Theoretical Computer Science ••• (••••) •••–•••

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

Deterministic versus randomized adaptive test cover ☆

Peter Damaschke *

Department of Computer Science and Engineering, Chalmers University, 41296 Göteborg, Sweden

A R T I C L E   I N F O

Article history:
Received 16 May 2016
Received in revised form 8 September 2016
Accepted 21 September 2016
Available online xxxx
Communicated by V.Th. Paschos

Keywords:
Combinatorial search
Randomization
Game theory
LP duality
Set cover
Fractional graph theory

A B S T R A C T

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.

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 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*.

☆ An early version appeared in the Proceedings of the *9th International Conference on Algorithms and Complexity CIAC 2015*, Paris, *Lecture Notes in Computer Science*, vol. 9079, Springer, 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.

* Fax: +46 31 772 3663.
E-mail address: ptr@chalmers.se.

http://dx.doi.org/10.1016/j.tcs.2016.09.019
0304-3975/© 2016 Elsevier B.V. All rights reserved.

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).

Please cite this article in press as: P. Damaschke, Deterministic versus randomized adaptive test cover, Theoret. Comput. Sci. (2016), http://dx.doi.org/10.1016/j.tcs.2016.09.019