *IEEE Trans. Inform. Theory* 10, 363–377.

Knill, E. 1995. Lower bounds for identifying subset members with subset queries. *Proc. 6th ACM-SIAM Symp. Discrete Algorithms*, 369–377.

Macula, A.J. 1996. A simple construction of d-disjunct matrices with certain constant weights. *Discrete Math.* 162, 311–312.

Macula, A.J. 1997. Error-correcting nonadaptive group testing with d^e-disjunct matrices. *Discrete Appl. Math.* 80, 217–222.

Macula, A.J. 1999a. Probabilistic nonadaptive group testing in the presence of errors and DNA library screening. *Ann. Comb.* 3, 61–69.

Macula, A. J. 1999b. Probabilistic nonadaptive and two-stage group testing with relatively small pools and DNA library screening. *J. Comb. Optim.* 2, 385–397.

Motwani, R., and Raghavan, P. 1995. *Randomized Algorithms*. Cambridge University Press, New York,.

Ngo, H.Q., and Du, D.Z. 2002. New constructions of non-adaptive and error-tolerance pooling designs. *Discrete Math.* 243, 161–170.

Park, H., Wu, W., Liu, Z., et al. 2003. DNA screening, pooling design and simplicial complex. *J. Comb. Optim.* 7, 389–394.

Address reprint requests to:
Dr. Yongxi Cheng
Department of Computer Science
Tsinghua University
Beijing 100084, China

E-mail: cyx@mails.tsinghua.edu.cn