Searching for an unknown edge in the graph and its tight complexity bounds

Abbas Cheraghi
Mathematics & Statistics Department,
Khansar, University of Isfahan,
Isfahan, Iran
a.cheraghi@khn.ui.ac.ir

Abstract—Key distribution patterns have been an active area of study in the last 20 years. Cover free family and disjunct system can be used to construct key distribution patterns and group testing algorithms, respectively; therefore they are dual incidence structures. In this paper, we present a simple proof of the tight bound on the number of tests in group testing problem for searching an edge in graphs. Our explicit new bound based on combinatorial object can be applied here as well, yielding key distribution patterns in which the minimum number of keys is a polynomial in the logarithm of the number of participants. Finally we show that our lower bound for the complexity is tight.

Keywords-component: Key distribution, group testing, Combinatorial search.

I. INTRODUCTION

In practically every field of human activity we encounter problems which require that we search for an unknown object. This may be a search in the literal sense of the word, e.g. the search for a faulty part in a mechanical device which caused it to malfunction, or for the exact place in a data file where a certain piece of information is stored. It may also be a search of a vaguer manner like the search for the meaning of a cryptically message or for the basic principles of a complex system. Most of these problems have one thing in common: The aim is to find the unknown object in as short a time or with as little cost as possible.

Graph searching has been subject to extensive study [2,9,10,11,16] and it fits into the broader class of pursuit evasion, search, rendezvous problems on which hundreds of papers have been written (see e.g., the book [2]). The problem was introduced by Parsons [12] and by Petrov [13] independently, and the original definition corresponds exactly to what we today call edge searching. In this setting, a team of searchers is trying to catch a fugitive moving along the edges of a graph. The fugitive is very fast and knows the moves of the searchers, whereas the searchers cannot see the fugitive until they capture him, i.e., when the fugitive is trapped and has nowhere to run. An edge is cleared by sliding a searcher from one endpoint to the other endpoint, and a vertex is cleared when a searcher is placed on it. The problem is to find the minimum number of searchers that can guarantee the capture of the fugitive, which is called the edge search number of the graph.

The edge searching problem on graphs is an extension of the classical group testing problem. Assume that we are given a graph \( G = (V, E) \) with vertex set \( V \) and edge set \( E \). Let \( G_S \) denote the subgraph of \( G \) induced by the set \( S \) of vertices. Our task is to identify a subset \( D \subseteq E \) of defective edges with a minimum number of edge tests, where an edge test takes an arbitrary subset \( S \subseteq V \) and asks whether the subgraph \( G_S \) contains a defective edge.

Chang and Hwang [5] first cast the problem of identifying two defective vertices in a complete bipartite graph. This problem can be treated as a special group testing problem of searching for a single edge on graphs. Aigner [1] was the first who consciously introduced the edge testing problem for a general graph, thus bringing the “graph” into focus. In this paper, we present a simple proof of the tight bound on the number of tests in group testing problem for searching an edge in graphs.

II. RELATED TOPICS

In this section, we study two topics: key distribution patterns (which are of interest in cryptography) and nonadaptive group testing algorithms.

Frameproof codes, as a set system, were introduced by Boneh and Shaw [4] as a method of “digital fingerprinting” which prevents a coalition of a specified size \( c \) from framing a user not in the coalition. Stinson and Wei [15] then gave a combinatorial formulation of the problem in terms of certain types of extremal set systems. In [14] \((i, j)\)-Cover-free family and \((i, j)\)-disjunct system are used to construct \((i, j)\)-key distribution pattern and a non adaptive group testing algorithm respectively, therefore they are dual incidence structures. We first define some terminology concerning set systems.

Definition 2.1. A set system is a pair \((X, \beta)\) where \(X\) is a set of elements called points, and \(\beta\) is a set of subsets of \(X\), the members of which are called blocks.
Definition 2.2. A set system \((X, \beta)\) is an \((i, j)\)-cover free family provided that, for any two disjoint subsets \(C_i, C_j\) of \(\beta\), where \(|C_i| \leq i\) and \(|C_j| \leq j\), it holds that \(\bigcap_{B \in C_i} B \not\subset \bigcup_{B \in C_j} B\).

An \((i, j)\)-cover-free family, \((X, \beta)\), will be denoted as an \((i, j)\)-CFF \((v, b)\) if \(|X| = v\) and \(|B| = b\).

Definition 2.3. Let \(v\) and \(b\) be positive integers. An \((i, j)\)-key distribution pattern is a method of distributing a set of \(v\) keys to a set of \(b\) users, such that any subset of \(i\) users can form a conference keys by combining the keys that they hold in common. Any conference key thus formed should be secure against a (disjoint) coalition of size at most \(j\). The key distribution pattern will be denoted as an \((i, j)\)-KDP \((v, b)\).

We now describe how key distribution patterns are constructed from cover free families.

Suppose that \((X, \beta)\) is an \((i, j)\)-CFF \((v, b)\), where \(i \geq 2\). For each \(x \in X\), let \(k_x\) be a key, chosen at random from a specified abelian group. Suppose we have a set of \(b\) participants, denoted \(u_b (B \in \beta)\), and each participant \(u_b\) is given the keys \(k_x (x \in B)\). Let \(C\) be a subset of \(i\) participants. Then, for any coalition \(D\) of size at most \(j\) that is disjoint from \(C\), there exists a key that is held by every member of \(C\) and by no member of \(D\). Now, suppose the conference key \(k_C\) is defined to be

\[ k_C = \sum_{x \in B \text{ for all } B \in C} k_x. \]

Then every member of \(C\) can compute the conference key \(k_C\), but the value of \(k_C\) cannot be computed by any coalition \(D\) of size at most \(j\).

Definition 2.4. A set system \((X, \beta)\) is an \((i, j)\)-disjunct system if, for any \(P, Q \subseteq X\) such that \(|P| \leq i\), \(|Q| \leq j\) and \(P \cap Q = \emptyset\), there exists a \(B \in \beta\) such that \(P \subseteq B\) and \(Q \setminus B = \emptyset\). An \((i, j)\)-disjunct system, \((X, \beta)\), will be denoted as an \((i, j)\)-DS \((v, b)\) if \(|X| = v\) and \(|B| = b\).

Cover free families and disjunct systems are dual incidence structures. We informally define nonadaptive group testing algorithms.

Definition 2.5. Suppose that \(X\) is a set of \(v\) samples that are to be tested positive or negative. Suppose that \(\beta\) is a set of subsets of \(X\) where each \(B \in \beta\) represents a subset of samples (called a group) that are to be combined and tested together. The testing procedure has the property that if a group contains at least one positive sample, then the test result for that group is positive. Suppose that the testing procedure allows the identification of the positive samples if the number positive samples are at most \(d\). Then the resulting scheme is called a nonadaptive group testing algorithm and is denoted by \(d\)-NAGTA \((v, b)\). (The term “nonadaptive” means that the tests performed are fixed ahead of time, and do not depend on the outcome of earlier tests. This is useful in practice due to simplicity, as well as the fact that a nonadaptive algorithm can be parallelized to any desired degree.)

Disjunct systems can be used to construct group testing algorithms. Suppose that \((X, \beta)\) is a \((i, d)\)-DS \((v, b)\), where \(X\) is the set of \(v\) samples and \(\beta\) is the set of \(b\) groups. It is clear that the samples that occur in no group that tests positive are in fact the negative samples. Thus, the positive samples are identified by this testing procedure, and we have a \(d\)-NAGTA \((v, b)\).

III. Tight Lower Bound on \(c(G)\)

Consider the following combinatorial sequential group testing problem for \(2\) defective items; assume that a set \(V\) of \(n\) items contains exactly \(2\) defective items. We want to interpret the search domain \(V\) as the vertex set of the graph \(G\) and search for two defective elements from \(V\), i.e., an unknown edge \(e^*\) in the edge set \(E\) of \(G\). The aim is to determine the unknown defective edge by sequentially choosing subsets \(U \subseteq V\) of \(V\) and asking questions of the form “Is at least one of the vertices of \(U\) an endpoint of \(e^*\)?”. Such questions are called tests and \(U\) is called a test set. Let \(c(A, G)\) denote the worst case number of tests required by an algorithm \(A\) to identify \(e^*\). The aim is to determine \(c(G) = \min_{A} c(A, G)\). The interested reader can find more details about group testing problems in [3,6,7,8]. We begin with a simple but useful result.

Lemma 3.1. Let \(H\) be a subgraph of \(G\). Then \(c(H) \geq c(G)\).

Proof. All of the tests and tests set in \(H\) are also in the \(G\) and so the minimal tests set on \(H\) is less or equal than the minimum tests set in \(G\).

Let us make a first observation on this problem. Suppose in the course of our algorithm we have arrived at a graph \(G(V, E)\) in which the unknown edge \(e^*\) lies. The next test \(A \subseteq V\) splits \(G\) into two parts \(G_0\) and \(G_1\) (Fig. 1). If the answer is “yes”, then \(e^*\) lies in \(G_1\) or in \(G_{A \cup V\setminus A}\) whereas if the answer is “no”, \(e^*\) lies in \(G_{V\setminus A}\). Thus \(G_1 = G_{1A} \cup G_{A \cup V\setminus A}\) and \(G_0 = G_{V\setminus A}\) in fact \(G_0\) is an induced subgraph of \(G\).

We conclude that any test corresponds to a partition of \(G\) into an induced subgraph and the remainder. It is convenient to interchange the roles of “yes” and “no” which obviously has no influence on \(c(G)\). That is, after any test \(A \subseteq V\) we receive as answer \(e^* \in G_1 = G_{A} \text{ or } e^* \in G_0 = G_{A \cup V\setminus A} \cup G_{V\setminus A}\). From now on, we will always consider this latter version of our problem. There is a famous information theoretic bound on \(c(G)\).

Proposition 3.2. Let \(G = (V, E)\) be a graph. The information-theoretic bound on \(c(G)\) is \(c(G) \geq \lceil \log_2 |E| \rceil\).

A proof can be found for example in [1].
In this paper, we present a simple proof of the tight bound on \( c(G) \) in group testing problem for searching an edge in graphs that is better than the famous information theoretic bound on \( c(G) \).

Let us first discuss some necessary results that we need for the proof of our new bound.

**Theorem 3.3.** Let \( G= (V,E) \) be a graph with vertex set \( V \) and edge set \( E \) and \( |E|/|V| > 2^k \) for \( k \geq 4 \), then \( c(G) \geq k+1 \).

**Proof.** By induction on \( k \). For \( k = 4 \), the only interesting case is \( |V| = 6 \). M. Aigner in [1] prove that \( c(K_6) = 5 \), thus \( c(G) \geq k+1 \). Let \( A \) be the first test set, so \( G \) splits into two parts induced subgraph \( G_A \) and the remainder \( G_0 = G \setminus A \). If \( |E(G_A)| + |V(G_A)| > 2^{k-1} \), so according to the induction \( c(G_A) \geq k \) because of the first question we have \( c(G) \geq k+1 \).

Otherwise, if \( |E(G_A)| + |V(G_A)| \leq 2^{k-1} \) by the definition of \( G_A \) and \( G_0 \) we have \( |E(G_0)| + |V(G_0)| > 2^{k-1} \) and again follows by induction applied to \( G_0 \) we have \( c(G_0) \geq 2k \). Then because of the first question “A”, we have \( c(G) \geq k+1 \).

**Theorem 3.4.** Let \( G \) be complete graph with \( n \) vertices. For every \( \binom{n+1}{2} > 2^k \) with \( k > 3 \) we have \( c(K_n) \geq k+1 \).

**Proof.** Suppose \( K_n \) be a complete graph with \( n \) vertices, thus
\[
|E(K_n)| + |V(K_n)| = \binom{n}{2} + n = \binom{n+1}{2}.
\]

Therefore by assumption, the assertion is trivial.

The preceding proposition shows the next result.

**Lemma 3.5.** \( c(K_n) \geq \left\lceil \log_2 \binom{n+1}{2} \right\rceil \) for all \( n \) with
\[
\binom{n}{2} < 2^k < \binom{n+1}{2}, \quad k \geq 4.
\]

Our explicit bound based on combinatorial object and related construction of key distribution and group testing pattern based on cover free family and its duality, can be applied here as well, yielding key distribution patterns in which the minimum number of keys is a polynomial in the logarithm of the number of participants.

**Theorem 3.6.** If \( G=(V,E) \) be a graph with vertex set \( V \) and edge set \( E \). Let \( k \) is an integer with \( 2^{k+1} \geq |E|/|V|^2 > 2^k \), then the lower bound on \( C(G) \) is \( c(G) \geq \left\lceil \log_2 |E|/|V|^2 \right\rceil \).

**Proof.** \( \left\lceil \log_2 |E|/|V|^2 \right\rceil = k+1 \), so the assertion is trivial.

The next example indicates that the lower bound presented in theorem 3.6. is sharp.

**Example 3.7.** Let \( G \) be a complete graph with \( n=2^l \), \( l \geq 3 \) vertices. We have
\[
\binom{n}{2} = \frac{2^l(2^l-1)}{2} = 2^{2l-1} - 2^{l-1} < 2^{2l-1}
\]
and
\[
\binom{n+1}{2} = \frac{2^l(2^l+1)}{2} = 2^{2l-1} + 2^{l-1} > 2^{2l-1},
\]
whence
\[
\binom{n+1}{2} < 2^{2l-1} < \frac{2^l(2^l+1)}{2}, \quad l \geq 3.
\]
Our result implies therefore
\[
c(K_n) \geq \left\lceil \log_2 \frac{2^l(2^l+1)}{2} \right\rceil = 2l.
\]

On the other hand, let us show \( c(K_n) \leq 2l \) for \( n=2^l \) and all \( l \geq 1 \). For \( l=1 \) this is obvious. Now we use induction on \( l \). We split the vertex set of \( K_n \) into two equal sized parts \( A \) and \( B \) with \( |A| = |B| = 2^{l-1} \). As first test set we take 1 and as second set we take 2. After these two tests we know that the unknown edge \( e^* \) lies in \( G_A = K_{2^{l-1}} \) or in \( G_B = K_{2^{l-1}} \) or in \( G_{AB} = K_{2^{l-2}} \). For the first two possibilities we have by induction \( c(G_A) \leq 2l-2 \), \( c(G_B) \leq 2l-2 \), and thus \( c(K_n) \leq 2l \). If \( e^* \) is in \( G_{AB} \) then we know that one end vertex \( u \) is in \( A \) while the other end vertex \( v \) is in \( B \). By the usual halving method, we can identify \( u \) with \( l-1 \) tests on \( A \), and similarly for \( v \). Thus we again obtain \( c(G_{AB}) \leq 2l-2 \), i.e. \( c(K_n) \leq 2l \). We have thus proved \( c(K_n) = 2l \) for \( l \geq 3 \).

**References**


