Deterministic versus Randomized Adaptive Test Cover*

Peter Damaschke†
Department of Computer Science and Engineering
Chalmers University, 41296 Göteborg, Sweden
ptr@chalmers.se

Abstract

In a combinatorial search problem with binary tests, we are given a set of elements (vertices) and a hypergraph of possible tests (hyperedges), and the goal is to find an unknown target element using a minimum number of tests. We explore the expected test number of randomized strategies. Our main results are that the ratio of the randomized and deterministic test numbers can be logarithmic in the number of elements, that the optimal deterministic test number can be approximated (in polynomial time) only within a logarithmic factor, whereas an approximation ratio 2 can be achieved in the randomized case, and that optimal randomized strategies can be efficiently constructed at least for special classes of graphs.

Keywords: combinatorial search, randomization, game theory, LP duality, set cover, fractional graph theory

1 Introduction

1.1 Combinatorial Search: Background

By combinatorial search we mean the identification of an unknown object by certain available tests, also called queries. Here we consider only tests that give binary answers. The setting can be conveniently described in terms of a hypergraph that specifies those available tests.

A *hypergraph* $\mathcal{H}$ is a set U of n *elements*, also called *vertices*, equipped with a family of subsets called the *edges*. One unknown element $u \in U$ is the *target*. A *searcher* can pick any edge T from $\mathcal{H}$ and ask whether $u \in T$. Therefore we also refer to the edges as *tests*. A test is *positive* if $u \in T$, and *negative* else. The searcher aims to identify u efficiently from the outcomes of carefully selected tests from $\mathcal{H}$. The primary goal is to minimize the number of tests that are needed.

Combinatorial search is a natural problem and arises in practice, e.g., in software testing and, most notably, in biological testing [2, 3]. A number of more specific, classic combinatorial search problems can also be formulated in the above way. Perhaps the foremost example is the group testing problem [10] which found numerous applications. Note that even problems like sorting by comparisons fit in this framework. (The elements are all permutations of a given set S of numbers, the target is the sorted sequence, and every test is a comparison of

*An early version appeared in the Proceedings of the *9th International Conference on Algorithms and Complexity CIAC 2015*, Paris, *Lecture Notes in Computer Science*, Springer, vol. 9079, pp. 182–193. This is a completely reworked version with strengthened results. In order to keep the focus on adaptive testing, some minor results are also left out.

†Tel. 0046 31 772 5405, Fax 0046 31 772 3663

two numbers.) One may also think of $\mathcal{H}$ as a system of binary attributes of objects, and then an efficient test strategy is also a concise *classification system*.

Besides deterministic search strategies, randomization has been used for many specific search problems. As a few examples, Quicksort is a famous sorting algorithm, and randomized constructions have been extensively studied for group testing, e.g., in [4, 5, 16].

The present work explores the power of randomization in combinatorial search on hypergraphs of arbitrary structure, with a focus on small tests. To the best of our knowledge, there is no study of the subject in this generality so far, therefore we hope to initiate a new research direction within an established field. We state the main results below in Section 1.3, as we need to give the necessary definitions first.

1.2 General Definitions

We define the problems and complexity measures that make up the subject of this work. More specific and technical definitions are given later when they are needed.

Target Search: Given a hypergraph $\mathcal{H}$ on a set U of elements and an unknown target $u \in U$, identify u by executing tests from $\mathcal{H}$.

The *rank* r of a hypergraph is the maximum size of its edges. Without loss of generality we can assume $r \leq \frac{1}{2}n$ in Target Search instances, since an edge and its complement represent the same test, with positive and negative answers swapped. Closely related to the above problem is:

Target Presence: Given a hypergraph $\mathcal{H}$ on a set U of elements and an unknown target $u \in U$, find some positive test in $\mathcal{H}$.

An equivalent formulation is: The searcher must confirm the presence of a target u, but is not required to identify u. At first glance, Target Presence may look less natural than Target Search, but for hypergraphs with rank r being small compared to n (especially, for fixed r) the two problems are asymptotically the same, because, once a positive test is found, there remain only r candidate elements for the target, thus we can also identify the target with $O(r)$ additional tests. Moreover it turns out that Target Presence is formally a bit easier to handle.

A hypergraph $\mathcal{H}$ is *separating* if for any two elements there exists an edge that contains exactly one of them. In other words, no two targets cause the same outcomes of all possible tests. In Target Search we will always implicitly assume that $\mathcal{H}$ is separating, since otherwise it would be impossible to distinguish all targets. A separating hypergraph may have one isolated element that belongs to no edge. But we also implicitly assume that no isolated element exists. For Target Presence this is a necessary restriction, and for Target Search, the special case that the isolated element is the target would pose only some trivial problems.

A search strategy can work in rounds, where all tests in a round are done in parallel, without waiting for each other's outcomes. An *adaptive* strategy performs only one test per round. The nonadaptive case of Target Search, with only one round, is known as the Test Cover problem. It can be rephrased as follows: Given a separating hypergraph $\mathcal{H}$, find a smallest subset of the edges of $\mathcal{H}$ that still form a separating hypergraph.

The complexity of Test Cover has been intensively studied in various directions [1, 3, 8, 9, 12, 14], whereas very little is known for strategies with more than one round; see [20] for some combinatorial results. It is worth noticing that adaptive strategies can save many tests compared to the one-round version, hence these two cases are quite different. Here is a simple example with rank $r = 2$. Let the hypergraph be a complete graph of n vertices. Then an adaptive strategy finds the target by roughly $n/2$ tests, since one can simply ask pairwise disjoint edges until a positive answer is obtained. Note that the tests need not separate all elements; they only need to separate the target from the other elements. In contrast, any

nonadaptive strategy must be a separating subset of the edges, which requires about $2n/3$ tests: A moment of thinking reveals that the optimal separating set is a graph where every connected component has two edges and three vertices; see also [13].

Notably, Test Cover for hypergraphs with fixed rank r found special interest, since already this restricted case has practical relevance [2, 7, 13]. It appears if every testable property is specific to a few elements only. (As an example from molecular biology, most antibodies used as test agents to identify proteins bind specifically to certain protein fragments [2].) The problem with rank $r = 2$ is known as TCP2.

Next, search strategies are *deterministic* if the choice of tests for each round is uniquely determined by the outcomes of earlier tests, whereas in the more general class of *randomized* strategies, the choice of tests for each round can, additionally, depend on random decisions of the searcher. We stress that the tests themselves behave deterministically; here we assume neither random errors nor a previously known target probability distribution. Adaptive strategies can be formally defined by the following points:

- The restriction of a hypergraph $\mathcal{H}$ with vertex set U to a subset $V \subset U$ is the hypergraph whose vertex set is V, and whose edges are all nonempty and distinct sets $V \cap T$, where T runs through the edges of $\mathcal{H}$.
- A randomized adaptive strategy A is a function that assigns to every hypergraph $\mathcal{H}$ with vertex set U, $|U| \geq 2$, a probability distribution on its edges.
- A deterministic adaptive strategy A is a function that assigns to every hypergraph $\mathcal{H}$ with vertex set U, $|U| \geq 2$, one of its edges T (or equivalently, a probability distribution where some edge T gets probability 1).
- The semantics is as follows: A draws an edge T according to the probability distribution and tests it. If the test T is positive (negative), then A continues on the restriction of $\mathcal{H}$ to T (to $V \setminus T$).

A deterministic strategy is *optimal* among all deterministic strategies if it minimizes the worst-case number of tests, and a randomized strategy is optimal if it minimizes the worst-case *expected* number of tests. Note that the worst case refers to maximization over all possible targets. For clarity we list the straightforward formal definitions of the test numbers below. Consider an instance of Target Search, that is, a separating hypergraph $\mathcal{H}$ without isolated element.

- Let $Det(\mathcal{H})$ and $Rand(\mathcal{H})$ denote the set of all deterministic and randomized search strategies, respectively, on $\mathcal{H}$. Note that $Det(\mathcal{H}) \subset Rand(\mathcal{H})$.
- For $A \in Rand(\mathcal{H})$ and an element $u \in U$, let $t_A(\mathcal{H}, u)$ denote the expected number of tests done by A if u is the target. If $A \in Det(\mathcal{H})$, this "expected" number is simply the deterministic number of tests.
- The (worst-case, expected) test number of strategy A is defined as $t_A(\mathcal{H}) := \max_{u \in U} t_A(\mathcal{H}, u)$.
- We define the optimal randomized and deterministic test numbers, respectively, as follows: $t_{rand}(\mathcal{H}) := \min_{A \in Rand(\mathcal{H})} t_A(\mathcal{H})$ and $t_{det}(\mathcal{H}) := \min_{A \in Det(\mathcal{H})} t_A(\mathcal{H})$.
- For Target Presence the definitions are similar, but we give all test numbers the superscript 1, for instance, we denote the optimal test numbers $t^1_{rand}(\mathcal{H})$ and $t^1_{det}(\mathcal{H})$.

In this paper we use the phrase *random order* with a specific meaning: A random order of m objects is an order where every fixed object is at expected position $\frac{1}{2}m + \frac{1}{2}$. This can be a random permutation, or (to save random bits) a random cyclic shift of a fixed order, or simply a fixed order and its reverse, both chosen with probability $\frac{1}{2}$. We will see later, within our search strategies, that this is a handy notation.

1.3 Contributions

We focus on adaptive testing. Here we highlight only the principal findings and skip smaller and auxiliary results.

- An obvious question is how much the test number can benefit from randomization. In other words: How large can the ratio $t_{det}(\mathcal{H})/t_{rand}(\mathcal{H})$ be? In Section 3 we first show that this ratio is at least 2 (minus a vanishing term) for all large enough hypergraphs of any fixed rank.

- At first glimpse one might intuitively think that 2 is already the largest possible ratio. But we will further see in Section 3 that $t_{det}(\mathcal{H})/t_{rand}(\mathcal{H})$ is not bounded by any constant. We present a sequence of hypergraphs where this ratio is logarithmic in the number of elements, and these hypergraphs are even "well behaved" in the sense that their randomized test complexity is only logarithmic, i.e., close to the information-theoretic lower bound.[1] Our construction uses, among precautions, finite projective spaces and their logarithmic integrality gap for the set cover problem. We conjecture that the logarithmic ratio is maximal, subject to constant factors.

- Section 4 deals with the complexity of computing optimal search strategies. We show NP-hardness of the deterministic problem for any fixed rank $r \geq 3$. This is not a surprise but raises an open question: We conjecture NP-hardness of the randomized problem, too. We also complement the hardness result with 2-approximability of the randomized test number.

- The case of graphs, i.e., hypergraphs of rank 2, was of particular interest already in the realm of Test Cover. Here we get partial results on randomized adaptive strategies, based on half-integral minimum fractional edge covers. Their structural properties are provided in Section 5. Then, Section 6 presents a lower bound and optimal strategies (matching this bound) for special graphs, including bipartite graphs and graphs that possess a perfect matching.

2 Preliminaries

2.1 Game-Theoretic Interpretation and Lower Bound

The following consideration is formulated for Target Presence, but it holds similarly for Target Search as well. In search problems it is common to imagine an adversary choosing the target. We will use that later for obtaining randomized lower bounds. (Some readers might even find the game interpretation as such more natural than the description in Section 1.2.)

Any randomized search strategy on a hypergraph $\mathcal{H}$ can be viewed as a probability distribution on the (finite) set of the deterministic strategies on $\mathcal{H}$, also called a *mixed strategy*. This allows us to apply the classic theory of zero-sum games. The player who wants to solve

[1] In the conference version we could only show ratio 4, and we needed an extra construction to obtain examples that also enjoy the latter property. In the new proof the logarithmic test number comes out more naturally.

TARGET PRESENCE follows a mixed strategy as above, and an adversary plays a mixed strategy, too, which is a probability distribution on U specifying the probability of each element being the target. We also refer to it as a *target distribution*. By von Neumann's minimax theorem, a special case of LP strong duality, we get:

Proposition 1. *There exists a pair of optimal mixed strategies such that the player does an expected number of at most $t^1_{rand}(\mathcal{H})$ tests whatever the target is, and every deterministic strategy needs an expected number of at least $t^1_{rand}(\mathcal{H})$ tests. Only deterministic strategies that attain this expected value can be involved with nonzero probability in an optimal mixed strategy of the player.*

The optimal strategies can be computed by a linear program (LP), with the caveat that the number of deterministic strategies to consider may not be polynomial in n. We formulate an immediate consequence of Proposition 1; the general reasoning is also known as Yao's minimax principle:

Proposition 2. *Let τ be any target distribution that gives every $u \in U$ the probability $\tau(u)$ of being the target. Then we have*

$$t^1_{rand}(\mathcal{H}) \geq \min_{A \in Det(\mathcal{H})} \sum_{u \in U} \tau(u) t^1_A(\mathcal{H}, u).$$

In words: The best (with respect to the target distribution) expected test number that can be achieved by a deterministic strategy lower-bounds the randomized test number.

2.2 Set Cover versus Target Search and Presence

A *set cover* in $\mathcal{H}$ is any subset of the edges whose union is U. Let $\nu := \nu(\mathcal{H})$ denote the minimum size (number of edges) of a set cover. A *fractional set cover* assigns a non-negative weight x_T to every edge T, such that every element belongs to edges with a total weight at least 1. A *fractional independent set* assigns a non-negative weight to every element, such that every edge contains elements with a total weight at most 1.

Finding a maximum-weight fractional independent set and a minimum-weight fractional set cover are the natural LP relaxations of the corresponding integral optimization problems, and they form a well-known pair of dual LPs, hence due to LP strong duality they achieve the same optimal value on any given hypergraph. We denote this value $\alpha := \alpha(\mathcal{H})$.

The reason why we introduced TARGET PRESENCE is that its test number is closely related to α and ν, which will also help investigate TARGET SEARCH. First consider deterministic strategies.

Proposition 3. *In any hypergraph $\mathcal{H}$ of rank r we have:*

$$\alpha \leq \nu = t^1_{det}(\mathcal{H}) \leq t_{det}(\mathcal{H}) + 1 \leq t^1_{det}(\mathcal{H}) + r = \nu + r.$$

$$t_{det}(\mathcal{H}) \leq \nu + r - 2.$$

Proof. Weak duality gives $\alpha \leq \nu$. Testing all edges of some minimum set cover solves TARGET PRESENCE, and there is no better strategy, since the first $\nu - 1$ tests could be negative. (In other words, an adversary can always choose an element covered last as the target.) Thus $\nu = t^1_{det}(\mathcal{H})$.

Next, consider any strategy for TARGET SEARCH. As soon as some test is positive, TARGET PRESENCE is solved. If all tests are negative until the target u is identified, then only one more edge containing u needs to be tested to solve TARGET PRESENCE. Thus $t^1_{det}(\mathcal{H}) \leq t_{det}(\mathcal{H}) + 1$.

To show $t_{det}(\mathcal{H}) \leq t^1_{det}(\mathcal{H}) + r - 1$ we start running an optimal strategy A for TARGET PRESENCE. If a positive test is encountered, there remain at most r candidate elements for

the target. Since $\mathcal{H}$ is separating, we always find some test being positive for some candidate element and negative for another one. Thus $r-1$ further tests suffice to pin down the target. If all tests are negative until only one test of A is left, we withhold this final test. Instead, we observe again that at most r candidate elements are left: If there were more, then A would need at least two further tests. Hence we can finish as in the former case.

The last inequality of this chain, $t_{det}(\mathcal{H}) \leq \nu + r - 1$, is now obvious. To show the slightly stronger bound $t_{det}(\mathcal{H}) \leq \nu + r - 2$ we test the first $\nu - 1$ edges of some minimum set cover, until a positive test is encountered. If all $\nu - 1$ tests are negative, then we know that the target is in the last edge. In both cases, similarly as above, at most $r-1$ further tests identify the target. □

For randomized strategies for Target Presence we observe the following simple and general bounds. Here $H_n := \sum_{i=1}^{n} \frac{1}{i}$ denotes the nth Harmonic number. Note that $H_n \sim \ln n$.

Proposition 4. *If a hypergraph $\mathcal{H}$ with rank r has a fractional independent set of total weight α, then $\frac{1}{2}\alpha - O(1) \leq t^1_{rand}(\mathcal{H}) \leq \frac{1}{2}\nu + \frac{1}{2} \leq \frac{1}{2}\alpha H_r + \frac{1}{2}$.*

Proof. To prove $\frac{1}{2}\alpha - O(1) \leq t^1_{rand}(\mathcal{H})$, we normalize the assumed fractional independent set, that is, divide all weights by α, such that the weights sum up to 1 and hence form a probability distribution on U. By Yao's minimax principle it suffices to show that every deterministic strategy A needs an expected number of at least $\frac{1}{2}\alpha - O(1)$ tests on this target distribution. As long as all tests are negative, strategy A tests the edges in some specific order. Note that every edge has a probability mass of at most $\frac{1}{\alpha}$. By an exchange argument, the expected number of tests until A hits the target is minimized if every tested edge covers new elements whose total probability mass equals the largest possible value $\frac{1}{\alpha}$. (For $x \geq y$ we have $ix + (i+1)y \leq iy + (i+1)x$. It follows that, for any sequence of numbers $x_1 \geq \ldots \geq x_n$, the permutation π of $\{1, \ldots, n\}$ that minimizes the sum $\sum_{i=1}^{k} i x_{\pi(i)}$ is the identical permutation with $\pi(i) = i$ for all i.) Applying the simple identity $\frac{1}{k}\sum_{i=1}^{k} i = \frac{1}{2k}k(k+1) = \frac{k+1}{2}$ to $k = \lceil \alpha \rceil$, the claimed lower bound on the expected test number follows.

Inequality $t^1_{rand}(\mathcal{H}) \leq \frac{1}{2}\nu + \frac{1}{2}$ is seen as follows. Let us test the edges of some minimum set cover $\mathcal{B}$ in random order (see Section 1.2). For the analysis we assign every element u arbitrarily to some edge $T \in \mathcal{B}$ with $u \in T$, called the designated edge of u. Due to the random order of $\mathcal{B}$, the designated edge of any fixed target u appears after an expected number of at most $\frac{1}{2}\nu + \frac{1}{2}$ tests. The last inequality $\nu \leq \alpha H_r$ is just the known integrality gap H_r for set cover; see Section 13.1.1 in [19]. □

Another chain of inequalities is concerned with the randomized test number of Target Search.

Proposition 5. *In any hypergraph $\mathcal{H}$ of rank r we have:*

$$t^1_{rand}(\mathcal{H}) \leq t_{rand}(\mathcal{H}) + 1 \leq t^1_{rand}(\mathcal{H}) + r \leq \frac{1}{2}\nu + \frac{1}{2} + r.$$

Proof. The arguments for the first two inequalities are the same as in the proof of Proposition 3, as they also apply to every possible computation path of a randomized strategy, and the last relation was shown in Proposition 4. □

3 Deterministic versus Randomized Test Number

Now we approach one of the main questions: How large can the ratio $t_{det}(\mathcal{H})/t_{rand}(\mathcal{H})$ be?

From Proposition 3 and 5 we have that $t_{det}(\mathcal{H}) \geq \nu - 1$ whereas $t_{rand}(\mathcal{H}) \leq \frac{1}{2}\nu + r - \frac{1}{2}$ for hypergraphs $\mathcal{H}$ of rank r. It follows:

Theorem 6. *Let r be any fixed rank, and let $\epsilon > 0$ be any fixed real number. Then all large enough hypergraphs $\mathcal{H}$ of rank r satisfy $t_{det}(\mathcal{H})/t_{rand}(\mathcal{H}) > 2 - \epsilon$.*

One might expect that treating edges of a set cover in random order, as in Proposition 4, is already the best use of randomization, which would imply that ratio 2 is the largest possible. But inspecting the proof of Proposition 4 again, we also observe that element u may appear earlier than in its designated edge, indicating that the ratio may possibly become larger. Indeed, we will show that the ratio can be logarithmic in n, in certain hypergraphs whose ranks are no longer fixed.

The idea for obtaining such examples is as follows: From Section 2.2 we know $t_{det}(\mathcal{H}) \geq \nu - 1$ and $t_{rand}(\mathcal{H}) \geq \frac{1}{2}\alpha - O(1)$. Hence we should look for hypergraphs that exhibit large set cover integrality gaps $\frac{\nu}{\alpha}$ and furthermore enable randomized strategies close to the mentioned lower bound. As building blocks we use hypergraphs described in Section 13.1.1 in [19], also known as *finite projective spaces*. For the sake of completeness we describe their construction.

For an integer d, consider a hypergraph $\mathcal{H}$ with $2^d - 1$ elements and edges, respectively, both indexed by the $2^d - 1$ binary vectors of length d without the zero vector. We apply algebraic operations over the field $GF[2]$. An element belongs to an edge if and only if the inner product of their vectors equals 1. Each element belongs to (slightly more than) half of the edges. Also remember from the statement of Target Presence that the target belongs to the vertex set. Thus a strategy for Target Presence that tests random edges succeeds after less than 2 expected tests, which means $t^1_{rand}(\mathcal{H}) < 2$. Any deterministic strategy may receive up to $d - 1$ negative answers. In fact, these answers are consistent since the corresponding homogeneous linear system with $d - 1$ equations in d variables retains a non-zero solution. Hence another test is needed, which implies $t^1_{det}(\mathcal{H}) = d$. (Note that d tests are also sufficient, even for Target Search, since an obvious strategy using unit vectors can figure out the components of the target vector one by one, and some test is positive.)

The example shows that, for Target Presence, randomized testing can be more efficient than deterministic testing by a logarithmic factor. To carry over this result to our actual problem, Target Search, we need yet another operation: Let $k\mathcal{H}$ denote the disjoint union of k copies of $\mathcal{H}$. Some of the following observations can be extended to any disjoint unions of hypergraphs, but we will need the special case $k\mathcal{H}$ only.

Lemma 7. *In any hypergraph $\mathcal{H}$ we have:*

$$t_{det}(k\mathcal{H}) \geq t^1_{det}(k\mathcal{H}) - 1 = k \cdot t^1_{det}(\mathcal{H}) - 1.$$

$$t_{rand}(k\mathcal{H}) \leq k \cdot t^1_{rand}(\mathcal{H}) - \frac{1}{2}k + \frac{1}{2} + t_{rand}(\mathcal{H}).$$

Proof. The deterministic lower bound follows immediately from Proposition 3, observing that $t^1_{det}(\mathcal{H}) = \nu(\mathcal{H})$.

Next we show the upper bound on $t_{rand}(k\mathcal{H})$. By the specification of Target Presence, one of the k copies of $\mathcal{H}$ contains the target. It might seem that we can order the copies randomly and search them one by one, but then it would be unclear when to stop and to proceed to the next copy. This thought suggests the following strategy. We put the k copies of $\mathcal{H}$ in random order (see Section 1.2), such that the copy containing the target is at expected position $\frac{1}{2}k + \frac{1}{2}$. Furthermore, we fix some randomized strategy A for Target Presence in $\mathcal{H}$, but we apply A to every copy of $\mathcal{H}$ in a round-robin fashion: We do the first step of A in all

k copies, then do the second step of A in all k copies, and so on. When the first positive test is done, we know which copy of $\mathcal{H}$ contains the target, and we search for it using a randomized strategy for TARGET SEARCH in $\mathcal{H}$. (The information from the negative tests already done in this copy could be used, but for simplicity we start the search from scratch.) This concludes the description of our strategy for $k\mathcal{H}$.

Note that A defines some randomized order of the edges of $\mathcal{H}$ (being tested until a positive outcome), and the full number k of tests is done each time, as long as the target is not yet covered. By linearity of expectation, the expected number of tests until the first positive outcome is $k(t^1_{rand}(\mathcal{H}) - 1) + \frac{1}{2}k + \frac{1}{2}$, and the assertion follows. □

In the sequel, log denotes the logarithm base 2. The following theorem says that, in some hypergraphs, the ratio of the deterministic and randomized complexity of TARGET SEARCH is logarithmic, and at the same time the randomized complexity is close to the (information-theoretic) logarithmic lower bound. In variant (ii) the randomized complexity is truly logarithmic, but the factor of $\log n$ in the ratio is smaller than in variant (i).

Theorem 8. *(logarithmic ratios)*
(i) For every $\epsilon > 0$ and $\delta > 0$ there exist hypergraphs $\mathcal{L}$ with $t_{rand}(\mathcal{L}) < \frac{3}{2}(\log n)^{1+\delta}$ and $t_{det}(\mathcal{L})/t_{rand}(\mathcal{L}) > (\frac{2}{3} - \epsilon)\log n$.
(ii) For every $\epsilon > 0$ and $\delta' > 0$ there exist hypergraphs $\mathcal{L}$ with $t_{rand}(\mathcal{L}) < (\frac{3}{2}\delta' + 1)\log n$ and $t_{det}(\mathcal{L})/t_{rand}(\mathcal{L}) > (\frac{2\delta'}{3\delta'+2} - \epsilon)\log n$.

Proof. We choose $\mathcal{H}$ as a finite projective space as above, and $\mathcal{L} = k\mathcal{H}$. We apply Lemma 7 and solve TARGET SEARCH on $\mathcal{H}$ deterministically; a randomized strategy cannot be faster there. Lemma 7 yields $t_{det}(k\mathcal{H}) \geq kd - 1$ and $t_{rand}(k\mathcal{H}) < 2k - \frac{1}{2}k + \frac{1}{2} + d = \frac{3}{2}k + \frac{1}{2} + d$. The number of elements in $k\mathcal{H}$ is $n := k(2^d - 1)$. For the asymptotic behavior for $n \to \infty$ we can neglect the constant summands and simplify the expressions to $t_{det}(\mathcal{L})/t_{rand}(\mathcal{L}) = \frac{2kd}{3k+2d}$ and $n = k2^d$. Note that $d = \log n - \log k$. By choosing $k = (\log n)^{1+\delta}$ we can make k grow faster than d, hence $t_{det}(\mathcal{L})/t_{rand}(\mathcal{L})$ tends to $\frac{2}{3}d$. Altogether this yields (i). Alternatively we may choose $k = \delta' \log n$ to obtain (ii). □

We conjecture that, up to constant factors, the logarithmic gap is the largest possible, that is, $t_{det}(\mathcal{H})/t_{rand}(\mathcal{H}) = O(\log n)$. A possible way to prove this could be to consider any optimal deterministic strategy as a binary tree (according to the test outcomes), define a suitable target distribution depending on the shape of this tree, and apply Yao's principle. However there seem to be technical challenges in this approach. Here we can confirm a tight logarithmic bound only for hypergraphs of fixed rank. (Also compare the second assertion in the next Proposition to the general lower bound 2 in Theorem 6).

Proposition 9. *For every fixed r, and for every $\epsilon > 0$, all large enough hypergraphs $\mathcal{H}$ of rank r satisfy $t_{det}(\mathcal{H})/t_{rand}(\mathcal{H}) < 2H_r - \epsilon \approx 1.4 \log r - \epsilon$. Moreover, $t_{det}(\mathcal{H})/t_{rand}(\mathcal{H}) \geq \log r$ holds in some hypergraphs of rank r.*

Proof. We get the upper bound from the inequalities in Propositions 3 and 4 via $t_{det}(\mathcal{H}) < \nu + r \leq H_r\alpha + r \leq 2H_r(t^1_{rand}(\mathcal{H}) + O(1)) + r \leq 2H_r(t_{rand}(\mathcal{H}) + O(1)) + r$. Divide by $t_{rand}(\mathcal{H})$ and observe that r was fixed. The lower bound follows from the construction in Theorem 8, applied with fixed $d = \Theta(\log r)$. □

4 Computational Complexity for Hypergraphs of Fixed Rank

As shown in [18], in hypergraphs of rank $r \geq 3$ it is NP-hard to approximate the size of a minimum set cover within a factor smaller than H_r. As usual, we refer to the problem as SET COVER. For $r = 2$ the problem is equivalent to computing a maximum matching and therefore

polynomial. The idea of the following proof is similar to the inapproximabilty result for TEST COVER in [2].

Theorem 10. *Deterministic* TARGET SEARCH *for hypergraphs $\mathcal{H}$ of any fixed rank $r \geq 3$ is NP-hard, and it is even NP-hard to approximate $t_{det}(\mathcal{H})$ within a factor smaller than H_r in polynomial time. On the positive side, $t_{det}(\mathcal{H})$ can be approximated within a factor H_r in polynomial time.*

Proof. By Proposition 3 we have $t^1_{det}(\mathcal{H}) = \nu$, and $t_{det}(\mathcal{H})$ and $t^1_{det}(\mathcal{H})$ differ by an additive term at most r. Now let $\mathcal{H}$ be an instance of SET COVER with rank r. We construct $k\mathcal{H}$ for some $k > 2r$. Due to the previous remarks, $t^1_{det}(k\mathcal{H}) = k\nu$ and consequently $k(\nu - \frac{1}{2}) < t_{det}(k\mathcal{H}) < k(\nu + \frac{1}{2})$. Thus we could recover ν from $t_{det}(k\mathcal{H})$. This way we have have reduced SET COVER to TARGET SEARCH in polynomial time. Non-approximability within a factor smaller than H_r follows from the result in [18] and the above simple facts on $t^1_{det}(\mathcal{H})$. For the positive result, simply apply the greedy algorithm for SET COVER [6]. □

Of course, we may also infer NP-hardness of the exact problem from non-approximability, but we inserted the above proof anyway, as it is an elementary argument that does not hinge on the deep non-approximability result.

Next, from Theorem 10 we cannot conclude anything about the exact computation of $t_{rand}(\mathcal{H})$. It is natural to conjecture NP-hardness for any fixed rank r as well, but we must leave this question open, even for $r = 2$. While the deterministic version of TARGET PRESENCE is basically SET COVER, we lack such a combinatorial characterization of optimal randomized strategies. We will come back to this point for $r = 2$ later on.

As opposed to $t_{det}(\mathcal{H})$ and $t^1_{det}(\mathcal{H})$ we can approximate $t_{rand}(\mathcal{H})$ and $t^1_{rand}(\mathcal{H})$ much better, within a factor 2. At first glance this may appear surprising, but the reason is that the randomized version is related to a fractional problem solvable by LP:

Proposition 11. *In arbitrary hypergraphs $\mathcal{H}$, a strategy for* TARGET PRESENCE *with expected test number at most α exists and can be computed in polynomial time. Thus, $t^1_{rand}(\mathcal{H})$ can be approximated within factor 2 in polynomial time.*

Proof. Let u be the unknown target. We compute a fractional set cover with optimal total weight α by an LP, and divide all edge weights by α to get a probability distribution on the edges. Then we repeatedly draw an edge from this probability distribution and test it, until we see a positive outcome. Since every element, in particular u, belongs to edges with total weight at least 1, we pick an edge containing u with probability at least $1/\alpha$ each time. Thus the expected number of tests is at most α. As seen in Proposition 4, $t^1_{rand}(\mathcal{H})$ is bounded from below by essentially $\frac{1}{2}\alpha$. □

From Propositions 5 and 11 it follows instantly:

Corollary 12. *As for* TARGET SEARCH *in arbitrary hypergraphs $\mathcal{H}$ of fixed rank, $t_{rand}(\mathcal{H})$ can be approximated within a factor $2 - o(1)$ in polynomial time.*

Looking back to the strategy in Proposition 11, it seems more rewarding to sample edges without repetition: Delete the edges tested negative, and sample from the remaining edges, with probabilities proportional to their weights. (One may even recalculate a minimum fractional set cover each time.) But in the worst case all negative edges might have very small weights, and then their deletion has only little effect on the test number. Thus it would be helpful to have fractional set covers which are not only optimal in terms of total weight, but have the further property that the nonzero edge weights are "large". However, it remains open whether an approximation ratio better than 2 can be obtained in this way.

5 On Fractional Set Covers in Graphs

Next we study the case of rank $r = 2$ closer, that is, when the hypergraphs are graphs. The motivations are twofold: The graph case has relevance and found special attention in the non-adaptive setting ([13] and others), which makes it only natural to look at the adaptive setting as well, and it is structurally interesting, in particular, it has nice relationships to fractional set covers.

The graph case appears naturally when each testable property is shared by at most two elements. In fact, for Target Presence it suffices to consider only 2-uniform hypergraphs where all edges have *exactly* two elements, that is, usual graphs $G = (V, E)$ with vertex set V and edge set E. This is not a restriction, since every edge with a single element contained in another edge of size 2 is useless and can be removed, and every element only contained in an edge of size 1 can be duplicated, without changing the problem. Also remember that Target Search and Target Presence differ (for $r = 2$) by at most one test, therefore it suffices to consider the latter problem. Thus, we are now concerned with $t^1_{rand}(G)$ in graphs G.

In this section we establish some structural and algorithmic properties of fractional set covers (edge covers) in graphs, which might also be of independent interest.

A *tour* in a graph is a sequence of edges where the end of every edge is the start of the next edge, and the last edge returns to the start vertex of the first edge. Note that edges may appear several times in a tour and may be traversed in arbitrary directions. The *length* of a tour is the number of edges, where repeatedly traversed edges are counted again. A tour of even (odd) length is called an *even (odd) tour*. An edge in a tour is *balanced* if it is traversed forth and back the same number of times, otherwise the edge is *unbalanced*. Finally, an *unbalanced tour* is a tour containing at least one unbalanced edge.

Lemma 13. *In a given graph we can find an unbalanced even tour, or report that no such tour exists, in linear time.*

Proof. Clearly, it suffices to consider connected graphs. We do breadth-first-search (BFS), starting in an arbitrary root vertex. BFS divides the set of vertices into layers, where each layer comprises all vertices with the same distance to the root. Furthermore, two types of edges exist: An inter-edge joins two vertices from consecutive layers, while an intra-edge joins two vertices from the same layer. The inter-edges form a bipartite subgraph. Any cycle in this subgraph is even, and is therefore an unbalanced even tour. Suppose that no inter-edges form a cycle. Then the following claims are easy to check. If at most one intra-edge exists, then the graph does not admit an unbalanced even tour. If at least two intra-edges exist, then we can form a tour that traverses two intra-edges exactly once and, apart from them, uses only inter-edges. Moreover, this tour is even, and the two inter-edges are unbalanced. Hence we obtain an unbalanced even tour. All mentioned case distinctions and operations can be performed in linear time by standard techniques. □

A *star* is a graph of at least two edges, where all edges have one common vertex, called the *center* of the star, while the other vertices are called *leaves* of the star. A *blossom* is a simple, i.e., not self-crossing cycle of odd length. (The name is adopted from the maximum matching algorithm in [11].)

A function with range $[0, 1]$ is *half-integral* if it attains only the values $0, \frac{1}{2}, 1$. Half-integrality is well studied for several optimization problems and is a tool for approximation algorithms (e.g., in [15]). The following result and the proof technique strongly resembles the considerations for fractional matchings and fractional transversals in [17], however this is not the structure we need, therefore we have to do the same for fractional set covers (synonymously, fractional edge covers). To our best knowledge, the following theorem did not appear in earlier literature (maybe because the corresponding integer problem, Edge Cover in graphs, is also solvable in polynomial time).

Theorem 14. *Every graph $G = (V, E)$ has an optimal fractional set cover with weights $x_e \leq 1$ for all $e \in E$, that enjoys the following properties, and such a fractional set cover can be obtained in polynomial time.*
Let Z, F, I denote the set of edges e with $x_e = 0$, $0 < x_e < 1$, $x_e = 1$, respectively. (Z and F stand for zero and fractional.) These symbols also denote the subgraphs spanned by the respective edge sets.
(i) The set cover is half-integral, that is, all edges $e \in F$ have $x_e = \frac{1}{2}$.
(ii) The connected components of I are stars and single edges.
(iii) The connected components of F are blossoms.
(iv) No two edges, one from F and one from I, are incident.
(v) No edge joins two leaves of stars (from I).
(vi) No alternating path of edges from Z and I (starting and ending in I) joins a blossom and a leaf of a star, or two blossoms.

Proof. We compute an optimal fractional set cover by an LP in polynomial time. Then we transform it step by step to obtain the desired properties. The transformation rules are applied as long as possible, and certain decreasing cardinalities ensure that the number of steps remains polynomial. First observe that $x_e \leq 1$ for all edges e, since otherwise we could reduce x_e to 1 and get a better valid solution, contradicting optimality.

Three or more edges of I can never form a path or cycle, since otherwise the weight of some edge could be reduced to 0, contradicting optimality. Hence I consists of vertex-disjoint stars and edges.

Let C be an unbalanced even tour in F; by Lemma 13 we can find some in polynomial time if some exists. We may change the weights of the edges in C alternatingly by some amount $+y$ and $-y$. This affects neither the sum of weights incident to any vertex, nor the total weight of the set cover. The net effect on x_e is the sum of all changes applied when e is traversed, hence it is zero if e is balanced. But C has some unbalanced edge where x_e actually changes. Specifically, we apply the smallest $y > 0$ such that some x_e becomes 0 or 1 while all other x_e values are still in $[0, 1]$. Thus we get another optimal solution where F has one edge less. Iterating the procedure we destroy all unbalanced even tours in F.

Next, consider any edge $f \in F$. Since the two vertices of f must be covered by weights at least 1, they are incident to further edges $e, g \notin Z$. If $e, g \in I$ then x_f can drop to 0, contradicting optimality. If $e \in I$ and $g \in F$ (the other case is symmetric), we can decrease x_f and increase x_g by the same amount, such that either f or g is no longer in F. Again, we get an optimal solution where F has one edge less. Iterating the procedure we destroy all pairs of incident edges with one edge from F and one from I.

Recall that F has no even cycle any more, since it would be an unbalanced even tour. Any two intersecting, i.e., not vertex-disjoint odd cycles can be connected to an unbalanced even tour in an obvious way. (In fact, since the two cycles are not identical, some edge needs to be traversed only once and is therefore unbalanced.) It follows that the odd cycles in F are vertex-disjoint. Any two disjoint odd cycles connected by a path (with all edges in F) can also be connected to an unbalanced even tour in an obvious way. It follows that every connected component of F has at most one odd cycle, and no even cycles. Furthermore, F cannot have an edge ending in a vertex v of degree 1 in F, since then v must be incident to some edge of I, which was already ruled out. It follows that every connected component of F is merely a blossom.

Now we give all edges in F the new weight $\frac{1}{2}$, whatever their old weights have been. This still yields a valid solution, since every vertex in the graph remains incident to edges of total weight at least 1. Furthermore, this change does not increase the total weight of the set cover, by the following argument. Consider any cycle in F, of length k. Edges of F are incident to further edges of Z only. Thus, vertices in the cycle can be covered using the weights of edges in the cycle only. Therefore, in any fractional set cover, each pair of incident edges in the cycle

must have weights with sum at least 1. Summing up these k constraints, where every edge is counted twice, we see that any fractional edge cover has total weight at least $\frac{1}{2}k$ in the cycle. Hence the old fractional set cover had a weight no better than the new one.

Let $uv \in Z$ be an edge between two leaves u and v of stars. Further, let c_u and c_v, respectively, be the centers of these stars. If $c_u = c_v =: c$, then we can give the edges uv, uc, vc the weight $\frac{1}{2}$ to obtain a fractional set cover with smaller weight, contradicting optimality. If $c_u \neq c_v$, then both centers are also incident to other edges of I. Hence we can give uc and vc the weight 0 and uv the weight 1, leading to the same contradiction.

Let P be an alternating path connecting a leaf u of a star with center c with a blossom. Then P necessarily starts and ends with an edge of Z, and edges from I and Z alternate. In particular, P is an odd path. Now we change all weights on P from 0 to 1 and vice versa. Since c is incident to some edge of I other than uc, we can also reduce x_{uc} to 0. This preserves the total weight, and the blossom from F becomes incident to an edge of I, which was already ruled out.

Let P be an alternating path connecting two blossoms. The detailed description of P is as above. We can combine the two blossoms and P to an an unbalanced even tour, and alternatingly add and subtract $\frac{1}{2}$ in such a way that, on P, weights are changed from 0 to 1 and vice versa. Again this reduces the number of edges in F.

Finally all listed properties are established. □

6 Application to Randomized Testing in Graphs

Now we make the connection between fractional set covers and searching in graphs G. First we prove a lower bound on $t^1_{rand}(G)$. Its strength compared to the general $\frac{1}{2}\alpha$ bound in Proposition 4 is the additional term for blossoms of fractional edges.

Lemma 15. *Let $G = (V, E)$ be a graph. Let β be the number of blossoms in some optimal fractional set cover according to Theorem 14. Then we have*

$$t^1_{rand}(G) \geq \frac{\alpha}{2} + \frac{\beta^2}{8\alpha} + \frac{1}{2}.$$

Proof. We take a half-integral set cover which has total weight α and the properties stated in Theorem 14. We define a target distribution τ as follows. Every vertex v in a blossom in F and every vertex v in an isolated edge in I (that is, not belonging to a star) gets $\tau(v) := \frac{1}{2\alpha}$. Every leaf v of a star in I gets $\tau(v) := \frac{1}{\alpha}$. Every center c of a star in I gets $\tau(c) := 0$. Indeed, these probabilities sum up to 1.

Let H be the subgraph of G induced by the vertices v with $\tau(v) = \frac{1}{2\alpha}$, and let h be the number of its vertices. A maximum matching in H has exactly $\frac{1}{2}(h - \beta)$ edges, such that exactly β vertices are orphaned (that is, not belonging to a matching edge). Namely, existence of such a matching is obvious, and a matching with fewer orphaned vertices would require an alternating path of edges from I and Z joining two blossoms, which does not exist due to (vi).

We derive our lower bound by Yao's minimax principle. Remember that any deterministic strategy for Target Presence is specified by the sequence of edges to be tested until success. Due to (v) and (vi), the two vertices of every edge in the graph have target probabilities of at most $\frac{1}{\alpha}$ together. (In particular, no edge joins a blossom and a leaf.) Due to the maximum matching size observed above, at least β vertices v with $\tau(v) = \frac{1}{2\alpha}$ are orphaned, such that v appears in a tested edge containing no other, yet untested vertex u with $\tau(u) = \frac{1}{2\alpha}$. (In other words, the edge actually tests only v, while u has already been excluded as the target.)

Altogether, in the best case, a deterministic strategy against τ can first test $\alpha - \frac{1}{2}\beta$ edges, each covering two new vertices with total probability mass $\frac{1}{\alpha}$, followed by β edges, each covering

only one new vertex v with $\tau(v) = \frac{1}{2\alpha}$. The expected number of tests evaluates to:

$$\frac{1}{\alpha}\sum_{i=1}^{\alpha-\frac{1}{2}\beta} i + \frac{1}{2\alpha}\sum_{j=1}^{\beta}\left(j+\alpha-\frac{1}{2}\beta\right)$$

$$\frac{1}{2\alpha}\left(\alpha-\frac{\beta}{2}\right)\left(\alpha-\frac{\beta}{2}+1\right) + \frac{1}{4\alpha}\beta(\beta+1) + \frac{\beta}{2\alpha}\left(\alpha-\frac{\beta}{2}\right)$$

$$= \frac{1}{2\alpha}\left(\alpha^2-\alpha\beta+\frac{\beta^2}{4}+\alpha-\frac{\beta}{2}\right) + \frac{\beta^2}{4\alpha} + \frac{\beta}{4\alpha} + \frac{\beta}{2} - \frac{\beta^2}{4\alpha}$$

$$= \frac{\alpha}{2} + \frac{\beta^2}{8\alpha} + \frac{1}{2}$$

□

Next we propose a method to tackle Target Presence in graphs, based on this structure, and derive some algorithmic results for it.

Given a graph $G = (V, E)$, we first compute a fractional set cover as in Theorem 14. We partition V into subsets that we call *circles*. (This resembles "cycles" but is not the same.) The 0th circle consists of all vertices in the edges of I. For every $i > 0$, the ith circle consists of all vertices in blossoms of exactly $2i + 1$ edges in F. Due to (iii) and (v) in Theorem 14, this is in fact a partitioning. For $i > 0$, let β_i denote the number of blossoms in the ith circle, and let α_i denote the total weight of the edges between vertices in the ith circle. Obviously, $\alpha_i = (i + \frac{1}{2})\beta_i$. Within every non-empty circle we proceed as follows.

Routine for the ith circle. If $i = 0$, we simply test all edges of I in random order. If $i > 0$, we take every blossom of F-edges and randomly choose one of its $2i + 1$ vertices as *left-over* vertex. The other $2i$ vertices of the blossom are covered by i edges of F in a unique way. We call them *full* edges. We test all $i\beta_i$ full edges from all β_i blossoms in random order, followed by any β_i edges that cover the left-over vertices, they are tested in random order, too.

By convention, all edges of I (that is, in the 0th circle) are full edges. Note that, if the target is in the ith circle with $i > 0$, it becomes a left-over vertex with probability $\frac{1}{2i+1}$.

Let us consider the special case of a pure ith circle, that is, a graph where only the ith circle is non-empty. Straightforward calculation shows that the expected test number of a pure ith circle matches the lower bound in Lemma 15. Thus our Routine is already an optimal strategy in this special case. This is also true for $i = 0$, and we use the latter fact in the following result:

Theorem 16. *We can compute $t^1_{rand}(G)$ and thus solve* Target Presence *in polynomial time, if the graph G is bipartite or has a perfect matching.*

Proof. The algorithm of Theorem 14 applied to a bipartite graph G yields $\beta = 0$ simply because G has no odd cycles. Hence only the 0th circle exists, and trivially, our Routine is then optimal. As for the second case, a perfect matching can be computed in polynomial time, and this is a set cover with $\alpha = \frac{1}{2}n$ and $\beta = 0$ which also satisfies the properties in Theorem 14. Hence the lower bound $\frac{1}{2}\alpha + \frac{1}{2}$ from Lemma 15 is valid. Testing the matching edges in random order achieves this expected test number. □

For more general graphs G we conjecture that the randomized schedules of the edges from the different circles, obtained by the Routine, can be aligned to an overall randomized schedule that is optimal. In more detail:

Let us work with a schedule that reserves $\alpha + \frac{1}{2}\beta$ positions for the edges to be tested in all circles. Consider any partitioning of the schedule that specifies which positions are allocated to

which circle. (The number of positions needed for the ith circle is uniquely determined by β_i.) Within every circle we use the Routine, where all full edges are tested prior to all randomly chosen left-over vertices, both in random order. For each circle we can easily calculate the expected position where the search succeeds, conditional on presence of the target in that circle. Clearly, we want to minimize the maximum of these expected positions. To this end we modify the partitioning by exchanging carefully chosen positions between circles, in such a way that the largest expectations decrease and smaller ones increase. Finally we may even mix different partitionings, that is, choose them with different probabilities. Intuitively, it should be possible to equalize the expected positions for all circles.

We illustrate the approach by the smallest graph that is not handled by Proposition 16. This example is the graph $G = (V, E)$ with $V = \{p, q, r, s, t\}$ and $E = \{pq, qr, rp, ps, st\}$, that is, a triangle with a path of two edges attached. In an optimal fractional set cover, the triangle is a blossom and st is a single edge in I. Note that $\alpha = 2.5$ and $\beta = 1$. The lower bound from Lemma 15 is 1.8. Our schedule of tested edges has three positions: one for the 0th circle and two for the 1st circle. If we place the 0th circle in the middle, symbolically $[1, 0, 1]$, then the worst case is to have the target in the 0th circle, such that 2 tests are needed. The 1st circle has a smaller expectation, since the target is at the last position only with probability $\frac{1}{3}$. Therefore let us also use the partitioning $[0, 1, 1]$ where the blossom is scheduled last. It is easy to calculate a mixture that equalizes the worst cases: We use $[1, 0, 1]$ with probability 0.8 and $[0, 1, 1]$ with probability 0.2. Now, if the target is in the 0th circle, the expected test number is $0.8 \cdot 2 + 0.2 \cdot 1 = 1.8$. If the target is in the 1st circle, the expected test number in $[1, 0, 1]$ is $\frac{2}{3} \cdot 1 + \frac{1}{3} \cdot 3 = \frac{5}{3}$, and the expected test number in $[0, 1, 1]$ is $\frac{2}{3} \cdot 2 + \frac{1}{3} \cdot 3 = \frac{7}{3}$. In total, the expected test number is $0.8 \cdot \frac{5}{3} + 0.2 \cdot \frac{7}{3} = 1.8$ again. Hence this strategy is optimal for the graph.

As in this example we can get provably optimal "ad-hoc" strategies also for other graphs. This supports the conjecture that the fractional set cover approach may lead to a polynomial-time algorithm for Target Presence in larger classes of graphs, or even all graphs. However, the challenge is to devise an efficient general equalizing procedure according to the idea. We must leave it open whether the problem is polynomial-time solvable or NP-complete in general graphs.

7 Further Research

The most intriguing open problems are the exact complexity of our randomized search problem in graphs, and the construction of approximation algorithms also for hypergraphs of any fixed rank.

The number β of blossoms played a central role in the graph case. Does β depend on arbitrary decisions of the algorithm in Theorem 14, or is β a graph invariant? If the latter is true, then β can be computed in polynomial time, and we can further ask: How are, in particular, the graphs with $\beta = 0$ characterized? (Note that Target Presence is polynomial for them.)

We have studied adaptive strategies only, but one can also consider searching in a prescribed number of rounds of parallel tests. Trivially, randomization is useless if only one round is permitted, but what about two rounds, etc.? A few results on combinatorial search in rounds are known in the deterministic setting only.

Finally, Target Search has a natural generalization: There may be a predefined partitioning of the set U of elements, and instead of identifying the target exactly, one may only want to find the partition containing the target. An example is the "order statistics" problem that asks for the kth smallest element in a set S of numbers. As in the sorting problem, the elements are the permutations of S, the tests are pairwise comparisons, but all permutations having the same number at position k form one partition.

Acknowledgment

The author thanks the referees for their constructive comments that helped improve the presentation.

References

[1] M. Basavaraju, M.C. Francis, M.S. Ramanujan, S., Saurabh, Partially polynomial kernels for set cover and test cover, in: A. Seth, N.K. Vishnoi (Eds.), FSTTCS 2013, LIPIcs, vol. 24, 2013, pp. 67–78.

[2] K.M.J. de Bontridder, B.V. Halldórsson, M.M. Halldórsson, C.A.J. Hurkens, J.K. Lenstra, R. Ravi, L. Stougie, Approximation algorithms for the test cover problem, Math. Progr. Series B 98 (2003) 477–491.

[3] K.M.J. de Bontridder, B.J. Lageweg, J.K. Lenstra, J.B. Orlin, L. Stougie, Branch-and-bound algorithms for the test cover problem, in: L. Epstein, P. Ferragina (Eds.), ESA 2002, LNCS, vol. 2461, Springer, 2002, pp. 223–233.

[4] Y. Cheng, J. Guo, F. Zheng, A new randomized algorithm for group testing with unknown number of defective items. J. Comb. Optim. 30 (2015) 150–159.

[5] M. Cheraghchi, A. Hormati, A. Karbasi, M. Vetterli, Group testing with probabilistic tests: Theory, design and application, IEEE Trans. Info. Theory (57 (2011), 7057–7067.

[6] V. Chvatal, A greedy heuristic for the set-covering problem, Math. Oper. Res. 4 (1979) 233–235.

[7] R. Crowston, G. Gutin, M. Jones, G. Muciaccia, A. Yeo, Parameterizations of test cover with bounded test sizes, Algorithmica 74 (2016) 367–384.

[8] R. Crowston, G. Gutin, M. Jones, S. Saurabh, A. Yeo, Parameterized study of the test cover problem, in: B. Rovan, V. Sassone, P. Widmayer (Eds.), MFCS 2012, LNCS, vol. 7464, Springer, 2012, pp. 283–295.

[9] P. Cui, A tighter analysis of set cover greedy algorithm for test set, in: B. Chen, M. Paterson, G. Zhang (Eds.), ESCAPE 2007, LNCS, vol. 4614, Springer, 2007, pp. 24–35.

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

[11] J. Edmonds, Paths, trees, and flowers, Canad. J. Math. 17 (1965) 449–467.

[12] T. Fahle, K. Tiemann, A faster branch-and-bound algorithm for the test-cover problem based on set-covering techniques, ACM J. Experim. Algor. 11 (2006)

[13] H. Fernau, D. Raible, A parameterized perspective on packing paths of length two, in: B. Yang, D.Z. Du, C.A. Wang (Eds.), COCOA 2008, LNCS, vol. 5165, Springer, 2008, pp. 54–63.

[14] G. Gutin, G. Muciaccia, A. Yeo, (Non-)existence of polynomial kernels for the test cover problem. Info. Proc. Letters 113 (2013) 123–126.

[15] D.S. Hochbaum, Solving integer programs over monotone inequalities in three variables: A framework for half integrality and good approximations, Eur. J. Oper. Res. 140 (2002) 291–321.

[16] M. Mázard, C. Toninelli, Group testing with random pools: Optimal two-stage algorithms, IEEE Trans. Info. Theory 57 (2011) 1736-1745.

[17] E.R. Scheinerman, D.H. Ullman, Fractional Graph Theory: A Rational Approach to the Theory of Graphs, Wiley Interscience, 1997.

[18] L. Trevisan, Non-approximability results for optimization problems on bounded degree instances, in: J.S. Vitter, P.G. Spirakis, M. Yannakakis (Eds.), STOC 2001, ACM, pp. 453–461.

[19] V.V. Vazirani, Approximation Algorithms, Springer, 2001.

[20] G. Wiener, Rounds in combinatorial search, Algorithmica 67 (2013) 315–323.

Appendix

On the practical side it may be valuable to know a good approximation algorithm for Target Presence (and thus also Target Search) in graphs. Since the complexity status is open, we add this preliminary result only as an appendix. The ratio is slightly better than 7/6 in the conference version.

Proposition 17. *In graphs G we can approximate $t^1_{rand}(G)$ within a factor at most $\frac{15}{13}$ in polynomial time.*

Proof. For large β (the cut-off point is specified below) we simply merge all circles as follows. First we test the full edges of all circles in random order, and then we test the left-over vertices of all circles in random order. (More precisely, we test further edges, each covering one left-over vertex.) Since there exist β left-over vertices, we have exactly $\alpha - \frac{1}{2}\beta$ full edges; this number is always integer. In the worst case, the target is in the 1st circle, thus it becomes a left-over vertex with probability $\frac{1}{3}$. Then the expected number of tests (neglecting constant summands) is at most

$$\frac{2}{3} \cdot \frac{1}{2} \left(\alpha - \frac{1}{2}\beta \right) + \frac{1}{3} \left(\alpha - \frac{1}{2}\beta + \frac{1}{2}\beta \right) = \frac{2}{3}\alpha - \frac{1}{6}\beta.$$

For small β we even merge both types of edges as follows: all full edges and left-over vertices are tested together in random order. Since their total number is $\alpha + \frac{1}{2}\beta$, trivially, the expected number of tests (neglecting constant summands again) is

$$\frac{1}{2}\alpha + \frac{1}{4}\beta.$$

The two expression are equal if $\beta = \frac{2}{5}\alpha$. Division by the lower bound from Lemma 15 yields the approximation ratio $\frac{15}{13}$. In fact, this is the worst-case ratio: The first upper bound decreases with β, hence the approximation ratio decreases with $\beta > \frac{2}{5}\alpha$. For the second upper bound it can be verified directly that the approximation ratio increases with $\beta < \frac{2}{5}\alpha$. □