1
Introduction

1.1 Group Testing

The theory of group testing has been well developed since 1943. Its combinatorial branch, the so-called *combinatorial group testing*, has been flourishing (see [1] for a general reference) due to its many applications to blood testing, chemical leak testing, electric shorting detection, codes, multi-access channel communication and AIDS screening. Recently, it has also been found to be useful in molecular biology. However, as true in each previous application, this new application also raises new models and new problems such that the classical theory needs to be modified, generalized or simply reinovated to handle them. This book is an attempt to contribute to the theory of group testing oriented towards its application to molecular biology.

We first give a brief description of the basic model of combinatorial group testing. There are n items each can be either *positive* (used to be called *defective*) or *negative* (used to be called *good*), while the number of positive items is upper bounded by an integer d. The problem is to identify all positive items. The tool of identification is the so-called *group tests*, while a group test is applicable to an arbitrary subset of items with two possible outcomes; a *negative outcome* indicates that all items in the subset are negative; a *positive outcome* indicates otherwise. The goal is to minimize the number of such tests in identifying the positive items. We will refer to this model as the $(n, \bar{d})$ model while the (n, d) model signifies that d is the exact number of positive items. It was proved [?] that the minimum number of tests differ by at most 1 between these two models, justifying the frequent use of the (n, d) model due to its mathematical simplicity.

There are two general types of group testing algorithms, sequential and nonadaptive. A *sequential* algorithm conducts the tests one by one and allow a later test to use the outcomes of all previous tests. A *nonadaptive* algorithm specifies all tests simultaneously, thus forbiding using the outcome information of one test to design another test. Sequential algorithms require fewer number of tests in general, since extra information allows for more efficient test designs. Nonadaptive algorithms can conduct all tests simultaneously, thus saving the time for testing if not the number of tests.

Historically, the main goal of group testing is to minimize the number of tests. Therefore sequential algorithms dominate the literature. Note that for the (n, d) model, the information-theoretical lower bound (heretofore referred to as the *information bound*) is

$$\lceil \log_2 \binom{n}{d} \rceil \sim d \log_2(n/d).$$

Since we can use the binary splitting algorithm to identify a positive item in at most $\lceil \log_2 n \rceil$ tests, $d\lceil \log_2 n \rceil$ tests suffices for the (n, d) model. By the closeness of $d\lceil \log_2 n \rceil$ with the information bound, we can say the (n, d) model is practically solved. On the other hand, the determination of exact optimal (n, d) algorithm is very difficult. Let $t(n, d)$ denote the minimum number of tests. $t(n, 1)$ is known to match the information bound, but $t(n, d)$ is not completely determined for any $d \geq 2$.

In the application to molecular biology, while it is still important to minimize the number of tests, two other goals emerge. In molecular biology, a group test corresponds to a PCR experiment (to be explained later) which could take several hours, where the items corresponding to the set of molecules under study could be in the tens of thousands. In some biological problem we have to identify different types (in thousands) of defectives while each type requires a separate group testing procedure, thus performing the tests sequentially, is impractical though it requires fewer tests in general. The focus then is *nonadaptive group testing algorithms*, also called *pooling designs* in the molecular biology literature, in which all tests are performed simultaneously (thus the information on the outcome of one test cannot benifit the selection of items in another test). Sometimes a pooling design cannot be found or consumes too many tests, then one has to settle for 2-stage or s-stage design for small s. In some other problems, sequential procedure can still be used, but the total time needed to identify the positive items must be considered along with the total number of tests.

Biological experients are known to be unreliable. So the second goal is to control the experimental error which has seldom been studied in the classical group testing literature. This can be done in two ways: The first is to build in error-tolerance in the pooling design so that even though errors occur, the positive items can still be correctly identified; the second is the ability to estimate the effect of errors and the probability of their occurance.

It is harder to construct nonadaptive group testing algorithms than sequential ones, For given n, often we do not know whether there exists a pooling design not exceeding t pools (tests). For that reason random pooling designs have been proposed which exist for all n and all t. Although random pooling designs are surprisingly efficient, they do not guarantee the identification of all positive items. Thus it is important to be able to estimate the probability of a given positive item not being identified, and the probability of all positive items being identified.

Group testing has been generalized to *group testing in graph* where an underlying graph G is given along with a set D of positive edges. The problem is to identify D by using a minimum number of edge-tests. An *edge-test* can be performed on any set S of vertices with two possible outcomes: *positive* if the induced subgraph of S contains an edge and *negative* if not. The notion of group testing on graph can be readily extended to hypergraph H. The classical group testing is a special case of group testing on hypergraph, where H is complete and each edge consists of a single vertex. It is well known [1] that there exists a one-one mapping between algorithms for the classical group testing with d positive items and algorithms for the group testing on hypergraph when each edge has d vertices and D consists of a single edge. We will see that some of the biological applications fall into this group testing on hypergraph framework.

Group testing has been extended to graph testing. A graph testing model has three parameters $(G, I(G^*), P)$ where G is a given graph , $I(G^*)$ is what we know about the hidden subgraph G^* to be identified, and P is what a test needs to contain to have a positive outcome. (If $I(G^*) = \emptyset$, then we can abbreviate to (G, P).) Typically, a test is a subset $S \subseteq V(G)$ which gives a positive outcome if and only if the induced subgraph $G(S)$ contains P, and G^* is obtained by deleting all elements which are known in priori (from biological knowledge) to be not in G^*. For example, the (n, d) group testing model can also be interpreted as a $(K_n, d$vertices, any positive vertex) model. The reason that G is always a complete graph in group testing is because any vertex known to be not in G^* does not need to be considered in G. Also, note that this graph representation of the group testing model is superfluous since we only deal with the vertices in K_n and in $G(S)$.

The second special case of graph testing is *edge-testing*. A typical model is $(G, I(G^*), \text{any edge in } E(G^*))$. Since the P part is always the same in edge-testing, we will omit it to save space. In particular, if the only information about G^* is that it has d (or at most d) edges, we write (G, d) (or $(G, \bar{d})$ or simply (G) if d is unknown. Such a model was first studied by Chang and Hwang [?] in the special case $(K_{m,n}, 1)$, where $K_{m,n}$ is the complete bipartite graph with m and n vertices, respectively, and an edge e is unknown positive. They proved that $t(K_{m,n}, 1) = \lceil \log_2 n \rceil$ (the information bound). Aihner [?] generalized $K_{m,n}$ to G and conjectured that $t(G, 1)$ differs from the information bound $\lceil |E| \rceil$ only by a constant. Du and Hwang [1] further conjectured that the constant is 1, which was proved by Damaschke [?].

We can generalize the above two model to the $(G, I(G^*), K_k \in G)$ model (the above two models correspond to $k = 1, 2$). But then G cannot represent all available information in $I(G^*)$. For example, suppose triangles *abd*, *bce*, *caf* are possibly in G^* but not triangle *abc*. Then G^* needs to include the first three triangles, hence to include edges *ab*, *bc*, *ca*. Consequently G also contains triangle *abc*, not reflecting our priori knowledge that triangle *abc* is not in G^*.

To remedy the ambiguity, the hypergraph H is used to replace G. Now H can contain the three huperedges *abd*, *bce*, *caf*, but not the hyperedge *abc*. Note that the hypergraph can also accomodate the case that the hyperedges do not have the same number of vertices.

Surprisingly, the (n, d) group testing model can also be represented as a $(K_n^d, 1)$ edge-testing model where K_n^d is the *complete* hypergraph with n vertices and $\binom{n}{d}$ edges each having d vertices. To see this, note that a test on a subset $S \subseteq V$ in the (n, d) model has two outcomes: either S contains no vertex in K_d , or the opposite. On the other hand, a test on $\bar{S}$ (the complementary set of S) in the $(K_n^d, 1)$ model also has two outcomes: either $\bar{S}$ contains the hyperedge e, which is equivalent to K_d, or the opposite. But, $\bar{S}$ contains e if and only if S contains no vertex of K_d. So, testing S in the first model is equivalent to testing $\bar{S}$ in the second model. Note that this coversion from G to H does not work if P is not K_d, for example, a star, since a subset (a hyperedge) does not reveal the ceter of as star.

Triesch [**?**] extended Dansschke's result to hypergraph; thus, effectively extending group testing from $G = K_n$ to general G, i.e., allowing us to specify which set of d vertices are candidates of G^*.

Anothe major generalization is to consider the d positive edges case. This represents a total breaking from a group testing model since the correspondence between group testing and graph testing is only for $d = 1$. Johnson [**?**] proved a conjecture of Du and Hwang [1] that $t(G, d)$ differs from the information bound $d \log_2 \lceil |E(G)|/d \rceil$ only by dc. Recently, Chen and Hwang [**?**] further extended this result to hypergraphs. In particular, they did it for the (H) model without assuming the knoeledge of G^* as all other above mentioned results do.

For the edge-testing model, Grebinski and Kucherov [**?**] first studied the case when there is a structural information about G^*. They consider the case that G^* is a hamiltonian cycle. Later, Alon, Beizel, Kasif and Sudarov [**?**], and Hwang and Lin [**?**] considered the case that G^* is a complete matching. Grebinski and Kucherov also considered the more general case when G^* is a graph of maximum degree Δ.

1.2 Nonadaptive Group Testing

A nonadaptive group testing scheme can be represented as a 0-1 matrix $M = (m_{ij})$ where columns are labeled by items and rows by tests. Thus m_{ij} signifies that test i contains item j. Sometimes it is more convenient to view a 0-1 column C_j as the subset $\{i \mid m_{ij} = 1\}$. Then we can talk about the union of several columns, which is nothing but the boolean sum of the corresponding 0-1 columns. Similarly, we can view a row as a subset of the column indices where its i-entries lie.

Given a set D of positive items, the outcomes of the tests can also be represented as a 0-1 vector which 0 indicates a negative outcome, i.e., all items in the test are negative, and 1 indicates a positive outcome, i.e., the test set contains at least one positive item. Note that the outcome vector $V(D)$ for given D is simply the union of columns in D.

A minimum requirement for M to be able to identify D is that $V(D) \neq V(D')$ for $D \neq D'$. A matrix with this property is called *$\bar{d}$-separable* if $|D| \leq d$ is assumed, and *d-separable* if $|D| = d$ is assumed. Although in theory, the vector $V(D)$ uniquely determines D, the actual decoding can be messy. Thus one can trade off stronger requirement for an easy decoding. M is called *d-disjunct* if no column is contained in the union of any d columns. Let $V(D)$ denote the outcome vector. Then the decoding for a column C is $C \in D$ if $C \subseteq V(D)$.

Let M be a d-separable, or $\bar{d}$-separable or d-disjunct matrix. Suppose a row R is contained in another row R'. Then we can replace R' by $R' \setminus R$ without losing any information (and may gain some) since R being negative implies testing R' is same as testing $R' \setminus R$, where R being positive implies testing R' provides no further information. Therefore the only case R implies testing R can contain a single 1-entry is that the column C incident to R contains no other 1-entry. We call R an isolated row and C an isolated column. If such a pair exists, we learn the state of C immediately, and can discard C and R from M since they have no effect on the testing of other columns except d needs to be replaced by $d - 1$ if c is positive. We assume throughout the book that M contains no isolated row or column unless specified otherwise.

We give examples of these matrices with $d = 1$.

1-separable. All we need us that columns are distinct. Let B_t denote the binary matrix consisting of all 2^t distinct t-dimensional vectors. Then for $n \leq 2^t$, any n columns constitute a 1-separable matrix for n items. The decoding is trivial, i.e., $C \in D$ if $C = V(D)$. For example, B_3 is

$$\begin{pmatrix} 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix}$$

The *weight* of a column is the number of 1's in it. Suppose there is an upper bound $\bar{w}$ of column weight. Consider the submatrix B_t^w consisting of all columns with weight $\leq \bar{w}$, there are

$$\sum_{i=0}^{\bar{w}} \binom{t}{i}$$

columns. For n not exceeding this number, any n columns constitute a 1-separable natrix satisfying $\bar{w}$.

Suppose there is an upper bound k of group size. We demonstrate a construction of a special case $n = g^{x+1}$ and $k = g^x$. The test are divided into groups where each group has $g - 1$ tests. In the first group the n items are partitioned into disjoint k-subsets, and each k-subset except the last constitutes a test. In the second group, the n items are re-partitioned into disjoint k-subsets such that each pair of k-subsets, one from the first group and one from the second group, intersect in exactly g^{x-1} items. Again, test every k-subset except the last. In general, in the ith group, $2 \leq i \leq x$, we require each k-subset to intersect every k-subset in previous groups in exactly g^{x-1} items. The resulting matrix is 1-separable with $(g-1)x$ tests of size k. For example, for $n = 27$ and $k = 9$, the matrix is

$$\begin{pmatrix}
1&1&1&1&1&1&1&1&1&&&&&&&&&&&&&&&&&& \\
&&&&&&&&&1&1&1&1&1&1&1&1&1&&&&&&&&& \\
1&1&1&&&&&&&1&1&1&&&&&&&1&1&1&&&&&& \\
&&&1&1&1&&&&&&&1&1&1&&&&&&&1&1&1&&& \\
1&&&1&&&1&&&1&&&1&&&1&&&1&&&1&&&1&& \\
&1&&&1&&&1&&&1&&&1&&&1&&&1&&&1&&&1&
\end{pmatrix}$$

For general n and k, the construction principle is the same except the partition would not be so balanced. In the formula for the number of tests, g will be replaced by $\lceil n/k \rceil$ and x by

$$\lceil \log_{\lceil n/k \rceil} n \rceil = \lceil \frac{\log n}{log \lceil n/k \rceil}.$$

$\bar{1}$-separable. The only difference from 1-separable is that the 0-vector must be deleted from M to be reserved for the outcome $D = \emptyset$.

1-disjunct. Spencer [**?**] proved that for given t of tests, the maximum number n of items in a 1-disjunct matrix is

$$\binom{t}{\lceil t/2 \rceil},$$

i.e., the matrix consists of all $lceilt/2\rceil$-subsets of the set $\{1, ..., t\}$. For n not exceeding that number, any n columns constitute a 1-disjunct matrix for n items. For $t = 5$, the maximum matrix is

$$\begin{pmatrix}
1&1&1&1&0&0&0&0&0&0 \\
1&0&0&0&1&1&1&0&0&0 \\
0&1&0&0&1&0&0&1&1&0 \\
0&0&1&0&0&1&0&1&0&1 \\
0&0&0&1&0&0&1&0&1&1
\end{pmatrix}$$

Suppose $\bar{w}$ is an upper bound of column weight. Parnes and Srinavasan [**?**] proved that the maximum number of columns is $\binom{t}{\bar{w}}$.

Suppose $\bar{k}$ is an upper bound of test size. Cai [**?**] proved that for $\bar{k} \leq \sqrt{2n}$, the minimum number of rows for given n is $\lceil 2n/\bar{k} \rceil$ obtained by the incidence matrix of a graph with $\lceil 2n/\bar{k} \rceil$ vertices, n edges and every vertex except one having $\bar{k}$ (easy to construct such a graph). For example, for $n = 12$ and $k = 3$, we have the graph in

Fig. **??** and the matrix as follows:

$$\begin{array}{c|cccccccccccc}
 & 12 & 13 & 14 & 23 & 24 & 34 & 56 & 57 & 58 & 67 & 68 & 78 \\
\hline
1 & 1 & 1 & 1 & & & & & & & & & \\
2 & 1 & & 1 & 1 & & & & & & & & \\
3 & & 1 & & 1 & & 1 & & & & & & \\
4 & & & 1 & & 1 & 1 & & & & & & \\
5 & & & & & & & 1 & 1 & 1 & & & \\
6 & & & & & & & 1 & & & 1 & 1 & \\
7 & & & & & & & & 1 & & 1 & & 1 \\
8 & & & & & & & & & & & 1 & 1
\end{array}$$

Note that $k = \binom{t-1}{\lfloor t/2 \rfloor - 1}$ for each row in Spencer's result. Therefore for $k \geq \binom{t-1}{\lfloor t/2 \rfloor - 1}$, Spencer's result remains to be the best construction. The best construction for the range $\sqrt{2n} < k < \binom{t-1}{\lfloor t/2 \rfloor - 1}$ remains an open problem.

1.3 Applications

A DNA sequence has four types of nucleotides, A, C, G, T as alphabets. In fact the normal state of a DNA sequence consists of two dual strains, where one strain can be obtained from the other by the mapping $A \to T$, $C \to G$, $G \to C$, $T \to A$. The two strains can be separated under proper heating, but they have a tendency to bind to each other when put together. This binding tendency is known as *hybridization*.

We can use hybridization to find out whether a DNA strain T contains a specific substrain S. Let S^{-1} denote the dual strain of S. Mix T and S^{-1}. If T contains S, then S^{-1} will hybridize with S. This hybridization effect is magnified by the PCR (polymer chain reaction) technique so that it becomes observable or measurable. A PCR essentially breaks up the hybridized pair to allow each half to hybridize with other copies of S and S^{-1} (assuming enough copies are provided), and the (break-up, hybridize) cycle reiterates until a huge PCR product is obtained. Hence the observation of a PCR product when S^{-1} is used as the *probe* implies the existence of S in T.

A PCR product can also grow in absence of S^{-1}. A *primer* is a short DNA sequence either preceding or succeeding the DNA sequence of interest, like S. Suppose we mix T with the two primers sandwiching S and S is not too long. Then a PCR product will grow whose length corresponds to the length of S. This length factor is critically used in the multiplex PCR.

In a multiplex PCR, many primers are used simultaneously in one test. Suppose the primers are ordered according to their positions in T. Then two primers are *adjacent* if there is no other primer between them, i.e., they are separated by one gap. In a multiplex test, a PCR product is obtained for each pair of adjacent primers. In theory, by counting the number of PCR products of different lengths, we know the number of pairs of adjacent primers. In practice, this information is not entirely accurate due to experimental errors and also to the possibilities that two PCR products have similar lengths.

For our purpose we classify PCR into two categories. The *Duality* PCR, using S^{-1} to catch S basically tests for the containment" relation. The other, the *primer* PCR, tests the adjacency relation between primers. Two primers are adjacent if there is no other primer between them. Using the primer PCR then we have a choice of whether to use the regular group testing model with only "yes" or "no" outcome, or the quantitative model which specifies the number of adjacent pairs of primers.

We now mention some specific applications of pooling design in molecular biology.

(i) Physical mapping. A clone library stores the DNA sequence, which we refer to as the *target sequence*, of a large molecule, f.i., a chromosone. Typically, the large molecule has to be broken into manageable sizes for easy storage, multiplication and study. The broken pieces are called *clones*. The cutting can be done either by a restriction enzyme which cuts whenever a given short DNA sequence, about six neucleotides and varying with enzymes, is encontered, or a random cutting called *shotgunning* which is effected by applying heat or shock. Regardless the cutting methods, the important things are: (a) one cannot control lengths of the cuts, (b) one can get different cuttings through using different enzymes or through shotgunning, (c) the order of the clones is lost through a cutting.

When we want to study the target sequence in its entirety, we have to reconstruct the clone sequence. This is a problem since even if we could read the clones, which we can't if the length of a clone is over 700, we would still be at a loss which clone should follow which. The prevailing method is to cut the target sequence several times into clones, each time with a different restriction enzyme or shotgunning to obtain different cuttings. Then we use the information provided by overlapping clones to reconstruct the target sequence.

As we said, we cannot read a clone in general. Thus we need to identify certain characteristics of a clone. One such characteristic is the possession of STSs (sequence-tagged sites) where each STS is a short DNA sequence (about 200 nucleotides), which appears uniquely in the target sequence. For each clone we identify the subset of STSs it contains. If two clones share an STS, then they must be adjacent in the target sequence. Note that when the clones are sequenced, the set of STSs making the clones also becomes a sequence marking the target sequence, which is usually referred to as a *physical mapping* of the target sequence. Fig. 1.1 illustrates the idea of using STSs to sequence the clones.

Suppose the target sequence has seven STSs: A, B, C, D, E, F, G. There are three cuttings resulting in a total of ten clones. Note that the second cutting cuts right into E, so E does not appear in any clone in the second cutting. Delete clones whose STS-set is a subset of another clone, and retain only one among those having the same STS-set. Draw an overlapping graph with the surviving clones as vertices and an edge between two clones if there is an overlap. A hamiltonian path of the

```
target Sequence   A       B       C       D       E       F   G
first cut         A       B   |   C       D   |   E       F   G
second cut        A   |   B       C   |   D       |       F   G
third cut         A       B       |       D       E   |   F   G
```

overlapping graph

$$\begin{tikzcd}
AB \arrow[r, dash] & BC \arrow[r, dash] & CD \arrow[d, dash] \\
 & EFG \arrow[r, dash] & DE
\end{tikzcd}$$

Figure 1.1: Using STSs to sequence

surviving clones then suggests a possible target sequence. Note that there may exist several hamiltonian path or none, and one hamiltonian path may suggests several target sequences.

Before this, we need to identify the STS-set for each clone. We do this by one STS at a time. Suppose we are doing A. Then we use A^{-1} as the probe in a pooling design in which a clone is positive if it contains A, and negative if not. Note that the number of positive clones cannot exceed the number of cutting (assuming no error), but can be less since an STS may be cut through in a cuting. Therefore we can set d to be the number of cutting. Also note that thousands of probes must be used to obtain enough information on the clones. Thus sequential procedures are impractical.

(ii) Contig sequencing

After we use the overlapping information to sequence the clones, it is still possible no hamiltonian path exists. Typically, the clones are sequence into a set of relatively long molecules, called *contigs*, separated by gaps. For example, in Fig.1.2.1, if the first cutting also cuts into E, then we will have two contigs, one consisting of $ABCDE$ and the other FG, separated by the gap from the end of E to the start of F. Multiplex PCR can be used to sequence the set of contigs. Suppose there are n contigs. The set of primers consists of the two end-subsequences (about 20 nucleotides long) of each contig. We call two contigs, or two primers, *adjacent* if ther are separated by a single gap.

For a pooling test we mix a subset of primers with the target sequence. If the mixure contains a pair of adjacent primers, then a PCR product will be produced whose length corresponds to the length of the gap between them. If the mixure contains several such pairs, then each pair yields a PCR product, and we can distingush them by their lengths in theory. This problem can be translated into a "group testing in graph" problem where each primer is a vertex, and each pair of adjacent primers is a positive edge (alternatively, one can also treat each contig as a vertex, and each pair of adjacent contigs a positive edge), while the underlying graph G is the complete graph. However, we do have the extra information that the subgraph consisting of the positive edges is a hamiltonian path (when contig are the vertices) and a complete matching (when primers are the vertices). Such a problem is also referred to as "learning a hidden subgraph".

Since the number of contigs is usually not too large and only one pooling is needed, sequential procedures can be considered for the contig sequencing problem.

(iii) Locating STS among ordered clones

Suppose the clones are ordered and we want to identify all clones containing a given STS. Due to the uniqueness of an STS in the tag sequence, clones containing the STS must be consecutively ordered, and their number is bounded by the number of cuttings generating these clones. This is a new group testing problem in which not only theobjects are linearly ordered, but also the positive objects are consecutive.

(iv) Gene detection

A human gene is often split into several disjoint parts, called *exons*, separated by gaps, called *inxons*. The DNA sequence of a gene, including all inxons, is first transcribed to RNA. Then the inxons are edited out before the corresponding proteins are made. We can backtrack this process to obtain the gene, called cDNA, with inxons edited out from the proteins. The problem is to identify the locations of the inxons in cDNA so to discover the composition of the target sequence (the original DNA sequence).

Primers are developed from the cDNA seqence. A test consists of a pair of primers, one forward and one backward, applying to both the cDNA and the DNA sequence. The test outcome is determined by the lengths of the PCR products yield in the cDNA interval (between the two primers) and the DNA interval. If the DNA length is longer than that of the cDNA, then existence of inxons in the DNA interval explains this difference. If there is no difference in length, then likely there exists no inxon in the DNA. Strangly, the nonexistence of PCR product in the DNA interval also suggests the existence of inxons and probably more than one, since the presence of internal inxons may interrupt the PCR product of the original interval.

The boundaries between exons and inxons are marked by some short, special combinations of nucleotides, but these combinations can also appear elsewhere accidentally. Suppose there are $n-1$ such combinations in the cDNA labeled by 1 to $n-1$ from the start to the end. We also add two labels 0 and n for the start and the end of the cDNA. Denote the interval between $i-1$ and i by I_i for $1 \leq i \leq n$. Then each inxon must occupy one of the n intervals. Call an interval positive if it contains an inxon. Then the problem is to identify all positive intervals. A group test here is a multiplex PCR test on any set S of consecutive I_i in the cDNA along with the target sequence. Let S' be the subsequence of the target sequence corresponding to S. Then mixing S with the target sequence has the effect of comparing S and S'. Clearly, S contains no inxon. So the comparison shows a difference, namely, S' has a PCR product, if and only if S' contains an inxon. Thus, a group testing algorithm, with tests restricted to sets of consecutive items, can be used to identify all positive I_i's.

Since we only need to do one probe, sequential procedure can be considered.

(v) The complex model

Suppose the items are molecules. Then a biological function may depend on the presence of a subset of molecules, called a *complex.* Therefore, the notion of a positive molecule is transformed to a *positive complex.* The complex model can also be easily fitted into the "group testing on hypergraph H" framework. Treating each molecule as a vertex, then a complex is simply an edge in H.

Thus the complex model intensifies the need of studying group testing on hypergraphs, where the classical group testing is the special class when the hypergraph is complete and each edge has a constant number of vertices. We will see that important breakthroughs have been made on this model which not only meets the real biological need, but also advanced the theory of group testing.

Although only one probing is required for this problem and thus sequential procedure can be considered, the number of tests is huge and time-consuming. Nonadaptive or k-round procedures will still be desired.

1.4 Two Simple Applications

The applications of group testing to problems (iii) and (iv) in the last section are relatively straightforward, and will be discussed in this section.

First, the gene detection problem. A typic pooling design may consist of several rounds of testing. Suppose we estimate the number of inxons to be k. The cDNA sequence is first partitioned into k equally spaced intervals. Primers at both ends of these intervals are developped, and each such pair is tested both DNA and cDNA. An interval is positive if there is a difference in the PCR product between cDNA and DNA. In the second round, we further divide each positive interval into sub-intervals to reduce the length of a positive interval. If the positive subintervals identified in the second round are still considered too long, further division are taken. When to stop depends on a balance of several conflicting goals: small positive intervals, small number of rounds, small number of tests and small number of primers (the same primer can be used in multi-rounds).

Next, the problem to identify all up-to-d consecutive positive clones in an ordered set of n clones. Colbourne [] gave a sequential algorithm with $\log n + \log d + c$ tests where t is a constant. We make a slight improvement by eliminating c.

Use the halving procedure to identify the first positive clone in $\log n$ tests. Suppose it is C_i. Then the only uncertainty is about clones $C_{i+1}, ..., C_{i+d-1}$. Use the halving procedure again, but in a reverse order, to identify the last positive clone, if any, in $\log d$ tests. If it does not exist, then C_i is the only positive clone. If C_{i+j} is it, then $\{C_i, ..., C_{i+j}\}$ is the set of positive clones. Colbourne also gave a nonadaptive algorithm. First, we introduce the notion of a Gray code G_n of order n which is a sequence of 2^n binary n-vectors such that two consecutive vectors differ only in one bit. G_1, G_2, G_3 are given below:

$$\begin{array}{ccc} 01 & \begin{array}{c}0011\\0110\end{array} & \begin{array}{c}00001111\\00111100\\01100110\end{array} \\ G_1 & G_2 & G_3 \end{array}$$

In general, G_n can be obtained from G_{n-1} by taking two copies of G_{n-1}, reversing the second copy and adding a first row which has 0s in the first copy and 1s in the second copy as illustrated below:

$$\begin{array}{c|cc} & & 0000000011111111 \\ 00001111 & 11110000 & 0000111111110000 \\ 00111100 & 00111100 & 0011110000111100 \\ 01100110 & 01100110 & 0110011001100110 \\ G_3 & G_3^{-1} & G_4 \end{array}$$

A matrix is called 2-consecutive if it can be identify all positive clones if there are exactly two and consecutive; it is called $\bar{2}$-consecutive if "exactly two" is replaced by "up to two". We now show how to construct 2-consecutive and $\bar{2}$-consecutive matrices from G_n.

2-Consecutive. Note that for any two consecutive columns, one is an subset of the other. Therefore the union of two consecutive columns always equals to the larger column. With the outcome vector V, we first identify column $C_i = V$. For $d = 2$, thus the other positive column is either C_{i-1} or C_{i+1}. So it suffices to add two rows such that $C_{i-1} \neq C_{i+1}$ in those two rows, and C_i does not contain both C_{i-1} and C_{i+1}. The following is such a matrix for eight items.

$$\begin{array}{cccccccc} 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \end{array}$$

$\bar{2}$-Consecutive. Then C_i itself can be the positive set. We need to add three rows to differentiate the three cases. One way is to let the i^{th} such row, $i = 1, 2$, have 1s in column j if $j \equiv i \pmod 3$. An example for $n = 8$ is

$$\begin{array}{cccccccc} 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0*1 & 0 & 0 & 1 & 0 & \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \end{array}$$

Note that the $D = \emptyset$ case is also taken care of since the 0-vector is not in the matrix.

If $d = 2$, use a $\bar{2}$-consecutive matrix to identify the positive clones. If $d > 2$, then partition the n clones consequtively into groups of $d - 1$ (the last group can have fewer). Call a group positive if it contains a positive clone. Then at most two consecutive groups can be positive. Treat each group as an objrect and use a $\bar{2}$-consecutive matrix to identify the positive group(s) if any, in at most

$$\left\lceil \log \left(\frac{n}{d-1} \right) \right\rceil + 3 \text{ tests.}$$

Colbourne proposed to add $2(d - 1)$ rows to identify which clones in the positive group(s) are positive. But actually, adding $d+1$ rows suffices where row i, $0 \leq i \leq d$, has 1 in column j if $j \equiv i \pmod{d+1}$.

1.5 Pooling Designs and Mathematics

Group testing has been traditionally associated with combinatorics and probability as the names "combinatorial group testing" and "probabilistic group testing" suggested. Its relation with "algorithm theory" is also obvious.

A pooling design is a combinatorial design whose blocks are the pools and whose items are the clones (or molecules), except that the requirements are different from the usual "balanced appearances" type. Nevertheless, the whole spectrum of combinatorial designs has lent its strength to the construction of pooling designs. With the close relation to combinatorial designs comes the unavoidable relation to coding theory. It must be said that the relation is not just to put known results in combinatorial design theory and coding theory in good use, but also that pooling designs suggest a new type of designs and also a new type of codes know as "superimposed codes".

As combinatorial designs draw heavily from algebraic and geometric structures, so do pooling designs. For example, lattices and simplicial complexes under very broad conditions can be transformed to pool designs. It is also well-known that the extremal set theory is closely related to nonadaptive group testing, hence pooling designs.

Finally, we point out two areas which have only played minor role in traditional group testing are of utmrst importance in pooling designs, and in the meantime offer great challenges. Group testing had been extended to graphs, but no real motivation to study that problem was known. Now we know the complex model corresponds exactly to group testing on hypergraph, while the contig sequencing problem and the gene detection problem correspond to the case when we know some property of the hidden subgraph. Therefore an urgency to develop a theory of group testing on graph suddenly emerges.

Traditionally, combinatorial group testing has very little to do with probability theory. Since the construction of pooling design is so much harder than sequential group testing algorithms, the importance of random designs is increased. It turns out that the probability analysis of random designs is a difficult problem, and so far excact analysis have been obtained only for the simple model.

To summarize, we see that group testing and pooling designs not only have surprisingly many applications to various fields, but also borrows heavily from many mathematical branches, and in the meantimes, proposed new problems to these branches. The mutually beneficial relation has just begun to unreel and there is a lot of work to be done.

1.6 An Outline of the Book

We will first introduce deterministic constructions of pooling designs. Chapter 2 reviews and updates of nonadaptive group testing, which guarantee to identify all positive items if that number does not exceed d, and their error-tolerant session. Chapter 3 reviews some traditional deterministic construction and Chapter 4 introduces the new construction methods, the partial order method with algebraic structure, and the simplicial complex method with geometric structure.

Then we discuss the more practical random designs. An important criterion here is to evaluate various probabilities that items are not correctly identified. Chapter 5 reviews some basic random designs whose probabilities of nonidentification are all solved. The emphasis here is to obtain the formulas which can be computed fastest, a real concern due to the large number of items. The problem of determining the design parameters to minimize the nonidentification probabilities is still open in general.

Chapter 6 studies a random design with a more structured sample space. The hypothesis is that a sensible subspace is better than the original space to construct a random design as there has been some evidence to it. We choose the subset-containment design and the random distinct k-set (a basic design given in Chapter 5) for such comparison since the former is constructed on a subspace of the latter. Unfortunately, the probability analysis for random designs are difficult and unsolved in general except for those given in Chapter 5. We present some approximation formulas for the subset-containment design given in the literature. For one set of parameters, exact formulas are known and compared to the random k-set design. A surprising finding is that the previously mentioned hypothesis does not hold.

Chapter 7 studies the array design actually used in some clone libraries. This design also leads to some new theoretical questions in the graph-design theory.

The next three chapters each covers an application mentioned in Sec. 1-2. Chapter 8 studies the complex model, Chapter 9 studies the contig-sequencing problem, and Chapter 10 studies the gene detection problem.

Chapter 11 presents a generalization of pooling design where an item can be positive, negative or inhibitive, meaning its presence in a pool preempts a positive outcome. The problem is still to identify the positive items. Several multi-stage pooling designs have been given in the literature. We focus on the recently found (one-stage) pooling design, whose construction, interestingly, uses the same mathematical tools as the regular pool design.

References

[1] D.-Z. Du and F.K. Hwang, *Combinatorial Group Testing and Its Applications*, World Scientific, 1999.