STRUCTURAL DIAGNOSIS OF WIRING NETWORKS: FINDING CONNECTED COMPONENTS OF UNKNOWN SUBGRAPHS*

WEIPING SHI† AND DOUGLAS B. WEST‡

Abstract. Given a graph $G = (V, \mathcal{E})$, we want to find the vertex sets of the components of an unknown subgraph $F = (V, E)$ of G, such that $E \subseteq \mathcal{E}$. We learn about F by sending an oracle a query set $S \subseteq V$, and the oracle tells us the vertices connected to S in F. The objective is to use the minimum number of queries to partition the vertex set V into components of F. In electronic circuit design, the problem is also known as structural diagnosis of wiring networks.

Key words. graph theory, lower bound, testing, diagnosis, component.

AMS subject classifications. 68Q25, 68R10, 05C85, 05C40, 94C12

1. Introduction.

1.1. Problem Formulation. Diagnosis of wiring networks is an important problem in the design and production of very large scale integration (VLSI), multi-chip module (MCM) and printed circuit board (PCB) systems [1, 4, 5, 7, 8, 12]. A wiring network consists of a set of nets. Each net contains a driver, a set of receivers and electric conductors that connect the driver and the receivers. The logic value (1 or 0) of a good net is set by its driver and observed by its receivers. When two or more nets are involved in a short fault, their receivers all receive the logical OR of the values of their drivers. To diagnose a wiring network, we send test vectors of 0's and 1's from the drivers, and observe the outputs from the receivers, to find all the short faults.

In this paper, we study *structural diagnosis*, that is the detection and location of all short faults between nets using the information regarding the particular routing of the nets. In contrast, *behavioral diagnosis* does not use any structural information and assumes all faults are possible. From the circuit layout information, we can find all places where a direct short fault may occur and represent the information as an undirected graph $G = (V, \mathcal{E})$, which we call the *adjacency graph*. Each vertex $v \in V$ represents a net and each edge $v_iv_j \in \mathcal{E}$ represents a potential direct short fault between nets v_i and v_j. Note that although G records only potential direct short faults between pairs of nets, we may have multiple-net short faults through sequences of direct shorts involving two nets each.

In any graph G, vertices v_i and v_j are *connected* if G contains a path from v_i to v_j. The *components* of a graph G are its maximal connected subgraphs. The vertex sets of the components are the equivalence classes of the connection relation, which we call the *connection classes* of G.

The actual presence of direct short faults among the nets can be viewed as a *fault graph* $F = (V, E)$, which is a subgraph of G. The vertex set of F is the same as the vertex set of G, the edge set of F is a subset of the edge set of G, but is not given to us. Each edge v_iv_j of F represents the presence of an actual direct short fault

* Research of the first author was supported in part by NSF grant MIP-9309120. Research of the second author was supported in part by NSA/MSP grant MDA904-93-H-3040.

†Department of Electrical Engineering, Texas A&M University, College Station, Texas 77843 (wshi@ee.tamu.edu).

‡Department of Mathematics, University of Illinois at Urbana-Champaign, Urbana, Illinois 61801 (west@math.uiuc.edu).

between nets v_i and v_j. Fig. 1.1 shows an adjacency graph G, a fault graph F, and the connection classes of F. In general, it is not possible to uniquely determine F. For example in Fig. 1.1, we cannot distinguish whether F has two edges or three edges among $\{v_1, v_2, v_3\}$.

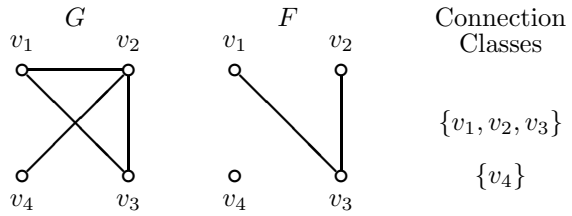

FIG. 1.1. *Adjacency graph G, fault graph F, and connection classes of F.*

Formally, the problem of structural diagnosis can be described as follows. Given an adjacency graph G, we want to find the connection classes of the unknown fault graph $F \subseteq G$. We obtain information about F only through queries to an oracle. For any query set $S \subseteq V$, the oracle tells us $Q(S)$, the set of vertices connected to vertices of S in the fault graph F. In other words, $Q(S)$ is the union of the connection classes that intersect S. In the example of Fig. 1.1, $Q(\{v_1\}) = \{v_1, v_2, v_3\}$ and $Q(\{v_2, v_4\}) = \{v_1, v_2, v_3, v_4\}$. The objective is to find the connection classes of F using the minimum number of queries. Diagnosing a wiring network corresponds to finding the connection classes of the fault graph F, and applying tests corresponds to querying the oracle.

1.2. Previous Results. Diagnosis algorithms may be adaptive or non-adaptive [1]. In an *adaptive* algorithm, each query is computed using the responses to previous queries. In a *non-adaptive* algorithm, all query inputs are decided before asking any queries. Adaptive algorithms can be used in applications where test vectors and results are sent and received though a external device. Non-adaptive diagnosis can be used in applications where tests are hardwired inside of the system.

There are three levels of diagnostic resolution. The highest level is *full diagnosis*: finding all the connection classes. The second level is *faulty net identification*: identifying all the nets involved in short faults. The lowest level is *fault detection*, detecting whether there is any fault at all. Faulty net identification and fault detection have been well understood [4, 8, 12]. In this paper, we study full diagnosis.

Table 1.1 and 1.2 summarize the previous best results. In the table, lg denotes the base 2 logarithm, n is the number of nets, G is the adjacency graph, and $\chi(G)$ is the chromatic number of G.

For behavioral diagnosis of n nets, which is the special case of structural diagnosis where $G = K_n$, Kautz [8] showed that $\lceil \lg n \rceil$ tests are necessary and sufficient to detect whether some short exists. He used the so-called *counting sequence* method, which is optimal for fault detection but does not provide faulty net identification. To identify all the nets involved in short faults, Cheng, Lewandowski, and Wu [4] proposed the *maximum anti-chain* method that uses $\lg n + \frac{1}{2} \lg \lg n + O(1)$ tests. In this method, the bit strings sent by the nets all contain an equal number of 0's and 1's. The maximum

TABLE 1.1
Best algorithms for behavioral diagnosis ($G = K_n$).

diagnostic resolution	**adaptive**	**best algorithm**	**number of tests**	**optimal**
fault detection	no	counting seq [8]	$\lceil \lg n \rceil$	yes
faulty net identification	no	max anti-chain [4]	$(1 + o(1)) \lg n$	almost
full diagnosis	no	walking 1 skip 1 [12]	$n - 1$	yes
	yes	divide conquer [12]	$\lceil \lg n \rceil$	yes

TABLE 1.2
Best algorithms for structural diagnosis (arbitrary G).

diagnostic resolution	**adaptive**	**best algorithm**	**number of tests**	**optimal**
fault detection	no	graph coloring [4, 7]	$\lceil \lg \chi(G) \rceil$	yes
faulty net identification	no	max anti-chain [4]	$(1 + o(1)) \cdot \lg \chi(G)$	almost
full diagnosis	no	decomposition [5]	?	?
	yes	?	?	?

anti-chain algorithm is proved to be optimal, or within 1 from optimal, under various restrictions on how the results are analyzed [4, 12]. For non-adaptive full diagnosis, a number of researchers proposed the *walking ones* method. In this method, each query contains a single vertex. After n queries, we can look at each output $Q(\{v_i\})$ to tell which vertices are shorted to v_i. Shi and Fuchs [12] proved that if we skip any one test from the walking ones algorithm, then it is optimal. For adaptive full diagnosis, Shi and Fuchs [12] presented a divide-and-conquer algorithm that uses $\lceil \lg n \rceil$ tests, which is optimal. Shi and West [13] also gave an optimal randomized algorithm that uses expected $\lg \lg n + \lg k$ queries, where k is the number of connection classes.

For structural diagnosis, where the adjacency graph G is an arbitrary graph, several researchers [4, 7] observed that $\lceil \lg \chi(G) \rceil$ tests are sufficient for fault detection, where $\chi(G)$ is the chromatic number of G. It can be shown that $\lceil \lg \chi(G) \rceil$ tests are also necessary: Consider an adversary that responds $Q(S) = S$ while potential edges remain, this result is equivalent to showing that the minimum number of bipartite subgraphs needed to cover G is $\lceil \lg \chi(G) \rceil$. Although these observations imply that finding the minimum of tests for fault detection is NP-hard, it relates the fault detection problem to the well-known graph theory problem where approximation algorithms are available. The maximum anti-chain algorithm is also applied to structural diagnosis [4], using $\lg \chi(G) + \frac{1}{2} \lg \lg \chi(G) + O(1)$ tests for faulty net identification. For full diagnosis, Feng, Huang, and Lombardi [5] proposed a graph decomposition algorithm.

1.3. Our Results. We present a variety of results for full diagnosis using graph theoretic techniques. In Section 2, we propose adaptive algorithms for structural diagnosis. Theorem 2.7 leads to a method of approximating the minimum number of

adaptive tests for arbitrary adjacency graphs. In Section 3, we consider the non-adaptive problem. We prove that deciding whether a set of queries can perform full diagnosis is NP-hard, even if $G = K_n$. Theorem 3.7 gives a method of approximating the minimum number of non-adaptive queries for arbitrary adjacency graphs. In Section 2 and Section 3, we also present optimal or near optimal adaptive and non-adaptive algorithms when the adjacency graphs are complete graphs, complete bipartite graphs, paths, trees, planar graphs, or random graphs.

2. Adaptive Algorithms. In this section we study adaptive algorithms for finding the connection classes of an unknown subgraph F of an arbitrary graph G. An adaptive algorithm A implicitly defines a decision tree. At each node of the decision tree, the algorithm selects a set of vertices S and makes a query. Upon receiving the set $Q(S)$ from the oracle, the algorithm selects a subtree. Each leaf of the decision tree corresponds to a partition of the vertex set $V(G)$ into connection classes. The decision tree cannot be represented explicitly because the number of leaves is at least the number of ways to partition $V(G)$, which is exponential.

DEFINITION 2.1. *Let $\mathbf{A}_G$ be the set of all adaptive algorithms that find the connection classes of an unknown subgraph of G. Let $q(A, F)$ be the number of queries used by algorithm A when the unknown subgraph is F. We define $q(G)$, the adaptive test number of a graph G, to be:*

$$q(G) = \min_{A \in \mathbf{A}_G} \max_{F \subseteq G} q(A, F).$$

In other words, $q(G)$ is the minimum number of tests necessary and sufficient for adaptive full diagnosis when the adjacency graph is G.

Graph G_1 is a *minor* of graph G_2 if G_1 can be obtained from G_2 by a sequence of edge deletions and edge contractions. For example, Fig. 2.1 shows that the complete graph K_4 is a minor of the complete bipartite graph $K_{3,3}$.

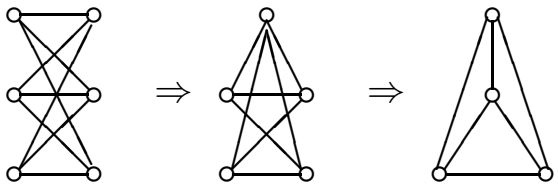

FIG. 2.1. *K_4 is a minor of $K_{3,3}$.*

LEMMA 2.2. *If G_1 is a minor of G_2, then $q(G_1) \leq q(G_2)$.*

Proof. If G_1 is obtained from G_2 by deleting (or contracting) an edge e, then the problem of finding connection classes for an unknown subgraph F_1 of G_1 is the problem of finding the connection classes of an unknown subgraph F_2 of G_2 with the additional information that F_2 does not contain (or does contain) edge e. With more information about F_2 within G_2, we cannot need more queries.

To see how to construct an algorithm for G_1 given an algorithm for G_2, assume G_1 is obtained from G_2 by contracting an edge v_iv_j to v_i', where $v_i, v_j \in V_2$ and

$v_i' \in V_1$. Any algorithm A_2 that finds the connection classes for G_2 can be modified to become an algorithm A_1 for G_1 as follows. Whenever A_2 makes a query S, A_1 makes a query S', where

$$S' = \begin{cases} S & \text{if } v_i \notin S \text{ and } v_j \notin S, \\ S \cup \{v_i'\} - \{v_i, v_j\} & \text{otherwise.} \end{cases}$$

□

LEMMA 2.3. *If the vertex sets of graphs G_1 and G_2 are disjoint, and $G_1 + G_2$ denotes the disjoint union of G_1 and G_2, then $q(G_1 + G_2) = \max\{q(G_1), q(G_2)\}$.*

Proof. Let V_1, V_2 be the vertex sets for G_1, G_2, and let A_1, A_2 be optimal algorithms for finding connection classes on these graphs. We define an algorithm on $G_1 + G_2$. Whenever A_1 wants to query $S_1 \subseteq V_1$ and A_2 wants to query $S_2 \subseteq V_2$, we query $S_1 \cup S_2$. Since $Q(S_1) = Q(S_1 \cup S_2) \cap V_1$ and $Q(S_2) = Q(S_1 \cup S_2) \cap V_2$, both A_1 and A_2 can proceed. □

DEFINITION 2.4. *For any graph $G = (V, \mathcal{E})$ and $S \subseteq V$, the S-connection graph G_S of G is the graph with vertex set S and edge set consisting of all pairs v_iv_j, such that $v_i, v_j \in S$ and G has a path from v_i to v_j that intersects S only at its endpoints.*

For example, Fig. 2.2 shows a graph G and its S-connection graph G_S. The concept of S-connection graph allows us to concentrate on the set of vertices S and simplify the rest of the graph to keep only the connection information.

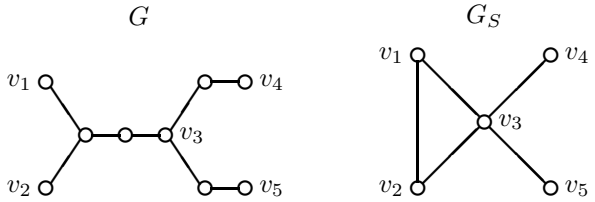

FIG. 2.2. *G_S is an S-connection graph of G for $S = \{v_1, v_2, v_3, v_4, v_5\}$.*

DEFINITION 2.5. *For any adjacency graph $G = (V, \mathcal{E})$ and $S \subseteq V$, the restricted connection class problem (G, S) is the problem of finding the connection classes of an unknown subgraph F of G, where F is restricted to subgraphs of G such that $Q(S) = V$. The number of queries required to solve this problem is*

$$q'(G, S) = \min_A \max_F q(A, F),$$

where F satisfies the restriction described above, A has the knowledge that F is restricted, and $q(A, F)$ is the number of queries used by A when the fault graph is F.

For example, consider the adjacency graph G in Fig. 1.1 and the restricted connection class problem $(G, \{v_2, v_4\})$. Since the restriction requires that $Q(\{v_2, v_4\}) = \{v_1, v_2, v_3, v_4\}$, we know the fault graph F must contain at least two edges among $\{v_1, v_2, v_3\}$. Therefore the restricted problem can be solved using only one query: $Q(\{v_2\})$.

Arguing as in the proof of Lemma 2.2, it follows that if G_1 is a minor of G_2 and both contain the vertex set S, then $q'(G_1, S) \leq q'(G_2, S)$.

LEMMA 2.6. *For any graph $G = (V, \mathcal{E})$ and a set of vertices $S \subseteq V$, $q'(G, S) \leq q(G_S)$.*

Proof. We first show for every fault graph F of G, there exists a fault graph H of G_S, such that the connection classes of H is the intersection of the connection classes of F with S. H consists of edges v_iv_j such that F has a path from v_i to v_j intersecting S only at its endpoints. The graph H incorporates the information of which pairs of vertices in F are in the same connection classes.

Let A be an optimal algorithm that solves the connection class problem for G_S. We use A to solve the restricted connection class problem (G, S). At each step, if A wants to make a query R on G_S, we make a query R on G and use $Q(R) \cap S$ as a simulated response on G_S. This is the correct response for an actual subgraph H of G_S whose connection classes are the intersections with S of the connection classes of the unknown subgraph F of G. Algorithm A uses the response $Q(R) \cap S$ to choose the next query to make on G_S. When A completes its work, it declares the connection classes for H. We claim that this partition of S permits us to determine the connection classes of F without further queries. If this claim is true, then we have used at most $q(G_S)$ queries to solve the restricted connection class problem.

Let C_i, C_j be any two connection class of F. Since $Q(S) = V$, $C_i \cap S \neq \emptyset$ and $C_j \cap S \neq \emptyset$. Also, $C_i \cap S$ and $C_j \cap S$ are connection classes in H. Thus we know the correct distribution of S among connection classes of F. Since A determines that C_i and C_j are distinct classes, some response contains one of them but not the other. Therefore, we learn that each vertex $v \in C_i$ is not connected to any vertex in C_j, for any $j \neq i$. □

For any graph $G = (V, \mathcal{E})$ and set $S \subseteq V$, the *induced subgraph* of G by S is the graph $G[S]$ whose vertex set is S and edge set is $\{v_iv_j\colon\ v_iv_j \in \mathcal{E} \text{ and } v_i, v_j \in S\}$. The graph obtained by deleting the vertices in S is $G - S$; thus $G[V - S] = G - S$.

THEOREM 2.7. *For any graph G with vertex set V,*

$$q(G) \leq 1 + \min_{S \subseteq V} \max\{q(G_S), q(G - S)\}.$$

Proof. For any algorithm that solves the connection class problem on G, let S be the set of vertices chosen by the algorithm to make the first query. The response $Q(S)$ partitions G into disjoint subgraphs $G[Q(S)]$ and $G - Q(S)$, such that the fault graph F has no edge between $Q(S)$ and $G - Q(S)$. Therefore,

$$\begin{aligned} q(G) &\leq 1 + \min_{S \subseteq V} \max_{Q(S)} \max\{q'(G[Q(S)], S), q(G - Q(S))\} \\ &= 1 + \min_{S \subseteq V} \max\{\max_{Q(S)} q'(G[Q(S)], S), \max_{Q(S)} q(G - Q(S))\} \\ &\leq 1 + \min_{S \subseteq V} \max\{q'(G, S), q(G - S)\}. \end{aligned}$$

In the last step, we used the fact that $G[Q(S)]$ is a minor of G, and that $G - Q(S)$ is a minor of $G - S$. From Lemma 2.6, the Theorem is proved. □

COROLLARY 2.8. *Let G be a graph with vertex set V. For any $S \subseteq V$, if $G_1, \ldots, G_k$ are components of $G - S$, then $q(G) \leq 1 + \max\{\lceil \lg |S| \rceil, q(G_1), \ldots, q(G_k)\}$.*

Proof. Since every S-connection graph is a minor of $K_{|S|}$, $q(G_S) \leq q(K_{|S|}) = \lceil \lg |S| \rceil$. From Lemma 2.3, $q(G - S) = \max\{q(G_1), q(G_2), \ldots, q(G_k)\}$. □

THEOREM 2.9. *For any graph G with vertex set V, if for every $S \subseteq V$, G_S is obtained by deleting or contracting each edge not in $G[S]$, then*

$$q(G) = 1 + \min_{S \subseteq V} \max\{q(G_S), q(G - S)\}.$$

Proof. Every optimal algorithm begins by making a query on some set S, after which it must solve the restricted problem $(G[Q(S)], S)$ and the usual connection class problem on $G - Q(S)$. It chooses S to minimize the worst-case subsequent number of queries. This yields the first equality below.

$$\begin{aligned} q(G) &= 1 + \min_{S \subseteq V} \max_{Q(S)} \{\max\{q'(G[Q(S)], S), q(G - Q(S))\}\} \\ &= 1 + \min_{S \subseteq V} \max\{\max_{Q(S)} q'(G[Q(S)], S), \max_{Q(S)} q(G - Q(S))\} \\ &\geq 1 + \min_{S \subseteq V} \max\{q'(G, S), q(G - S)\}. \end{aligned}$$

In the last step, we choose $Q(S) = V$ for the first term, and we choose $Q(S) = S$ for the second term. Then, since G_S is a minor of G, we have $q'(G, S) \geq q'(G_S, S)$. Finally, we observe $q'(G_S, S) = q(G_S)$ since the restricted problem (G_S, S) is actually the unrestricted problem on G_S. Therefore,

$$q(G) \geq 1 + \min_{S \subseteq V} \max\{q(G_S), q(G - S)\}.$$

The other direction of the inequality is Theorem 2.7. □

Theorem 2.7 and Corollary 2.8 can be used to design approximation algorithm for general graphs, such as the one at the end of this section. Theorem 2.9 can be used to obtain exact expressions for the adaptive test number on some classes of graphs.

COROLLARY 2.10. *For the complete graph K_n on n vertices, $q(K_n) = \lceil \lg n \rceil$.*

Proof. This was first proved by Shi and Fuchs [12]. Here we obtain it from Theorem 2.9. Each S-connection graph of K_n is $K_{|S|}$, which is a minor of K_n. Also $K_n - S = K_{n-|S|}$. Therefore $q(K_n) = 1 + q(K_{\lceil n/2 \rceil}) = \lceil \lg n \rceil$. □

COROLLARY 2.11. *For the n-vertex path P_n, $q(P_n) = \lceil \lg \lg(n+1) \rceil$.*

Proof. Each S-connection graph is $P_{|S|}$, which is a minor of P_n. The graph $G - S$ has $n - |S|$ vertices. From the pigeonhole principle, at least one component of $G - S$ is a path of at least $(n - |S|)/(|S| + 1)$ vertices. On the other hand, if the vertices of S are evenly spaced among the n vertices, then every component in $G - S$ contains at most $(n - |S|)/(|S| + 1)$ vertices. Therefore

$$q(P_n) = 1 + \min_{S \subseteq V} \max\{q(P_{|S|}), q(P_{\lceil (n-|S|)/(|S|+1) \rceil})\}.$$

Solving the equation $|S| = (n - |S|)/(|S| + 1)$ gives $|S| = \sqrt{n+1} - 1$. Therefore, $q(P_n) = 1 + q(P_{\lceil \sqrt{n+1}-1 \rceil})$ which yields $q(P_n) = \lceil \lg \lg(n+1) \rceil$. □

THEOREM 2.12. *If G is a tree of n vertices, then $q(G) \leq \lg \lg n + 3$, and the queries can be constructed in polynomial time.*

Proof. By removing a single vertex, an n-vertex tree can be partitioned into components having at most $n/2$ vertices each. (Pick any vertex v in the tree, if some component in $G - \{v\}$ has more than $n/2$ vertices, move to the neighbor of v in that component, and repeat until all components have size at most $n/2$.)

We iteratively place such splitting vertices into S until each remaining component has at most $2\sqrt{n}$ vertices. This process is modeled by a decomposition tree T. The parents of leaves in T correspond to connected subgraphs of G with at least $2\sqrt{n}$ vertices, so there are at most $\sqrt{n}/2$ of them. When the leaves of T are deleted, we have a tree with at most $\sqrt{n}/2$ leaves and thus fewer than $\sqrt{n}$ vertices, each corresponding to a vertex of S.

If G_S at this point is not a tree, as in Fig. 2.2, we add additional vertices of G to S. Let G' denote the subgraph of G that is the union of all paths in G joining vertices of G. We add to S all vertices that have degree at least 3 in G' (in Fig. 2.2, one vertex is added). Since G' has fewer than $\sqrt{n}$ leaves, we add fewer than $\sqrt{n}$ vertices to S. The final set S has fewer than $2\sqrt{n}$ vertices. The graph G_S is the graph obtained from G' by contracting an edge incident to a vertex of degree 2 (unless both endpoints are in S) until no further such operations are available.

Let $f(n) = \max q(G)$, where the maximum is taken over all n-vertex trees. By Theorem 2.7, $f(n) \le 1 + f(2\sqrt{n})$. This recurrence yields $f(n) \le \lg\lg n + f(8) = \lg\lg n + 3$. □

Theorem 2.12 is essentially best possible, because the test number of the path is within three of this bound.

THEOREM 2.13. *If G is the complete bipartite graph $K_{m,n}$, then*

$$q(G) = \lceil \lg(\min\{m,n\} + 1) \rceil.$$

Proof. Assume without loss of generality that $m \le n$. If we pick any $\lceil m/2 \rceil$ vertices in the partite set of size m to use as S in Theorem 2.7, then

$$\begin{aligned} q(K_{m,n}) &\le 1 + \max\{q(K_{\lceil m/2 \rceil}), q(K_{\lfloor m/2, n})\} \\ &\le 1 + q(K_{\lfloor m/2 \rfloor \rfloor, n}). \end{aligned}$$

With $q(K_{1,n}) = 1$, the recurrence yields $q(K_{m,n}) \le \lceil \lg(m+1) \rceil$.

On the other hand, contracting $m-1$ edges of a matching in $K_{m,n}$ yields a minor K_{m+1}, as illustrated in Fig. 2.1. From Lemma 2.2, $q(K_{m,n}) \ge q(K_{m+1}) = \lceil \lg(m+1) \rceil$. □

Next consider G is a planar graph. Planar graphs arise naturally when the routing is planar and direct short faults occur only between wires that are close together [7].

THEOREM 2.14. *(Planar Separator Theorem, Lipton-Tarjan [10]) Let G be an n-vertex planar graph. In $O(n)$ time we can partition $V(G)$ into three sets A, B, C such that 1) no edge has one endpoint in A and the other endpoint in B, 2) $|A|, |B| \le 2n/3$, and 3) $|C| \le \sqrt{8n}$.*

COROLLARY 2.15. *Let G be an n-vertex planar graph. In $O(n)$ time we can find a set $S \subset V(G)$ such that $|S| \le (12 + 6\sqrt{2})\sqrt{n}$ and each component of $G - S$ has at most $n/4$ vertices.*

Proof. From Theorem 2.14, $V(G)$ can be partitioned into A, B and C such that there is no edge between A and B, $|A|, |B| \le 2n/3$, and $|C| \le \sqrt{8n}$. We call such a set C a *separator*. The sizes of A and B are bounded by αn and $(1-\alpha n)$ for some $1/3 \le \alpha \le 2/3$. Recursively find separators C_A for $G[A]$ and C_B for $G[B]$, we have $|C_A \cup C_B| \le \sqrt{8\alpha n} + \sqrt{8(1-\alpha)n} < \sqrt{8n}\sqrt{2}$. We apply Theorem 2.14 recursively for 4 levels, reducing all components among the remaining vertices to order at most $(2/3)^4 n = 16n/81 < n/4$. Let S be the union of all the separators found in this tree of separations. We have

$$|S| \le \sqrt{8n}(1 + 2^{1/2} + 2^{2/2} + 2^{3/2}) = \sqrt{n}(12 + 6\sqrt{2}).$$

Since each C can be found in time linear in the number of vertices, the total time to find S is $O(n)$. □

THEOREM 2.16. *If G is a planar graph of n vertices, then $q(G) \le \frac{1}{2}\lg n + O(1)$, and each query can be constructed in $O(n)$ time.*

Proof. Let $f(n) = \max q(G)$, where the maximum is taken over all n-vertex planar graphs. By Corollaries 2.8 and 2.15,

$$f(n) \leq 1 + \max\{\lg(\sqrt{n}(12 + 6\sqrt{2})), f(n/4)\}.$$

By induction it can be shown $f(n) \leq \frac{1}{2}\lg n + c + 1$, where $c = \lg(12 + 6\sqrt{2})$. □

We do not know whether Theorem 2.16 is best possible. The best lower bound is $\lg\lg n$ when G is a path of n vertices. Note that the S-connection graph of a planar graph need not be planar. We leave as an open problem to close the gap. Please note that since no planar graph contains K_5 or $K_{3,3}$ as a minor, it is not possible to prove any nontrivial lower bound using Lemma 2.2.

Random graphs [2] are generated by letting each edge occur with probability 1/2. When we say *almost every* graph has property X, it means the probability for a random graph on n vertices to have property X intends to 1 as n goes to infinity. We show next that solving the connection class problem for random graphs is almost as hard as that for complete graphs.

THEOREM 2.17. *For almost every graph G, $q(G) \geq \lg n - \frac{1}{2}\lg\lg n + O(1)$. This bound also holds for every graph with at least $n^2/4$ edges.*

Proof. Bollobás, Catlin and Erdős [3] proved that almost every n-vertex graph has K_m as a minor, where $m = (1 + o(1))n/\sqrt{\lg n}$. Lemma 2.2 then yields $q(G) \geq \lg m = \lg n - \frac{1}{2}\lg\lg n + O(1)$ almost always. In fact *every* graph of n vertices and about $n^2/4$ edges has K_m as a minor, where $m = (1 + o(1))n/\sqrt{\lg n}$ (see Bollobás [2], page 279). □

In general, computing $q(G)$ appears to be NP-hard, but we have not proved this. Observe that the minimum number of queries to determine whether $F = \overline{K}_n$ equals $\lceil \lg \chi(G) \rceil$. Thus $q(G) \geq \lceil \lg \chi(G) \rceil$, but it is still possible that $q(G)$ is easier to compute.

It is important to clarify that the arguments of all Theorems in this section give procedures for generating the first query for the algorithms whose number of queries satisfies the resulting bounds, but the arguments are not recursive algorithms for finding the connection classes. For example in the proof of Theorem 2.7, we assumed the worst-case that $Q(S) = V$ and $Q(S) = G - S$, but the actual response is not necessarily the worst-case. Thus we must take $Q(S)$ into account to decide each query to solve a particular instance.

The proof in Theorem 2.7 can be used to design a heuristics for general graphs: Find a vertex separator S, construct G_S and the components in $G - S$, then recurse for G_S and $G - S$ in parallel. The key is to find a vertex separator S so that $\max\{q(G_S), q(G - S)\}$ is minimized. In general, when S is small (large), $q(G_S)$ is small (large) while $q(G - S)$ is large (small). Therefore, we may experiment on the size of S until we balance $q(G_S)$ and $q(G - S)$.

Algorithm 1 is the adaptive diagnosis algorithm. It iteratively maintains a *component structure* $P = \{(G_i, R_i) : i = 1, 2, \ldots, t\}$, where $\{G_1, G_2, \ldots, G_t\}$ is a collection of adjacency graphs whose vertex sets form a partition of V, and $\{R_1, R_2, \ldots, R_t\}$ is a collection of "representative" subsets of vertices such that $R_i \subseteq V(G_i)$ and $Q(R_i) = V(G_i)$. In other words, each (G_i, R_i) is a restricted connection class problem. This property of R_i's implies that each G_i is a union of components. Algorithm 1 initializes with the component structure $P = \{(G, V(G))\}$, and then refines the partition to reduce the size of each R_i. When the algorithm terminates, every R_i is reduced to a single vertex, and therefore every $V(G_i)$ is one connection class. The hardest step is step 4, where we have to use the Theorems in this section to find the set S_i.

Algorithm 1. Adaptive full diagnosis.
Input: Adjacency graph G.
Output: All connection classes.
1: $P \leftarrow \{(G, V(G))\}$.
2: **Repeat**
3: **For** each $(G_i, R_i) \in P$ **do**
4: Find $S_i \subset R_i$.
5: Query $\cup S_i$, with result $Q \leftarrow Q(\cup S_i)$.
6: $P' \leftarrow \emptyset$.
7: **For** each $(G_i, R_i) \in P$ **do**
8: $U_i \leftarrow Q \cap V(G_i)$.
9: **For** each component C of $G_i[U_i]$ **do**
10: $P' \leftarrow P' \cup \{(C, S_i \cap V(C))\}$
11: **For** each component C of $G_i - U_i$ **do**
12: $P' \leftarrow P' \cup \{(C, R_i \cap V(C))\}$.
13: $P \leftarrow P'$.
14: **Until** $|R_i| = 1$ for all $(G_i, R_i) \in P$.
14: Report each $V(G_i)$ as one connection class.

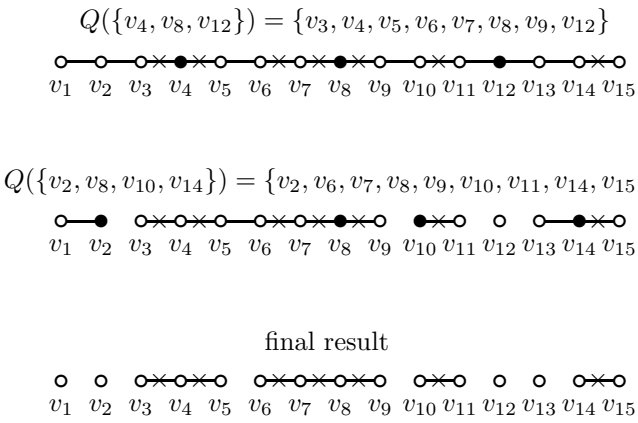

FIG. 2.3. *Find connection classes of $F \subset P_{15}$ in 2 queries.*

For example, Fig. 2.3 considers $G = P_{15}$, and uses Corollary 2.11 to generate the queries. The edges in the adjacency graph G are shown, and the edges in the fault graph F are marked with "×". The dark vertices are vertices in the query set S. After the first query, we have five subgraphs containing potential components, including two in $G[Q(S)]$ and three in $G - Q(S)$. Each subgraph is a path. We use Corollary 2.11 to generate the next query for each subgraph. Because the 7-vertex component in $G[Q(S)]$ was generated by two vertices of S, we know that all but one of its edges are faults, and one additional query at v_8 finishes the restricted problem. After the second query, we have all the connection classes of F.

3. Non-Adaptive Algorithms. In this section we study non-adaptive algorithms for solving the connection class problem. A non-adaptive algorithm T is

a sequence of queries $S_1, S_2, \ldots, S_t$, and a subroutine that analyzes the responses $Q(S_1), Q(S_2), \ldots, Q(S_t)$ to derive the connection classes. The query sets $S_1, S_2, \ldots, S_t$ are decided before asking any queries. We say that a non-adaptive algorithm T solves the connection class problem for G if, for every $F \subseteq G$, T finds the connection classes of F.

DEFINITION 3.1. *Let $\mathbf{T}_G$ be the set of all non-adaptive algorithms that solve the connection class problem for G. Let the number of queries used by an algorithm T be $t(T)$. The non-adaptive test number $t(G)$ of a graph G is the minimum number of non-adaptive queries that always suffices to solve the connection class problem of G:*

$$t(G) = \min_{T \in \mathbf{T}_G} t(T).$$

A sequence of queries $S_1, S_2, \ldots, S_t$ defines a *sequential test vector* $X(v_i)$ for each vertex v_i. $X(v_i)$ is a t-bit binary vector $x_{i1}x_{i2}\cdots x_{it}$ where $x_{ij} = 1$ if $v_i \in S_j$, and $x_{ij} = 0$ otherwise. Conversely, sequential test vectors $X(v_1), \ldots, X(v_n)$, where $X(v_i) = x_{i1}x_{i2}\cdots x_{it}$, define a set of queries $S_1, S_2, \ldots, S_t$ by $S_j = \{v_i : x_{ij} = 1\}$. We will also use sequential test vectors to describe the queries in this section, because the vector language underscores the history of queries on each vertex. In Fig. 3.1, each row is a sequential test vector, and each column corresponds to one query.

	test vectors				queries
$X(v_1)$	0	0	1	1	$S_1 = \{v_2, v_5\}$
$X(v_2)$	1	0	0	1	$S_2 = \{v_3\}$
$X(v_3)$	0	1	1	0	$S_3 = \{v_1, v_3, v_5\}$
$X(v_4)$	0	0	0	0	$S_4 = \{v_1, v_2\}$
$X(v_5)$	1	0	1	0	

FIG. 3.1. *Sequential test vectors and queries.*

It is well known that the sequence of queries $S_1, S_2, \ldots, S_n$, where $S_i = \{v_i\}$, (so called the walking ones sequence) is sufficient to solve the connection class problem. It is also known that a sequence of queries is sufficient if it contains the walking ones sequence as a subsequence or is diagonally independent. It is less obvious that the queries in Fig. 3.1 can also perform full diagnosis for K_5.

LEMMA 3.2. *(Shi-Fuchs [12]) A necessary and sufficient condition for sequential test vectors $X(v_1), \ldots, X(v_n)$ to solve the connection class problem on K_n is that for any disjoint nonempty vertex sets U, V, $\vee_{u \in U} X(u) \neq \vee_{v \in V} X(v)$, where the operation $\vee$ is the bit-wise Boolean OR.*

Lemma 3.2 does not give an efficient method to check whether the set of queries can can perform full diagnosis, because the number of such V_1 and V_2 subsets is exponential in terms of n. Next we prove that it is unlikely that there exists any polynomial time verifiable characterization.

THEOREM 3.3. *It is NP-hard to tell whether a set of sequential test vectors can solve the connection class problem for G, even if $G = K_n$.*

Proof. By Lemma 3.2, it is sufficient to show the following problem is NP-hard.

Problem Π: Given a set of t-bit binary vectors $T = \{X_1, \ldots, X_n\}$, are there disjoint non-empty subsets of indices I and J such that $\vee_{i \in I} X_i = \vee_{j \in J} X_j$?

We use a reduction from *Not-All-Equal 3SAT* problem, which is a known NP-complete problem [6], to prove Π is NP-hard. Due to the page limit, the details of the reduction are omitted and can be found in [11]. □

THEOREM 3.4. *A necessary and sufficient condition for sequential test vectors $X(v_1), \ldots, X(v_n)$ to perform full diagnosis for G is that $\vee_{v\in U} X(v) \neq \vee_{v\in W} X(v)$ whenever U, W are disjoint nonempty subsets of $V(G)$ such that $G[U]$, $G[W]$, and $G[U \cup W]$ are connected graphs.*

Proof. If the condition fails and there are U and W as described, then consider fault graphs F_1 and F_2 of G. Graph F_1 has components $G[U]$, $G[W]$, and the rest are isolated vertices. Graph F_2 has component $G[U \cup W]$, and the rest are isolated vertices. Clearly, F_1 and F_2 have different connection classes. However the responses from the oracle for F_1 and F_2 are the same.

Conversely, suppose that the condition holds. First partition the set of vertices into disjoint subsets according the response of each query as follows. Before the ith step, suppose that we have partitioned $V(G)$ into $V_1, \ldots, V_m$. The response of the next query $Q(S_i)$ further partitions each V_j into $V_j \cap Q(S_i)$ and $V_j - Q(S_i)$. When we finish all the queries, report each maximal vertex set inducing a connected subgraph of G that lies in a single V_i as one connection class of the fault graph F. We show that this algorithm works correctly.

If C is a connection class of F, then $C \subseteq Q(S)$ or $C \cap Q(S) = \emptyset$ for every $S \subseteq V(G)$. Thus all of C remains in the same V_k, and all of C will be reported to be in the same connection class.

If connection classes $C_1, \ldots, C_k$ of F are reported as being a single connection class D, then $G[D]$ is connected, because each set reported as a connection class induces a connected subgraph of G. Also, at each iteration the algorithm leaves the entire set D unpartitioned. Thus each $Q(S_i)$ contains all or none of D. Thus S_i contains a vertex of some C_j if and only if it contains a vertex of each C_j. In particular, $\vee_{v\in C_j} X(v)$ is the same for all j. Letting $U = C_1$ and $W = C_2 \cup \ldots \cup C_k$ yields a violation of the condition. □

Ideas like those of Section 2 allow us to compute non-adaptive test numbers of some graphs.

LEMMA 3.5. *If G_1 is a minor of G_2, then $t(G_1) \leq t(G_2)$.*

Proof. Similar to Lemma 2.2. □

LEMMA 3.6. *If graphs G_1 and G_2 are disjoint, then*

$$t(G_1 + G_2) = \max\{t(G_1), t(G_2)\}.$$

Proof. Similar to Lemma 2.3. □

THEOREM 3.7. *For a graph G with vertex set V,*

$$t(G) \leq 1 + \min_{S\subseteq V}\{t(G_S) + t(G - S)\}.$$

Proof. For the set S achieving the minimum, we make the queries consisting of S, a minimum set of queries for G_S, and a minimum set of queries for $G - S$. Each resulting sequential test vector $X(v)$ is the concatenation of one truth bit for $v \in S$, the sequential test vector for v in the query set for G_S, and the sequential test vector for v in the query set for $G - S$.

Let U, W be disjoint nonempty subsets of $V(G)$ such that $G[U]$, $G[W]$, and $G[U \cup W]$ are connected. It suffices to show that $\vee_{v\in U} X(v) \neq \vee_{v\in W} X(v)$.

If U, W both lie outside S, then U, W and $U \cup W$ induce connected subgraphs in $G - S$. Thus the last $t(G - S)$ bits of the sequential test vectors yield the desired non-equality.

If exactly one of U, W intersects S, then query S yields the desired non-equality, since $\vee_{v \in U} X(v)$ and $\vee_{v \in W} X(v)$ differ in the first bit.

Finally, suppose that both U and W intersect S. Let $U' = U \cap S$ and $W' = W \cap S$. By construction, $U', W', U' \cup W'$ all induce connected subgraphs of G_S. Thus the $t(G_S)$ bits of the sequential test vectors corresponding to G_S yield the desired non-equality. □

The algorithm of Feng, et al [5] is a special case of Theorem 3.7, by letting G_S to be $K_{|S|}$.

Shi and Fuchs [12] proved $t(K_n) = n-1$, using general consequences of Lindström [9] and Tverberg [14]. We now consider other families of graphs.

THEOREM 3.8. *If G is the complete bipartite graph $K_{m,n}$, then $t(G) = \min\{m, n\}$.*

Proof. Similar to Theorem 2.13. □

THEOREM 3.9. *For the n-vertex path P_n, $t(P_n) = \lceil \log n \rceil$.*

Proof. For the upper bound, pick the center vertex as S in Theorem 3.7. Then $t(P_n) \leq 1 + t(P_{\lfloor n/2 \rfloor})$. Solving the recurrence relation with $t(P_1) = 0$ gives $t(P_n) \leq \lfloor \lg n \rfloor$.

For the lower bound. let U be the set of $\lceil n/2 \rceil$ vertices closest to one end of the path, and let W be the set of the remaining $\lfloor n/2 \rfloor$ vertices closest to the other end. Note that U, W, and $U \cup W$ all induce connected subgraphs. For sequential test vectors solving the connection class problem, we must have $\vee_{v \in U} X(v) \neq \vee_{v \in W} X(v)$. Let j be a coordinate where they differ. Then elements of U (or W) have test vectors that all are 0 in coordinate j. That means we can solve the connection class problem for U (or W) without using query S_j. Therefore $t(P_{\lfloor n/2 \rfloor}) \leq t(P_n) - 1$ and $t(P_1) = 0$. Solving the recurrence relation yields $t(P_n) \geq \lfloor \lg n \rfloor$. □

THEOREM 3.10. *If G is a tree of n vertices, then $t(G) \leq \lfloor \lg n \rfloor$, and the queries can be constructed in $O(n \log n)$ time.*

Proof. As in Theorem 2.12, each n-vertex tree can be divided into subtrees of order at most $\lfloor n/2 \rfloor$ by removing one vertex. Let $f(n) = \max t(G)$, where the maximum is taken over all n-vertex trees. From Theorem 3.7, $f(n) \leq 1 + f(\lfloor n/2 \rfloor)$. Solving the recurrence relation with $f(1) = 0$ yields $f(n) \leq \lfloor \lg n \rfloor$. The time to find each separator is linear in the number of vertices. The depth of the recursion is $O(\log n)$. Therefore the total time to generate the queries is $O(n \log n)$. □

The upper bound in Theorem 3.10 holds with equality for paths, by Theorem 3.9.

THEOREM 3.11. *If G is an n-vertex planar graph, then $t(G) = O(\sqrt{n})$, and the set of queries can be computed in polynomial time.*

Proof. Let $f(n) = \max t(G)$, where the maximum is taken over all n-vertex planar graphs G. From Theorem 2.14 and Theorem 3.7, $f(n) \leq \sqrt{8n} + f(2n/3)$, which yields $f(n) = O(\sqrt{n})$. The time to find each separator is $O(n)$. The time to find all separators is $O(n \log n)$ since the depth of the recursion is $O(\log n)$. □

THEOREM 3.12. *For almost every graph G, $t(G) \geq (1 + o(1))n/\sqrt{\lg n}$.*

Proof. Similar to Theorem 2.17. □

Let $\chi(G)$ be the chromatic number of G. Since $\lg \chi(G)$ queries are necessary to tell whether $F = \overline{K}_n$, at least this many queries are needed in the worst case to find all the connection classes. Therefore $t(G) \geq \lg \chi(G)$. If the conjecture of Hadwiger [3] is true, then $t(G) \geq \chi(G) - 1$. Hadwiger's conjecture states that each graph G has

$K_{\chi(G)}$ as a minor.

4. Discussion. We presented new adaptive and nonadaptive algorithms for interconnect diagnosis. The adaptive algorithms reduce the number of tests exponentially compared with traditional non-adaptive algorithms. We also show that structural information can further reduce the number of tests drastically, for certain sparse graphs such as planar graphs. For dense graphs, there is not much gain using structural diagnosis.

TABLE 4.1
Number of tests for full diagnosis of special families of adjacency graphs.

adaptive		K_n	$K_{m,n}$ $m \le n$	path	tree	planar graph	random graph
yes	lower bound	$\lg n$	$\lg(m+1)$	$\lg\lg(n+1)$	$\lg\lg(n+1)$	$\lg\lg(n+1)$	$\lg n - \frac{1}{2}\lg\lg n$
	upper bound	$\lg n$	$\lg(m+1)$	$\lg\lg(n+1)$	$\lg\lg n + 3$	$\frac{1}{2}\lg n$	$\lg n$
no	lower bound	$n-1$	m	$\lfloor \lg n \rfloor$	$\lfloor \lg n \rfloor$	$\lfloor \lg n \rfloor$	$n/\sqrt{\lg n}$
	upper bound	$n-1$	m	$\lfloor \lg n \rfloor$	$\lfloor \lg n \rfloor$	$O(\sqrt{n})$	$n-1$

Our results for special families of graphs are summarized in Table 4.1. In the table, n is the number of vertices of G except for $K_{m,n}$, and all fractions are rounded to the ceiling unless otherwise specified. The *lower bound* for a family is the maximum number of tests necessary for any graph in that family. The *upper bound* for a family is the number of tests sufficient for all graphs in that family. The computation time for generating the queries is low-order polynomial for all the algorithms in Table 4.1. The results for K_n were first given by Shi and Fuchs [12], but we include here for completeness.

To use Theorem 2.7 and 3.7 for general graphs, it is crucial that we find good multi-way vertex separators. Unfortunately, there are few practical algorithms for finding good vertex separators of general graphs.

REFERENCES

[1] M. Abramovic, M. A. Breuer and A. D. Friedman. *Digital System Testing and Testable Design.* Computer Science Press, 1990.
[2] B. Bollobás. *Random Graphs.* Academic Press, New York 1985.
[3] B. Bollobás, P. Catlin and P. Erdős. Hadwiger's conjecture is true for almost every graph. *Europ. J. Combinatorics*, Vol. 1, 1980, 195-199.
[4] W.-T. Cheng, J. L. Lewandowski and E. Wu. Optimal diagnostic methods for wiring interconnects. *IEEE Trans. Computer-Aided Design*, Vol. 11, No. 9, Sept. 1992, 1161–1166.
[5] C. Feng, W. Hang and F. Lombardi. A new diagnosis approach for short faults in interconnects, *Proc. 1995 Fault Tolerant Computing Symp.*, 331–339.
[6] M. R. Garey and D. S. Johnson, *Computers and Intractability – A Guide to the Theory of NP-Completeness,* Freeman, New York, 1979.
[7] M. Garey, D. Johnson, and H. So. An application of graph coloring to printed circuit testing. *IEEE Trans. Circuits and Systems*, Vol. CAS-23, No. 10, Oct. 1976, 591–599.
[8] W. H. Kautz. Testing for faults in wiring networks. *IEEE Trans. Computers*, Vol. C-23, No. 4, April 1973, 358–363.

[9] B. Lindström. A theorem on families of sets. *J. Combinatorial Theory (A)*, Vol. 13, pp. 274–277, 1972.

[10] R. J. Lipton and R. E. Tarjan. A separator theorem for planar graphs. *SIAM J. Computing*, Vol. 36, No. 2, April 1979, 177–189.

[11] W. Shi. Complexity of finding two disjoint subsets that have the same unions. Technical Report TAMU-ECE-2001-03, Department of Electrical Engineering, Texas A&M University, April 2001.

[12] W. Shi and W. K. Fuchs. Optimal interconnect diagnosis of wiring networks. *IEEE Trans. on VLSI*, Vol. 3, No. 3, Sept. 1995, 430–436.

[13] W. Shi and D. B. West. Diagnosis of wiring networks: An optimal randomized algorithm for finding connected components of an unknown graph. *SIAM J. Computing,* Vol. 28, No. 5, 1999, 1541-1551.

[14] H. Tverberg. On equal unions of sets. *Studies in Pure Mathematics,* edited by L. Mirsky, Academic Press, 249–250, 1971.