Available online at www.sciencedirect.com
SCIENCE DIRECT
Discrete Applied Mathematics 148 (2005) 273–277
DISCRETE APPLIED MATHEMATICS
www.elsevier.com/locate/dam

Note

A competitive algorithm to find all defective edges in a graph

Frank K. Hwang

Department of Applied Mathematics, National Chiao-Tung University, Hsin-Chu, Taiwan, Republic of China

Received 23 July 2003; received in revised form 25 November 2004; accepted 7 February 2005
Available online 27 April 2005

Abstract

Consider a graph $G(V, E)$ where a subset $D \in E$ is called the set of defective edges. The problem is to identify D with a small number of edge tests, where an edge test takes an arbitrary subset S and asks whether the subgraph $G(S)$ induced by S intersects D (contains a defective edge).

Recently, Johann gave an algorithm to find all d defective edges in a graph assuming $d = |D|$ is known. We give an algorithm with d unknown which requires at most $d(\lceil \log_2 |E| \rceil + 4) + 1$ tests. The information-theoretic bound, knowing d, is about $d \log_2(|E|/d)$. For d fixed, our algorithm is competitive with coefficient 1.

Keywords: Group testing; Graph testing; Competitive algorithm

1. Introduction

The edge-test problem, sometimes called group testing on graphs, is an extension of the classical group-testing problem that seeks to identify a subset D of defective vertices among a given set V by taking an arbitrary subset S of V and asking whether S intersects D. Chang and Hwang [2] considered the problem of identifying two defective vertices, one in an m-set and the other in a disjoint n-set. Construct a complete bipartite graph with the m-set and the n-set as the two parts, then the two defective vertices can be represented by an edge connecting them. Asking whether $G(S)$ contains a defective vertex is the same as asking whether the complementary graph of $G(S)$ contains a defective edge. Thus the problem studied in [2] can be treated as the first group-testing problem on graphs.

E-mail address: fhwang@math.nctu.edu.tw.

0166-218X/$ - see front matter
doi:10.1016/j.dam.2005.02.010

Aigner [1] was the first one who consciously introduced the edge-testing problem by studying a general graph and thus bringing the “graph” into focus. Note that

$$\log_2 \binom{|E|}{d} \sim d \log_2 \frac{|E|}{d}$$

is the information-theoretic lower bound of finding the d defective edges. Let $M(G, d)$ denote the minimum number of (edge) tests guaranteed to identify the d defective edges in $G(V, E)$. Aigner [6] conjectured

$$M(G, 1) = \lceil \log_2 |E| \rceil + c,$$

where c is a constant.

Damaschke [3] proved

$$M(G, 1) \leqslant \lceil \log_2 |E| \rceil + 1$$

and showed that this result is sharp for general G. Triesch [6] generalized the result to hypergraphs (with rank r) by proving

$$M(G, 1) \leqslant \lceil \log_2 |E| \rceil + r - 1.$$

Recently, Johann [5] made a breakthrough by proving

$$M(G, d) \leqslant d \left(\left\lceil \log_2 \frac{|E|}{d} \right\rceil + 7 \right),$$

proving a conjecture of Du and Hwang [4] that

$$M(G, d) = d \left(\left\lceil \log_2 \frac{|E|}{d} \right\rceil + c \right).$$

This proof is ingenious but slightly complicated.

All the above results assume that d is known. This assumption somewhat restricts their applicability. In this paper, we discard this assumption and show that for all d, our algorithm needs at most $d(\lceil \log_2 E \rceil + 4) + 1$ tests. Our proof is simpler than Johann’s, hence could be more amenable to an extension to r-graphs.

2. The algorithm

The intricacy of the algorithm is to meet two seemingly contradicting goals: one to identify all defective edges and the other not to keep repeatedly identifying the same defective edges (thus wasting tests). This can be accomplished by removing a defective edge once identified. However, unlike the vertex-testing model where a defective vertex can be simply removed, an edge in the edge-testing model can be removed only by removing its two end vertices, which are also end vertices of other edges. Thus, uncoordinated removal of vertices of a defective edge is not allowed. The correct strategy is to create the right environment and timing under which removals are allowed.

Our algorithm is much like Johann's, except a bit simpler. The algorithm also consists of a partition stage and a search stage. In the partition stage V is partitioned into $V_1, V_2, \ldots$ such that no V_i contains a defective edge. Some defective edges are identified along the way with its two vertices assigned to different V_i and V_j. In the search stage, all remaining defective edges are to be identified. Since such a defective edge must have its two vertices in different V_i and V_j, we have to conduct tests of the type $A \cup B$ with $A \subseteq V_i$ and $B \subseteq V_j$. But then $A \cup B$ may contain an identified defective edge. We adopt two rules to prevent this from happening:

(i) Allow at most one of A and B to be nonsingleton.
(ii) Suppose $A = \{v\}$. Remove all $u \in B$ from B if (u, v) is an identified defective edge.

Note that u is only temporarily removed for this particular A, and is put back to B as soon as A changes.

We will now describe the details of the algorithm. First we introduce the halving procedure as a subroutine of the algorithm. For a set S of n elements, the halving procedure tests a subset S' of $\lceil \frac{n}{2} \rceil$ elements. If S' is positive, iterate the procedure on S'; if negative, iterate on $S \backslash S'$.

Johann commented that Triesch's procedure for $r = 2$, with a little modification, can be used to identify a single defective edge in G in $\lceil \log_2 |E| \rceil + 1$ tests even though G has many defective edges. Since this is important to us, we will present her idea in detail.

Construct a vertex cover of E by first taking a vertex v_1 with maximum degree, then a vertex v_2 of maximum degree after v_1 and all edges incident to it are deleted, and so on. Suppose the vertex-cover C contains c vertices. Then we test a subset $V \backslash \{v_1, \ldots, v_k\}$ for some $k < c$. If negative, we iterate the same procedure on $\{v_1, \ldots, v_k\}$. If positive, we test a smaller subset $V \backslash \{v_1, \ldots, v_{k'}\}$ with $k' > k$. Continue in this manner until finally we identify a v_i such that $V \backslash \{v_1, \ldots, v_{i-1}\}$ is positive but $V \backslash \{v_1, \ldots, v_i\}$ is negative. Hence V_i must be a vertex of a defective edge. Identify a defective edge $\{v_i, u\}$ with $u \in V \backslash \{v_1, \ldots, v_i\}$ by the halving procedure. We will refer to this procedure as the TJ procedure. Triesch and Johann proved that, by using the Kraft's inequalities, a binary tree which determines the values of $k, k', \ldots$ such that $\lceil \log_2 |E| \rceil + 1$ tests suffice can be constructed.

Algorithm

The partition stage:

Step 1: Set $V_1 = V,\ V_2 = \cdots = V_d = \phi,\ I = \phi$ (I is the set of identified defective edges).

Step 2: Test V_1. If positive, then

- Use the TJ procedure to identify a positive edge (v, u) where $v \in C$.
- Use the join subroutine to assign v to some V_i, $i > 1$.
- Set $V_1 = V_1 \backslash \{v\}$, $V_i = V_i \cup \{v\}$ and $I = I \cup \{(v, u)\}$. If $|V_1| \geqslant 2$, go back to step 2.

Step 3: If one of the V_j, $j > 1$, is nonempty, we enter the search stage.

Step 4: Stop with no defective edge identified.

The join subroutine:

Suppose v is the vertex to be assigned.

Step 1: Set $i = 2$.

Step 2:
- If $(v, u) \in I$ for some $u \in V_i$, set $V_i' = \{u \in V_i : (v, u) \notin I\}$.
- Test $v \cup V_i'$. If positive, use the halving procedure to identify a defective edge (v, u).
- Set $I = I \cup \{(v, u)\}$, $i = i + 1$ and go back to step 2.

Step 3: Add v to V_i.

The search stage:

Suppose the partition stage yields nonempty $V_1, \ldots, V_m$ for some $m \geqslant 2$.

Step 1: Set $j = 2$.

Step 2: For each vertex v in V_j, let $V(v) = \{u \in \bigcup_{i=1}^{j-1} V_i : (v, n) \in E \backslash I\}$. Test $v \cup V(v)$. If negative, go to the next v. If positive, use the halving procedure (with v attached to every test) to identify a defective edge (v, u). Set $V(v) = V(v) \backslash u$. If $V(v) \neq \phi$, go back to step 2. If $V(v) = \phi$, go to the next v

Step 3: Set $j = j + 1$. If $j \leqslant m$, go back to step 2.

Step 4: Stop.

Theorem. *The above is an algorithm which identifies all positive edges in at most* $d(\lceil \log_2 |E| \rceil + 4) + 1$ *tests.*

Proof. Each defective edge is identified by the TJ-procedure in $\lceil \log_2 |E| \rceil + 1$ tests, or the halving procedure in $\lceil \log_2 |E| \rceil$ tests. We also associate the positive test which initiates the TJ-procedure or the halving procedure to the identification procedure. Thus, the *d* defective edges cost a total of at most $d(\lceil \log_2 |E| \rceil + 2)$ tests.

Negative tests which occurred in the TJ procedure or the halving procedure are already counted in the $\lceil \log_2 |E| \rceil + 1$ tests. We count other negative tests. The partition stage stops with a negative test on V_1. Each join-subroutine ends with a negative test to assign v. Since each v to be assigned corresponds to a distinct defective edge, at most $d + 1$ negative tests occur at the partition stage.

Since each vertex in $\bigcup_{j=2}^{m} V_j$ represents a distinct defective edge, there are at most *d* of them. In the search stage, each such v starts a testing process which ends whenever a negative test occurs (not counting the negative tests in the halving procedure). Therefore, at most *d* negative tests occur. Thus, the total number of tests is at most $d(\lceil \log_2 |E| \rceil + 2) + d + 1 + d = d(\lceil \log_2 |E| \rceil + 4) + 1$. □

Acknowledgements

The author wishes to thank the reviewers for many constructive suggestions.

References

[1] M. Aigner, Search problem on graphs, Discrete Appl. Math. 14 (1986) 215–230.
[2] G.J. Chang, F.K. Hwang, A graph testing problem, SIAM J. Algebraic Discrete Methods 1 (1980) 21–24.

[3] P. Damaschke, A tight upper bound for group testing in graphs, Discrete Appl. Math. 48 (1994) 101–109.
[4] D.Z. Du, F.K. Hwang, Combinational Group Testing and its Applications, second ed., World Scientific, Singapore, 2000.
[5] P. Johann, A group testing problem for graphs with several defective edges, Discrete Appl. Math. 117 (2002) 97–108.
[6] E. Triesch, A group testing problem for hypergraphs of bounded rank, Discrete Appl. Math. 66 (1996) 165–188.