

# Introduction

Group testing has been around for fifty years. While traditionally group testing literature employs probabilistic models, the combinatorial model has cut its own share and becomes an important part of the literature. Furthermore, combinatorial group testing has tied its knots with many computer science subjects: complexity theory, computational geometry and learning models among others. It has also been used in multiaccess communication and coding.

## 1.1 The History of Group Testing

Unlike many other mathematical problems which can trace back to earlier centuries and divergent sources, the origin of group testing is pretty much pinned down to a fairly recent event-World War II, and is usually credited to a single person-Robert Dorfman. The following is his recollection after 50 years (quoted from a November 17, 1992 letter in response to our inquiry about the role of Rosenblatt):

“The date was 1942 or early '43. The place was Washington, DC in the offices of the Price Statistics Branch of the Research Division of the office of Price Administration, where David Rosenblatt and I were both working. The offices were located in a temporary building that consisted of long wings, chock-full of desks without partitions.

The drabness of life in those wings was relieved by occasional bull sessions. Group testing was first conceived in one of them, in which David Rosenblatt and I participated. Being economists, we were all struck by the wastefulness of subjecting blood samples from millions of draftees to identical analyses in order to detect a few thousand cases of syphilis. Someone (who?) suggested that it might be economical to pool the blood samples, and the idea was batted back and forth. There was lively give-and-take and some persiflage. I don't recall how explicitly the problem was formulated there. What is clear is that I took the idea seriously enough so that in the next few days I formulated the underlying probability problem

and worked through the algebra (which is pretty elementary). Shortly after, I wrote it up, presented it at a meeting of the Washington Statistical Association, and submitted the four-page note that was published in the *Annals of Mathematical Statistics*. By the time the note was published, Rosenblatt and I were both overseas and out of contact.”

We also quote from an October 18, 1992 letter from David Rosenblatt to Milton Sobel which provides a different perspective.

“It is now (Fall of 1992) over fifty year ago when I first invented and propounded the concepts and procedure for what is now called “group testing” in Statistics. I expounded it in the Spring of 1942 the day after I reported for induction during World War II and underwent blood sampling for the Wasserman test. I expounded it before a group of fellow statisticians in the division of Research of the Office of Price Administration in Washington, D.C. Among my auditors that morning was my then colleague Rorbet Dorfman.”

Considering that fifty years have lapsed between the event and the recollections, we find the two accounts reasonably close to each other. Whatever discrepancies there are, they are certainly within the normal boundary of differences associated with human memory. Thus that “someone” (in Dorfman’s letter) who first suggested to pool blood samples could very well be David Rosenblatt. It is also undisputed that Dorfman alone wrote that seminal report [1] published in the Notes Sections of the journal *Annals of Mathematical Statistics* which gave a method intended to be used by the United States Public Health Service and the Selective Service System to weed out all syphilitic men called up for induction. We quote from [1]:

“Under this program each prospective inductee is subjected to a ‘Wasserman-type’ blood test. The test may be divided conveniently into two parts:

1. A sample of blood is drawn from the man,
2. The blood sample is subjected to a laboratory analysis which reveals the presence or absence of “syphilitic antigen.” The presence of syphilitic antigen is a good indication of infection.

When this procedure is used,  $n$  chemical analyses are required in order to detect all infected members of a population of size  $n$ .

The germ of the proposed technique is revealed by the following possibility. Suppose that after the individual blood sera are drawn they are pooled in groups of, say, five and that the groups rather than the individual sera are subjected to chemical analysis. If none of the five sera contributing to

the pool contains syphilitic antigen, the pool will not contain it either and will test negative. If, however, one or more of the sera contain syphilitic antigen, the pool will also contain it and the group test will reveal its presence (the author inserted a note here saying that diagnostic tests for syphilis are extremely sensitive and will show positive results for even great dilutions of antigen). The individuals making up the pool must then be retested to determine which of the members are infected. It is not necessary to draw a new blood sample for this purpose since sufficient blood for both the test and the retest can be taken at once. The chemical analysis requires only small quantities of blood.”

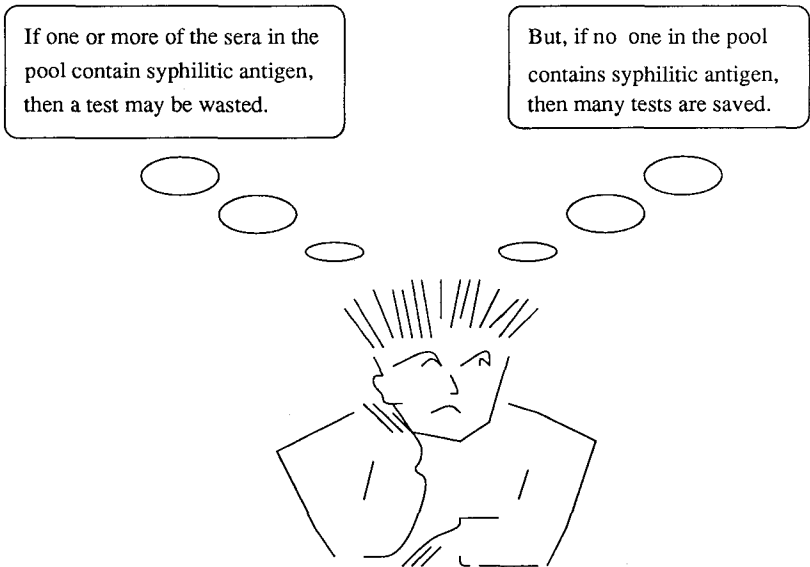


Figure 1.1: The idea of group testing.

Unfortunately, this very promising idea of grouping blood samples for syphilis screening was not actually put to use. The main reason, communicated to us by C. Eisenhart, was that the test was no longer accurate when as few as eight or nine samples were pooled. Nevertheless, test accuracy could have been improved over years, or possibly not a serious problem in screening for another disease. Therefore we quoted the above from Dorfman in length not only because of its historical significance, but also because at this age of a potential AIDS epidemic, Dorfman’s clear account of applying group testing to screen syphilitic individuals may have new impact to the medical world and the health service sector.

Dorfman's blood testing problem found its entrance into a very popular textbook on probability as an exercise; Feller's 1950 book "*An Introduction to Probability Theory and Its Application*, Vol. I" [2] and thus might have survived as a coffee break talk-piece among the academic circle in those days. But by and large, with the conclusion of the Second World War and the release of millions of inductees, the need of group testing disappeared from the Selective Service and the academic world was ready to bury it as a war episode. The only exception was a short note published by Sterrett [9] in 1951 based on his Ph.D. dissertation at University of Pittsburgh. Then Sobel and Groll [8], the two Bell Laboratories Scientists, gave the phrase "group testing" new meaning by giving the subject a very thorough treatment and established many new grounds for future studies in their 74-page paper. Again, they were motivated by practical need, this time from the industrial sector, to remove all leakers from a set of  $n$  devices. We quote from Sobel and Groll:

"One chemical apparatus is available and the devices are tested by putting  $x$  of them (where  $1 \leq x \leq n$ ) in a bell jar and testing whether any of the gas used in constructing the devices has leaked out into the bell jar. It is assumed that the presence of gas in the bell jar indicates only that there is at least one leaker and that the amount of gas gives no indication of the number of leakers."

Sobel and Groll also mentioned other industrial applications such as testing condensers and resistors, the main idea is very well demonstrated by the Christmas tree lighting problem. A batch of light bulbs is electrically arranged in series and tested by applying a voltage across the whole batch or any subset thereof. If the lights are on, then whole tested subset of bulbs must all be good; if the lights are off, then at least one bulb in the subset is defective.

Call the set of defectives among the  $n$  items the *defective set*. Dorfman, as well as Sobel and Groll, studied group testing under probabilistic models, namely, a probability distribution is attached to the defective set and the goal is to minimize the expected number of tests required to identify the defective set. Katona [5] first emphasized the combinatorial aspects of group testing. However, his coverage was predominantly for the case of a single defective and he considered probability distributions of defectives. In this volume a more restrictive viewpoint on *combinatorial group testing* (CGT) is taken by completely eliminating probability distributions on defectives. The presumed knowledge on the defective set is that it must be a member, called a *sample*, of a given family called a *sample space*. For example, the sample space can consist of all  $d$ -subsets of the  $n$  items, when the presumed knowledge is that there are exactly  $d$  defectives among the  $n$  items. This sample space is denoted by  $S(d, n)$ . An item is said to be defective (or good) *in* a sample if it is in (or not in) the sample. The goal in CGT is to minimize the number of tests under the worst scenario. A best algorithm under this goal is called a *minimax algorithm*. The reason that probabilistic models are excluded is not because they are less important or less interesting, but simply

because there is so much to tell about group testing and this is a natural way to divide the material.

Li [6] was the first to study CGT. He was concerned with the situation where industrial and scientific experiments are conducted only to determine which of the variables are important. Usually, only a relatively small number of critical variables exists among a large group of candidates. These critical variables are assumed to have effects too large to be masked by the experimental error, or the combined effect of the unimportant variables. Interpreting each variable as an item, each critical variable as a defective, and each experiment as a group test, a large effect from an experiment indicates the existence of a critical variable among the variables covered by the experiment. Li assumed that there are exactly  $d$  critical variables to start with, and set to minimize the worst-case number of tests.

Since Li, CGT has been studied along side with PGT for those classical applications in medical, industrial and statistical fields. Recently, CGT is also studied in complexity theory, graph theory, learning models, communication channels and fault tolerant computing. While it is very encouraging to see a wide interest in group testing, one unfortunate consequence is that the results obtained are fragmented and submerged in the jargons of the particular fields. This book is an attempt to give a unified and coherent account of up-to-date results in combinatorial group testing.

## 1.2 The Binary Tree Representation of a Group Testing Algorithm and the Information Lower Bound

A *binary tree* can be inductively defined as a node, called the *root*, with its two disjoint binary trees, called the *left* and *right* subtree of the root, either both empty or both nonempty. Nodes occurring in the two subtrees are called *descendants* of the root (the two immediate descendants are called *children*), and all nodes having a given node as a descendant are ancestors (the immediate ancestor is called a *parent*). Two children of the same parent are *siblings*. Nodes which have no descendants are called *leaves*, and all other nodes are called *internal nodes*. The path length of a node is the number of that node's ancestors. A node is also said at *level*  $l$  if its path length is  $l - 1$ . The *depth* of a binary tree is the maximal level over all leaves.

Let  $S$  denote the sample space. Then a group testing algorithm  $T$  for  $S$  can be represented by a binary tree, also denoted by  $T$ , by the following rules:

- (i) Each internal node  $u$  is associated with a test  $t(u)$ ; its two links associated with the two outcomes of  $t(u)$  (we will always designate the negative outcome by the left link). The *test history*  $H(u)$  of a node  $u$  is the set of tests and outcomes associated with the nodes and links on the path of  $u$ .
- (ii) Each node  $u$  is also associated with an event  $S(u)$  which consists of all the members of  $S$  consistent with  $H(u)$ .  $|S(v)| \leq 1$  for each leaf  $v$ .

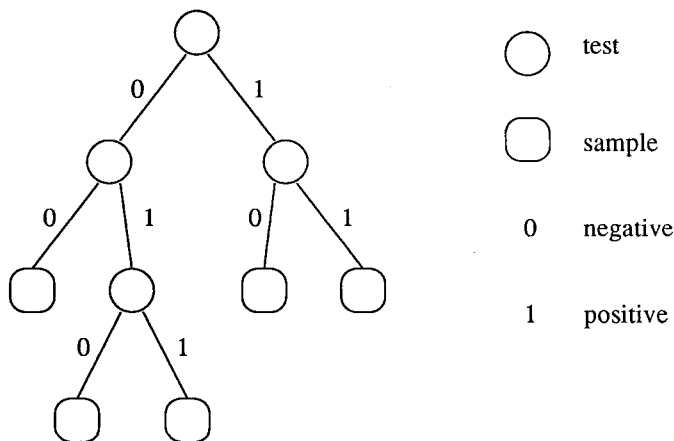


Figure 1.2: The binary tree representation.

Since the test  $t(u)$  simply splits  $S(u)$  into two disjoint subsets, every member of  $S$  must appear in one and only one  $S(v)$  for some leaf  $v$ .

An algorithm is called *reasonable* if no test whose outcome is predictable is allowed. For a reasonable algorithm each  $S(u)$  is split into two nonempty subsets. Thus  $|S(v)| = 1$  for every leaf  $v$ , i.e., there exists a one-to-one mapping between  $S$  and the leaf-set.

Let  $p(s)$  denote the path of the leaf  $v$  associated with the sample  $s$  and let  $|p(s)|$  denote its length. Then

$$M_T(S) = \max_s |p(s)|$$

is the depth of  $T$ . Since the same subset will not be tested more than once in a reasonable algorithm, the number of tests, consequently the number of reasonable algorithms, is finite. Therefore we can define

$$M(S) = \min_T M_T(S).$$

An algorithm which achieves  $M(S)$  is called a *minimax algorithm* for  $S$ . The goal of CGT is to find a minimax algorithm for a given  $S$ ; usually, we settle for a good heuristic. The determination of  $M(S)$  and obtaining good bounds for it are challenging problems and unsettled for most  $S$ .

Let  $\lceil x \rceil$  ( $\lfloor x \rfloor$ ) denote the smallest (largest) integer not less (greater) than  $x$ . Also let  $\log x$  mean  $\log_2 x$  throughout unless otherwise specified.

**Theorem 1.2.1**  $M(S) \geq \lceil \log |S| \rceil$ .

*Proof:* Consider a group testing algorithm  $T$  for the problem  $S$ . At each internal node  $u$  of  $T$ , the test  $t(u)$  splits  $S(u)$  into two disjoint subsets according to whether the outcome is negative or positive. The cardinality of one of the two subsets must be at least half of  $|S(u)|$ . Therefore  $M(S) \geq \lceil \log |S| \rceil$ .  $\square$

We will refer to the bound in Theorem 1.2.1 as the *information lower bound*.

**Lemma 1.2.2**  $\sum_{s \in S} 2^{-|p(s)|} = 1$  for every reasonable algorithm.

*Proof.* True for  $|S| = 1$ . A straightforward induction on  $|S|$  proves the general case.  $\square$

Theorem 1.2.1 and Lemma 1.2.2 are well known in the binary tree literature. The proofs are included here for completeness.

A group testing algorithm  $T$  for the sample space  $S$  is called *admissible* if there does not exist another algorithm  $T'$  for  $S$  such that

$$|p(s)| \geq |p'(s)| \quad \text{for all } s \in S$$

with at least one strict inequality true, where  $p'(s)$  is the path of  $s$  in  $T'$ .

**Theorem 1.2.3** A group testing algorithm is admissible if and only if it is reasonable.

*Proof.* That “reasonable” is necessary is obvious. To show sufficiency, suppose to the contrary that  $T$  is an inadmissible reasonable algorithm, i.e., there exists  $T'$  such that

$$|p(s)| \geq |p'(s)| \quad \text{for all } s \in S$$

with at least one strict inequality true. Then

$$\sum_{s \in S} 2^{-|p'(s)|} > \sum_{s \in S} 2^{-|p(s)|} = 1,$$

a contradiction to Lemma 1.2.2.  $\square$

From now on only admissible, or reasonable, algorithms are considered in the volume. The word “algorithm” implies an admissible algorithm.

### 1.3 The Structure of Group Testing

The information lower bound is usually not achievable. For example, consider a set of six items containing exactly two defectives. Then  $\lceil \log |S| \rceil = \lceil \log \binom{6}{2} \rceil = 4$ . If a subset of one item is tested, the split is 10 (negative) and 5 (positive); it is 6 and 9 for a subset of two, 3 and 12 for a subset of three, and 1 and 14 for a subset of four. By Theorem 1.2.1, at least four more tests are required.

The reason that information lower bound cannot be achieved in general for group testing is that the split of  $S(u)$  at an internal node  $u$  is not arbitrary, but must be realizable by a group test. Therefore it is of importance to study which types of splitting are permissible in group testing. The rest of this section reports work done by Hwang, Lin and Mallows [4].

While a group testing algorithm certainly performs the tests in the order starting from the root of the tree proceeding to the leaves, the analysis is often more convenient if started from the leaves (that is the way the Huffman tree - a minimum weighted binary tree - is constructed). Thus instead of asking what splits are permissible, the question becomes: for two children nodes  $x, y$  of  $u$ , what types of  $S(x)$  and  $S(y)$  are permitted to merge into  $S(u)$ .

Let  $N$  denote a set of  $n$  items,  $D$  the defective set and  $S_0 = \{D_1, \dots, D_k\}$  the initial sample space. Without loss of generality assume  $\cup_{i=1}^k D_i = N$ , for any item not in  $\cup_{i=1}^k D_i$  can immediately be identified as good and deleted from  $N$ . A subset  $S_i$  of  $S_0$  is said to be *realizable* if there exists a group testing tree  $T(A)$  for  $S_0$  and a node  $u$  of  $T$  such that  $S(u) = S_i$ . A partition  $\pi = (S_1, \dots, S_m)$  of  $S_0$  is said to be *realizable* if there exists a group testing tree  $T$  for  $S_0$  and a set of  $m$  nodes  $(u_1, \dots, u_m)$  of  $T$  such that  $S(u_i) = S_i$  for  $1 \leq i \leq m$ . Define  $\|S\| = \cup_{D_i \in S} D_i$ ,  $\overline{\|S\|} = N \setminus \|S\|$ , the complement of  $\|S\|$ , and  $\hat{S} = \{A \mid A \subseteq \|S\|, A \supseteq \text{some } D_i\}$ , the closure of  $S$ . Furthermore, let  $\pi = (S_1, S_2, \dots, S_m)$  be a partition of  $S_0$ , i.e.,  $S_i \cap S_j = \emptyset$  for all  $i \neq j$ , and  $\cup_{S_i \in \pi} S_i = S_0$ . Then  $S_i$  and  $S_j$  are said to be *separable* if there exists an  $I \subseteq N$  such that  $I \cap D = \emptyset$  for all  $D \in S_i$ , and  $I \cap D \neq \emptyset$  for all  $D \in S_j$  (or vice versa).

Given  $\pi$ , define a directed graph  $G_\pi$  by taking each  $S_i$  as a node, and a directed edge from  $S_i$  to  $S_j$  ( $S_i \rightarrow S_j$ ),  $i \neq j$ , if and only if there exists  $A_i \in \hat{S}_i$ ,  $A_j \in \hat{S}_j$  such that  $A_i \subseteq A_j$ .

**Theorem 1.3.1** *The following statements are equivalent:*

- (i)  $S_i$  and  $S_j$  are not separable.
- (ii)  $S_i \rightarrow S_j \rightarrow S_i$  in  $G_\pi$ .
- (iii)  $\hat{S}_i \cap \hat{S}_j \neq \emptyset$ .

*Proof:* By showing (i)  $\Rightarrow$  (ii)  $\Rightarrow$  (iii)  $\Rightarrow$  (i).

(i)  $\Rightarrow$  (ii): It is first shown that if  $S_i \not\rightarrow S_j$ , then  $I = \overline{\|S_j\|} \cap \|S_i\| \neq \emptyset$  and  $I$  separates  $S_i, S_j$ . Clearly, if  $I = \emptyset$ , then  $\|S_i\| \subseteq \|S_j\|$ ; since  $\|S_i\| \in \hat{S}_i$  and  $\|S_j\| \in \hat{S}_j$ , it follows that  $S_i \rightarrow S_j$ , a contradiction. Thus  $I \neq \emptyset$ . Also,  $I \cap D = \emptyset$  for all  $D \in S_j$  since  $I \subseteq \overline{\|S_j\|}$ . Furthermore, if there is a  $D \in S_i$  such that  $I \cap D = \emptyset$ , then from the definition of  $I$ ,  $\overline{\|S_j\|} \cap \|S_i\| \cap D = \overline{\|S_j\|} \cap D = \emptyset$  and hence  $D \subseteq \|S_j\| \in \hat{S}_j$ . This implies  $S_i \rightarrow S_j$ , again a contradiction. Therefore  $I \cap D \neq \emptyset$  for all  $D \in S_i$  and  $I$  separates  $S_i, S_j$ . The proof is similar if  $S_j \not\rightarrow S_i$ .

(ii)  $\Rightarrow$  (iii): Suppose  $S_i \rightarrow S_j \rightarrow S_i$ . Let  $A_i, A'_i \in \hat{S}_i$  and  $A_j, A'_j \in \hat{S}_j$  such that  $A_i \subseteq A_j$  and  $A'_i \supseteq A'_j$ . Furthermore, let  $u = \|S_i\| \cap \|S_j\|$ . Then  $u \neq \emptyset$ . Since

$A_i \subseteq A_j \subseteq \| S_j \|$ ,  $A_i \subseteq \| S_i \| \cap \| S_j \| = u \subseteq \| S_i \|$  and thus  $u \in \hat{S}_i$ . A similar argument shows  $u \in \hat{S}_j$ . Therefore  $\hat{S}_i \cap \hat{S}_j \neq \emptyset$ .

(iii)  $\Rightarrow$  (i): Suppose  $w \in \hat{S}_i \cap \hat{S}_j$  ( $w \neq \emptyset$ ) but  $S_i$  and  $S_j$  are separable. Let  $I \subseteq N$  be a subset such that  $I \cap D_i \neq \emptyset$  for all  $D_i \in S_i$  and  $I \cap D_j = \emptyset$  for all  $D_j \in S_j$ . But  $w \in \hat{S}_i \Rightarrow$  there exists a  $D_i \in S_i$  such that  $D_i \subseteq w$ , hence  $I \cap w \supseteq I \cap D_i \neq \emptyset$ . On the other hand  $w \in \hat{S}_j \Rightarrow w \subseteq \| S_j \| \Rightarrow I \cap w = \emptyset$ , a contradiction. Therefore,  $S_i$  and  $S_j$  are not separable.

**Theorem 1.3.2**  $\pi$  is realizable if and only if  $G_\pi$  does not contain a directed cycle.

*Proof.* Suppose  $G_\pi$  has a cycle  $C$ . Let  $\pi'$  be a partition of  $S_0$  obtained by merging two (separable) sets  $S_i$  and  $S_j$  in  $\pi$ . From Theorem 1.3.1,  $G_\pi$  cannot contain the cycle  $S_i \rightarrow S_j \rightarrow S_i$  for otherwise  $S_i$  and  $S_j$  are not separable, therefore not mergeable. Since every edge in  $G_\pi$  except those between  $S_i$  and  $S_j$  is preserved in  $G_{\pi'}$ ,  $G_{\pi'}$  must contain a cycle  $C'$ . By repeating this argument, eventually one obtains a partition with no two parts separable. Therefore  $\pi$  is not realizable.

Next assume that  $G_\pi$  contains no cycle. The graph  $G_\pi$  induces a partial ordering on the  $S_i$ 's with  $S_i < S_j$  if and only if  $S_i \rightarrow S_j$ . Let  $S_j$  be a minimal element in this ordering. Then  $I = \| S_j \|$  separates  $S_j$  from all other  $S_i$ 's. Let  $G_{\pi'}$  be obtained from  $G_\pi$  by deleting the node  $S_j$  and its edges. Then clearly,  $G_{\pi'}$  contains no cycle. Therefore the above argument can be repeated to find a set of tests which separate one  $S_i$  at a time.  $\square$

Unfortunately, local conditions, i.e., conditions on  $S_i$  and  $S_j$  alone, are not sufficient to tell whether a pair is validly mergeable or not. ( $S_i$  and  $S_j$  is *validly mergeable* if merging  $S_i$  and  $S_j$  preserves the realizability of the partition.) Theorem 1.3.1 provides some local necessary conditions while the following corollaries to Theorem 1.3.2 provide more such conditions as well as some sufficient conditions.

**Corollary 1.3.3** Let  $M$  be the set of  $S_i$ 's which are maximal under the partial ordering induced by  $G_\pi$ . (If  $|M| = 1$ , add to  $M = \{S_i\}$  an element  $S_j$  such that  $S_j < S_i$  and  $S_j \not\prec S_k$  for all other  $S_k$ .) Then every pair in  $M$  is validly mergeable.

**Corollary 1.3.4** Let  $\pi$  be realizable. Then the pair  $S_i, S_j$  is validly mergeable if there does not exist some other  $S_k$  such that either  $S_i \rightarrow S_k$  or  $S_j \rightarrow S_k$ .

**Corollary 1.3.5**  $S_i$  and  $S_j$  are not validly mergeable if there exists another  $S_k$  such that either  $S_i \rightarrow S_k \rightarrow S_j$  or vice versa.

**Corollary 1.3.6** Let  $\pi$  be the partition of  $S_0$  in which every  $S_i$  consists of just a single element. Then  $\pi$  is realizable.

Let  $S \subset S_0$  ( $S \neq \emptyset$ ) and let  $\pi = \{S, S_1, S_2, \dots, S_m\}$  be the partition of  $S_0$  where each  $S_i$  consists of a single element  $D_i$ . The following theorem shows a connection between the realization of  $S$  and the realization of  $\pi$ .

**Theorem 1.3.7** *The following statements are equivalent:*

- (i)  $S$  is realizable.
- (ii)  $\hat{S} \cap S_0 = S$ .
- (iii)  $\pi$  is realizable.

*Proof.* We show (i)  $\Rightarrow$  (ii)  $\Rightarrow$  (iii)  $\Rightarrow$  (i).

(i)  $\Rightarrow$  (ii): Since  $S \subseteq S_0$ , clearly  $\hat{S} \cap S_0 \supseteq S$ . So only  $\hat{S} \cap S_0 \subseteq S$  needs to be shown. Let  $D \in \hat{S} \cap S_0$ , then  $D \in \hat{S}$  and hence there is some  $D' \in S$  such that  $D' \subseteq D \subseteq \parallel S \parallel$ . Suppose  $D \notin S$ . Then  $D = S_i$ , for some  $i$ . But  $D \in \hat{S}$  implies  $S_i \rightarrow S$  and  $D' \subseteq D$  implies  $S \rightarrow S_i$ . Hence  $S$  is not realizable by Theorem 1.3.2.

(ii)  $\Rightarrow$  (iii): If  $\pi$  is not realizable, then  $G_\pi$  contains a directed cycle. Since each  $S_i$  is a single element,  $\hat{S}_i = S_i$  and hence any directed cycle must contain  $S$ , say  $S_1 \rightarrow S_2 \rightarrow \dots \rightarrow S_k \rightarrow S \rightarrow S_1$ . But  $S_1 \rightarrow S_2 \rightarrow \dots \rightarrow S_k \rightarrow S$  implies  $S_1 \rightarrow S$  since  $S_i$ 's are single elements. By Theorem 1.3.1,  $S \rightarrow S_1 \rightarrow S$  implies  $\hat{S} \cap \hat{S}_1 = \hat{S} \cap \{D_1\} \neq \emptyset$ . This implies  $D_1 \in \hat{S} \cap S_0$ . Since  $D_1 \in S$ ,  $\hat{S} \cap S_0 \neq S$ .

(iii)  $\Rightarrow$  (i): Trivially true by using the definition of realizability.  $\square$

## 1.4 Number of Group Testing Algorithms

One interesting problem is to count the number of group testing algorithms for  $S$ . This is the total number of binary trees with  $|S|$  leaves (labeled by members of  $S$ ) which satisfy the group testing structure. While this problem remains open, Moon and Sobel [7] counted for a class of algorithms when the sample space  $S$  is the power set of  $n$  items.

Call a group *pure* if it contains no defective, and *contaminated* otherwise. A group testing algorithm is *nested* if whenever a contaminated group is known, the next group to be tested must be a proper subset of the contaminated group. (A more detailed definition is given in Section 2.3.) Sobel and Groll [8] proved

**Lemma 1.4.1** *Let  $U$  be the set of unclassified items and suppose that  $C \subset U$  is tested to be contaminated. Furthermore, suppose  $C' \subset C$  is then tested to be contaminated. Then items in  $C \setminus C'$  can be mixed with items in  $U \setminus C$  without losing any information.*

*Proof.* Since  $C'$  being contaminated implies  $C$  being contaminated, the sample space given both  $C$  and  $C'$  being contaminated is the same as only  $C'$  being contaminated. But under the latter case, items in  $C \setminus C'$  and  $U \setminus C$  are indistinguishable.  $\square$

Thus under a nested algorithm, at any stage the set of unclassified items is characterized by two parameters  $m$  and  $n$ , where  $m \geq 0$  is the number of items in a

contaminated group and  $n$  is the total number of unclassified items. Let  $f(m, n)$  denote the number of nested algorithms when the sample space is characterized by such  $m$  and  $n$ . By using the “nested” property, Moon and Sobel [7] obtained

$$\begin{aligned} f(0, 0) &= 1 \\ f(0, n) &= \sum_{k=1}^n f(0, n-k)f(k, n) \text{ for } n \geq 1, \\ f(1, n) &= f(0, n-1) \text{ for } n \geq 1, \\ f(m, n) &= \sum_{k=1}^{m-1} f(m-k, n-k)f(k, n) \text{ for } n \geq m \geq 1, \end{aligned}$$

where  $k$  is the size of the group to be tested. Recall that the Catalan numbers

$$C_k = \frac{1}{k} \binom{2k-2}{k-1}$$

satisfy the recurrence relation

$$C_k = \sum_{i=1}^{k-1} C_i C_{k-i}$$

for  $k \geq 2$ . Moon and Sobel gave

**Theorem 1.4.2**

$$\begin{aligned} f(0, n) &= C_{n+1} \prod_{i=1}^{n-1} f(0, i) \quad \text{for } n \geq 2, \\ f(m, n) &= C_m \prod_{i=1}^m f(0, n-i) \quad \text{for } 1 \leq m \leq n. \end{aligned}$$

*Proof.* Theorem 1.4.2 is easily verified for  $f(0, 2)$  and  $f(1, n)$ . The general case is proved by induction

$$\begin{aligned} f(0, n) &= \sum_{k=1}^n f(0, n-1)f(k, n) \\ &= \sum_{k=1}^n \left( C_{n-k+1} \prod_{i=1}^{n-k-1} f(0, i) \right) (C_k \prod_{i=1}^k f(0, n-i)) \\ &= C_{k+1} \prod_{i=1}^{n-1} f(0, i). \\ f(m, n) &= \sum_{k=1}^{m-1} f(m-k, n-k)f(k, n) \\ &= \sum_{k=1}^{m-1} \left( C_{m-k} \prod_{i=1}^{m-k} f(0, n-k-i) \right) (C_k \prod_{i=1}^k f(0, n-k)) \\ &= C_m \prod_{i=1}^m f(0, n-i). \quad \square \end{aligned}$$

Define  $F(n) = f(0, n)$ . The following recurrence relations follow readily from Theorem 1.4.2 and the definition of the Catalan numbers.

**Corollary 1.4.3** *If  $n \geq 1$ , then*

$$F(n) = \frac{C_{n+1}}{C_n} F^2(n-1) = \frac{2(2n-1)}{n+1} F^2(n-1).$$

**Corollary 1.4.4** *If  $n \geq 2$ , then*

$$F(n) = C_{n+1}C_nC_{n-1}^2C_{n-2}^4 \cdots C_2^{2^{n-2}}.$$

**Corollary 1.4.5** *If  $n \geq 1$ , then*

$$F(n) = 4^{2^n - 1} \prod_{i=1}^n \left\{ 1 - \frac{3}{2(i+1)} \right\}^{2^{n-i}}.$$

The first few values of  $F(n)$  are

$n$	1	2	3	4	5	6
$F(n)$	1	2	10	280	235,220	173,859,840,000

The limiting behavior of  $F(n)$  can be derived from the formula in Corollary 1.4.5.

**Corollary 1.4.6**

$$\lim_{n \rightarrow \infty} \{F(n)\}^{2^{-n}} = 4 \prod_{i=1}^{\infty} \left\{ 1 - \frac{3}{2(i+1)} \right\}^{2^{-i}} = 1.526753 \cdots$$

*More generally, it can be shown that*

$$F(n) = \frac{1}{4} \alpha^{2^n} \left\{ 1 + \frac{3/2}{n} + o\left(\frac{1}{n}\right) \right\}$$

*as  $n \rightarrow \infty$ , where  $\alpha = 1.526753 \cdots$ ; in particular,  $F(n) > \frac{1}{4} \alpha^{2^n}$  for  $n \geq 1$ .*

The proof of this is omitted.

## 1.5 A Prototype Problem and Some Basic Inequalities

We first describe a prototype CGT problem which will be the focus of the whole book. Then we discuss generalizations and special cases.

Consider a set  $I$  of  $n$  items known to contain exactly  $d$  defectives. The defectives look exactly like the good items and the only way to identify them is through testing, which is error-free. A test can be applied to an arbitrary subset of the  $n$  items with two possible outcomes: a *negative* outcome indicates that all items in the subset are good, a *positive* outcome indicates the opposite, i.e., at least one item in the subset is defective (but not knowing which ones or how many are defective). The goal is to find an algorithm  $A$  to identify all defectives with a small  $M_A(S) = M_A(d, n)$ . This is the prototype problem. It is also called the  $(d, n)$ -problem to highlight the two parameters  $n$  and  $d$ . In the literature the  $(d, n)$  problem has sometimes been called the *hypergeometric group testing problem*. We now reserve the latter term for the case

that a uniform distribution is imposed on  $S$ , since then the probability that a subset of  $k$  items containing  $x$  defectives follows the hypergeometric distribution

$$\frac{\binom{d}{x} \binom{n-d}{k-x}}{\binom{n}{k}}.$$

Note that the hypergeometric group testing problem under this new definition belongs to PGT, not CGT.

Since  $M(n, n) = M(0, n) = 0$ , whenever the  $(d, n)$  problem is studied, it is understood that  $0 < d < n$ . Hu, Hwang and Wang [3] proved some basic inequalities about the  $M$  function which are reported in this section.

**Lemma 1.5.1**  $S \subset S'$  implies  $M(S) \leq M(S')$ .

*Proof.* Any algorithm for  $S'$  is also an algorithm for  $S$  where all paths except those of  $s \in S' \setminus S$  are preserved. Therefore, the maximum path length of the algorithm on  $S$  cannot exceed that on  $S'$ .  $\square$

**Corollary 1.5.2**  $M(d, n) \leq M(d, n + 1)$ .

*Proof.* Add an imaginary good new item  $I_{n+1}$  to the item set. Then each sample in  $S(d, n)$  is augmented to a sample in  $S(d, n + 1)$  since  $I_{n+1}$  of the former can only be good.

Note that adding a good item to any group does not affect the test outcome. Therefore, no additional piece of good item is actually needed; an imaginary piece of good item will do.  $\square$

Let  $M(m; d, n)$  denote the minimum number of tests necessary to identify the  $d$  defectives among  $n$  items when a particular subset of  $m$  items is known to be contaminated.

**Theorem 1.5.3**  $M(m; d, n) \geq 1 + M(d - 1, n - 1)$  for  $m \geq 2$  and  $0 < d < n$ .

*Proof.* By Lemma 1.5.1

$$M(m; d, n) \geq M(2; d, n) \quad \text{for } m \geq 2.$$

Let  $T$  be an algorithm for the  $(2; d, n)$  problem, and let  $M_T(2; d, n) = k$ , i.e.,  $k$  is the maximum path length of a leaf in  $T$ . Let  $I_1$  and  $I_2$  denote the two items in the contaminated group.

**Claim.** Every path of length  $k$  in  $T$  includes at least one test that contains either  $I_1$  or  $I_2$ .

*Proof of claim.* Suppose to the contrary that for some leaf  $v$  the path  $p(v)$  has length  $k$  and involves no test containing  $I_1$  or  $I_2$ . Since no test on  $p(v)$  can distinguish  $I_1$  from  $I_2$ , and since  $\{I_1, I_2\}$  is a contaminated group,  $I_1$  and  $I_2$  must both be defective in the sample  $s(v)$ . Let  $u$  be the sibling node of  $v$ . Then  $u$  is also a leaf since  $v$  has maximum path length. Since  $p(u)$  and  $p(v)$  have the same set of tests,  $I_1$  and  $I_2$  must also be both defective in the sample  $s(u)$ . Since  $s(u) \neq s(v)$ , there exists indices  $i$  and  $j$  such that  $I_i$  is defective and  $I_j$  is good in the sample  $s(u)$ , while  $I_i$  is good and  $I_j$  is defective in  $s(v)$ . Let  $w$  denote the parent node of  $u$  and  $v$ ; thus  $S(w) = \{s(u), s(v)\}$ . Then no test on  $p(w)$  can be of the form  $G \cup \{I_i\}$ , where  $G$ , possibly empty, contains only items classified as good in  $s(u)$ , since such a test must yield a positive outcome for  $s(u)$  and a negative outcome for  $s(v)$ , and hence would have separated  $s(u)$  from  $s(v)$ . Define  $s$  to be the sample identical to  $s(u)$  except that  $I_2$  is good and both  $I_i$  and  $I_j$  are defective. Then  $s$  can be distinguished from  $s(u)$  only by a test containing  $I_2$ , which by assumption does not exist on  $p(w)$ , or by a test of the form  $G \cup \{I_j\}$ , whose existence has also been ruled out. Thus  $s \in S(w)$ , and  $u$  and  $v$  cannot both be leaves, a contradiction that completes the proof of the claim.

By renaming the items if necessary, one may assume that every path of length  $k$  in  $T$  involves a test that contains  $I_1$ . Add an imaginary defective to the  $(d-1, n-1)$  problem and label it  $I_1$ , and map the  $n-1$  items of the  $(d-1, n-1)$  problem one-to-one to the items of the  $(d, n)$  problem except  $I_1$ . Then the modified  $T$  can be used to solve the  $(d-1, n-1)$  problem except that every test containing  $I_1$  is skipped since the positive outcome is predictable. But each path of length  $k$  in  $T$  contains such a test. Hence the maximum path length in applying  $T$  to the  $(d-1, n-1)$  problem is  $k-1$ , i.e.,

$$M_T(2; d, n) \geq 1 + M_T(d-1, n-1) .$$

Since  $T$  is arbitrary, the proof is complete.  $\square$

**Corollary 1.5.4**  $M(n-1, n) = n-1$ .

*Proof.* Trivially true for  $n=1$ . For  $n \geq 2$

$$M(n-1, n) = M(n; n-1, n) \geq 1 + M(n-2, n-1) \geq n-1 + M(0, 1) = n-1 . \quad \square$$

**Theorem 1.5.5**  $M(d, n) \geq 1 + M(d-1, n-1) \geq M(d-1, n)$  for  $0 < d < n$ .

*Proof.* By noting

$$M(d, n) = M(n; d, n) ,$$

the first inequality follows from Theorem 1.5.3. The second inequality is trivially true for  $d=1$ . The general case is proved by using the induction assumption

$$M(d, n) \geq M(d-1, n) .$$

Let  $T$  be the algorithm which first tests a single item and then uses a minimax algorithm for the remaining problem. Then

$$\begin{aligned} M(d-1, n) &\leq M_T(d-1, n) \\ &= 1 + \max\{M(d-1, n-1), M(d-2, n-1)\} \\ &= 1 + M(d-1, n-1). \quad \square \end{aligned}$$

**Lemma 1.5.6**  $M(d, n) \leq n-1$ .

*Proof.* The individual testing algorithm needs only  $n-1$  tests since the state of the last item can be deduced by knowing the states of the other items and knowing  $d$ .  $\square$

**Lemma 1.5.7** Suppose that  $n-d > 1$ . Then  $M(d, n) = n-1$  implies  $M(d, n-1) = n-2$ .

*Proof.* Suppose to the contrary that  $M(d, n-1) \neq n-2$ . By Lemma 1.5.6, this is equivalent to assuming  $M(d, n-1) < n-2$ . Let  $T$  denote an algorithm for the  $(d, n)$  problem which first tests a single item and then uses a minimax algorithm for the remaining problem. Then

$$\begin{aligned} M(d, n) &\leq M_T(d, n) \\ &= 1 + \max\{M(d, n-1), M(d-1, n-1)\} \\ &= 1 + M(d, n-1) \quad \text{by Theorem 1.5.5} \\ &< n-1, \end{aligned}$$

a contradiction to the assumption of the lemma.  $\square$

**Theorem 1.5.8**  $M(d, n) = M(d-1, n)$  implies  $M(d, n) = n-1$ .

*Proof.* Suppose  $n-d = 1$ . Then Theorem 1.5.8 follows from Corollary 1.5.4. The general case is proved by induction on  $n-d$ . Note that  $M(d, n) = M(d-1, n)$  implies

$$M(d, n) = 1 + M(d-1, n-1)$$

by Theorem 1.5.5. Let  $T$  be a minimax algorithm for the  $(d, n)$  problem. Suppose that  $T$  first tests a group of  $m$  items. If  $m > 1$ , then

$$\begin{aligned} M_T(d, n) &\geq 1 + M(m; d, n) \\ &\geq 2 + M(d-1, n-1) \quad \text{by Theorem 1.5.3,} \end{aligned}$$

a contradiction to what has just been shown. Therefore  $m = 1$  and

$$\begin{aligned} M(d, n) &= 1 + \max\{M(d, n-1), M(d-1, n-1)\} \\ &= 1 + M(d, n-1) \end{aligned}$$

by Theorem 1.5.5 and the fact  $d < n - 1$ . It follows that

$$M(d-1, n-1) = M(d, n-1).$$

Hence

$$M(d-1, n-1) = n-2$$

by induction. Therefore

$$M(d, n) = 1 + M(d-1, n-1) = n-1.$$

**Lemma 1.5.9** *Suppose  $M(d, n) < n - 1$ . Then*

$$M(d, n) \geq 2l + M(d-l, n-l) \text{ for } 0 < l \leq d < n.$$

*Proof.* By Corollary 1.5.4 and Theorem 1.5.5,  $M(d, n) < n - 1$  implies  $n - d > 1$ . First consider the case  $l = 1$ .

Let  $T$  be a minimax algorithm for the  $(d, n)$  problem which first tests a set of  $m$  items. If  $m > 1$ , then Lemma 1.5.9 is an immediate consequence of Theorem 1.5.3. Therefore assume  $m = 1$ . Suppose to the contrary that

$$M_T(d, n) < 2 + M(d-1, n-1).$$

Then

$$\begin{aligned} 1 + M(d-1, n-1) &\geq M_T(d, n) \\ &= 1 + \max\{M(d, n-1), M(d-1, n-1)\} \\ &= 1 + M(d, n-1) \end{aligned}$$

by Theorem 1.5.5. Therefore

$$M(d-1, n-1) = M(d, n-1) = n-2$$

by Theorem 1.5.8. Consequently,

$$M(d, n) = 1 + M(d, n-1) = n-1,$$

a contradiction to the assumptions of the lemma. Thus Lemma 1.5.9 is true for  $l = 1$ . The general case is proved by a straightforward induction argument (on  $l$ ).  $\square$

**Corollary 1.5.10**  $M(d, n) \geq \min\{n-1, 2l + \lceil \log \binom{n-l}{d-l} \rceil\}$  for  $0 < l \leq d < n$ .

## 1.6 Variations of the Prototype Problem

While we define a group testing algorithm to be sequential in nature, one can also study *nonadaptive algorithms* in which all tests must be specified simultaneously without the knowledge of any test outcomes. In general, one can consider *multistage algorithms* in which tests are divided into several stages and only test outcomes of previous stages are assumed known. In practice the choice of a sequential algorithm, a nonadaptive algorithm, or a multistage algorithm is a trade-off between time and the number of tests required. However, the binary tree representation is valid only for sequential procedures.

When  $d$  is not exactly known, we will keep the notation of  $M_T(d, n)$  except replacing  $d$  by some other parameters characterizing the sample space. In particular,  $\bar{d}$  denotes that  $d$  is an upper bound of the number of defectives and a blank for  $d$  denotes that nothing is known about the defective set. Sometimes the sample space is induced from another space through a set of tests and their outcomes. Such a space may be easier to characterize as the original space tagged with a test history.

Let  $i$  denote the outcome that the tested subset contains  $i$  defectives, and let  $i^+$  denote at least  $i$  defectives. The binary outcome  $(0, 1^+)$  of group testing can be extended to  $k$ -nary outcomes. For example, multiaccess channels in computer communication have ternary outcome  $(0, 1, 2^+)$ . This has been further generalized to  $(k + 1)$ -nary outcome  $(0, 1, \dots, k - 1, k^+)$ . In particular, when  $k = n$  (the number of items), then each test reveals the exact number of defectives contained therein. Other types of  $k$ -nary outcome are also possible. For example, a check bit in coding often provides the binary outcome (even, odd).

Various kinds of restrictions have been considered. In the line group testing problem the  $n$  items are arranged on a line and only subsets of consecutive items from top of the line can be tested. This situation may be appropriate when one inspects items on an assembly line. Similarly, one can study circular group testing and in general, graphical group testing in which only certain subgroups can be tested. In the last problem, the goal can also change from identifying  $n$  defective nodes to identifying a prespecified subgraph. Other restrictions may concern the size of testable subsets, or the memory space in implementing an algorithm. The searched items and the searching space can also be geometrical objects.

One big assumption in the prototype problem which may not hold in reality is that tests are error-free. Recently, this assumption has been dropped in the study of learning models which can be considered as an extension of combinatorial group testing.

The prototype problems as well as all of its variations will be the subject matter in subsequent chapters.

## References

- [1] R. Dorfman, The detection of defective members of large populations, *Ann. Math. Statist.* 14 (1943) 436-440.
- [2] W. Feller, *An Introduction to Probability Theory and Its Applications*, Vol. 1, (John Wiley, New York, 1950).
- [3] M. C. Hu, F. K. Hwang and J. K. Wang, A boundary problem for group testing, *SIAM J. Alg. Disc. Methods* 2 (1981) 81-87.
- [4] F. K. Hwang, S. Lin and C. L. Mallows, Some realizability theorems group testing, *SIAM J. Appl. Math.* 17 (1979) 396-400.
- [5] G. O. H. Katona, Combinatorial search problem, in *A Survey of Combinatorial Theory*, Ed. J. N. Srivastava et al, (North-Holland, Amsterdam, 1973).
- [6] C. H. Li, A sequential method for screening experimental variables, *J. Amer. Statist. Assoc.* 57 (1962) 455-477.
- [7] J. W. Moon and M. Sobel, Enumerating a class of nested group testing procedures, *J. Combin. Theory*, Series B 23 (1977) 184-188.
- [8] M. Sobel and P. A. Groll, Group testing to eliminate efficiently all defectives in a binomial sample, *Bell System Tech. J.* 28 (1959) 1179-1252.
- [9] A. Sterrett, On the detection of defective members of large populations, *Ann. Math. Statist.* 28 (1957) 1033-1036.