

Introduction

We introduce the notion of group testing with its many industrial applications, and its recent application to various molecular biology screening designs, usually referred to as “pooling designs”. We also summarize several mathematical topics which play an important role in developing the theory of pooling designs.

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 [8] 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 renovated 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 specifies that d is the exact number of positive items. It was proved [11] that the minimum number of required 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 forbidding 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 permit to conduct all tests simultaneously, thus saving the time for testing if not the number of tests. Between the sequential and the nonadaptive algorithms, there are the s -stage algorithms for which all tests in a stage must be specified simultaneously, but the stages are sequential. In particular, an s -stage algorithm is said to be *trivial* if in the last stage every test contains only an individual item.

Historically, the main goal of group testing is to minimize the number of tests. Therefore sequential algorithms have dominated 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 suffice 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, a group testing algorithm is called a *pooling design* and the composition of each test a *pool*. 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*, in which all tests are performed simultaneously (thus the information on the outcome of one test cannot benefit the selection of items in another test), although sometimes a pooling design has to settle for 2-stage or s -stage design for small s . Occasionally, sequential procedures 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 experiments 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 occurrence.

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 extended to graph testing. A graph testing model has three parameters $(G, I(G^*), P)$ where G is a given graph with vertex-set $V(G)$ and edge-set $E(G)$, $I(G^*)$ is what we know about the hidden subgraph G^* to be identified, and P is the property of a test on a subgraph that is required to have a positive outcome. (If $I(G^*) = \emptyset$, then we can abbreviate to (G, P) .) Such a problem is also known as “*learning a hidden subgraph*”. Typically, a test on a subset G' gives a positive outcome if and only if G' contains P . For example, the (n, d) group testing model can also be interpreted as a $(K_n, |G^*|)$ is a set of d positive vertices, any positive vertex) model, where K_n is the complete graph with n vertices. We may call it the *vertex-testing* model. Of course, this graph representation of the group testing model is superfluous since we only deal with the vertices in K_n and in G' .

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 [5] 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 the unknown positive. They proved that $t(K_{m,n}, 1) = \lceil \log_2 n \rceil$ (the information bound). Aigner [2] generalized $K_{m,n}$ to G and conjectured that $t(G, 1)$ differs from the information bound $\log_2 |E(G)|$ only by a constant. Du and Hwang [8] further conjectured that the constant is 1, which was proved by Damaschke [7].

We can generalize the above two models to the $(G, \text{a } d\text{-set of } K_k, K_k \in G^*)$ model (the above two models correspond to $k = 1, 2$). For better representation, the hypergraph H is used to replace G . Then G^* is a d -set of $E(H)$. When P is any positive edge, we represent this model by $(H : d)$. Note that the hypergraph can also accommodate the case that the hyperedges do not have the same number of vertices. Triesch [21] extended Damaschke’s result to the $(H : 1)$ model.

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 conversion from G to H does not work if P is not K_d , for example, P is a star, since a subset (a hyperedge) does not reveal the center of a star.

The $(H : d)$ model represents a total breaking from the group testing model since the correspondence between group testing and graph testing is only for $(H : 1)$. Johann [12] proved a conjecture of Du and Hwang that $t(G, d)$ differs from the information bound $d \log_2 [|E(G)|/d]$ only by dc . Recently, Chen and Hwang [6] further extended this result to hypergraphs. In particular, they did it without assuming the knowledge of d as all other above mentioned results do.

For the edge-testing model, Grebinski and Kucherov [9] first studied the case when there is a structural information about G^* . They consider the case that G^* is a Hamiltonian cycle. Later, Alon, Beigel, Kasif, Rudth and Sudarov [1], and Hwang and Lin [10] 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 (or binary) matrix $M = (m_{ij})$ where columns are labeled by items and rows by tests. Thus m_{ij} specifies that test i contains item j . Sometimes it is more convenient to view a 0-1 column C_j as the incidence vector of subset $\{i \mid m_{ij} = 1\}$. Then we can talk about the union $U(s)$ of a set s of columns, which is nothing but the boolean sum of the corresponding 0-1 columns. Similarly, we can view a row as the incidence vector of a subset of the column indices where its 1-entries lie.

Given a set D of positive items, the outcomes of the tests can also be represented as a 0-1 vector where 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 can be represented by $U(D)$, the union of the columns in D .

A minimum requirement for M to be able to identify D is that $U(D) \neq U(D')$ 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 $U(D)$ uniquely determines D , the actual decoding can be messy. Thus one can trade off weaker requirement for an easy decoding. M is called d -disjunct if no column is contained in the union of any other d columns. Then the decoding for a column C is $C \in D$ if $C \subseteq U(D)$. For convenience, we use d -separating as a generic term for d -separable, \bar{d} -separable or d -disjunct.

Let M be a d -separating 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 the same as testing $R' \setminus R$, where R being positive implies testing R' provides no further information. Therefore, we may assume throughout the book that M contains no row properly contained by another row unless specified otherwise. Consequently, if a row R contains a single 1-entry, then the column C incident to R contains no other 1-entry. We call R an isolated row

$\bar{1}$ -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 [19] proved that for given t of tests, the maximum number n of items in a 1-disjunct matrix is

$$\binom{t}{\lfloor t/2 \rfloor},$$

i.e., the matrix consists of all $\lfloor t/2 \rfloor$ -subsets of the set $\{1, \dots, t\}$. 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. Schultz, Parnes and Srinivasan [17] proved that the maximum number of columns is $\binom{t}{\bar{w}}$.

Suppose k is an upper bound of test size. 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. For $n > k^2/2$, Cai [3] constructed a 1-disjunct matrix with $\lceil 2n/k \rceil$ rows obtained by the incidence matrix of a graph with $\lceil 2n/k \rceil$ vertices, n edges and every vertex except possibly one having degree k (it is easy to construct such a graph). For example, for $n = 12$ and $k = 3$, we have the graph in Fig. 1.1 and the matrix as

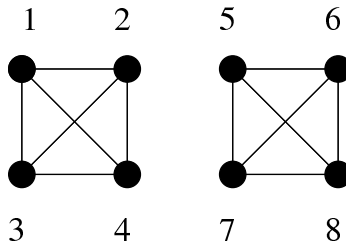


Figure 1.1: A 3-regular graph.

follows:

	12	13	14	23	24	34	56	57	58	67	68	78
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	1

For $k^2/2 \geq n \geq \binom{k}{2}$, Ramsey and Roberts [16] proved $t(d, n, k) = k + 1$. Finally, for $\binom{k}{2} > n \geq 2k$, Knudgen, Mubayi and Tetali [13] proved

$$\min\{w \mid n \leq \binom{w}{\lfloor wk/n \rfloor}\} \geq t(d, n, k) \geq \min\{w \mid n \leq \binom{w}{\lceil wk/n \rceil}\}.$$

1.3 Applications in Molecular Biology

An eukaryotic cell has a nucleus. The nucleus contains chromosomes which carry the genetic instructions for making living organism. Cells in different kinds of organisms contain different numbers of chromosomes. Each human cell contains 23 pairs of chromosomes. Each chromosome is a packed DNA and is 5,000 times shorter than the extended form of DNA (Fig. 1.2).

Each DNA consists of two chains, entwined forming a double spiral to form a DNA double helix. Each chain is a sequence of four types of nucleotides, A, C, G, T , and hence can be seen as a string of alphabets $\{A, C, G, T\}$. The two strings stay together in double helix with a duality relation where one string can be obtained from the other by the mapping $A \rightarrow T, C \rightarrow G, G \rightarrow C, T \rightarrow A$. The two strings 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 string T , called a *target sequence*, contains a specific substring S . Let S^{-1} denote the complementary strand 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 is essentially a technology to make copies of existing DNA pieces by using hybridization. A *primer* is a short DNA sequence either preceding or succeeding the DNA sequence S (S^{-1}) of interest. If S exists in T , the hybridization effect would make a primer of a S^{-1} to grow into a S^{-1} along S under action of enzyme. Heating is able to break up the hybridized pair to allow each half to grow with other copies of S and S^{-1} . This (break-up, hybridize) cycle reiterates until a huge PCR product is obtained (Fig. 1.3). Hence the observation of a PCR product when primers of S^{-1}

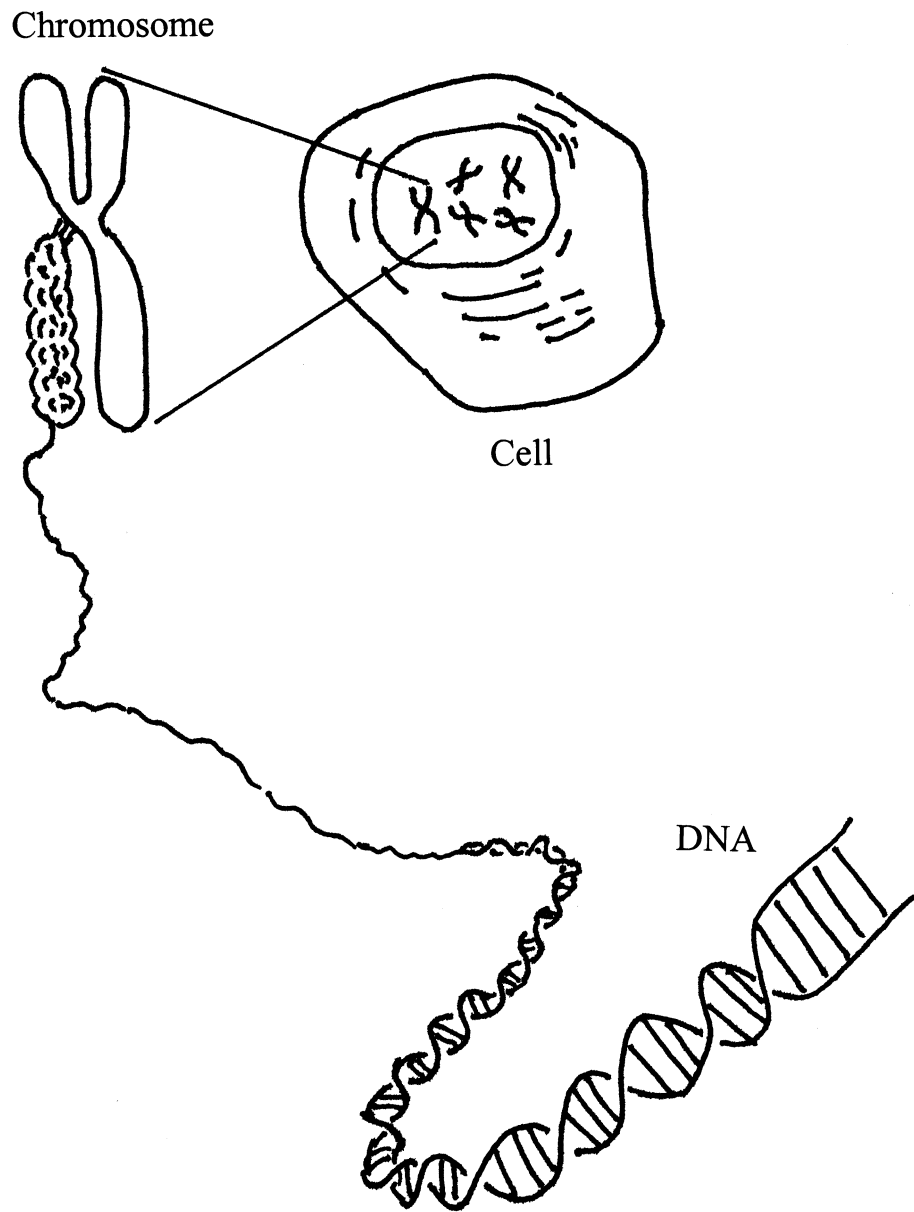


Figure 1.2: Each chromosome is a packed DNA.

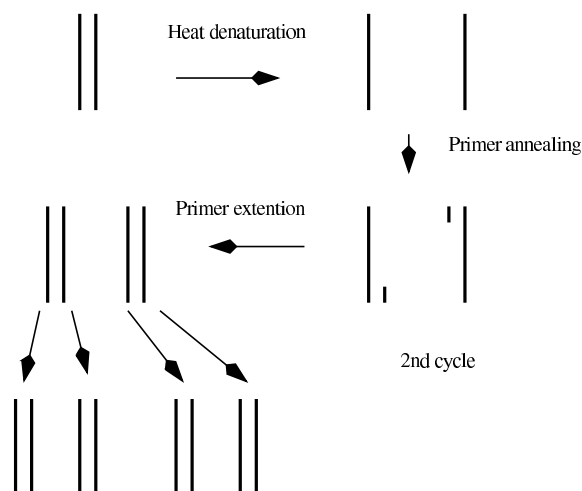


Figure 1.3: Polymer Chain Resaction (PCR).

are used as the *probe* implies the existence of S in T . In fact, suppose we mix T with the two primers sandwiching S^{-1} 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 set P of primers are ordered according to their positions in T (we do not know the order). Then two primers are *adjacent* if there is no other primer in P between them and they are not too far apart in T . 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 due 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. Using the multiplex PCR 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 designs in molecular biology.

(i) Physical mapping. A clone library stores the DNA sequence T of a large molecule, such as a chromosome. 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 nucleotides and varying with enzymes, is encountered, 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 several shotgunnings, (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 order of clones. 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 as to 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 marking the clones also becomes a sequence marking the target sequence, which is usually referred to as a *physical mapping* of the target sequence. Figure 1.4 illustrates the idea of using STSs to sequence the clones.

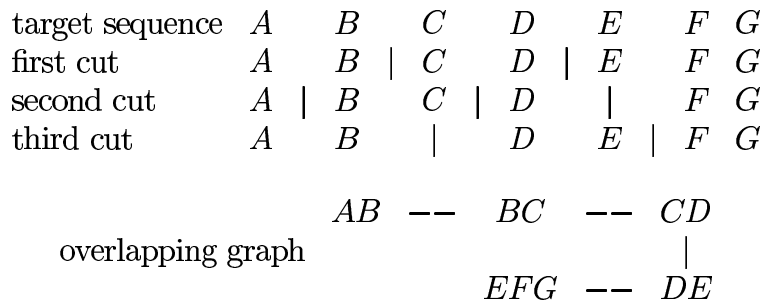


Figure 1.4: Using STSs to sequence.

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 surviving clones then suggests a possible target sequence. Note that there may exist several Hamiltonian paths or none, and one Hamiltonian path may suggest 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 cuttings (assuming no error), but can be less since an STS may be separated in a cutting or contaminated. Therefore we can set d to be the number of cuttings. 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 sequenced into a set of relatively long molecules, called *contigs*, separated by gaps. For example, in Fig. 1.4, 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 they are separated by a single gap.

For a pooling test we mix a subset of primers with the target sequence. If the mixture 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 mixture contains several such pairs, then each pair yields a PCR product, and we can distinguish them by their lengths in theory. This problem can be translated into an edge testing 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 contigs are the vertices) or a perfect matching (when primers are the vertices).

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 the objects are linearly ordered, but also the positive objects are consecutive.

(iv) Finding exon boundaries in cDNA

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

of the target sequence. Here the existence of an exon boundary is equivalent to the existence of an intron.

A solution to this problem was given in [22]. A suitable number of primers was developed from the cDNA sequence, "suitable" in the sense that the distance between each pair of adjacent primers satisfies the design specification. A probe is a pair of primers applying to both the cDNA interval (between the two primers) and the target sequence. The test outcome is obtained by comparing the lengths of the PCR products from the cDNA interval and the target sequence. If the latter is longer, it must be because the target sequence contains introns though their exact number and locations are unknown, while the cDNA obviously does not. Note that two primers too far apart cannot produce a PCR product. Hence the length of the cDNA interval for a test may be restricted.

Call an interval positive if it contains an intron. Then the problem is to identify all positive intervals, a group testing problem, except tests are restricted to sets of consecutive items. One would like to minimize both the number of rounds of testing and the number of distinct primers, since we need to do n probings, sequential procedure can take too much time.

(v) Protein-Protein Interaction

Protein-protein interactions are critical in many biological processes, such as the formation of macromolecular complexes and the transduction of signals in biological pathways. The interaction is usually between a bait protein and a prey protein. Therefore, to identify all protein-protein interactions, we are facing a group testing problem in bipartite graph with bait proteins as one vertex set and prey proteins as the other vertex set [20]. Interactions are edges between the two vertex sets.

(vi) Non-unique Probe Selection

To identify closely related virus subtypes in a biological sample, it is hard to find unique probes (i.e., each hybridizes only one target). In this situation, Schliep, Torney and Rahmann [18] suggested to use non-unique probes with group testing techniques. In the case that the number of targets in a sample is expected to be small, the number of probes can also be reduced. For example, on a data set of 285 rDNA sequences, they were able "to identify 660 sequences, a substantial improvement over a prior approach using unique probes which only identified 408 sequences."

(vii) 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. We will see that important breakthroughs have been made on this model which not only meets the real biological need, but also advances the theory of group testing.

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

1.4 Pooling Designs for 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, for the exon boundary problem, Xu *et al.* [22] suggested the following procedure. A typical pooling design for this problem may consist of several rounds of testing. Suppose we estimate the number of inxons to be k . The cDNA sequence is first partitioned into n equally spaced intervals. Primers at both ends of these intervals are developed. A \bar{d} -separable matrix is used to identify the positive intervals. 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 divisions 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). Note that the multi-round structure is a balance between a sequential procedure, which uses too many rounds, and a 1-round procedure, which uses too many tests and primers.

Next comes the problem to identify all up-to- d consecutive positive clones in an ordered set of n clones. Colbourne [4] 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 $C_{i+1}, \dots, C_{i+d-1}$. 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 the last positive clone, then $\{C_i, \dots, C_{i+j}\}$ 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:

		00001111
01	0011	00111100
	0110	01100110
G_1	G_2	G_3

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:

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

A matrix is called 2-consecutive if it can identify all positive clones provided there are exactly two which are 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 a 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 $d = 2$, 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} . One way is to let the i th such row, $i = 1, 2$, has 1s in column j if and only if $j \equiv i \pmod{4}$. The following is such a matrix for eight items.

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

$\bar{2}$ -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

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

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 consecutively 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 object 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 , $1 \leq i \leq d + 1$, has 1 in column j if $j \equiv i \pmod{d + 1}$.

Let $n(d_{cons}, t)$ denote the maximum n such that there exists a $t \times n$ d -consecutive matrix. Similarly, $n(\bar{d}_{cons}, t)$ is defined. Since in the $\bar{2}$ -consecutive problem, n columns generate $2n - 1$ sample points whose outcome vector must be all distinct in the $(\bar{2}_{cons}, t)$ space. Hence $n \leq 2^{t-1}$. Müller and Jimbo [15] showed that this upper bound can always be achieved except for $t = 3$ by a recursive construction. Let $M_t = (C_1, C_2, \dots, C_n)$, $n = 2^{t-1}$, denote such a matrix of order t . Then M_2, M_4, M_5 are given explicitly, while for $t \geq 4$,

$$M_{t+2} = \begin{pmatrix} 0 & 0 & C_n & \cdots & C_1 & C_2 & \cdots & C_{n-1} & C_n & C_{n-1} & \cdots & C_2 & C_1 & C_2 & \cdots & C_n \\ 0 & 1 & 0 & & 0 & 1 & & 1 & 1 & 1 & & 1 & 0 & 0 & & 0 \\ 1 & 0 & 1 & & 1 & 0 & & 0 & 0 & 1 & & 1 & 0 & 0 & & 0 \end{pmatrix}.$$

They also showed similar constructions work under constant column weight.

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 known as “superimposed codes”.

As combinatorial designs draw heavily from algebraic and geometric structures, so do pooling designs. Mathematical structures are always helpful in understanding, motivating and finding pooling designs. For example, look at again the construction of 1-separable matrix for $n = g^{x+1}$ items and group size $k = g^x$ in Section 1.2. From the previous description, it is not so easy to see why the construction works, where the idea comes from and how to do specifically. Now, let us represent each item by a $(x + 1)$ -dimensional vector with all components chosen from g elements $0, 1, \dots, g - 1$. Then g^{x+1} items would form a $(x + 1)$ -dimensional cube. For each coordinate, there are g parallel hyperplanes, perpendicular to it, which partition all items into g subsets of k items. Performing tests on $g - 1$ of them would reveal which hyperplane contains the unique positive item. So, these k items form a group in the previous description. Clearly, two non-parallel hyperplanes intersect with g^{x-1} items and the positive item would be identified when all coordinates are determined. It can also be easily seen that

if all g subsets in each group are tested, then these tests correspond to a $\bar{1}$ -separable matrix.

We will find that finite fields, linear spaces in finite field, lattices, t -designs and packing provide very helpful tools for pool designs. It is also well-known that the extremal set theory is closely related to nonadaptive group testing, hence pooling designs.

There are two areas which have only played minor role in traditional group testing, but are of utmost importance in pooling designs, and in the meantime offer great challenges. Group testing have been extended to graphs, but no real motivation to study that problem is known. Now we know that 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 exact analysis have been obtained only for some simple models.

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 meantime, propose new problems to these branches. The mutually beneficial relation has just begun to unreel and there is a lot of work to be done.

For convenience of the reader, we collect some basic knowledge about finite fields simplicial complexes and linear spaces over finite fields in the rest of this section.

A finite non-empty set F with two binary operations, addition $+$ and multiplication \cdot , is called a *finite field*, if the following hold

- (1) F is a commutative group with addition, that is,
 - (1a) $x + y = y + x$ for all $x, y, z \in F$,
 - (1b) $x + (y + z) = (x + y) + z$ for all $x, y, z \in F$,
 - (1c) F contains an element 0 such that for every element x in F , $x + 0 = 0 + x = x$, and there exists element $-x$ satisfying $x + (-x) = 0$;
- (2) F is a commutative semi-group and $F - \{0\}$ is a commutative group with multiplication, that is,
 - (2a) $x \cdot y = y \cdot x$ for all $x, y, z \in F$,
 - (1b) $x \cdot (y \cdot z) = (x \cdot y) \cdot z$ for all $x, y, z \in F$,
 - (1c) $F - \{0\}$ contains an element 1 such that for every element x in $F - \{0\}$, $x \cdot 1 = 1 \cdot x = x$, and there exists element x^{-1} satisfying $x \cdot x^{-1} = 1$;
- (3) $x \cdot (y + z) = x \cdot y + x \cdot z$ for all $x, y, z \in F$;
- (4) $0 \cdot x = 0$ for all $x \in F$.

The *order* of a finite field is the number of elements in the field. It is a well-known

fact that a finite field of order q exists if and only if q is a prime power. Moreover, under isomorphism, the finite field of order q is unique for each prime power q , denoted by $GF(q)$ and called the *Galois field* of order q .

When q is a prime, $GF(q)$ consists of all residues modulo q , that is, all elements in Z_q . When $q = p^n$ for a prime p and $r \geq 2$, irreducible polynomial $p(x)$ of degree n .

In fact, for any positive integer n , there exists an irreducible polynomial $p(x)$ of degree n over $GF(p)$, such that the set $\{x, x^2, \dots, x^{p^n-2}, x^{p^n-1} = 1\}$ modulo $p(x)$ is the set of all non-zero polynomials of degree less than n , which is also the set of non-zero elements of $GF(p^n)$. In other words, the multiplication group of $GF(p^n)$ is cyclic. The set of scalars (polynomials of degree zero) is a cyclic subgroup of order $p - 1$, in the form $\{x^m, x^{2m}, \dots, x^{(p-1)m}\}$ where $m = \frac{p^n-1}{p-1}$. We call such $p(x)$ a *primitive polynomial*.

Denote $[t] = \{1, 2, \dots, t\}$ and a subset of k elements is called a k -subset. A $t - (v, k, \lambda)$ design is a family \mathcal{F} of k -subsets of $[v]$ such that every t -subset is contained in at most λ members, called *block* of \mathcal{F} . It is easily verified that

$$b = |\mathcal{F}| = \frac{\binom{v}{t}\lambda}{\binom{k}{t}} \quad (1.5.1)$$

and each element of $[v]$ appears in

$$r = \frac{\binom{v-1}{t-1}\lambda}{\binom{k-1}{t-1}} \quad (1.5.2)$$

blocks. Also, it is easily verified that a t -design is a t' -design for $t' < t$.

A $2 - (v, k, \lambda)$ design is often referred to as a (v, b, r, k, λ) -*block design* in the literature with b and r as given above. A $t - (v, k, 1)$ design is called a *Steiner* (t, v, k) *packing*.

A $t - (v, k, \lambda)$ design becomes a $t - (v, k, \lambda)$ *packing* if the condition that every t -subset appears in exactly λ blocks is replaced by at most λ blocks.

When we use the $t - (v, k, \lambda)$ design or packing in the construction of pooling design, we often have to replace t by T to avoid a conflict with using t as the number of rows in the test matrix.

1.6 An Outline of the Book

We will first introduce deterministic constructions of pooling designs. Chapter 2 reviews and updates the theory of nonadaptive group testing, which guarantee to identify all positive items if that number does not exceed d , and their error-tolerant version. 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 5 also studies pooling designs obtained by mixing a random matrix with a deterministic matrix. It even includes totally deterministic matrices, but used beyond their design specifications; hence identification of positive clones can be stated only in probability terms.

The next two chapters each covers an application mentioned in Section 1.3. Chapter 6 studies the complex model, Chapter 7 studies the contig-sequencing problem. The general technique is to use the graph-testing model.

Chapter 8 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 pooling design.

Chapter 9 studies the array design actually used in some clone libraries. This design also leads to some new theoretical questions in the graph-design theory.

In Chapter 10, we consider problems related to non-unique probe selection, including computational complexity issues on pooling designs.

References

- [1] N. Alon, R. Beigel, S. Kasif, S. Rudth and B. Sudarov, Learning a hidden matching.
- [2] M. Aigner, Search problems in graphs, *Disc. Appl. Math.*, 14 (1986) 215-230.
- [3] M. C. Cai, On the problem of Katona on minimal completely separating systems with restrictions, *Disc. Math.*, 48 (1984) 121-123.
- [4] C. Colbourn, Group testing for consecutive positives, *Ann. Combinatorics*, 3 (1999) 37-41.
- [5] G. J. Chang and F. K. Hwang, A group testing problem, *SIAM J. Alg. Disc. Meth.*, 1 (1980) 21-24.
- [6] T. Chen and F. K. Hwang, A competitive algorithm in searching for many edges in a hypergraph, 2003, preprint.

- [7] P. Damaschke, A tight upper bound for group testing in graphs, *Disc. Math.*, 48 (1994) 101-109.
- [8] D.-Z. Du and F.K. Hwang, *Combinatorial Group Testing and Its Applications (2nd edition)*, World Scientific, 1999.
- [9] V. Grebinski and G. Kucherov, Optimal reconstruction of graphs under additive model, *Algorithmica*, 28 (2000) 104-124.
- [10] F. K. Hwang and W. D. Lin, The incremental group testing model for gap closing in sequencing long molecules, *J. Combin. Opt.*, 7 (2003) 327-337.
- [11] F. K. Hwang, T. T. Song and D.-Z. Du, Hypergeometric and generalized hypergeometric group testing, *SIAM J. Alg. Disc. Methods*, 2 (1981) 426-428.
- [12] P. Johann, A group testing problem for graphs with several defective edges, *Disc. Appl. Math.*, 117 (2002) 99-108.
- [13] A. Knudgen, D. Mubayi and P. Tetali, Minimal completely separating systems of k -sets, *J. Combin. Thy, Series A*, 93 (2001) 192-198.
- [14] J. H. van Lint and R. M. Wilson, *A Course in Combinatorics*, Cambridge University Press, Cambridge, 1992.
- [15] M. Müller and M. Jimbo, Consecutive positive delectable matrices and group testing for consecutive positives, *Disc. Math.*, 279 (2001) 369-381.
- [16] C. Ramsey and I. T. Roberts, Minimal completely separating systems of sets, *Austral. J. Combin.*, 13 (1996) 129-151.
- [17] D. J. Schultz, M. Parnes and R. Srinivasan, Further applications of d -complete designs to group testing, *J. Combin. Inform. & Syst. Sci.*, 8 (1993) 31-41.
- [18] A. Schliep, D. C. Torney, S. Rahmann, Group testing with DNA chips: generating designs and decoding experiments, *Proceedings of the 2nd IEEE Computer Society Bioinformatics Conference*, 2003.
- [19] J. Spencer, Minimal completely separating systems, *J. Combin. Thy.*, 8 (1970) 446-447.
- [20] N. Thierry-Mieg, L. Trilling and J.-L. Roch, A novel pooling design for protein-protein interaction mapping, manuscript, 2004.
- [21] E. Triesch, A group testing problem for hypergraphs of bounded rank, *Disc. Appl. Math.*, 66 (1996) 185-188.

- [22] G. Xu, S.-H. Sze, C.-P. Liu, P.A. Pevzner and N. Arnheim, Gene hunting without sequencing genomic clones: finding exon boundaries in cDNA, *Genomics*, 47 (1998) 171-179.