CHALMERS
UNIVERSITY OF TECHNOLOGY

UNIVERSITY OF GOTHENBURG

Searching for Search Strategies

Solutions to Strict Group Testing Problem Instances

Master's thesis in Computer Science – algorithms, languages and logic

DAVID GRANKVIST

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2017

Master's thesis 2017

Searching for Search Strategies

Solutions to Strict Group Testing Problem Instances

DAVID GRANKVIST

Department of Computer Science and Engineering
Division of Computing Science
Chalmers University of Technology and University of Gothenburg
Gothenburg, Sweden 2017

Searching for Search Strategies
Solutions to Strict Group Testing Problem Instances
DAVID GRANKVIST

Supervisor: Peter Damaschke, Computer Science and Engineering
Examiner: Patrik Jansson, Computer Science and Engineering

Master's Thesis 2017
Department of Computer Science and Engineering
Division of Computing Science
Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Typeset in LaTeX
Gothenburg, Sweden 2017

Searching for Search Strategies
Solutions to Strict Group Testing Problem Instances
DAVID GRANKVIST
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg

Abstract

Group testing is the problem of identifying a set of d defectives amongst a much larger set of n elements. To this end, the searcher is able to perform tests on groups of elements, each test revealing whether a defective is present within a group. In multi-stage group testing, the search can be subdivided into s stages where all tests within the same stage are run in parallel. This allows for adaptations to be made mid-search based on partial knowledge, potentially reducing the total number of tests. In strict group testing, there exists the additional possibility that more than d defectives exist. In such scenarios, the searcher is obligated to report this fact.

The individual computation steps of group testing, the tests, are assumed to be incredibly costly, implying that search strategies can not be evaluated based solely on asymptotic optimality. Therefore, the approach is to solve specific problem instances optimally by devising tailor-made strategies. The goal of this thesis work is to devise such strategies for previously unsolved multi-stage strict group testing instances. Part of this goal, as well as being a goal on its own, is the study of related subproblems.

The chosen methodology to accomplish the goals is rather simplistic. Each derivation consists of making assumptions about the number of tests required for the given instance, and then applying various theoretical tools in order to systematically narrow down the search space.

Multiple instances of both the main problem and the related subproblems are solved, with the latter being a key step in accomplishing the former. Moreover, the subproblems are studied further in order to gain insights for future work. In conclusion, it is illustrated in this thesis work both that simple reduction arguments are effective in this context and that it is worthwhile to study subproblems as independent instances.

Keywords: group testing, algorithms, combinatorics, combinatorial design, hypergraph, optimization

Acknowledgements

First of all, I would like to express my sincere gratitude towards my academic supervisor Peter Damaschke for his invaluable guidance, feedback and encouragement throughout this thesis project. I would also like to thank the examiner Patrik Jansson for enthusiastically carrying insightful discussions. Furthermore, thank you Markus Otterberg, Annika Johansson and Roland Hellström Keyte for providing constructive criticism from the perspective of students. Finally, a thanks of immeasurable proportions to my family and friends for their love and support.

David Grankvist, Gothenburg, May 2017

Contents

1

Introduction

This thesis work takes on the challenge of expanding the set of solved instances of the strict group testing problem. To this end, the theoretical foundation from recent papers is applied in order to systematically eliminate vast regions of the problem search space and finally reveal optimal solutions.

The project of which the present thesis is an end product corresponds to 60 hec (higher education credits) rather than the typical amount of 30 hec. The reason for this is to allow for in-depth investigations of broad open questions within the field. However, for the sake of coherence, the focus throughout this report lies on areas within which future research looks promising.

This chapter is arranged as follows. First, in order to explain the group testing problem, the reader is provided with illustrative examples, a list of real-world applications and an informal problem definition. Then the state of the field of strict group testing is outlined, followed by the purpose and scope of this thesis work. Finally, the remainder of the thesis is outlined.

1.1 Examples

This section presents two artificial examples of real-world applications of group testing, in order to provide the reader with intuition for the problem. See section 1.2 for a list of actual group testing applications.

1.1.1 Preventing a pandemic

As a conceptual example, consider the following. Suppose that city X is facing the threat of a potential pandemic. With the aid of the world's most prominent epidemiologists and statisticians, it is predicted that within the next few days, at most d of the n inhabitants of X will be infected, where $d \ll n$. The medical staff of the various hospitals of X label a given citizen as either infected or healthy by taking a blood sample and performing a series laboratory analyses on it.

Unfortunately, this labeling procedure is quite costly in terms of hospital resources due to a shortage of both personnel and medical instruments throughout the city. Consequently, it is impossible in practice to label one citizen at a time until every single citizen blood sample has been examined. Even if this were possible, it would be catastrophically resourceful since it is known that only a small fraction of the population is infected.

There is, however, a silver lining: traces of the disease are very easily detected, even in very large blood samples. This allows for the possibility of combining blood samples into larger ones in order to label groups of citizens simultaneously. Ideally, a given combined blood sample yields a negative result, hence classifying the corresponding group of citizens as healthy. Otherwise, at least one individual in the group is infected and further investigations are required. The core issue, then, becomes to subdivide the n inhabitants of X into the minimum number of groups such that, regardless of the test outcomes, the at most d infected individuals are identified. Finding this subdivision is a group testing problem.

1.1.2 Alternative example

In the previous section, the very basic principles of the group testing problem are illustrated by the means of an artificial real-world example considering the prevention of a pandemic. Note, however, that group testing is a more general problem that is not necessarily restricted to the context of biological testing. Here an alternative real-world application and its analogies to the previous example are briefly outlined.

Consider a massive storage facility for gas containers (analogous to citizens), where there is a small, but nonzero, probability that a given gas container is leaking (analogous to a potentially infected individual). The group testing procedure in this case could be to move a portion of the gas containers to a sealed off area, in which sensitive instruments could find traces of the gas in the air (analogous to combining blood samples into larger ones).

1.2 Applications of group testing

The preceding subsections aim to provide the reader with intuition for the group testing problem and its applicability. This section further elaborates on the latter by listing various real-world applications.

Applications of group testing include:

- searching for syphilis [17] or HIV [25] infection traces in blood samples.
- DNA sequencing in molecular biology [4].

- Built-In Self-Test (BIST) diagnosis in engineering [18].
- various subjects relating to communication networks, such as broadcasting [19], conflict resolution in multiple access channels [20, 21], identifying the dead sensors in a mobile ad hoc wireless network [20] and detecting Denial of Service (DoS) attacks [26].
- additional computer science problems, such as dynamically gathering statistics (in for example data mining) [22], corruption-localizing hashing [23] and hard disk integrity for forensics investigations [24].

1.3 Problem definition

An instance of the group testing problem consists of n elements along with the knowledge that at most d of these elements are defective elements, or defectives. The set of defectives can be identified through a series of binary group tests, or pools. A pool is positive if it contains at least one defective, and negative otherwise. It is typically the case that $d \ll n$, because then testing elements individually is highly inefficient. The solution is an optimal search strategy of binary group tests, meaning a minimum set of pools, such that the defectives are identified regardless of the test outcomes. Hence the solution to the problem is in itself an algorithm.

In strict group testing, there exists the additional possibility that more than d defectives exist. Then the goal of the searcher is to either identify a group of up to d defectives, or to report that more than d defectives exist. To motivate this extension to the problem, consider real-world scenarios in which inaccurate predictions can be made. For example, in the case of preventing a pandemic, the possibility exists that the scientists in question underestimated the contagiousness of the disease.

In multi-stage (possibly strict) group testing, the search can be divided into s stages where the tests belonging to the same stage are run in parallel, and their outcomes are revealed to the searcher in subsequent stages. This can capture time constraints. For example, suppose that, in some application, each test takes one day to execute, and that the search must be completed during the course of five days. Suppose further that multiple tests can be executed on the same day. Then one option is to run all of the tests on the first day, in order to easily satisfy the time constraint. However, one can typically reduce the total number of tests by revealing the outcome of some of the tests before proceeding with the search, because partial knowledge of the solution allows for more intelligent decision-making.

This thesis work considers finding solutions to instances of the strict multi-stage group testing problem. In particular, the focus is on the case of two defectives and two stages, as well as a related subproblem (see section 1.6).

1.4 Previous work

Much research has been made within the field of group testing as a whole, see for example the textbooks [3] and [4]. However, the more restrictive problem of strict group testing, as well as its multi-stage counterpart, are under-researched [1]. Some key results are outlined in this section.

The case of one single defective is completely solved in [2] (see also [9] for partial results and [10] for a related concept). The case of one single stage is the size of a d-disjunct matrix [1]. Bounds for d-disjunct matrices are presented in [11] and research into specific instances is presented in [12].

In [13], bounds for the case of two stages are presented. Furthermore, it is known that in the case of two stages $O(d \log n)$ tests are sufficient [5] (improved upon in [6]). For one stage, however, $O(d \log n)$ tests are not sufficient [7, 8].

The case of two defectives and two stages is provided with bounds in [1] (see also [14] for earlier results).

In [1], strict group testing is expressed in terms of candidate hypergraphs (see section 3.7), which in turn is related to the famous NP-hard Set Basis and Graph Coloring problems (see for example [15] and [16]). This broad picture of strict group testing, accompanied with its game-theoretic interpretation (see [2] and section 3.4), provide a solid theoretical foundation for solving new instances (as illustrated in [1]).

1.5 Purpose

This thesis work aims to apply the theoretical foundation laid out in [1] and the game-theoretic interpretation summarized in [2] in order to solve new strict group testing instances.

Part of this goal is to study special cases of candidate hypergraphs (see section 3.7) in order to break down strict group testing instances into smaller components. This includes both an examination of specific candidate hypergraph instances and to at least partially investigate the additivity-related conjecture for strict nonadaptive group testing on candidate hypergraph products that is suggested in [1].

1.6 Scope

As pointed out in section 1.4, strict group testing is under-researched. Therefore, the goal of solving new instances is a very broad one.

In terms of multi-stage strict group testing problem instances, the focus here is on the case of two defectives and two stages. There are several reasons for this. Firstly, due to the small number of defectives, the problem is inherently structurally simple compared to the general case (this, of course, by no means implies that it is trivially solved). Moreover, since the single-defective case is completely solved, incrementing the number of defectives is a natural direction to take.

The reason for studying the case of two stages is partly for the structural simplicity, but also because it is an interesting case in itself. For example, as previously noted, $O(d \log n)$ tests are sufficient in the case of (at least) two stages. This indicates that this is a very useful case in the context of practical applications with time limitations.

On a further note, the combined case of two defectives and two stages is studied in [1]. This includes upper bounds which limit the search space for a number of instances, as well as solutions to a number of smaller instances. The availability of such theoretical tools further justify research into this specific case.

A common scenario encountered in the two-defective case is that two disjoint pools have positive results. In terms of candidate hypergraphs (see section 3.7), this is represented as a bipartite graph. The investigation of candidate hypergraphs in this thesis focus entirely on this special case. Only a few initial instances of bipartite candidate graphs have been solved [1].

1.7 Thesis outline

In the subsequent chapter, the more general field of combinatorial design theory is briefly presented, along with a few historical key points, leading up to the more modern problem of group testing.

In chapter 3, the reader is provided with the theoretical foundation on which this thesis is based on. First, group testing is defined in a more precise manner, and its variations are discussed. Then some fundamental properties of the problem are shown, such as the relation to game theory and bounds on the number of tests. After this, the reader is introduced to the more general problem of solving strict group testing on a candidate hypergraph. This this followed by an outline of further theoretical tools. Finally, the chapter is concluded with a discussion of the computational complexity of the problem at hand.

Chapter 4 discusses how the concepts from chapter 3 are applied when searching for search strategies. It is explained how various theoretical tools can be applied and how they relate to one another.

Chapter 5 considers the scientific accomplishments this thesis work. The selected instances to solve are motivated and then outlined along with the accompanying strategies. Finally, the corresponding derivations are presented.

In chapter 6, the methods and the attained results are discussed. The significance of the results is questioned and the methodology is evaluated. The remainder of the chapter considers potential future work.

Chapter 7 takes on a philosophical perspective by discussing the inherent limitations of the group testing model in when faced with ethical dilemmas.

Finally, chapter 8 summarizes the thesis as whole, along with some conclusive remarks.

2

Background

In this chapter the reader is presented with a background to the field of combinatorial design theory, of which group testing is a subfield. The aim is to explain how group testing and its related field relate mathematics as a whole, as well as providing a historical context.

2.1 Combinatorial design theory

Group testing is a special case of the more general scientific field of combinatorial design theory. The reader is referred to text books such as [27] for an extensive overview of combinatorial design theory. For the purposes of this text, combinatorial design theory can, in essence, be seen as the study of the arrangement of finite sets according to certain rules. Such arrangements are called combinatorial designs. Interestingly, the problems of study in combinatorial design theory are ostensibly simplistic, even in their formal statements, to the extent that they can typically be grasped by a reader without a rich mathematical background; they often resemble problems from recreational mathematics. The solutions, however, are far from trivial, and can rely on highly abstract mathematical concepts such as groups (not to be confused with the groups in group testing), rings and fields, in addition to common university-level topics such as linear algebra, number theory and combinatorics. Combinatorial design theory is a tremendously motivated field, with applications in tournament scheduling, lotteries, mathematical biology, algorithm design and analysis, networking cryptography and, of course, group testing.

2.2 Combinatorial designs throughout history

It is out of the scope of this paper to provide an extensive outline of the entire history of combinatorial design theory. Rather, this section brings up a few examples of combinatorial designs throughout history, most notably the advent of group testing.

2.2.1 Luoshu square

A very early example of a combinatorial design is the luoshu square, with its first occurrences possibly dating as far back as to the third millennium B.C.E. [28]. The luoshu square is an example of a magic square, which is a famous recreational mathematics puzzle. The objective is to fill an n by n grid with the numbers 1 through n^2, with the constraint that the sum of the n numbers in any row, column or diagonal must have the same value. The particular case of the luoshu square is a 3 by 3 grid. It is however of much higher cultural significance than a mere recreational activity, and its history touches on cosmology, mythology, philosophy, religion, occult practices, mathematics, architecture and music [28].

2.2.2 Latin squares

In the 18th century, the Swiss mathematician and scientist Leonhard Euler (1707-1783) made incredible scientific contributions. One of his mathematical problems of study was that of the Latin square, called so due to Euler's own choice of symbols [29]. A Latin square is an n by n grid where each position in the grid is to be filled with one of n symbols such that each symbol occurs exactly once in every row and column.

2.2.3 Kirkman's schoolgirl problem

In the 19th century, the British mathematician Thomas Penyngton Kirkman (1806-1895) posed the following combinatorics problem [30].

> Fifteen young ladies in a school walk out three abreast for seven days in succession: it is required to arrange them daily so that no two shall walk twice abreast.

This problem is referred to as Kirkman's schoolgirl problem, and was a popular problem of study. So popular, in fact, that it drew much attention from another Kirkman paper [31].

2.2.4 Group testing during the Second World War

During the Second World War, The United States Public Health Service and the Selective Service were faced with the problem of identifying syphilitic soldiers [17]. Traces of syphilis could be find by performing tests on blood samples from individual soldiers. Problematically, however, performing such a test was significantly resourceful. Moreover, only a very small portion of the massive group of soldiers was likely to be infected, which would result in a great deal of unnecessary laboratory procedures.

In 1943, the American statistician Robert Dorfman (1916-2002) published the paper [17] with a proposed solution to this problem. Dorfman observed that diagnostic tests for the syphilitic antigen are extremely sensitive and will show positive results for even great dilutions of antigen. This allows for the possibility of combining blood samples into larger ones by pouring them together, and then examine these larger blood samples. If organized properly, such combinations can greatly reduce the total number of tests. Due to this paper, Robert Dorfman is commonly credited as the founder of group testing [3].

3

Theory

The search for search strategies relies on a considerable amount of mathematical theory. This chapter covers the theoretical foundation on which this thesis work relies. First, group testing is defined in mathematical terms and related to game theory. Then some fundamental bounds on the number of tests required are presented. This is followed by a more general notion of the problem in terms of hypergraphs, which in turn relates to techniques for devising solutions. This, in turn, is followed by an outline of additional theoretical tools which are useful for solving group testing instances. Finally, the chapter is concluded with a discussion of the complexity of the problem at hand.

3.1 Problem definition

The group testing problem considers the following setting. A set of n elements is given, each with the binary label of either being a defective or non-defective. Let P denote the set of defectives. It is known that $|P| \leq d$, where the typical case is that $d \ll n$. A pool, or test, takes a subset Q of the elements and queries whether $Q \cap P \neq \emptyset$. Q is called positive if $Q \cap P \neq \emptyset$, and negative otherwise. Hence if Q is negative, all elements in Q are immediately revealed to be non-defective. If Q is positive, then at least some possibilities of P are ruled out. The goal is to find the set P by using a minimum number of tests. More precisely, the goal is to use the minimax number of tests (see section 3.4).

It is often convenient to refer to a specific element as being the ith element according to some arbitrary order. This order, however, must be consistent; the ith element must be unambiguously defined. Let v_i denote the ith element in this arbitrary, yet specific order. Formally,

Definition 1. *Let ω be a bijection from the element set to the set $\{1, 2, \ldots n\}$ and let v_i denote the element u such that $\omega(u) = i$. Moreover, let v_i be referred to as the ith element.*

The preceding definition allows for alternative way of denoting a pool Q, called its

indicator vector. The indicator vector representing Q is a binary vector of n slots which has a 1 at slot i if $v_i \in Q$, and 0 otherwise. Let this vector be denoted by the corresponding binary sequence (in the order specified by the bijection ω) surrounded by square brackets. For example, if $n = 5$, then $\{v_1, v_3\} = [10100]$.

3.2 Temporal considerations

A relevant question to ask is at which point in time during the search the outcome of a given pool becomes known to the searcher. A succinct answer is that it depends on which group testing model is preferred in the given context. In nonadaptive group testing, all of the tests are run in parallel. Hence, in this case, the outcome of one pool does not affect the construction of any other. The downside of this is that searcher is not able to gather partial knowledge of P during the search, which implies an excessive amount of testing. A nonadaptive search strategy is also called a pooling design.

In contrast to this is adaptive group testing, where tests are run sequentially. In this case, the construction of a subsequent pool depends on the outcome of all previous pools. While this maximizes the searcher's instantaneous knowledge during the search, the downside of this model is that it does not capture real-life time constraints very well. If each test is time-consuming, parallelization becomes a necessity.

The compromise between nonadaptive and adaptive group testing is multi-stage group testing. In this case, the search can be divided into at most s stages, in which sets of tests can be run simultaneously. This can be seen as a tree of nonadaptive strategies, where a given stage branches out in the possible outcomes of its set of pools. Interestingly, already $s = 2$ is sufficient to achieve $O(d \log n)$ tests, which is asymptotically optimal [5].

3.3 Strict and hypergeometric group testing

As stated in section 3.1, it is assumed that $|P| \leq d$. However, blind faith in this assumption is not always applicable. Recall chapter 1 in which a group of epidemiologists and statisticians predict that at most d infected citizens will exist in the near future. Here an inaccurate prediction can have dire consequences, which is one possible motivation of the strict group testing model. In strict group testing, there exists the additional possibility that $|P| > d$. The goal of the searcher then becomes to either identify at most d defectives, or simply report that more than d defectives exist.

In contrast, the model of blind faith is called hypergeometric group testing. This of course does not render this model useless, merely context-dependent; in a scenario with less emphasis on error aversion, one might benefit from this less restrictive model.

3.4 Game-theoretic interpretation

In this section, group testing is related to game theory. There are two benefits in this: 1. it allows for some precise definitions within the scope of group testing and 2. it provides a way of reasoning when solving group testing problems.

A two-player zero-sum game is, in very basic terms, defined as follows. There exist two players, called the player and the adversary, respectively. The specific details of how the game is played are irrelevant, except the following. The player seeks to maximize a utility value t using a strategy S and the adversary seeks to minimize t using a strategy A. In other words, any point of utility gained for the player is a loss for the adversary, and vice versa. Hence the sum of their utilities is zero; there is no mutual benefit of the players.

In the context of group testing, the player searches for P with strategy S, attempting to minimize the number of tests, and in response the adversary chooses P with strategy A in order to maximize the number of tests in strategy S. Define $t(A, S)$ to be the number of tests used when strategies of S and A counteract. Now we define the test number $t(n, d, s)$.

Definition 2. *For a strict group testing instance with parameters n, d and s, let the test number $t(n, d, s)$ be defined as $t(n, d, s) = max_S min_A t(A, S)$.*

Note that the order of the *max* and *min* applications is irrelevant [2]. Intuitively, $t(n, d, s)$ is the number of tests of a search strategy that has the best possible worst-case scenario. In the hypergeometric case, define analogously $t_h(n, d, s) = max_S min_A t_h(A, S)$.

3.5 Monotone properties

This section briefly justifies the monotone properties of the strict group testing numbers (the arguments for the hypergeometric case are completely analogous). These constantly serve as convenient lower and upper bounds by relating instances

to each other. Let $n' \geq n$, $d' \geq d$ and $s' \leq s$. Then

$$t(n, d, s) \leq t(n', d', s')$$

Proof. Assume that $t(n, d, s) > t(n', d', s')$. Consider the strict group testing instance with parameters n, d and s. By way of contradiction, the allegedly optimal strategy for this instance can be outperformed as follows. Firstly, add $n' - n$ auxiliary elements which are chosen to be negative. Secondly, observe that the searcher is not obliged to utilize all s stages, but can choose to use only s' of them. Finally, apply the strategy of $t(n', d', s')$ tests to this modified instance, with one slight adaption. If $d < |P| \leq d'$, simply discard the solution and report that $|P| > d$. It follows that the instance with parameters n, d and s can be solved using a $t(n', d', s')$-test strategy, which in turn implies that fewer than $t(n, d, s)$ tests are required. This contradicts the definition of $t(n, d, s)$. □

Moreover, it is the case that

$$t(n, d, n) \leq t(n', d', n')$$

Note that the n and n' stages, respectively, refer to adaptive group testing. To see this, consider the strategy of testing each element individually in its own stage. Even in this naive approach at most n (or n') stages are used. Hence $t(n', d', n') \geq t(n, d, n') = t(n, d, n)$.

3.6 Information-theoretic lower bound

When designing a strategy for a specific group testing instance, the typical initial step is to establish bounds based on some special symmetries of the given problem. However, prior the discovery of such symmetries, it is useful to have some reliable, albeit less efficient, bounds on the search space, based on the more fundamental properties of the group testing problem as a whole. One example of such a bound is the monotonicity of the testing numbers, which allows for a comparison between the problem at hand and previously solved instances. The downside of this approach is its reliance on having solved other instances. For example, suppose that one studies a family of instances which share some symmetries, but are far apart in the search space. In such cases monotone bounds are not necessarily efficient. In this section we show that there in fact exists a very simple lower bound which does not rely on having solved other instances. This bound is called the information-theoretic bound.

The information-theoretic bound is simply $\log_2$ of the number of possible answers

that the given search strategy can provide. More precisely, an answer identifies which elements belong to the true set of defectives P. In the context of strict group testing, all of the scenarios where $|P| > d$ are grouped together as if being equal; the strictness criterion adds only one further possible answer. Hence, in the strict case, the number of possible answers X is the number of possible sets of size at most d plus the additional possibility that $|P| > d$. That is, $X = \sum_{j=0}^{d} \binom{n}{j} + 1$. Hence the information-theoretic bound is $t(n, d, s) \geq \log_2 X$, as shown later in this section. First, however, the overall proof approach is laid out.

Due to the monotonicity of test numbers, it suffices to argue that $t(n, d, n) \geq \log_2 X$. That is, it suffices to argue that an adaptive strategy uses at least $\log_2 X$ tests. In order to conveniently argue about an arbitrary adaptive strategy α, one can view α as a binary search tree; this is at least a more concrete interpretation than the abstract notion of a game-theoretic strategy. Recall that by definition any pool has two possible outcomes: positive or negative. Furthermore, the tests of α are run sequentially since α is adaptive. These properties allows for a binary tree representation of α as follows. Let Q denote any pool of α and let Q^+ and Q^- denote the subsequent pool when Q has a positive or negative outcome, respectively. Let Q be represented by a node and let Q^+ (arbitrarily) be the left child and Q^- the right child. Similarly, the subtrees of Q^+ and Q^- show how the strategy proceeds. The leaf nodes of this tree represent the X possible testing outcomes and are in some sense auxiliary, since they do not represent pools. This tree representation of α bridges the gap between strategies and logarithms and allows for a way of arguing about the information-theoretic bound, as is shown below.

$$t(n, d, s) \geq \log_2 X$$

Proof. It suffices to show that $t(n, d, n) \geq \log_2 X$. Consider the adaptive strategy α. The number of tests α needs to reach a particular outcome is the number of edges of the path from the root node to the corresponding leaf. Hence the test number $t(n, d, n)$ is the length of the longest such path. First suppose that the α tree is a complete binary tree. Then, if the root node level is referred to as level 0, the number of nodes on level i is 2^i. Hence, if α is complete, the length of any path from the root node to level i has length $\log_2$ of the number of nodes in level i. This is also true for the bottom level, which means that $\log_2$ of the number of outcomes reflects any root-to-leaf path length in the complete binary tree. Hence, if α is complete, the $t(n, d, n) = \log_2 X$.

Now suppose that α is not complete. Then it does not follow that there are 2^i nodes on level i, implying that $\log_2$ is not the path length to a particular level. It remains to show that even in this case $\log_2 X$ is a lower bound for $t(n, d, n)$. Consider the following procedure. Add auxiliary nodes to α in order to make it a complete binary tree. For each path in α that has y fewer levels than the worst-case scenario than the deepest path, that is the worst-case scenario, add two auxiliary complete subtrees with y levels. With other words, unnecessary tests are added until all root-to-leaf paths in the tree are as long as in the worst-case scenario. Let Y denote the number

of leaves in the resulting complete binary tree. Now $\log_2 Y$ is equal to the length of the longest root-to-leaf path in the original α tree, that is $t(n, d, n) = \log_2 Y$. Since $\log_2$ is monotone it follows that $X \leq Y \implies \log_2 X \leq \log_2 Y$, which shows that $\log_2 X$ is a lower bound in this case as well. Thus, $t(n, d, n) \geq \log_2 X$, implying that $t(n, d, s) \geq \log_2 X$. □

As a final remark, note that $\lceil \log_2 X \rceil$ is a lower bound. If $\log_2 X$ is integer, then this follows directly. Now suppose $\log_2 X$ is not integer. Then $t(n, d, s) \geq \log_2 X > \lfloor \log_2 X \rfloor$. It is known that $t(n, d, s)$ is integer by its definition, from which it follows that $t(n, d, s) \geq \lfloor \log_2 X \rfloor + 1 = \lceil \log_2 X \rceil$, which we wanted to show.

3.7 Candidate hypergraphs

This section considers the representation of strict group testing as a candidate hypergraph. This allows for useful observations to be made about both the problem at hand and its related subproblems.

3.7.1 Definition

As many computer science problems do, group testing has connections to graph-theoretic concepts. Recall that a graph consists of a set V of vertices (or nodes) and a set E of edges, which are (possibly directed) pairs of vertices. A hypergraph is a generalization of this concept where each edge is an arbitrary set of vertices, rather than a pair. More precisely, a hypergraph H is a pair (V, E) where $E \subseteq Pow(V)$ (with $Pow(V)$ denoting the power set of V).

In the context of group testing, a useful hypergraph to consider is the candidate hypergraph. A candidate element is an element that has not yet appeared in a negative pool. A candidate set is a set of up to d elements that may be the true set of desired elements P, according to previous test results. Thus, the instantaneous knowledge during a search can be represented as a hypergraph consisting of the candidate elements along with all of the candidate sets. This hypergraph is called the candidate hypergraph. For example, prior the search, the vertex set V consists of all elements and the edge set E consists of all $\sum_{j=0}^{d} \binom{n}{j}$ sets of up to d elements. Note that the additional possibility (in the strict setting) that $|P| > d$ is not captured by the candidate hypergraph, because then the searcher would always begin with $E = Pow(V)$, which does not meaningfully represent the problem at hand.

The candidate hypergraph representation of the problem has a further strength: it allows for a formulation of a more generalized group testing problem on a candidate hypergraph. This is extremely useful, both in the study of special cases of the problem (see section 3.7.2) and in the connection to well-known NP-hard problems

(see section 3.7.3). Analogous to $t(n, d, 1)$, the nonadaptive test number for solving strict group testing on a candidate hypergraph is defined in game-theoretic parlance (see section 3.4).

Definition 3. *For a strict group testing represented by the candidate hypergraph H, let the nonadaptive test number $t(H)$ be defined as $t(H) = max_S min_A t(A, S)$.*

3.7.2 Products and sums

The candidate hypergraph representation of the group testing problem gives rise to the question of which special cases of the problem are of interest. Suppose that some category C of candidate hypergraphs appears frequently during searches. If C happens to be well-studied, the searcher's efforts are alleviated. This section considers two such categories: products and sums. Formally, the product of two hypergraphs is defined as follows

Definition 4. *Let $H = (V, E)$ and $H' = (V', E')$ be two hypergraphs. Then their product $H \times H'$ is defined as $H \times H' := (V \cup V', \{a \cup b \mid (a, b) \in E \times E'\})$, where $E \times E'$ refers to the Cartesian product.*

In words, the resulting vertex set is the union of the two vertex sets and the resulting edge set is the set of all sets that can be formed by taking exactly one edge from each of E and E' and joining these two edges.

The (arguably) most intuitive case of a candidate hypergraph product is when two disjoint pools are both positive, so the searcher knows that the set of defectives P contains elements from both of those pools. The study of candidate hypergraph products in this thesis work focuses entirely on (strict and nonadaptive) solutions to the bipartite candidate graph $K_{x,y}$.

Note that, for the hypergraph $H \times H'$, there exists the possible strategy to solve H and H' in isolation and combine the results afterwards. This gives the following upper bound.

Lemma 1. *For two hypergraphs H and H', it is the case that $t(H \times H') \leq t(H) + t(H')$.*

Little is known regarding lower bounds to such problems, though it is conjectured in [1] that $t(H \times H') \geq t(H) + t(H') - 1$.

Even though candidate hypergraph products are under-researched as a whole, the following special case has been completely solved.

Theorem 1 [1, Theorem 8]**.** *Let G_1 be the candidate hypergraph with hyperedges $\{v_1\}$ and $\{v_2\}$ (that is, exactly one of these 2 elements is defective), and G_2 be an arbitrary candidate hypergraph. Then nonadaptive strict group testing on $G_1 \times G_2$ needs $t_2 + 2$ tests.*

Another useful class of subproblems to study is that of candidate hypergraph sums. Define the (disjoint) sum of two candidate hypergraphs $H = (V, E)$ and $H' = (V', E')$ as $H + H' = (V \cup V', E \cup E')$, where $V \cap V' = \emptyset$. This represents an *or* relation regarding the set P of defectives: either $P \in E$ or $P \in E'$, but never both (unless $|P| > d$).

The study of products and sums (and possibly other special cases) is useful for two main reasons. Firstly, as previously mentioned, it is convenient to be prepared for a frequently occurring scenario. Secondly, it gives the possibility to decompose a larger problem into smaller, more manageable parts. Say, for example, that due to practical reasons the maximum pool size is restricted. In such cases, one can still choose to solve sections of the problem optimally, even though the solution may be suboptimal in its entirety.

3.7.3 Set bases and conflict graphs

This section considers the relation of group testing to both the Set Basis Problem and the ostensibly different, yet strongly related, Graph Coloring Problem on a (later to be defined) conflict graph. These relations provide theoretical tools which are crucial in the derivation of the test numbers in the present thesis.

The Set Basis Problem is the following. Given a set of sets $S = \{S_0, S_1 \dots S_l\}$, find a minimum set of sets $B = \{B_0, B_1 \dots B_m\}$ such that each set S_i is the union of some of the sets B_j. The set of sets B is called a (minimum) set basis of S. The Set Basis Problem relates to strict group testing on a candidate hypergraph as follows.

Theorem 2 [1, Theorem 1]**.** *A pooling design is strict if and only if the complement of every candidate set is the union of some pools.*

In other words, the set of pools must be a minimum set basis of the set of candidate complements. This observation can rule out many suboptimal pooling designs (as seen in chapter 5).

The relation to graph coloring is less straightforward and a few steps are needed in order to define the graph to color, called the conflict graph. First, note that an alternate formulation of Theorem 2 is the following.

Corollary 1 [1, Corollary 5]**.** *A pooling design is strict if, and only if, for every pair of a candidate set C and a candidate element $v \notin C$, some pool Q exists such that $v \in Q$ and $C \cap Q = \emptyset$.*

Corollary 1 serves as an intermediate step when relating group testing to graph coloring by relating pairs (C, v) to the original problem. Each such pair is a node in the conflict graph, though expressed in different terms, as is defined below. Recall from section 3.1 that v_i denotes the ith element in a specific order of the elements.

Definition 5. *Let (C, v) be a pair of a candidate set C and an element $v \notin C$. Then*

the partial vector p representing (C, v) is defined as having the following entries p_i

$$p_i = \begin{cases} 0, & v_i \in C \\ 1, & v_i = v \\ *, & otherwise \end{cases}$$

where $*$ is a special value called an open value.

The notation for partial vectors is similar to that of indicator vectors (see section 3.1), except that there exists a third possible value '$*$'. More precisely, a partial vector p is denoted as its in-order (increasing i) sequence of ternary values p_i, surrounded by square brackets. For example, if $n = 7$, then the pair $(\{v_1, v_2, v_5\}, v_4)$ corresponds to the partial vector $[00{*}10{**}]$. Two partial vectors are said to be in conflict if they have a 0 and 1 at the same position. This allows for a definition of the conflict graph.

Definition 6. *Let H be a candidate hypergraph. Then its conflict graph $G = (V, E)$ is defined as follows. V is the set of all partial vectors corresponding to the H instance and E is the set of pairs of conflicting partial vectors.*

Finally, this conflict graph is related to the problem at hand as follows. A pool Q is said to cover a partial vector p if every non-open value of p is equal to that of the indicator vector of Q. For example, the pool $[001]$ covers $[0{*}1]$ but not $[01{*}]$. Note that Corollary 1 states that every partial vector must be covered by some pool. Further note that if two partial vectors are covered by the same pool, then they can not be in conflict. Hence, each pool covers an independent set in the conflict graph. Then the nonadaptive strict group testing problem becomes the problem of covering the conflict graph vertices with a minimum number of independent sets. This is exactly the problem of graph coloring, which gives us the following result. Recall that the chromatic number of a graph is the minimum number of colors needed to ensure that no two adjacent vertices share the same color.

Theorem 3 [1, Theorem 6]**.** *Solving the strict group testing problem nonadaptively for a given candidate hypergraph is equivalent to coloring the conflict graph. Consequently, the conflict chromatic number equals the number of pools required.*

Not only does Theorem 3 provide an interesting connection to another NP-hard problem, but it also allows for reasoning about group testing using the perspicuous notation of partial and indicator vectors.

3.8 Additional theoretical tools

This section considers theoretical tools that are not as fundamental as those previously brought up. Nevertheless, they are frequently applied in the derivations of test numbers in this thesis work.

As discussed in section 3.6, it is useful to have bounds available that do not rely solely on monotone properties. One such bound is provided by Lemma 2. The lemma is expressed in terms of antichains, which we define here. Two sets S and S' are called incomparable if neither $S \subseteq S'$ nor $S' \subseteq S$. An antichain is a set of incomparable sets. The bound is the following

Lemma 2 [1, Lemma 11]**.** *If $m > 2^r$ candidate sets remain which form an antichain, then at least $r + 2$ more tests are needed, even adaptively.*

Despite being a rather simple statement, this lemma does in fact often provide more efficient means for reducing the search space than the information-theoretic bound. Note that as the lemma considers candidate sets in general, it is applicable in the context of any candidate hypergraph rather than the original group testing problem statement. Further note that Lemma 2 is applicable in any stage during the search.

The following two lemmas are simply game-theoretic monotonicity observations. Again, note that the statements consider group testing on arbitrary candidate hypergraphs.

Lemma 3 [1, Lemma 12]**.** *Suppose that the adversary, in response to a given deterministic test strategy, applies a test answering strategy A that enforces t tests in the worst case. If the searcher replaces some pool Q, that is negative (positive) in A, with a subset (superset) of Q, then still at least t tests are needed in the worst case.*

Lemma 4 [1, Lemma 13]**.** *Suppose the adversary reveals the outcome of some pools in a stage, and then allows the searcher to redesign the remaining pools of this stage from scratch. If t further tests are not sufficient despite redesign, then they are not sufficient for the original problem instance either.*

The remaining two theoretical tools considered in this section consider specific group testing instances rather than the more general candidate hypergraphs. As test numbers for the case of two defectives and stages are part of the main focus of this thesis, the following proposition is highly relevant.

Proposition 1 [1, Proposition 15]**.** *For $n \leq \binom{m}{2} + 3$ we have $t(n, 2, 2) \leq m + 3$.*

Finally consider the case of a single defective. As mentioned in section 1.4, this problem has been completely solved for any number of stages. Even though Theorem 4 aims to summarize the solutions to all instances with one defective and two stages, it is mainly used in this thesis work as a convenient reference for single-stage test numbers. In Theorem 4, define $L(n) = \lceil \log_2 n \rceil$.

Theorem 4 [2, Theorem 6]**.** *For all n, the optimal deterministic strict group testing strategy for 1 defective in 2 stages requires $t(n, 1, 2) = L(n)+1$ tests in the worst case. For almost all n, the second stage is actually needed to accomplish this test number, with the following exceptions where one stage suffices: $n = 1, 2, 3, 5, 6, 9, 10, 17, 18, 19, 20, 33, 34, 35, 65, 66, 67, 68, 69, 70$. Moreover, more stages do not improve the worst-case test number, that is, we have $t(n, 1, 2) = t(n, 1, s)$ for all n and all $s \geq 2$.*

3.9 Complexity

Algorithm complexity analysis typically consists of establishing bounds in terms of asymptotic growth, expressed with big O notation. Even though such bounds give a rough idea of performance for large problem instances, sometimes one needs to be more specific; the actual number of computation steps may need to be provided. Moreover, it is not necessarily optimal to apply a general approach to a specific problem instance. If individual computation steps are very expensive by some measure, then it is relevant to develop tailor-made approaches, even for small instances of the problem at hand. This thesis work considers such tailor-made approaches to instances of the strict group testing problem.

The problem of finding an optimal tailor-made search strategy is of staggeringly high computational complexity, because the number of feasible solutions is doubly exponential in the problem size. To see this, consider an instance with n elements and at most d defectives, and suppose that we know that exactly t tests are required. For a given pool Q, each element is either contained in Q or not. Supposing that $Q \neq \emptyset$, this gives $2^n - 1$ possibilities for the construction of Q. There are t such pools Q, so, assuming that no two pools are equal, this gives a total number of $\prod_{i=1}^{t}(2^n - i)$ possible pooling designs. It follows from $t \ll 2^n$ that $2^n - i \approx 2^n$, implying that $\prod_{i=1}^{t}(2^n - i) \approx (2^n)^t$. In the case of $s \geq 2$ it is known that $t \in O(d \log n)$ [5], while for $s = 1$, even further tests are required in the general case [7, 8]. Clearly, even $(2^n)^{O(d \log n)}$ grows very rapidly with n and d.

Even though the observations above give some indication of the complexity in terms of the search space size, there are further complications to consider. Firstly, it was assumed that t was known. Typically, considerable effort is put into determining the exact value of t. In fact, in case of this thesis work, the main challenge was to determine t, while the accompanying strategies were trivial modifications of strategies of previously solved instances. This is not entirely representative, though, as finding t does not necessarily imply finding the accompanying strategy.

Secondly, in the case of multi-stage strategies one must decide not only which elements are included in which pools, but also in which stage a given pool is to be placed. For example, suppose that there are x possible sets of t pools and that one is allowed to use up to s stages. Each strategy of exactly $i \leq s$ stages has at least one pool in each stage by definition. Hence the number of i-stage strategies is equal to

the number of ways to distribute the remaining $t - i$ pools among the i stages. This gives i^{t-i} possibilities. Hence, there are $\sum_{i=1}^{s} i^{t-i}$ ways of distributing a given set of t pools across the s stages. Finally, multiply this by x to take into account all of the possible sets of t pools, resulting in $x \sum_{i=1}^{s} i^{t-i}$ possible multi-stage strategies. Recall that x is very large, so multiplying it by even a small constant greatly increases the number of possibilities.

There is yet another dimension to the problem which one must take into account. So far in this section, the focus has been entirely on the magnitude of the search space. However, a hugely important question is the following: Given a t-test strategy, how does one verify that this strategy is a feasible solution to the group testing problem instance? One must be certain that the set of up to d defectives is always identified and there are $\sum_{i=0}^{d} \binom{n}{i}$ possible such sets (plus one possibility in the case of strict group testing where additional defectives is a possibility). Even considering the 2^t possible test outcomes quickly becomes computationally intractable as t increases.

As indicated in this subsection, applying a computational approach to the group testing problem is a challenge of considerable difficulty. This, however, does not prevent one from applying theoretical tools in order to narrow down the vast space of possibilities. The latter approach is the focus of this thesis work, as elaborated on in the following chapter.

4

Methodology

This chapter considers the question of how the problems at hand are approached. The aim is to explain in more concrete terms how the theory from the previous chapter is applied in this thesis work in order to solve strict group testing instances. The main theoretical tools are covered in order of application along with illustrative examples.

4.1 Proof approach

The approach, in its most basic terms, is the following. Faced with the question whether t or $t+1$ tests are required, assume that t tests are sufficient and systematically rule out all possible pooling designs using the tools from chapter 3, thus concluding that $t+1$ tests are required. Of course, one can not in general be certain that t tests are insufficient, but this happens to be the case for the particular instances considered in this thesis. Even in cases where the contrary is true, the procedure described here can be used in order to reduce the search space of t-test strategies.

The reduction process is now described in more concrete terms. Note first that the applications of Proposition 1 are self-explanatory and Theorem 4 is used as a convenient reference for looking up (small) $t(n,1,1)$ numbers (both Proposition 1 and Theorem 4 are stated in section 3.8). The first step taken here is to consider the possible sizes of a single pool in the very first stage. The number of possible pool sizes can in many cases be greatly reduced using a combination of Lemma 3 and Lemma 4. Consider the following illustrative example. The goal is to show that $t(10,1,1) = 5$. Suppose that one of the pools Q has size 3. It is known from Theorem 4 (see section 3.8) that $t(7,1,1) = 5$. Now, if Q is negative, Lemma 4 reveals that one is required to use $t(7,1,1) = 5$ further tests, regardless of what the remainder of the pooling design looks like. Hence Q can not be part of a 5-test strategy. Furthermore, Lemma 3 reveals that $|Q| > 3$. Similarly, if $|Q| \geq 7$, Q may instead be positive. Intuitively, it is inefficient to have small negative pools or large positive pools.

Once the number possible pool sizes has been reduced, one possible next step is to see what happens in a subsequent stage. Here it is often useful to apply Lemma 2 (see section 3.8). For example, all candidate sets of size 2 form an antichain, which is exactly the observation made in this thesis work as Lemma 2 is applied. More precisely, suppose that $d = 2$, that the goal is to use 6 tests, where 3 tests are performed in the first stage. Now, if $5 > 2^2$ candidate 2-sets remain in the second stage, then Lemma 2 reveals that $2 + 2 = 4$ further tests are required. Hence one immediately realizes that either stage 1 of the proposed strategy is incorrectly designed, or that in reality 7 tests are required in total.

In situations where Lemma 2 is not applicable due to a shortage of candidate sets, one possible direction is to consider the remaining subproblem in terms of candidate hypergraphs. In such cases, Theorem 1 (see section 3.7.2) can be a useful tool to find lower bounds, since that theorem considers small candidate hypergraphs which are typically subproblems of the problem at hand.

If all of the previous steps have been applied and some subproblem still remains, the final step is to 'pull out the big guns' and apply either Theorem 2 or Theorem 3 (see section 3.7.3). Due to their generality, there are many ways in which one can apply these theorems. One strength of Theorem 2 is that it is stated in terms of the original problem, in contrast to Theorem 3 which considers an equivalent problem phrased in quite different terms. For example, by applying Theorem 2 one may arrive at conclusions such that an element v is contained in at most 4 pools, because otherwise some candidate set complement is not a union of pools. On the other hand, one strength of Theorem 3 is that it is often convenient to argue about perspicuously notated partial vectors rather than sets that intersect in complicated ways. For example, if only one further test can be made and two conflicting partial vectors are still uncovered, then either the current strategy is suboptimal or further tests are required in order to solve the problem instance. Conclusively, it is useful to alternate between these two perspectives while searching for search strategies.

5

Results

The scientific accomplishments of this thesis work are presented in this chapter. First, the results are explained on a higher level and the particular choices of instances to solve are motivated. This is followed by an outline of the solutions to the selected instances, that is the discovered test numbers along with their accompanying strategies. The remainder of this chapter justifies the results with mathematical proofs. For the sake of readability, the proofs are subdivided into sections with descriptive headlines. The more involved derivations include context-specific lemmas.

5.1 Overview of results

The main goal of this thesis work is to find solutions instances of the strict group testing problem for the case of two defectives and two stages, that is $t(n, 2, 2)$, and the respective strategies. The solutions for a few such problem instances are presented in this chapter, namely the instances with $n = 14, 20, 21, 30, 31$. The motivation for this ostensibly counter-intuitive selection is the following. In [1], the upper bound stated in Proposition 1 is derived and then applied along with monotone lower bounds in order to find multiple $t(n, 2, 2)$ numbers (the strategies follow directly from the bound derivation), with the largest instance being $n = 24$. However, Proposition 1 alone is not sufficient for solving all instances in this range, leaving the instances with $n = 14, 19, 20, 21$ open. In the present paper, all of these except $n = 19$ are solved, as well as the additional instances with $n = 30, 31$. The motivation for the latter selection is that the newly attained monotone bounds allow for simple derivations.

An essential step for finding the $t(n, 2, 2)$ numbers in the present paper is to solve nonadaptive strict group testing problem on bipartite candidate graphs, that is to find $t(K_{x,y})$ numbers and the respective strategies. These investigations do not merely aim to find $t(K_{x,y})$ numbers as a means of finding $t(n, 2, 2)$ numbers, but also to study the bipartite candidate graph further in order to gain insights for future work. The particular instances solved as part of this thesis work are those within the range of $4 \leq x, y \leq 6$, as well as those of the form $K_{3,x}$ with $5 \leq x \leq 35$. Again, the selection requires justification. Prior this thesis work, all instances up to the

size of $K_{3,4}$ were solved, while any instance beyond this remained open [1]. Hence $K_{3,5}$ and $K_{4,4}$ are reasonable starting points. Further previously solved instances are those of the forms $K_{1,x}$ and $K_{2,x}$, for all x. Hence a natural continuation is to consider the case of $K_{3,x}$.

A minor goal is to at least partially investigate in which cases candidate hypergraph product test numbers are additive, that is when, for two candidate hypergraphs H and H', it is the case that $t(H \times H') = t(H) + t(H')$. No generalized results for this are provided here, but it is shown that many $K_{3,x}$ instances are additive, which possibly indicates that $t(K_{3,x}) = 3 + t(x, 1, 1)$ for $x \geq 5$.

5.2 Solutions to selected instances

This section serves as a list of the accomplishments of this thesis work. Test numbers of instances that have equivalent strategies are grouped together, followed by a description of that strategy.

$\mathbf{t(K_{3,5}) = t(K_{3,6}) = 7}$.
$\mathbf{t(K_{3,7}) = t(K_{3,8}) = t(K_{3,9}) = t(K_{3,10}) = 8}$
$\mathbf{t(K_{4,5}) = t(K_{5,5}) = t(K_{4,6}) = t(K_{5,6}) = t(K_{6,6}) = 8}$.
$\mathbf{t(K_{3,11}) = t(K_{3,12}) = \cdots = t(K_{3,20}) = 9}$.
$\mathbf{t(K_{3,21}) = t(K_{3,22}) = \cdots = t(K_{3,35}) = 10}$
Strategy: Solve the two disjoint parts as independent one-defective instances.

$\mathbf{t(K_{4,4}) = 7}$.
Strategy: Add one individual test to the $K_{3,4}$ strategy presented in [1].

$\mathbf{t(14, 2, 2) = 9}$.
$\mathbf{t(20, 2, 2) = t(21, 2, 2) = 10}$.
$\mathbf{t(30, 2, 2) = t(31, 2, 2) = 11}$.
Strategy: See [1] for the strategy justifying the bound in Proposition 1.

5.3 Derivations for bipartite candidate graphs

This section considers derivations for test numbers of the form $t(K_{x,y})$. First, the reader is provided with a quick reference to the previously known such test numbers, since these are frequently referred to within the derivations. This is followed by a section covering some key terminology and notation. The subsequent subsections consider the actual derivations.

5.3.1 Previously solved instances

For $x > 1$, the solved special cases of arbitrary size are $t(K_{1,x}) = t(x, 1, 1)$ [1] and $t(K_{2,x}) = 2+t(x, 1, 1)$, the latter being a special case of Theorem 1. Note that in the case of $K_{1,x}$, one defective is already identified, implying that the problem reduces to the one-defective case. The other known $t(K_{x,y})$ numbers [1] are summarized in the following lemma.

Lemma 5. *For $1 < x, y \leq 3$, it is the case that $t(K_{x,y}) = t(x, 1, 1) + t(y, 1, 1)$. In addition, $t(K_{3,4}) = 6$.*

This lemma serves as a convenient reference in the derivations. Note that for $n \leq 4$ it is the case that $t(n, 1, 1) = n$, by Theorem 4.

5.3.2 Terminology and notation

In $K_{x,y}$, an element is said to be 'on the left side' or a 'left-side element' if it belongs to the vertex set of which the variable x refers to and 'on the right side' or a 'right-side element' otherwise. Similarly, a pool consisting solely of left-side elements is referred to as being 'on the left side' or a 'left-side pool', with analogous definitions for the case of right-side elements. To avoid confusion, a graph $K_{x,y}$ is consistently denoted as such rather than $K_{y,x}$. In that way, the subscript to the left of the comma refers to the left side, and vice versa. This informal terminology serves as a mere convenience.

Recall from section 3.1 that v_i denotes the ith element in a specific order. A pool Q is called a v_i-pool if $v_i \in Q$. Throughout the remainder of section 5.3, it is assumed that for $1 \leq i \leq x$, v_i is on the left side and for $x < i \leq y$, v_i is on the right side.

Before some notation regarding indicator vectors and partial vectors is outlined, the reader is provided with a brief reminder of these two notions and their relation. An indicator vector (see section 3.1) is a representation of a pool Q as a binary sequence, where the ith digit is 1 if $v_i \in Q$ and 0 otherwise. A partial vector (see section 3.7.3) is a ternary sequence representing a pair (C, v) of a candidate set C and an element $v \notin C$, where the ith slot is 1 if $v_i = v$, 0 if $v_i \in C$ and the open value '$*$' otherwise. A pool covers a partial vector if there exists no slot where one has a 1 and the other has a 0. Solving strict group testing nonadaptively on a candidate hypergraph is equivalent to covering all partial vectors with a minimum number of pools, by Theorem 3.

Let '$-$' denote an unknown value in an indicator vector or a partial vector. This is not to be confused with the open value '$*$'. For example, $[---]$ can refer to any partial (or indicator) vector, while $[--*]$ specifically refers to a partial vector with an open value in its third slot. Let '$\ldots$' denote that a (known or unknown) portion of a partial (or indicator) vector is not explicitly stated.

5.3.3 The case of three left-side elements

This section considers derivations for test numbers of the form $t(K_{3,x})$. First, two smaller instances are solved. Part of the corresponding argument is then generalized to a lemma, which is then applied in the derivations of further test numbers. Finally, the section is concluded with additional generalizations which may be useful for future studies into the $K_{3,x}$ problem.

$\mathbf{t(K_{3,5}) = t(K_{3,6}) = 7}$.

Proof. For the upper bound, $t(K_{3,6}) \leq t(3,1,1)+t(6,1,1) = 3+4 = 7$, by Lemma 1 and Theorem 4. It remains to show that $t(K_{3,5}) \geq 7$. Assume by way of contradiction that $t(K_{3,5}) \leq 6$. Consider a pool Q.

Pool sizes

Suppose first that $|Q| = 1$. If Q is a negative left-side pool, then there must exist enough tests to solve the remaining subproblem $K_{2,5}$, by Lemma 4. Hence the total number of tests required is at least $1 + t(K_{2,5}) = 1 + 2 + t(5,1,1) = 7$, by Theorem 1 and Theorem 4. Similarly, if Q is a negative right-side pool, then at least $1+t(K_{3,4}) = 1+6 = 7$ tests are required, by Lemma 5. It follows that $|Q| \geq 2$.

Suppose that $|Q| = 2$. If Q is a positive left-side pool, then the remaining subproblem contains $K_{2,5}$, and the situation is equivalent to having a negative left-side pool of size 1. If Q is a negative right-side pool, then $K_{3,3}$ remains to be solved, by Lemma 4, which requires $1 + t(K_{3,3}) = 1 + 6 = 7$ tests in total, by Lemma 5. Similarly, if Q is negative and contains one element from each side, then $1+t(K_{2,4}) = 1+2+t(4,1,1) = 7$ tests are needed, by Theorem 1 and Theorem 4. It follows that $|Q| \geq 3$.

Suppose that $|Q| = 3$. A left-side Q gives no information. If Q is a positive right-side pool, then the situation is equivalent to the case of a negative right-side pool of size 2. Suppose that Q is a positive pool consisting of 2 left-side elements and 1 right-side element. If Q is positive due to a defective left-side element, then Q is a superset of a positive left-side pool of size 2. Since the latter has already been ruled out, it follows from Lemma 3 that Q can be ruled out as well.

Before proceeding with the last case of $|Q| = 3$, the previous observations will be used along with Lemma 3 to show that $|Q| \leq 3$. Suppose that $|Q| \geq 4$ and that Q is positive. Q is necessarily a superset of either a left-side pool of size 2 or a right-side pool of size 3. In any of these cases, a positive result implies at least 7 tests in total. It follows from Lemma 3 that a strategy including Q can not be an improvement. Hence it must be the case that $|Q| \leq 3$, implying that $|Q| = 3$.

The only remaining Q to consider has 1 left-side element and 2 right-side elements. However, this case is nontrivial to rule out. If Q is negative, then it follows from Lemma 4 and Lemma 5 that $1 + t(K_{2,3}) = 6$ tests are required, which is acceptable. The positive case is not as easily examined and will therefore be left open. Ruling out this form of Q will be done in several steps.

Pool properties

By Theorem 3, every partial vector must be covered by some pool. Consider which pools are required to do so. As previously shown, a pool Q has 1 left-side element and 2 right-side elements. In terms of indicator vectors, Q has one 1 marking a left-side element, two 1's marking right-side elements and 0's everywhere else. Consider the 3 'families' of the forms $[100\ldots]$, $[010\ldots]$ and $[001\ldots]$. Without loss of generality, suppose that the family $[100\ldots]$ consists only of the pool $[1001\ldots]$. Then no pool covers the partial vector $[1{-}{-}0{*}{*}{*}{*}]$. By symmetry, each family must consist of at least 2 pools. Moreover, if any family contains more than 2 pools, then more than $3 * 2 = 6$ pools are used, contradicting that only 6 tests are needed. Hence, the pooling design consists of the 6 aforementioned pools.

Consider the family $[100\ldots]$. There exists at least one index $i \geq 4$ such that the two pools share a 0, because the family contains at most 4 of the 5 right-side elements. Again without loss of generality, this 0 is chosen to be at the 8th slot, so the $[100\ldots]$ pools are both of the form $[100\ldots0]$.

Further slots can be revealed by observing the following. Let Q and Q' denote the two pools of the form $[100\ldots]$. Since $Q \neq Q'$, there must exist some index $i \geq 4$ where Q has a 0 and Q' has a 1. Hence one can arbitrarily choose $i = 7$, so $Q = [100\ldots00]$ and $Q' = [100\ldots10]$. Moreover, note that Q has two further 1's and that Q' has two further 0's. Hence there must exist at least one index $j \geq 4$ where Q has a 1 and Q' has a 0. Arbitrarily choose $j = 6$ so $Q = [100{-}{-}100]$ and $Q' = [100{-}{-}010]$.

Let $Q' = [1000101]$, since Q' must contain exactly one further element. Then either $Q = [10001100]$ or $Q = [10010100]$. Call these cases Case 1 and Case 2, respectively. Recall that every candidate set complement must be a union of pools, by Theorem 2.

Case 1

Assume that the $[100\ldots]$ family consists of $[10001010]$ and $[10001100]$. Consider the candidate complement $C' = \{v_2, v_5\}^{\complement}$. By Theorem 2, C' must be a union of pools. Clearly, $v_1 \in C'$, implying that C' is a union of pools of the form $[100\ldots]$. However, $v_5 \in [100\ldots]$ by the Case 1 assumption and $v_5 \notin C'$ by definition of C'. Hence C' is not a union of pools, which violates the requirement stated in Theorem 2.

Case 2

Assume that the $[100\ldots]$ family consists of $[10001010]$ and $[10010100]$. Consider the candidate complement $C' = \{v_1, v_8\}^{\complement}$. By Theorem 2, C' must be a union of pools. Recall that the pooling design consists solely of the three families $[100\ldots]$, $[010\ldots]$ and $[001\ldots]$. Combining this knowledge with the facts that $v_2 \in C'$ and $v_8 \notin C'$, it follows that C' is a superset of the pool $[010\ldots0]$. Now consider the pools that, by Theorem 2, form the union $C'_1 = \{v_3, v_k\}^{\complement}$, $k < 8$. No pools of the form $[001\ldots]$ are part of this union because $v_3 \notin C'_1$. By definition, $v_8 \in C'_1$, implying that at least one pool part of the union must contain v_8. Neither of the $[100\ldots0]$ pools contain v_8, so a pool of the form $[010\ldots1]$ must exist. The pool

$[010\ldots1]$ has exactly 2 right-side elements, so it must have a 1 at some index $k' < 8$. Note that $k' \neq k$, because $v_k \notin C_1'$ and $[010\ldots1] \subseteq C_1'$. Finally, consider the pools that, by Theorem 2, form the union $C_2' = \{v_3, v_{k'}\}^{\complement}$. Analogous to the case of C_1', some pool of the form $[010\ldots1]$ is required. Since only 2 pools $[010\ldots]$ can exist and one of them is of the form $[010\ldots0]$, it must be the case that there exists a pool $Q'' = [010\ldots1]$ such that $Q'' \subseteq C_1'$ and $Q'' \subseteq C_2'$. This is impossible, because $v_{k'} \in Q''$ and $v_{k'} \notin C_2'$.

It is shown above that the assumptions in both Case 1 and Case 2 lead to contradictions. Hence, the pool Q cannot exist. Since particular choice of Q was chosen without loss of generality, it follows that there exists no pool consisting of exactly 1 left-side element and 2 right-side elements. It has already been shown that this is the only possible structure of a pool in a 6-test strategy, implying that no 6-test strategy can exist. This contradicts the initial assumption that $t(K_{3,5}) \leq 6$, implying that $t(K_{3,5}) \geq 7$. □

The preceding derivation contains several generalizable observations. Specifically, the 3 'families' consisting of exactly 2 pools each emerge in a more general context. The following lemma considers the existence and properties of those pools.

Lemma 6. *Suppose that* $t(K_{3,x}) = 2+t(x,1,1) = t(K_{3,x-1}) = 3+t(x-1,1,1)$*, where* $x \geq 5$*. Then the following is true for the pooling design of* $K_{3,x}$*. For each* $i \leq 3$*, there exist exactly 2 pools* Q_i *and* Q_i' *containing* v_i*. Moreover,* Q_i *and* Q_i' *contain no further left-side elements,* $|Q_i|, |Q_i'| > 1$ *and* $Q_i \cap Q_i' = \{v_i\}$*. The remaining* $t(x,1,1) - 4$ *pools contain only right-side elements.*

Proof. Consider a pool Q part of the strategy for the $K_{3,x}$ instance. Suppose that Q contains at least one left-side element. In a similar manner to the case of $x = 5$, various constructions of Q are considered and ruled out. If $|Q| = 1$ and Q is negative, then there must exist enough tests to solve the remaining $K_{2,x}$ instance, by Lemma 4. This implies a total number of $1 + t(K_{2,x}) = 1 + 2 + t(x,1,1) = 3 + t(x,1,1)$ tests, by Theorem 1. Hence $|Q| \geq 2$.

Suppose that $|Q| = 2$. If Q is a positive left-side pool, then the situation is equivalent to that of $|Q| = 1$. By Lemma 3, the same is true for any positive pool with at least 2 left-side elements. It follows that Q consists of exactly 1 left-side element along with at least 1 right-side element.

Suppose that $v_1 \in Q$ and that Q is positive due to v_1 being a defective. Then Q gives no information about the right-side defective, implying that at least $t(x,1,1)$ further tests are required, by Lemma 4. Hence at most 2 pools contain v_1. If only one such pool Q exists and $v_i \in Q$, $i > 3$, then $C' = \{v_2, v_i\}^{\complement}$ is not a union of pools, because $v_1 \in C'$ and $v_i \notin C'$. Hence, by Theorem 2 and the previous observations, exactly 2 pools contain v_1. Let Q' denote the second such pool. Suppose that $v_i \in Q \cap Q'$. Then, once more, C' is not a union of pools. Hence, by Theorem 2, there exist exactly 2 pools contain v_1, and these pools are right-side disjoint. By symmetry, the same is true for v_2 and v_3.

Since there are exactly 6 pools with a left-side element and it is assumed that $t(K_{3,x}) = 2 + t(x, 1, 1)$, there must exist exactly $2 + t(x, 1, 1) - 6 = t(x, 1, 1) - 4$ right-side pools in the pooling design for $K_{3,x}$. □

The remainder of this section focuses entirely on listing proofs for specific test numbers, as well as some conclusive remarks.

$\mathbf{t(K_{3,7}) = t(K_{3,8}) = t(K_{3,9}) = t(K_{3,10}) = 8}$

Proof. For the upper bound, $t(K_{3,10}) \leq t(3, 1, 1) + t(10, 1, 1) = 8$, by Lemma 1 and Theorem 4. It remains to show that $t(K_{3,7}) \geq 8$. Assume that $t(K_{3,7}) \leq 7$. Then $t(K_{3,7}) = 7$, because $t(K_{3,7}) \geq t(K_{3,6}) = 7$. By Lemma 6, exactly 6 of the pools contain a left-side element, implying that the 7th pool is a right-side pool. Let R denote the right-side pool.

Pool sizes and intersections

Consider the v_1-pools. By Lemma 6, there exists two such pools Q_1 and Q_1' which are right-side disjoint. Suppose that $|Q_1 \setminus \{v_1\}| = 4$. Suppose further that Q_1 is positive and Q_1' is negative. Then the remaining instance is $K_{2,4}$, implying that at least $2 + t(K_{2,4}) = 2 + 2 + 4 = 8$ tests are needed in total, by Lemma 4 and Theorem 1. By Lemma 3, if Q_1 is positive and $|Q_1 \setminus \{v_1\}| \geq 4$, then at least 8 tests are required as well. Hence $|Q_1 \setminus \{v_1\}| \leq 3$. By symmetry, for all $i \leq 3$, a v_i-pool contains at most 4 right-side elements.

Consider the v_1- and v_2-pools. Suppose that $Q_1 \cap Q_2 = \{v_4, v_5\}$. By Theorem 2, the candidate set complement $C' = \{v_3, v_4\}^{\complement}$ is a union of pools. Note that $v_5 \in C'$, so there must exist at least one pool Q such that $Q \subseteq C'$ and $v_5 \in Q$. Then $Q \notin \{Q_1, Q_2\}$, because, $v_4 \notin C'$ and $v_4 \in Q_1 \cap Q_2$. It must also be the case that $Q \notin \{Q_1', Q_2'\}$, because $v_5 \in Q_1 \cap Q_2$ implies that $v_5 \notin Q_1' \cup Q_2'$, by Lemma 6. Furthermore, Q is not a v_3-pool, because $Q \subseteq C'$ and $v_3 \notin C'$. The only remaining possibility is that $Q = R$. Now consider the pools that, by Theorem 2, form the union $C'' = \{v_3, v_5\}^{\complement}$. This includes at least one v_4-pool Q'. Then $Q' \notin \{Q_1', Q_2'\}$, because $v_4 \notin Q_1' \cup Q_2'$. Moreover, $Q' \notin \{Q_1, Q_2, R\}$, because otherwise Q' would be a v_5-pool, contradicting that $v_5 \notin C''$. Finally, since $v_3 \notin C''$, Q' can not be a v_3-pool either. It follows that Q' does not exist, contradicting the initial assumption that $Q_1 \cap Q_2 = \{v_4, v_5\}$. By symmetry, any 2 pools with different left-side elements can share at most 1 element.

Consider R. If $|R| = 2$ and R is negative, then there must exist enough tests to solve the remaining $K_{3,5}$ instance, by Lemma 4. Hence at least $1 + t(K_{3,5}) = 8$ tests are required in total. The case of a positive R with $|R| = 5$ is equivalent. By Lemma 3, if R is negative and $|R| \leq 2$ or R is positive and $|R| \geq 5$, then at least 8 tests are required as well. Hence $3 \leq |R| \leq 4$. Without loss of generality, assume that $R = [000000{-}111]$.

With the preceding observations in mind, the problem at hand is split into two cases: either no pool with exactly 1 element from each side exists, or at least one

such pool exists. Call these cases Case 1 and Case 2, respectively.

Case 1

Assume that there exists no pool with exactly one element from each side. Recall that $R = [000000{-}111]$. Suppose that $v_4 \in Q_1 \cap Q_2$. Suppose further that $v_5 \in Q_2$. Then $v_5 \notin Q_1$, because $|Q_1 \cap Q_2| \leq 1$, as previously noted. Consider the pools that, by Theorem 2, form the union $\{v_1, v_5\}^{\complement}$. This includes at least one v_4-pool. The v_1-pools and Q_2 can be ruled out due to the inclusion of v_1 and v_5, respectively. Since $v_4 \in Q_2$, it follows from Lemma 6 that $v_4 \notin Q_2'$. Clearly, $v_4 \notin R = [000000{-}111]$. Hence v_4 is contained in a v_3-pool, and by Lemma 6 at most one such pool can exist. Since it is assumed that $v_4 \in Q_1 \cap Q_2$, it follows that v_4 is in exactly one v_i-pool for each $i \leq 3$.

By symmetry, v_5 and v_6 are also in exactly one v_i-pool for each $i \leq 3$. This implies that $\{v_4, v_5, v_6\} \subseteq Q_i \cup Q_i'$. Then either $|\{v_4, v_5, v_6\} \cap Q_i| \geq 2$ or the analogous case for Q_i'. Suppose that $\{v_4, v_5, v_6\} \subseteq Q_i$. Then, there exists some $j \leq 3$, $j \neq i$, such that $|Q_i \cap Q_j| \leq 2$ (or the same for Q_j'), and it has already been shown that $|Q_i \cap Q_j| \geq 1$. Hence $|\{v_4, v_5, v_6\} \cap Q_i| \leq 2$. It follows that, for each $i \leq 3$, one v_i-pool contains two of v_4, v_5 and v_6, and the other one contains the third.

Let S denote the set of the 3 pools containing 2 of v_4, v_5 and v_6. Then any 2 pools in S have different left-side elements and share exactly one element in $\{v_4, v_5, v_6\}$, due to the observations above.

Recall that, by Theorem 3, every partial vector must be covered by some pool. Let p_i denote the partial vector of the form $[---****10*]$ which has a 0 at index $i \leq 3$. Clearly, the right-side pool $R = [000000{-}111]$ does not cover any such partial vector due to the 1 at index 9. Every remaining pool is a v_j-pool with $j \leq 3$. Since a v_j-pool can only cover p_i if $j \neq i$, it follows that at least 2 pools Q and Q' are required in order to cover all partial vectors $[---****10*]$. Then $v_8 \in Q \cap Q'$. Recall that any 2 pools in S share exactly one element in $\{v_4, v_5, v_6\}$. Hence, since $|Q \cap Q'| \leq 1$, it follows that $Q, Q' \notin S$.

Now consider the partial vectors of the form $[---****01*]$. As in the case of $[---****10*]$, two pools Q'' and Q''' with different left-side elements must cover these, and hence share v_9. Since $v_9 \notin Q, Q'$, it follows that $\{Q, Q'\} \cap \{Q'', Q'''\} = \emptyset$. Moreover, no pool in S shares v_9 with another pool, so $Q'', Q''' \notin S$. Clearly, $R \notin S$. Hence there exist a total number of $5 + |S| = 5 + 3 = 8$ pools, which contradicts the initial assumption that $t(K_{3,7}) \leq 7$.

Case 2

Let Q_1 and Q_1' denote the 2 v_1-pools and assume without loss of generality that $|Q_1 \setminus \{v_1\}| = 1$. Suppose that $|Q_1' \setminus \{v_1\}| \leq 2$. Then $Q_1 \cup Q_1'$ contains exactly 1 left-side element and at most 3 right-side elements. This implies that if both of Q_1 and Q_1' are negative, then at least the candidate graph $K_{2,4}$ remains, implying at least $2 + t(K_{2,4}) = 2 + 2 + 4 = 8$ tests in total, by Lemma 4 and Theorem 1. Hence $|Q_1' \setminus \{v_1\}| \geq 3$. It has already been shown that pools with left-side elements have

at most 3 right-side elements, implying that $|Q'_1 \setminus \{v_1\}| = 3$.

Case 2: The size of the right-side pool

Recall that $R = [000000{-}111]$. Assume by way of contradiction that $R = [0000001111]$. Suppose that there exists a pool Q such that $R \cap Q = \emptyset$. If R is positive and Q is negative, then the remaining subproblem contains $K_{2,4}$, implying that at least $2 + t(K_{2,4}) = 2 + 2 + 4 = 8$ tests are needed in total, by Lemma 4 and Theorem 1. Thus, $R \cap Q \neq \emptyset$. In particular, this is true for Q_1 and Q'_1. Then $(Q_1 \setminus \{v_1\}) \subset R$ and $|(Q'_1 \setminus \{v_1\}) \setminus R| \leq |Q'_1 \setminus \{v_1\}| - 1 = 2$. Then, for some right-side element u it must be the case that $u \notin Q_1 \cup Q'_1 \cup R$, because $|R^{\complement} \setminus \{v_1, v_2, v_3\}| = 3$. Note that $u \in Q, Q'$ for two arbitrary pools Q and Q'. To see this, recall that $|Q| > 1$, and note that if Q is the only u-pool, then the partial vector having a 1 at the index corresponding to u, but a 0 at some other index corresponding to an element in Q, then that partial vector is not covered by any pool. Hence further u-pools must exist, by Theorem 3. This is summarized in a nested lemma to be referred to at a later stage.

Lemma A. For any element u, there exist at least two pools Q and Q' such that $u \in Q, Q'$.

Since no further right-side pools or v_1-pools exist, so both of Q and Q' have a left-side element. Then, since $u \in Q, Q'$, it follows from Lemma 6 that Q and Q' have different left-side elements. Suppose without loss of generality that $Q = Q_2$ and $Q' = Q_3$.

Let v be a right-side element such that $v \neq u$. Suppose that $v \in Q$. Consider the pools that, by Theorem 2, form the union $\{v_3, v\}^{\complement}$. This includes at least one u-pool, and this pool contains neither of v_3 and v. The latter condition immediately rules out Q and Q'. Moreover, no further u-pools exist, because $u \notin Q_1 \cup Q'_1 \cup R$ and, since $Q = Q_2$ and $Q' = Q_3$, it follows from Lemma 6 that $u \notin Q'_2 \cup Q'_3$. Hence it must be the case that $v \notin Q$, implying that $Q = \{v_2, u\}$. But then $R \cap Q = \emptyset$, which contradicts the assumption that $R = [0000001111]$. Hence $R = [0000000111]$.

Note that the same argument can be applied to Q'. This pattern is useful at a later stage of this proof and is therefore stated here as the following nested lemma.

Lemma B. If u is a right-side element such that $u \notin Q_1 \cup Q'_1 \cup R$ and $u \in Q, Q'$, then Q and Q' have different left-side elements and no further right-side elements.

Consider the right-side elements that are not in R. There exists two possibilities: either every right-side element outside of R is in some v_1-pool, or at least one is not. Call these cases Case 2.1 and Case 2.2, respectively.

Case 2.1

Suppose that $\forall i : 4 \leq i \leq 7 \implies v_i \in Q_1 \cup Q'_1$. Assume without loss of generality that $Q_1 = [1001000\ldots]$. Then $Q'_1 = [1000111\ldots]$, by the Case 2.1 assumption and Lemma 6. Recall that every partial vector must be covered by some pool, by

Theorem 3. Consider the partial vector $p = [*0{*}{*}10\ldots]$. Since $v_2 \in Q_2, Q_2'$, $v_5 \notin Q_1$ and $v_6 \in Q_1'$, it follows that p must be covered by a v_3-pool. Specifically, this pool must have the form $[001{-}10{-}\ldots]$. Analogously, the pool $[*0{*}{*}01\ldots]$ must be covered by $[001{-}01{-}\ldots]$. But then the v_3-pools share 2 right-side elements, which has already been shown to be impossible.

Case 2.2

It follows from above that $\exists i : 4 \leq i \leq 7 \implies v_i \notin Q_1 \cup Q_1'$. Assume without loss of generality that Q_1 and Q_1' are both of the form $[1000\ldots]$. Then $v_4 \notin Q_1 \cup Q_1' \cup R$. Let $v_4 \in Q, Q'$ for two arbitrary pools Q and Q'. Two such pools do indeed exist, by Lemma A. Moreover, it follows from Lemma B that Q and Q' contain no further right-side elements. Hence one can assume that $Q = [0101000000]$ and $Q' = [0011000000]$.

Recall that every partial vector must be covered by some pool, by Theorem 3. Let $p = [0{*}{*}\ldots]$ and let p have both a 0 and a 1 in the last 3 slots. Due to the leading 0, the pools $[100\ldots]$ can not cover p. Since a 0 exists among the last 3 slots of p, $R = [\ldots111]$ can not cover p either. The pools Q and Q' are both of the form $[\ldots000]$, and therefore unable to cover p because of the 1 in one its last 3 slots. Then, since 5 pools have been ruled out, only $7 - 5 = 2$ pools remain. Note that there are 6 partial vectors of the same form as p, because the last 3 slots all have different values (0, 1 and $*$), and there are $3! = 6$ permutations of that sequence. Hence there exists a pool Q'' that covers at least $6/2 = 3$ such partial vectors. Clearly, the last 3 slots of Q'' contain, because otherwise none of the partial vectors in question would be covered. Without loss of generality, let $Q'' = [\ldots01{-}]$. Note that regardless of the value of the final slot, only one further partial vector will be covered. Hence some partial vectors are left uncovered, implying that more than 7 tests are required, by Theorem 3.

It follows from above that no 7-test strategy exists, which contradicts the initial assumption that $t(K_{3,7}) \leq 7$. □

$\mathbf{t(K_{3,11}) = t(K_{3,12}) = \cdots = t(K_{3,20}) = 9}$.

Proof. For the upper bound, $t(K_{3,20}) \leq t(3,1,1) + t(20,1,1) = 9$, by Lemma 1 and Theorem 4. It remains to show that $t(K_{3,11}) \geq 9$. Assume that $t(K_{3,11}) \leq 8$. Then $t(K_{3,11}) = 8$, because $t(K_{3,11}) \geq t(K_{3,10}) = 8$. By Lemma 6, there exist $t(11,1,1) - 4 = 2$ right-side pools. Let these be denoted as R_1 and R_2, respectively.

Pool sizes and intersections

Consider a right-side pool R. Suppose that $|R| = 4$ and that R is negative. Then there must exist enough tests to solve the remaining $K_{3,7}$ instance, by Lemma 4. Hence $1 + t(K_{3,7}) = 9$ are required in total. It follows from Lemma 3 that the same is true for any R with $|R| \leq 4$. The case of a positive R with $|R| = 7$ is equivalent, and it follows from Lemma 3 that any R with $|R| \geq 7$ can be ruled out. Hence $5 \leq |R| \leq 6$.

Consider how the right-side pools intersect. Suppose that $R_1 \cap R_2 = \emptyset$, that R_1 is positive and that R_2 is negative. Then the instance $K_{3,5}$ remains, and at least $2 + t(K_{3,5}) = 9$ tests are needed in total, by Lemma 4. Note that $2 + t(K_{3,4}) = 2 + 6 = 8$, by Lemma 5. It follows that $|R_1 \cap R_2| \geq 1$. Similarly, if at least one of them contains 6 elements, then $|R_1 \cap R_2| \geq 2$. The upper bound relies on some further observations.

Consider the two right-side elements u and v, where $u \neq v$. Suppose that $u \notin R_1 \cup R_2$. Suppose without loss of generality that Q_1 is the only u-pool. Then $\{v_1, v\}^{\complement}$ is not a union of pools, violating the condition stated in Theorem 2. Hence u is in at least 2 pools. Suppose without loss of generality that Q_1 and Q_2 are the only u-pools. Then $v \in Q_1$ implies that $\{v_2, v\}^{\complement}$ is not a union of pools, violating the condition stated in Theorem 2. Hence, if exactly 2 u-pools exist, then both of them contain no other right-side elements. Assuming this is the case, and that $u \in Q_1, Q_2$, suppose that Q_1 and Q_2 are negative and the v_3-pools are positive. Then the total number of tests required is at least $4 + t(10, 1, 1) = 9$, by Lemma 4 and Theorem 4. It follows by symmetry that any right-side element that is not in a right-side pool is contained in exactly 3 pools. In Lemma 6 it is stated that for each $i \leq 3$ there exist exactly 2 v_i-pools, and that these are right-side disjoint. Hence, if $u \notin R_1 \cup R_2$, then u is in exactly one v_i-pool for each $i \leq 3$. A direct consequence is that at most 2 such right-side elements can exist. These observations are summarized in the following nested lemma.

Lemma A. If $u \notin R_1 \cup R_2$, then there exists one v_i-pool containing u for each $i \leq 3$. Moreover, at most one further such element exists.

The latter observation can be rephrased as $11 - |R_1 \cup R_2| \leq 2$, implying that $|R_1 \cup R_2| \geq 9$. This in turn gives an upper bound for $|R_1 \cap R_2|$. Note that $|R_1 \cup R_2| = |R_1| + |R_2| - |R_1 \cap R_2|$, implying that $|R_1 \cap R_2| = |R_1| + |R_2| - |R_1 \cup R_2|$. Hence $|R_1 \cap R_2| \leq |R_1| + |R_2| - 9$. Recall that $|R_1 \cap R_2| \geq 1$ and that the right-side pool size is either 5 or 6. These properties are summarized in the following nested lemma.

Lemma B. There exist two right-side pools R_1 and R_2 such that each one has size 5 or 6. Moreover, $1 \leq |R_1 \cap R_2| \leq |R_1| + |R_2| - 9$

The following derivation considers two main cases: either the right-side pools are both of the form $[0000011\ldots]$, or not. The former assumption is made without loss of generality. Call these cases Case 1 and Case 2, respectively.

Case 1

Assume that the right-side pools are both of the form $[0000011\ldots]$. Then, by Lemma A, each of v_4 and v_5 is in its own 3 pools that contain left-side elements. This implies that the v_1-pools are of the form $Q_1 = [10001{-}{-}\ldots]$ and $Q_1' = [10010{-}{-}\ldots]$, and the analogous case for the v_2- and v_3-pools. Recall that by Theorem 3, every partial vector needs to be covered by some pool. Consider $p = [0{*}{*}{*}{*}10\ldots]$. Since $v_7 \in R_1 \cap R_2$, it follows that at least one v_i-pool, $i \leq 3$ covers p. Without loss of generality, let this pool be $Q_3 = [0010110\ldots]$.

Consider the partial vectors of the form $p' = [---*10*\dots]$. At least 2 pools of the form $[----10-\dots]$ are necessary, because any of the three unknown slots of p' can be a 0. Neither of R_1, R_2 and Q'_i, $i \leq 3$, is a v_5-pool and can therefore be ruled out. Moreover, Q_3 is a v_6-pool and can therefore be ruled out as well. Hence the only option is to have $Q_1 = [100010-\dots]$ along with $Q_2 = [010010-\dots]$.

At least one pool must cover $p'' = [**0**10\dots]$, and this pool must be of the form $[--0--10\dots]$. Note that Q_3 and Q'_3 are v_3-pools and that R_1, R_2 are v_7-pools, so all of these 4 pools can be ruled out. The remaining pools are the two pairs $i < 3$, $Q_i = [--0010-\dots]$ and $Q'_i = [--010--\dots]$. By symmetry, any of the Q'_i, $i < 3$ can be chosen to cover p''. Hence Q'_1 is arbitrarily selected, implying that $Q'_1 = [1001010\dots]$

Consider the 2 pools covering $[---1*0*\dots]$. Neither of R_1, R_2, Q_1, Q_2 and Q_3 are v_4-pools, and Q'_1 is a v_6-pool. Hence the only remaining option is to let $Q'_2 = [010100-\dots]$ and $Q'_3 = [001100\dots]$. But then no pool covers $[0***01*\dots]$. It follows from Theorem 3 that further tests are needed, contradicting the Case 1 that both right-side pools are of the form $[0000011\dots]$.

Case 2

Case 2: Properties of right-side pools

Consider R_1 and R_2. Recall that Lemma B states that each right-side pool has size 5 or 6 and that $1 \leq |R_1 \cap R_2| \leq |R_1| + |R_2| - 9$. Suppose that $|R_1| = 6$ and that either $|R_2| = 6$ and $|R_1 \cap R_2| = 3$ or $|R_2| = 5$ and $|R_1 \cap R_2| = 2$. In both of these cases, $|R_1 \cap R_2| \geq 2$ and $|R_1 \cup R_2| = 9$. Hence $11 - 9 = 2$ right-side elements are in neither of R_1 and R_2. It follows that, without loss of generality, that the right-side pools are on the form $[0000011\dots]$. This is exactly Case 1, which has already been ruled out.

Suppose now that $|R_1| = |R_2| = 6$ and that $|R_1 \cap R_2| = 1$. Then $|R_1 \setminus R_2| = 5$. Suppose that R_1 is positive and R_2 is negative. Then the instance $K_{3,5}$ remains, and $2 + t(K_{3,5}) = 2 + 7 = 9$ tests are required in total, by Lemma 4.

It follows from the previous observations that $|R_1 \cap R_2| = 1$ and $|R_1 \cup R_2| \leq 10$. Then $|R_1 \setminus R_2| \geq 4$ and at least 1 right-side element is in neither of R_1 and R_2. Hence, without loss of generality, $R_1 = [00000000\dots]$ and $R_2 = [00001111\dots]$.

Consider Q_i, $i \leq 3$. Suppose that $Q_i \cap (R_2 \setminus R_1) = \emptyset$. Suppose further that Q_i and R_1 are negative, while R_2 is positive. Then, since $R_1 = [00000000\dots]$ and $R_2 = [00001111\dots]$, it follows that the instance $K_{2,4}$ remains, implying a total number of $3+t(K_{2,4}) = 3+2+t(4,1,1) = 9$, by Lemma 4, Theorem 1 and Theorem 4. It follows that $Q_i \cap (R_2 \setminus R_1) \neq \emptyset$.

Case 2: Properties of right-side elements

Let $5 \leq j \leq 8$ and recall that $v_j \in R_2 \setminus R_1$. Suppose that v_j is in exactly 2 further pools Q and Q'. Then Q and Q' are the only pools that can cover partial vectors

which have a 1 at slot j and a 0 at some k, $5 \leq k \leq 8$, $k \neq j$. Hence all such partial vectors are only covered if $v_k \notin Q \cup Q'$. That is, Q and Q' contain no further elements from $R_2 \setminus R_1$. But then, by symmetry, there must exist two distinct pools Q and Q' for each j, implying a total number of $2 * 4 = 8$ such pools. By Lemma 6, only 6 such pools can exist, which is a contradiction. Hence each v_j is in either 1 or 3 such pools. In the former case, suppose that $v_j \in Q_i$, $i \leq 3$. Then $\{v_i, v_k\}^{\complement}$ is not a union of pools, implying that further tests are required, by Theorem 2. Hence, each v_j is in R_2 as well as 3 additional pools. This is summarized is a nested lemma.

Lemma C. If $v \in R_2 \setminus R_1$, then there exist 3 additional v-pools.

Case 2: Revealing indicator vector slots

Consider the pooling design in its entirety, in particular the 8 initial slots of every indicator vector. Since $v_4 \notin R_1 \cup R_2$, it follows from Lemma A that for each $i \leq 3$ there exists one v_i-pool that contains v_4. Since any two v_i-pools are right-side disjoint, by Lemma 6, it follows that $Q_i = [1000{-}{-}{-}{-}\dots]$ and $Q'_i = [1001{-}{-}{-}{-}\dots]$ for each ≤ 3. Moreover, since $R_2 = [00001111\dots]$ and $Q_i \cap (R_2 \setminus R_1) \neq \emptyset$ by Lemma C, one can assume without loss of generality that $v_6 \in Q'_1$. Since Lemma 6 states that the v_1-pools are right-side disjoint, it follows that $Q_1 = [100010{-}{-}\dots]$ and $Q'_1 = [100101{-}{-}\dots]$. This, in turn, implies that the partial vectors $[{-}{-}{-}0{*}1{*}{*}\dots]$ can only be covered by Q_2 and Q_3. Hence $Q_2 = [0100{-}1{-}{-}\dots]$ and $Q_3 = [0010{-}1{-}{-}\dots]$, and Lemma 6 implies that v_6 is in neither of Q'_2 and Q'_3.

The partial vector $[0{*}{*}{*}10{*}{*}\dots]$ must be covered by either Q'_2 or Q'_3. This choice can be made arbitrarily, because, for $1 < i \leq 3$ it is the case that $Q_i = [{-}{-}{-}0{-}1{-}{-}\dots]$ and $Q'_i = [{-}{-}{-}1{-}0{-}{-}\dots]$. Hence the situation is entirely symmetric. Without loss of generality, let $v_5 \in Q'_2$. By Lemma 6, it follows that $v_5 \notin Q_2$, so $Q_2 = [010001{-}{-}\dots]$ and $Q'_2 = [010110{-}{-}\dots]$.

Note that $v_4 \notin Q'_3$ because Q'_3 must cover the partial vector $[0{*}{*}10{*}{*}{*}\dots]$. Recall that if $v \in R_2 \setminus R_1$, then v is in 3 additional pools, by Lemma C. This is applicable to v_5, with Q_3 being the third such v_5-pool.

Consider Q_3 and observe that $|Q_3 \cap (R_2 \setminus R_1)| \leq 2$. To see this, suppose that R_1 is negative and that both of R_2 and Q_3 are positive. If $|Q_3 \cap (R_2 \setminus R_1)| \geq 3$, then the instance $K_{3,3}$ remains, implying that the total number of tests is $3 + t(K_{3,3}) = 3 + 3 + 3 = 9$, by Lemma 4, Lemma 5 and Theorem 4. It follows that Q_3 contains no further elements within $R_2 \setminus R_1$, implying that $Q_3 = [00101100\dots]$. Then $Q'_3 = [00110011\dots]$, by Lemma C.

Consider the partial vectors $[{-}{-}{-}1{*}{*}0{*}\dots]$. It follows from the previous observations that these can only be covered by Q'_1 and Q'_2. Then both Q_1 and Q_2 contain v_7, by Lemma C. But then there exists no pool that covers $[{*}0{*}{*}1{*}0{*}\dots]$ By Theorem 3, further tests are required. This implies that Case 2 is an impossibility as well.

It from follows above that no matter which two right-side pools are chosen, 8 pools

will be insufficient. It is already known that a pooling design of 8 pools must contain two right-side pools, which contradicts the initial assumption that $t(K_{3,11}) \leq 8$. □

$\mathbf{t(K_{3,21}) = t(K_{3,22}) = \cdots = t(K_{3,35}) = 10}$

Proof. For the upper bound, $t(K_{3,35}) \leq t(3,1,1) + t(35,1,1) = 10$, by Lemma 1 and Theorem 4. It remains to show that $t(K_{3,21}) \geq 10$. Assume that $t(K_{3,21}) \leq 9$. Consider a right-side pool R. If $|R| = 11$ and Q is positive, then $1 + t(K_{3,11}) = 10$ further tests are required, by Lemma 4. By Lemma 3, the same is true for any size $|R| \geq 11$. If $|R| = 10$ and R is negative, then again $1 + t(K_{3,11}) = 10$, by Lemma 4. By Lemma 3, the same is true for any size $|R| \leq 10$. Hence $10 < |R| < 11$. No such integer $|R|$ exists, so R can not exist. By Lemma 6, there must exist some right-side pool. This contradicts the initial assumption that $t(K_{3,21}) \leq 9$. □

This section is concluded with several remarks that may be useful in future work. First, the preceding derivation is generalized. Assume that $t(K_{3,x}) = 2 + t(x,1,1)$ and let $x' < x$ be the smallest integer such that $t(K_{3,x'}) = 2 + t(x,1,1)$. Then $t(x',1,1) = t(x,1,1) - 1$. Consider a right-side pool R. If $|R| = x'$ and R is positive, then $1 + t(K_{3,x'}) = 3 + t(x,1,1)$ tests are required, by Lemma 4. Lemma 3 rules out any size $|R| \geq x'$. If $|R| = x - x'$ and R is negative, then $1 + t(K_{3,x'}) = 3 + t(x,1,1)$ tests are required, by Lemma 4. Lemma 3 rules out any size $|R| \leq x - x'$. Hence $x - x' < |R| < x'$.

Analogous to x', define x'' to be the smallest integer such that $t(K_{3,x''}) = 1 + t(x,1,1)$. Consider any pool Q. Suppose that Q has $x'' + c$ right-side elements, $c \geq 0$. Suppose further that Q shares at most c right-side elements with some other pool Q', and that Q is positive and Q' is negative. Then, by Lemma 4, at least $2 + t(K_{3,x''}) = 3 + t(x,1,1)$ tests are needed. Hence Q shares at least $c + 1$ right-side elements with any pool. If Q and Q' instead share x'' right-side elements and both are positive, then again $2 + t(K_{3,x''}) = 3 + t(x,1,1)$ tests are needed, by Lemma 4. Hence Q shares less than x'' right-side elements with any other pool.

Further observations can be made regarding the pool sizes. Let t_a denote $t(a,1,1)$. Suppose that $t(K_{3,x}) = 2 + t_x$. Then, by Lemma 6, for each $i \leq 3$ there exists exactly 2 v_i-pools Q_i and Q_i' which are right-side disjoint. The remaining $t_x - 4$ pools are right-side pools. Suppose that exactly one defective exists on each side and that v_1 is the left-side defective. In order to rule out v_j, $j \leq 2$, as a defective, at least one v_j-pool must be negative. Hence it can be assumed that Q_2 and Q_3 are negative.

Now consider the right-side defective and let $y = |(Q_2 \setminus \{v_2\}) \cup (Q_3 \setminus \{v_3\})|$, that is the union of the right-side elements of Q_2 and Q_3. Since both of the v_1-pools are positive due to v_1, they provide no information regarding the right-side defective. Hence there exist $x - y$ potential right-side defectives, which must be identified using at most $2 + t_x - 4 = t_x - 2$ tests. Suppose that $x - y \geq x'$. Then at least $t_{x'} = t_x - 1$ more tests are required, which is beyond the budget. Hence $x - y \leq x' - 1$, implying that $y \geq x - x' + 1$. Since the choice of Q_2 and Q_3 was arbitrary, it follows that the

union of any two pools with different left-side elements contains at least $x - x' + 1$ right-side elements.

In fact, the same is true for two pools with the same left-side element. Let the two v_1-pools be negative and suppose that $x - y$ right-side elements are not in any v_1-pool. Then what remains is a $K_{2,x-y}$ instance, implying $2+t(K_{2,x-y}) = 2+2+t_{x-y} = 4 + t_{x-y}$ tests in total, by Lemma 4 and Theorem 1. Then, if $x - y \geq x'$, it follows that $4 + t_{x-y} \geq 4 + t_{x'} = 3 + t_x$. Hence $x - y \leq x' - 1$, implying that $y \geq x - x' + 1$.

Consider Q_i and Q_j. It follows from the previous observations and the definition of y that $|(Q_i \setminus \{v_i\}) \cup (Q_j \setminus \{v_j\})| \geq x - x' + 1$. Then, if $|Q_i \setminus \{v_i\}| < x - x' + 1$, there must exist $x - x' + 1 - |Q_i \setminus \{v_i\}|$ right-side elements that are in Q_j but not in Q_i. Hence, if $|Q_i \setminus \{v_i\}| \leq \frac{x-x'+1}{2}$, then $|Q_j \setminus \{v_j\}| \geq \frac{x-x'+1}{2}$. This implies that there always exist at least 5 pools with one left-side element and at least $\frac{x-x'+1}{2}$ right-side elements.

5.3.4 Various small instances

$\mathbf{t(K_{4,4}) = 7}$.

Proof. For the upper bound, $t(K_{4,4}) \leq t(K_{3,4}) + 1 = 7$, because there exists the possible strategy to add one test to the $K_{3,4}$ strategy, and $t(K_{3,4}) = 6$, by Lemma 5. It remains to show that $t(K_{4,4}) \geq 7$. Suppose that $t(K_{4,4}) \leq 6$. Consider a pool Q. Suppose $|Q| = 2$. If Q is a positive left-side pool, then $1 + t(K_{2,4}) = 1 + 2 + 4 = 7$ tests are needed, by Theorem 1 and Theorem 4. By Lemma 3, the same is true for $|Q| \geq 2$. The right-side case is equivalent. Consider a negative pool Q consisting of exactly one element from each side. The remaining problem is $K_{3,3}$, implying $1 + t(K_{3,3}) = 1 + 3 + 3 = 7$ tests in total, by Lemma 5 and Theorem 4. Lemma 3 rules out any other case where $|Q| \leq 2$. Hence no pool Q exists, contradicting the initial assumption that $t(K_{4,4}) \leq 7$. □

$\mathbf{t(K_{4,5}) = t(K_{5,5}) = t(K_{4,6}) = t(K_{5,6}) = t(K_{6,6}) = 8}$.

Proof. For the upper bound, $t(K_{6,6}) \leq t(6,1,1) + t(6,1,1) = 8$, by Lemma 1 and Theorem 4. It remains to show that $t(K_{4,5}) \geq 8$. Assume that $t(K_{4,5}) \leq 7$.

Pool sizes

Consider a pool Q. If $|Q| = 1$ and Q is negative, then either $1 + t(K_{3,5}) = 8$ or $1 + t(K_{4,4}) = 8$ tests are required, by Lemma 4. Hence $|Q| \geq 2$. Suppose that $|Q| = 4$ and that Q is positive. The case of a left-side Q gives no information and a right-side Q is equivalent to the case of a negative right-side Q of size 1. Hence, Q contains at most 3 elements from the same side, by Lemma 3. Suppose that Q contains 3 left-side elements and that Q is positive. This is equivalent to the case of a negative left-side Q of size 1. It follows from Lemma 3 that Q contains at most 2 left-side elements.

Properties of elements

Consider an arbitrary element v_i. Suppose that only one pool Q contains v_i. Recall that $|Q| > 1$, implying that some further element v_j, $j \neq i$, is in Q. Then the indicator vector of Q has value 1 at the indices i and j. Then no pool covers the partial vector with a 1 at i and a 0 at j. Hence it must be the case that further v_i-pools exist.

Let v be a left-side element and suppose that at least 4 v-pools exist. Then, if all of them are positive, at least $4 + t(5, 1, 1) = 8$ tests are needed, by Lemma 4 and Theorem 4. Hence at most 3 v-pools exist.

The preceding observations are stated in the following lemma, which will be referred to later.

Lemma A. For each element v, there exist at least 2 v-pools. Moreover, if v is on left side, then there exist at most 3 v-pools.

Pools with left-side elements

Suppose that no pool contains more than 1 left-side element. Then $2 * 4 = 8$ tests are required to include all left-side elements in 2 pools. Hence some pool contains at least 2 left-side elements. Combining this observation with Lemma A, it follows that there exists a Q pool that contains exactly 2 left-side elements. Moreover, there exists at least one more such pool, as is shown below.

Let Q be a pool containing exactly 2 left-side elements, and without loss of generality let $Q = [1100\ldots]$. Suppose that there exist no further pools with 2 left-side elements. Recall that each left-side element must be in at least 2 pools, by Lemma A. This implies that there exist 2 pools of the form $[0001\ldots]$ and 2 pools of the form $[0010\ldots]$. Then the last 2 pools are of the forms $[0100\ldots]$ and $[1000\ldots]$. It follows that $[0100\ldots]$ is the only pool that covers partial vectors of the form $[01{*}{*}\ldots]$. This in turn implies that $[0100\ldots]$ has a 0 at any right-side slot, by Theorem 3. But then $[0100\ldots]$ is a left-side pool of size 1, which has already been shown to be impossible. Consequently, apart from Q, there exists at least one pool which contains 2 left-side elements. This is summarized in the following nested lemma.

Lemma B. Let Q be a pool that contains 2 left-side elements. There exist 2 such pools.

There are two main cases to consider: either Q contains at most 1 right-side elements, or Q contains at least 2 right-side elements. Call these cases Case 1 and Case 2, respectively.

Case 1

Assume that a pool Q contains exactly 2 left-side elements and at most 1 right-side elements. Let, without loss of generality, $Q = [1100{-}0000]$.

Let Q' be a pool such that $Q' \neq Q$ and Q' contains 2 left-side elements. Note

that Q' does indeed exist such a pool, by Lemma B. Suppose that Q and Q' share either 0 or 2 left-side elements. In the former case, let the smallest of the pools be negative and the other one positive, and the in latter case let both of them be positive. Then at least the instance $K_{2,4}$ remains, implying a total number of $2 + t(K_{2,4}) = 2 + 2 + t(4, 1, 1) = 8$ tests, by Lemma 4, Theorem 1 and Theorem 4. Hence Q and Q' share exactly 1 left-side element. Without loss of generality, let $Q' = [0110\ldots]$.

Suppose that two further pools contain v_3. Then, if Q is negative and the 3 v_3-pools are positive, $4+t(4, 1, 1) = 8$ tests are required in total, by Lemma 4 and Theorem 4. Hence only one further v_3-pool Q'' exists, by Lemma A. If Q' and Q'' share 2 left-side elements, then $2 + t(K_{2,5}) = 2 + 2 + t(5, 1, 1) = 8$ tests are needed, by Lemma 4, Theorem 1 and Theorem 4. It follows that $Q'' = [-01-\ldots]$. Then only Q'' covers partial vectors of the form $[*01*\ldots]$, because $Q = [1100\ldots]$, $Q' = [0110\ldots]$ and no further v_3-pools exist. In order to cover all such vectors, Q'' must have 0s in all right-side slots. Hence Q'' is a left-side pool, by Theorem 3. Then $|Q''| = 2$, implying that Q'' and Q share exactly 1 element. Hence $Q'' = [1010\ldots]$.

Consider v_4. Recall that each left-side element is in at least 2 pools, by Lemma A. This implies that there exist 2 pools of the form $[--01\ldots]$.

It follows from the previous observations that part of the pooling design consists of $Q = [1100-0000]$, $Q'' = [101000000]$ and 2 pools $[--01\ldots]$. Suppose that Q and Q' are negative and that the 2 v_4-pools are positive. Then $4 + t(4, 1, 1) = 8$ tests are required, by Lemma 4 and Theorem 4, which contradicts the Case 1 assumption that Q contains at most 1 right-side element.

Case 2

Let Q be a pool containing 2 left-side elements and either 2 or 3 right-side elements. Without loss of generality, let $Q = [110011\ldots]$.

Suppose that some pool Q' consists entirely of left-side elements. Then $|Q'| = 2$, as has already been shown. Then it follows from the Case 2 assumption that Q' contains at least one right-side element, which is a contradiction. Hence no such Q' can exist.

Considers the partial vectors $[01**\ldots]$. As implied by the previous observation, there exists no pool of the form $[01--00000]$. Since, by Theorem 3, all of the partial vectors must be covered, it follows that there exist at least 2 pools of the form $[01--\ldots]$. The situation is completely analogous for the partial vectors $[10**\ldots]$, implying the existence of 2 pools of the form $[10--\ldots]$.

Note that, since it is stated in Lemma A that any left-side element is in at most 3 pools, it follows from above that there exist 2 pools of the form $[00--\ldots]$. If both of them are of the form $[0011\ldots]$ and are positive, then $2+t(K_{2,5}) = 2+2+t(5, 1, 1) = 8$ tests are required, by Lemma 4, Theorem 1 and Theorem 4. Hence it can be assumed without loss of generality that there exists a pool $[000-\ldots]$.

It follows from the previous observations that the pooling design consists of $Q = [110011\ldots]$, 2 pools $[10\ldots]$, 2 pools $[01\ldots]$, 1 pool $[000-\ldots]$ and 1 pool $[00--\ldots]$. Theorem 3 is applied multiple times below, in order to reveal slots by arguing that some partial vector must be covered.

Note that for each partial vector covered by Q, there exists a partial vector that is almost identical, apart from having a 0 at index 5 or 6. Hence each such vector must covered by some further pool. This is the case for both $[*10*0\ldots]$ and $[1*0*0\ldots]$, implying the existence of the pools $[010{-}0\ldots]$ and $[100{-}0\ldots]$.

Consider the partial vectors $[*01*\ldots]$. No pool has the form $[-01-00000]$, so at least 2 pools must cover such vectors. Due to the previously revealed indicator vector slots, it follows that the pooling design contains the pools $[101-\ldots]$ and $[001-\ldots]$.

Consider the partial vectors $[0*1*\ldots]$. The pool $[001-\ldots]$ contains at least one right-side element, so there must exist at least one more pool covering such vectors. The only option is $[011-\ldots]$.

It follows from the previous observations that $[*10**0\ldots]$ and $[1*0**0\ldots]$ must be covered by $[100{-}00\ldots]$ and $[010{-}00\ldots]$, respectively. But then both of $[**0*10\ldots]$ and $[**0*01\ldots]$ must be covered by the same pool $[000-\ldots]$, which is a contradiction. Hence Case 2 is contradictory as well.

It follows from above that no pool with 2 left-side elements can exist, which contradicts Lemma B. This in turn contradicts the initial assumption that $t(K_{4,5}) \leq 7$. □

5.4 Derivations for two defectives and two stages

$\mathbf{t(14, 2, 2) = 9}$.

Proof. For the upper bound, $t(15, 2, 2) = 9$ [1]. It remains to show that $t(14, 2, 2) \geq 9$. Assume $t(14, 2, 2) \leq 8$ and consider the pools of stage 1.

Pool sizes

Consider a stage 1 pool Q. If $|Q| = 4$ and Q is negative then $1 + t(10, 2, 2) = 9$ tests are needed, by Lemma 4 and that $t(10, 2, 2) = 8$ is known from [1]. Then the same is true for $|Q| \leq 4$, by Lemma 3. Hence $|Q| \geq 5$.

Suppose that $|Q| = 7$ and that Q is positive. Then the number of remaining candidate sets of size 2 is $\binom{7}{2} + 7(14 - 7) = 70 > 64 = 2^6$. Hence at least $1 + 6 + 2 = 9$ tests are needed in this case, by Lemma 2 and Lemma 4. Thus, $|Q| \leq 6$.

Single-pool stage

Suppose that stage 1 consists of a single pool Q. If $|Q| = 6$ and Q is negative, then the stage 2 pools must identify the defectives among the remaining 8 elements, implying $1 + t(8, 2, 1) = 9$ tests, where $t(8, 2, 1) = 8$ is known from [1]. It follows from Lemma 3 that the same is true for $|Q| \leq 6$. Since 6 is the maximum pool size, it follows that stage 1 consists of at least 2 pools.

Pairs of pools

Consider two stage 1 pools Q and Q'. It is shown below that, in most cases, the presence of Q and Q' implies that at least 9 tests are required. Let c denote the number of candidate sets of size 2 that remain after the outcome of Q and Q' has been revealed.

Suppose that $Q \cap Q' = \emptyset$, $|Q| = |Q'| = 6$ and that both of Q and Q' are positive. Then every candidate set consists of exactly one element from each of Q and Q'. Then $c = 6 * 6 = 36 > 32 = 2^5$, implying that at least $2 + 5 + 2 = 9$ tests are required, by Lemma 4 and Lemma 2.

Suppose that $Q \cap Q' \neq \emptyset$, $|Q| = |Q'| = 6$ and that both of Q and Q' are positive. If $|Q \cap Q'| = 1$, then $c = 5 * 5 + 5 * 1 + 5 * 1 + 3 * 1 = 38 > 32 = 2^5$, because each candidate element is in either 1, 2 or 0 pools. If $|Q \cap Q'| > 1$, even more possibilities exist. Hence at least $2 + 5 + 2 = 9$ tests are required, by Lemma 4 and Lemma 2.

Suppose that $Q \cap Q' \neq \emptyset$, $|Q| = 5$, $|Q'| = 6$ and that both of Q and Q' are positive. Similar to the case above, $c = 4 * 5 + 4 * 1 + 5 * 1 + 4 * 1 = 33 > 32 = 2^5$, implying that $2 + 5 + 2 = 9$ tests are required, by Lemma 4 and Lemma 2.

Suppose that $|Q \cap Q'| \geq 2$, $|Q| = |Q'| = 5$ and that both of Q and Q' are positive. As in the previous two cases, $c = 3 * 3 + 3 * 2 + 3 * 2 + 2 * 6 = 33 > 32 = 2^5$, implying that $2 + 5 + 2 = 9$ tests are required, by Lemma 4 and Lemma 2.

It follows from the preceding observations that it is either the case that $|Q| = 6$, $|Q'| = 5$ and $Q \cap Q' = \emptyset$, or it must be that $|Q| = |Q'| = 5$ and $|Q \cap Q'| \leq 1$.

Triples of pools

Consider a triple of pools in stage 1, denoted as Q, Q' and Q''. Note that the properties of pairs of pools can be applied to pairs within a triple.

Note that most one pool has size 6. This implies that at least 2 of the pools have size 5. Let $|Q| = |Q'| = 5$. Then $|Q \cap Q'| \leq 1$. Suppose $|Q''| = 6$. Then $Q'' \cap (Q \cup Q') = \emptyset$, because pools of size 6 are disjoint from pools of size 5. But this is impossible, because $|(Q \cup Q')^{\complement}| \leq 14 - 9 = 5 < |Q''|$. It follows that all pools have size 5 and that any 2 of them share at most 1 element. Here, let c denote the number of candidate sets of size 2 that remain after the outcomes of Q, Q' and Q'' have been revealed.

Suppose that $|Q \cap Q'| = 1$ and $Q'' \cap (Q \cup Q') = \emptyset$. Suppose further that Q is negative and that Q' and Q'' are positive. Then each candidate element is in exactly one of Q' and Q''. Hence $c = 4 * 5 = 20 > 16 = 2^4$, implying that $3 + 4 + 2 = 9$ tests are

required, by Lemma 4 and Lemma 2.

Suppose that $Q \cap Q' = \emptyset$, $|Q'' \cap Q| = 1$ and $|Q'' \cap Q'| = 1$. Suppose further that Q is negative and that Q' and Q'' are positive. Then each candidate element is in either 1, 2 or 0 pools. Hence $c = 3 * 4 + 3 * 1 + 4 * 1 + 1 * 1 = 20 > 16 = 2^4$, implying that $3 + 4 + 2 = 9$ tests are required, by Lemma 4 and Lemma 2.

Suppose that any 2 pools share 1 element and that exactly 2 of them are positive. Analogously to the previous case, $c = 3 * 3 + 3 * 1 + 3 * 1 + 1 * 2 = 17 > 16 = 2^4$, implying that $3 + 4 + 2 = 9$ tests are required, by Lemma 4 and Lemma 2.

It follows from the previous observations that the only possible triple is the case of $|Q \cap Q' \cap Q''| = 1$.

Quadruplets of pools

Consider a quadruplet of pools. Any quadruplet must contain the triplet Q, Q' and Q'', where $|Q| = |Q'| = |Q''| = 5$ and $|Q \cap Q' \cap Q''| = 1$. Let v denote this shared element. The fourth pool Q''' must be a v-pool in order to form a triple with two of the other pools. Moreover, v is the only element which Q''' can share. However, this is impossible, because then $|(Q \cup Q' \cup Q'')^\complement| = 1$ and $|Q'''| = 5$. Hence no quadruplet can exist.

The stage 1 pools

The following has been shown. Stage 1 consist of at least 2 pools and at most 3 pools. If it consists of exactly 2 pools Q and Q', then it is either the case that $|Q| = 6$, $|Q'| = 5$ and $Q \cap Q' = \emptyset$, or it must be that $|Q| = |Q'| = 5$ and $|Q \cap Q'| \leq 1$. If a third pool Q'' exists, then the latter conditions for Q and Q' apply, as well as the additional condition that $|Q \cap Q' \cap Q''| = 1$. Suppose that, in any of these cases, exactly 2 pools are positive. Then the remaining instance contains $K_{4,4}$, implying a total number of $2 + t(K_{4,4}) = 2 + 7 = 9$ tests, by Lemma 4. Hence, it has been demonstrated that no 8-test strategy can exist, which contradicts the initial assumption that $t(14, 2, 2) \leq 8$. □

$\mathbf{t(20, 2, 2) = t(21, 2, 2) = 10}$.

Proof. The approach is similar to the $t(14, 2, 2)$ case. For the upper bound, $t(22, 2, 2) = 10$ is known from [1]. Assume $t(20, 2, 2) \leq 9$ and consider the stage 1 pools.

Pool sizes

Let Q be a stage 1 pool. If $|Q| = 6$ and Q is negative, then $1 + t(14, 2, 2) = 10$ tests are required, by Lemma 4. The same is true for $|Q| \leq 6$, by Lemma 3. If $|Q| = 9$ and Q is positive, then there exist $\binom{9}{2} + 9(20 - 9) = 135 > 128 = 2^7$ candidate sets of size 2, implying a total number of $1 + 7 + 2 = 10$ tests, by Lemma 2 and Lemma 4. The same is true for $|Q| \geq 9$, by Lemma 3. Hence $7 \leq |Q| \leq 8$.

Single-pool stage

Suppose that stage 1 consists of a single pool Q. If $|Q| = 8$ and Q is negative 8-pool, then $1 + t(12, 2, 1) \geq 1 + t(10, 2, 1) = 10$ [1] tests are required, by Lemma 4. The same is true for $|Q| \leq 7$, by Lemma 3. Hence stage 1 contains at least 2 pools.

Pairs of pools

Consider the stage 1 pools Q and Q'. Let c denote the number of candidate sets of size 2 that remain after the outcome of Q and Q' has been revealed.

Suppose that $|Q| = |Q'| = 8$ and $Q \cap Q' \neq \emptyset$. Suppose further that Q and Q' are positive. If $|Q \cap Q'| \leq 1$, then $c = 7 * 7 + 7 * 1 + 7 * 1 + 1 * 5 = 68 > 64 = 2^6$, because each candidate element is in 1, 2 or 0 pools. Then $2 + 6 + 2 = 10$ tests are required, by Lemma 4 and Lemma 2. Increasing the number of shared elements gives a similar situation. Hence any 2 pools of size 8 are disjoint.

Suppose that $|Q| = 7$, $|Q| = 8$ and $Q \cap Q' = \emptyset$. Suppose further that Q is negative and Q' is positive. Then $c = \binom{8}{2} + 8 * 5 = 68 > 64 = 2^6$. Then $2 + 6 + 2 = 10$ tests are required, by Lemma 4 and Lemma 2. Hence pools of different sizes are not disjoint.

Suppose that $|Q| = |Q'| = 7$ and $|Q \cap Q'| \geq 3$. Suppose further that both of them are positive. If $|Q \cap Q'| = 3$, then $c = 4 * 4 + 4 * 3 + 4 * 3 + 3 * 9 + 3 = 70 > 64 = 2^6$, because each candidate element is in either 1, 2 or 0 pools. Then $2+6+2 = 10$ tests are required, by Lemma 4 and Lemma 2. This does not improve as the number of shared elements increases. Hence two pools of size 7 share at most 2 elements.

Suppose that $|Q| = 7$, $|Q| = 8$ and $|Q \cap Q'| \geq 2$. Suppose further that both of them are positive. As in the previous case, $c = 6*5+6*2+5*2+2*8+1 = 67 > 64 = 2^6$. Then $2 + 6 + 2 = 10$ tests are required, by Lemma 4 and Lemma 2. This does not improve as the number of shared elements increases. Hence two pools of different sizes share at most 1 element. Since it is known that two such pools are not disjoint, it follows that they share exactly 1 element.

Triples of pools

Consider a triple of pools Q, Q' and Q''. The properties of pairs are applied in order to rule out multiple triples.

Suppose that $|Q| = |Q'| = 8$. Then $Q \cap Q' = \emptyset$, implying that $|Q \cup Q'| = 16$. Then only $|Q'' \setminus (Q \cup Q')| = 4$. Since $|Q''| \geq 7$, it follows that at least one pool shares 2 elements with Q''. But it has already been shown that Q'' shares exactly 1 element with any other pool. Hence any triple contains at most one pool of size 8.

Suppose that $Q \cap Q' \neq \emptyset$ and $Q'' \cap (Q \cup Q') = \emptyset$. The latter can only be true if $|Q| = |Q'| = |Q''| = 7$. Let Q be negative and let Q' and Q'' be positive. Then, since $|Q \cap Q' \neq \emptyset| \leq 2$, it follows that $c \geq 5 * 7 = 35 > 32 = 2^5$. Then $3 + 5 + 2 = 10$ tests are required, by Lemma 4 and Lemma 2.

Suppose that $Q \cap Q' = \emptyset$ and $|Q'' \cap (Q \cup Q')| \geq 2$. Then $|Q| = |Q'| = 7$. Let Q be

negative and let Q' and Q'' be positive. If $|Q''| = 8$, then $c = 6*6+6*1+6*1 = 48 > 32 = 2^5$, because pools of different sizes share exactly 1 element. Then $3+5+2 = 10$ tests are required, by Lemma 4 and Lemma 2. The latter is true for $|Q''| = 7$, by Lemma 3.

Suppose that $|Q \cap Q'| = 2$, $|Q'' \cap (Q \cap Q')| = 1$ and that Q'' shares no further elements. Then $|Q| = |Q'| = 7$. Let Q and Q' be positive and let Q'' be negative. Then $c \geq 5*5+5*1+5*1 = 35 > 32 = 2^5$. Then $3+5+2 = 10$ tests are required, by Lemma 4 and Lemma 2.

Let $S = Q \cap Q' \cap Q''$ and suppose that $L \neq \emptyset$. Suppose further that if $v \notin L$, then there exists only 1 v-pool. If $|S| = 1$ and let exactly 2 pools be positive. Then $c \geq 6 * 6 = 36 > 32 = 2^5$. If $|S| = 2$, then all of the 3 pools have size 7. Let all 3 be positive. Then $c = 5 * 2 + 5 * 2 + 5 * 2 + 2 * 3 + 1 = 37 > 32 = 2^5$. Hence, in both cases, $3 + 5 + 2 = 10$ tests are required, by Lemma 4 and Lemma 2.

Suppose that any 2 pools share a set of elements that is not contained by any further pools. If any 2 pools share 1 element and exactly 2 are positive, then $c \geq 5 * 5 + 5 * 1 + 5 * 1 = 35 > 32 = 2^5$ candidate 2-sets remain. Now suppose $|Q \cap Q'| = 2$ and that Q'' shares 1 with each. Let the only Q'' be negative. Then $c = 4 * 4 + 4 * 2 + 4 * 2 + 2 * 2 + 1 = 37 > 32 = 2^5$. Now suppose $|Q \cap Q'| = 2$, $|Q \cap Q''| = 1$ and $|Q' \cap Q''| = 2$. Then each pool shares 2 elements with some other pool, implying that they are all of size 7. Again, let only Q'' be negative. Then $c = 4*3+4*2+3*2+2*4+1 = 35 > 32 = 2^5$. In all of these cases, $3+5+2 = 10$ tests are required, by Lemma 4 and Lemma 2

The only case that remains is that any 2 pools share 2 elements that are not in any further pools. Then all pools have size 7. This triple can not be ruled out in the same manner as the others and is temporarily considered valid.

Quadruplets of pools

Consider a quadruplet of pools. Any three of these must form the only possible triple. Hence any 2 pools must share a set of 2 elements that are in no further pools, and every pool has size 7. Thus, each pool in the triple contains a set S of 3 elements that no other pool in the triple contains. Then it must be the case for the fourth pool Q''' that $|Q''' \cap S| = 2$, because Q''' must form a valid triple with any 2 pools. Then, if exactly 3 of the pools are positive, there exist $2 * 1 * 3 + 2 * 2 * 3 = 18 > 16 = 2^4$ candidate sets of size 2, implying that $4+4+2 = 10$ tests are required, by Lemma 2 and Lemma 4. Hence no quadruplet exists, and it follows that at most 3 pools can exist in stage 1.

The stage 1 pools

Suppose that exactly 3 pools exist in stage 1. If exactly one of these is positive, then at least the instance $K_{3,5}$ remains, implying that at least $3 + t(K_{3,5}) = 10$ tests are required, by Lemma 4. Hence stage 1 consists of at most 2 pools.

It has already been shown that stage 1 contains at least 2 pools. Hence it follows that

stage 1 contains exactly 2 pools. But in any such case, two positive pools imply that at least the instance $K_{5,5}$ remains, from which it follows that $2 + t(K_{5,5}) = 10$ tests are required, by Lemma 4. Hence no 9-test strategy can exist, which contradicts the initial assumption that $t(20, 2, 2) \leq 9$. □

$\mathbf{t(30, 2, 2) = t(31, 2, 2) = 11}$.

Proof. For the upper bound, $31 \leq \binom{8}{2} + 3$ implies that $t(31, 2, 2) \leq 8 + 3 = 11$, by Proposition 1. It suffices to show $t(30, 2, 2) \geq 11$. Assume that $t(30, 2, 2) \leq 10$. Consider a pool Q. If $|Q| = 10$ and Q is negative, then $1 + t(20, 2, 2) = 11$ tests are required tests, by Lemma 4. This is also true for $|Q| \leq 10$, by Lemma 3. Hence the minimum pool size is 11. If $|Q| = 11$ and Q is positive, then there exist $\binom{11}{2} + 11(30 - 11) = 264 > 256 = 2^8$ candidate sets of size 2, implying a total number of $1 + 8 + 2 = 11$ tests, by Lemma 4 and Lemma 2. This is also true for $|Q| \geq 11$, by Lemma 3. Hence no 10-test strategy exists, contradicting the initial assumption that $t(30, 2, 2) \leq 10$. □

6

Discussion

This chapter discusses the results of this thesis work. The significance of the accomplishments is questioned, as well as the strengths and weaknesses of the selected methodology. In addition, possible future directions are suggested considering theoretical tools and automated procedures.

6.1 Significance of results

In chapter 5, several new test numbers $t(n, 2, 2)$ and $t(K_{x,y})$ are presented. Unfortunately, however, very little insight is provided regarding these test numbers in general; the derivations rely mainly on instance-specific symmetries. Even though this is the case, the context-dependent arguments demonstrate multiple ways in which the various theoretical tools from [1] can be applied. In particular, both Theorem 2 and Theorem 3 have very high levels of abstraction and can be applied in completely different ways. For example, conflict graphs can potentially be analyzed from a graph-theoretic perspective.

The observation that comes closest to a more general result is the conjecture that $t(K_{3,x}) = 3 + t(x, 1, 1)$ for $x \geq 5$, as indicated by every instance $5 \leq x \leq 35$. Further insights into the special case of $K_{3,x}$ are provided by Lemma 6 as well as a few observations about pool sizes (see the last few paragraphs of section 5.3.3). The $K_{3,x}$ is rather restrictive and not necessarily inherently significant, but a slightly generalized version of the problem could prove useful. Furthermore, the analysis of this problem provided by this thesis work illustrates to some extent how generalizations can be made regarding special classes of candidate hypergraph products.

On a further note, consider the limited size of the instances solved within this thesis work. Recall from the introductory chapter that group testing can be applied as a pandemic-preventing procedure. Problem instances rooted in such real-world applications are of massive proportions, making it incredibly difficult to find optimal solutions. However, it is still possible to apply strategies for smaller instances in an hierarchical manner. For example, if n is very large, then the set of elements can be subdivided into x disjoint groups G of size n/x. Then G can be treated as a single

defective in an instance of x elements, calling G positive if it contains at least one defective element from the original problem. After each defective group has been identified, the remaining problem can be subdivided further if necessary. Note that there many ways in which one can subdivide the problem. The observation made here is simply that even strategies for small instances can be placed in the context of arbitrarily large ones, though optimality can not be expected.

6.2 Evaluation of methodology

All of the proofs in chapter 5 apply the same procedure: assume that the lowest possible number of tests suffices and then derive a contradiction by systematically eliminating all possible strategies. This, despite its simplicity, is demonstrated to be an effective approach to solving new instances.

Reduction-oriented methods do however have some flaws. As pointed out in the section 6.1, the solutions presented in chapter 5 provide little generalization; many proof details are rather tailor-made for specific instances. This is much due to the fact that Theorem 2 and Theorem 3 are frequently applied in the context of global symmetries of a given problem instance, rather than local properties which can be found in arbitrarily large instances.

A second downside of a systematic elimination process is its inherent tediousness, which is potentially overwhelming. This motivates investigations into automated procedures, which is discussed in section 6.3.2.

6.3 Future directions

This section considers possible future directions that relate rather strongly to this thesis. Note, however, as pointed out in the introductory chapter, that strict group testing is under-researched; there are many other problems within strict group testing that one may investigate

6.3.1 Theoretical tools

A very natural extension to the study of candidate hypergraphs in this paper would be to completely solve the instances of the form $K_{3,x}$. One potential approach for this would be to look for patterns similar to those in Lemma 6, by either deriving new test numbers or 're-derive' existing ones using more generalized methods. If $K_{3,x}$

is shown to be additive, one could attempt to generalize this result into something similar to Theorem 1.

6.3.2 Automated procedures

This thesis work relies on many tedious case distinctions which could, at least to some extent, be automated. For example, it would be useful to have an automated procedure which, given a table of known test numbers, first applies Lemma 3 and Lemma 4 to reduce the number of possible pool sizes, followed by numerous applications of Lemma 2 on single pools, pairs, triples, etc. in order to rule out as many cases as possible. This would be a useful preprocessing module before resorting to more sophisticated methods.

It is much less obvious how one would automate applications of Theorem 2 or Theorem 3. Due to the convenient notation regarding coloring a conflict graph, attempting to automate applications of Theorem 3 is (arguably) a more natural step to take. For example, rather than generating a graph, one could represent a partial vector as a triple (i, j, k) of the indices of its 3 non-open values. The pooling design could for example be represented as a matrix in which every row is an indicator vector. The number of columns would then depend on which elements are relevant for the automated derivation in question.

The preceding remarks are purely speculative, and it is rather unclear how one would carry out such an approach in a computationally tractable manner. As has been noted throughout this chapter, most applications of Theorem 3 in this paper are tailor-made, based on the symmetries of the given problem instance. Such symmetries are not as easily discovered by a preprogrammed procedure; its functionality would rather be preprocessing followed by backtracking, the latter being analogous to deriving contradictions. Nevertheless, even a simplistic implementation focusing entirely on preprocessing could significantly reduce the workload, since the search space grows so rapidly.

7

Ethical Considerations

In this chapter, group testing is examined from the perspective of moral philosophy. Rather than mindlessly advancing the state of science and technology for its own sake, it is important to frequently take a step back and reflect on the ethical implications of one's research. This chapter aims to provide such reflections on group testing in the particular case of biological testing.

7.1 Subject motivation

One possible question to pose regarding the ethical considerations is why the field of group testing should be studied in the first place; why should social institutions invest its precious resources in this particular type of research? As have been seen the in previous chapters, a classic motivating real-world application is biological testing. Throughout this text, the various ethical issues that arise within this particular application are discussed. Even though many other 'needles in a haystack' problems can be formulated as group testing, as was seen in the introductory chapter, the particular case of biological testing is interesting due to the fact that it is so strongly related to saving lives. Before moving on to various ethical considerations, the reader is briefly reminded of the underlying optimization problem.

Consider a large set of people among which a disease has only recently started to spread. To check whether a given individual is healthy, a biological sample is drawn, which in turn is tested through a series of highly resourceful laboratory procedures. The procedures may be costly because, for example, the examination times are lengthy, implying that important instruments are occupied. Furthermore, only a small fraction of the individuals has been infected, implying a great deal of unnecessary testing. It is however possible to combine a group of samples into one, which in turn can be examined. Then, if the examination of such a combination yields a negative result, it follows directly that the corresponding group of people is healthy. If the result is positive, then at least some information is given. The core issue, then, is to find the optimal way combining the biological samples, which is a group testing problem.

At first glance, the biological testing problem above may appear to be important for only a small portion of the population, the individuals with the disease. However, there are cases where the efficiency aspect is crucial for society. Consider a scenario where a few individuals suffer from a highly contagious disease, which may cause a pandemic if the individuals are not placed in quarantine as soon as possible. It is further known that the infectees belong to some subset of the population, but that this subset is too large to be placed into quarantine. In such a situation, it is of great importance that the testing procedure is highly efficient.

7.2 Ethical dilemmas in biological testing

7.2.1 Quarantine

A modified approach to the group testing scenario in the previous section could be to perform tests until it is certain that the number of infected individuals is at most the number of people that can possibly be isolated from the population. However, this raises further ethical questions. Suppose that, due to space limitations, the individuals in the isolation facility can not be placed in separate rooms. If the disease is lethal and difficult to cure, then some isolated healthy persons may be sentenced to death. However, if one performs more tests in order to avoid isolating healthy individuals, then the disease might spread more, thus possibly sentencing even more people to death.

7.2.2 Prioritized infectees

Further ethical issues arise in multi-stage group testing. Typically, it is desirable find a strategy that minimizes the total number of tests. However, the order in which individuals are tested is of ethical significance. Suppose that some individuals are less likely to survive due to, for example, allergies or heart issues. Even though it is necessary to find all of the infected individuals, it is crucial that the particularly sensitive individuals are tested as soon as possible. Enforcing pools that more quickly identify sensitive individuals may lead to an overall suboptimal pooling design in terms of the total number of tests to perform, but such a compromise may need to be considered in some situations.

7.2.3 The fallibility of tests

Another important aspect of group testing is the accuracy of test results. An optimal pooling design relies on completely accurate tests, but in real-world applications tests may be erroneous; physical experiments typically suffer from some measurement

error. This is particularly problematic if one encounters a negative pool. The typical action is to discard all of the supposedly negative elements. However, if the test has a failure probability, then discarding the elements may have severe repercussions, for example that some infected individuals remain untreated. A precautionary action would be to repeat negative tests until the desired level of confidence of the result has been reached. The obvious drawback of such an action is that in the context of group testing it is known that the individual tests are very expensive to perform. Furthermore, assuming that one is more interested in minimizing the number of tests than the number of stages, repeating negative tests requires additional stages, since one clearly needs to perform a given test before the result is known. If a smaller number of stages is of higher priority, then one could have equivalent tests in the same stage. In either case, this gives rise to a dilemma. If tests are repeated, then the overall testing procedure will be very accurate but also very costly. If tests are not repeated, the testing procedure will be efficient, but the result may be inaccurate. In particular, the sequence of actions performed in a multi-stage strategy may be greatly affected by inaccurate tests, leading to a completely wrong conclusion.

7.2.4 Future formulations

To conclude this section, it is noted that group testing research has important practical implications, but that it also gives rise to ethical dilemmas. Such issues include prioritizing fairness or accuracy rather than optimality aspects. To cope with such issues, one would need to do research into different group testing problem formulations, with more sophisticated goal functions to optimize. Such goal functions could for example assign weights to elements or pools to highlight the importance of fairness or accuracy. Another, somewhat simpler, possibility is to have ternary tests, in order make a distinction between defectives and 'important' defectives, for example individuals with allergies. Of course, such alternatives would be mathematical models of the ethical dilemmas rather than solutions, but an accurate problem formulation is a good starting point. An acceptable solution to one of the ethical dilemmas is context-dependent, and a sophisticated mathematical model would bridge the gap between real-world applications and the abstract world of algorithms.

8

Conclusion

This thesis work considers finding solutions to larger strict group testing problem instances than have previously been solved. The approach to accomplish this is to make use of the theoretical foundation presented in [1] by applying various theorems, lemmas and propositions, as well as decomposing problem instances into subproblems of a certain form. These subproblems are also studied independently.

The focus is on the case of two defectives and two stages, the test numbers $t(n, 2, 2)$, as well as the strongly related subproblem of strict nonadaptive group testing on a bipartite candidate graph, that is the test numbers $t(K_{x,y})$. The derivations for the $t(n, 2, 2)$ and $t(K_{x,y})$ numbers are systematic reduction arguments.

Several $t(n, 2, 2)$ and $t(K_{x,y})$ numbers are presented, and it is conjectured that $t(K_{3,x}) = 3+t(x, 1, 1)$ for $x \geq 5$, that is the test number $t(K_{3,x})$ is additive except for particularly small x. These results strongly indicate that the theoretical foundation from [1] can be applied in order to find further test numbers. Furthermore, essential steps in the $t(n, 2, 2)$ derivations rely on knowing $t(K_{x,y})$, which motivates further studies into strict group testing on candidate hypergraph products, as well as other subproblems of strict group testing.

In conclusion, the goal of this thesis is accomplished, at least to a certain degree, and its work can likely be extended upon in the future, possibly by the means of automated methods.

Bibliography

[1] Damaschke P., Sheikh Muhammad A., Wiener G. Strict group testing and the set basis problem. J. Combin. Theory A 126, (2014) 70-91

[2] Damaschke P., Sheikh Muhammad A., Triesch E., Two new perspectives on multi-stage group testing, Algorithmica 67, (2013) 324–354

[3] Du, D.Z., Hwang, F.K.: Combinatorial Group Testing and Its Applications. Series on Appl. Math. 18, World Scientific (2000)

[4] Du, D.Z., Hwang, F.K.: Pooling Designs and Nonadaptive Group Testing. Series on Appl. Math. 18, World Scientific (2006)

[5] De Bonis, A., Gasieniec, L., Vaccaro, U.: Optimal two-stage algorithms for group testing problems. SIAM J. Comp. 34, 1253–1270 (2005)

[6] Eppstein, D., Goodrich, M.T., Hirschberg, D.S.: Improved combinatorial group testing algorithms for real-world problem sizes. SIAM J. Comp. 36, (2007) 1360–1375

[7] Dyachkov, A.G., Rykov, V.V.: Bounds on the length of disjunctive codes. Problems of Info. Transmission (in Russian) 18, (1982) 7–13

[8] Ruszinkó, M.: On the upper bound of the size of the r-cover-free families. J. Combin. Theory A 66, (1994) 302–310

[9] M.T. Goodrich, D.S. Hirschberg, Improved adaptive group testing algorithms with applications to multiple access channels and dead sensor diagnosis, J. Comb. Optim. 15, (2008) 95–121

[10] J. Spencer, Minimal completely separating systems. J. Comb. Theory 8, (1970) 446–447

[11] W.H. Kautz, R.C. Singleton, Nonrandom binary superimposed codes, IEEE Trans. Info. Theory 10, (1964) 363–377

[12] S.H. Huang, F.K. Hwang, When is individual testing optimal for nonadaptive group testing? SIAM J. Discr. Math. 14, (2001) 540–548

[13] A.G. D'yachkov, I.V. Vorobyev, N.A. Polyanskii, V.Y. Shchukin, Bounds on the rate of superimposed codes, Preprint (2014) arXiv 1401.0050 [cs.IT]

[14] A.J. Macula, G.R. Reuter, Simplified searching for two defects, J. Statist. Planning Infer. 66, (1998) 77–82

[15] L.J. Stockmeyer, The set basis problem is NP-complete, Tech. Report RC-5431, IBM, (1975)

[16] T. Jensen, B. Toft, Graph coloring problems, Wiley, New York (1995).

[17] Dorfman, R.: The Detection of Defective Members of Large Populations. The Annals of Math. Stat. 14, (1943) 436–440

[18] Kahng, A.B., Reda, S.: New and Improved BIST Diagnosis Methods from Combinatorial Group Testing Theory. IEEE Trans. CAD of Integr. Circuits and Systems 25, (2006) 533–543

[19] Clementi, A.E.F., Monti, A., Silvestri, R.: Selective Families, Superimposed Codes, and Broadcasting on Unknown Radio Networks. In: SODA 2001. ACM/SIAM (2001) 709–718

[20] De Bonis, A., Vaccaro, U.: Constructions of Generalized Superimposed Codes with Applications to Group Testing and Conflict Resolution in Multiple Access Channels. Theor. Comp. Sc. 306, (2003) 223–243

[21] Goodrich, M.T., Hirschberg, D.S.: Improved Adaptive Group Testing Algorithms with Applications to Multiple Access Channels and Dead Sensor Diagnosis. J. Comb. Optim. 15, (2008) 95–121

[22] Cormode, G., Muthukrishnan, S.: What's Hot and What's Not: Tracking Most Frequent Items Dynamically. ACM Trans. Database Systems 30, (2005) 249–278

[23] A. De Bonis, G. Di Crescenco, Combinatorial group testing for corruption localizing hashing, in: B. Fu, D.Z. Du (Eds.) COCOON 2011, LNCS 6842, Springer, HeidelBerg (2011) 579–591

[24] J. Fang, Z.L. Jiang, S.M. Yiu, L.C.K. Hui, An efficient scheme for hard disk integrity check in digital forensics by hashing with combinatorial group testing, Int. J. Digital Content Technol. Appl. 5, (2011) 300–308

[25] S.A. Zenios and L.M. Wein, Pooled testing for HIV prevalence estimation: Exploiting the dilution effect, Stat.Med. 17, (1998) 1447–1467

[26] Y. Xuan, I. Shin, M.T. Thai, T. Znati, Detecting application denial-of-service attacks: A group-testing-based approach, IEEE Trans. Par. Distr. Syst. 21, (2010) 1203–1216

[27] Stinson, D.R.: Combinatorial Designs: Construction and Analysis. Springer, New York (2003)

[28] Swetz, F.J.: The Legacy of the Luoshu. Wellesly, MA: A.K. Peters / CRC Press (2008)

[29] Wallis, W.D.; George, J.C.: Introduction to Combinatorics. CRC Press (2011)

[30] Graham, R.L.; Grötschel, M; Lovász, L: Handbook of Combinatorics, 2, The MIT Press, Cambridge, MA (1995)

[31] Cummings, L.D.: An undervalued Kirkman paper. Bulletin of the American Mathematical Society (1918)