Sequential Fault Diagnosis in Combinational Networks

ISRAEL KOREN, MEMBER, IEEE, AND ZVI KOHAVI, MEMBER, IEEE

Abstract—**The problem considered in this paper is that of generating sequential decision trees (SDT's) for fault diagnosis in digital combinational networks. Since in most applications of the decision tree the final conclusion will be that the network is failure-free, we are interested mainly in decision trees containing minimal fault detection paths. Such a procedure will reduce the cost of verifying the proper operation of the network.**

The faults under consideration are assumed to be single, permanent, stuck-at type faults. *A priori* probabilities are assigned to the nonequivalent faults and the generated decision tree is based upon these probabilities. It is suggested in this paper that the *a priori* probability p_i assigned to the fault f_i should be proportional to the number of faults in the equivalence class of f_i.

A procedure for generating the required decision tree for fan-out-free networks is presented. The procedure generates the tests directly from the structure of the network instead of selecting them from a given fault table. The generated decision tree contains a minimal detection path, i.e., a minimal number of tests required to locate the failure-free network. The decision tree yields a nearly minimal weighted average number of tests required to locate a fault. The average is weighted by the *a priori* probabilities of occurrence of the faults.

A lower bound for this average is derived in Section III enabling adequate evaluation of the generated decision tree.

Index Terms—***A priori* probability of occurrence, combinational logic networks, fault diagnosis, minimal detection set, sequential decision tree (SDT).**

I. Introduction

ALTHOUGH the failures in a digital network are rare, we have to check out frequently the proper operation of the network, i.e., detect and locate any possible failure in order to maintain high reliability of the network.

The tests for detecting and locating the possible faults can be arranged either as a sequential decision tree or as a fixed (preset) set of distinguishing tests [1]–[3]. Sequential diagnosis is preferred because the average number of tests required to locate a fault can be reduced using this method rather than the fixed diagnosis.

In designing a sequential decision tree (SDT) we have two objectives:

1) Reducing the cost of applying the SDT which is proportional to the average number of tests applied.

2) Reducing the computation time needed for generating the SDT.

In most applications of the SDT the final conclusion will be that the network is failure-free. Therefore we have to minimize the number of tests in the detection path in order to achieve the first objective. The second objective is achieved by generating the required tests rather than selecting them from a given fault table.

The fault table is used in most existing methods for selecting locating tests. From the fault table a minimal or nearly minimal set of locating tests is obtained, usually by using weighting functions [1]–[3], [8], [9]. For large networks these methods become inefficient since the computation time and the size of computer memory required increase rapidly. The procedure presented in this paper does not require a fault table and the diagnosis tests are generated directly from the structure of the network. Most existing methods for fault diagnosis first apply a set of detection tests and whenever any of the detection tests fails a set of location tests is applied, e.g., Su and Cho [4]. The limitation of such methods is that they do not use the information gained by the success of the detection tests prior to the failing one. Our objective is to overcome this disadvantage by generating an SDT in which the fault detection tests become an integral part of the fault location procedure.

II. Basic Definitions and Notations

The Network

The network under discussion has n primary inputs $x_1, x_2, \cdots, x_n$ and one primary output $y = y(x_1, x_2, \cdots, x_n)$. There are r lines in this network, $x_1, x_2, \cdots, x_r$, where x_r is the output line. Numbers are assigned to the lines in the network in the usual way, i.e., the number assigned to an output line of a gate is always greater than the numbers assigned to its input lines.

We assume that the network consists of monotone gates [5] which can be defined by one set of values. Let i and j be the inputs of a monotone gate and let k be its output. This gate will be defined by the row $g_i g_j g_k$ where each g is 0 or 1. This row means that $g_i g_j$ is the only combination for which the output of the gate is g_k. For any other combination the output is $\bar{g}_k$, e.g., a two input NOR gate is defined

Manuscript received May 1, 1975; revised January 10, 1976.

I. Koren was with the Department of Electrical Engineering, Technion–Israel Institute of Technology, Haifa, Israel. He is now with the Department of Electrical Engineering and Computer Science, University of California, Santa Barbara, CA 93106.

Z. Kohavi is with the Department of Computer Science, University of Utah, Salt Lake City, Utah 84112, on leave from the Department of Electrical Engineering, Technion–Israel Institute of Technology, Haifa, Israel.

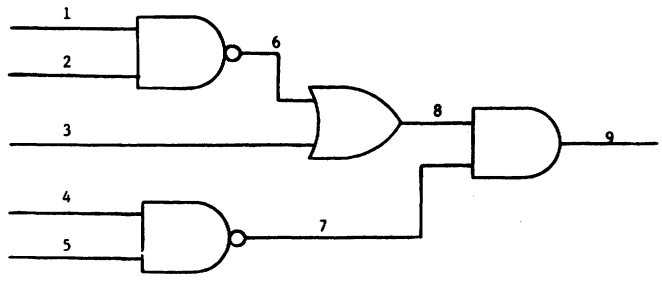


Fig. 1. Tree network.

by the row 001. From this row the Boolean equation of the gate can easily be derived in the following way:

$$x_k^{g_k} = x_i^{g_i} \cdot x_j^{g_j},$$

where $x^0 = \bar{x}$ and $x^1 = x$, e.g., the Boolean equation derived from the row 001 is $x_k^1 = x_i^0 \cdot x_j^0 = \bar{x}_i \cdot \bar{x}_j$, i.e., $x_k = \overline{x_i + x_j}$.

An immediate result is that a network consisting of monotone gates can be described by a gate table where each row corresponds to one gate in the network.

Example: The gate table describing the network in Fig. 1 is given in Table I.

One bit appears in each column corresponding to a primary input or output in the gate table, two bits appear for internal lines. A greater number of bits in these two cases indicates the presence of fan out.

The Faults

The possible faults in the network are assumed to be single, permanent, stuck-at (s-a) type faults in any one of the r lines. Some of these $2r$ faults may not be distinguishable, i.e., they are equivalent [6].

Example: In a monotone gate, defined by the row $g_i g_j g_k$, whose inputs are not fan-out lines, the following faults are equivalent: x_i s-a-$\bar{g}_i$, x_j s-a-$\bar{g}_j$ and x_k s-a-$\bar{g}_k$.

The equivalence relation between the faults partitions the set of $2r$ possible faults into q disjoint equivalence classes. From each equivalence class we choose a representative fault and to these q faults we add f_0 to denote the failure-free network. Our aim is to distinguish among $f_0, f_1, f_2, \cdots, f_q$.

It has been a long established practice to assign equal probabilities of occurrence to all distinguishable faults [1]. However, since different faults occur at different frequencies, different *a priori* probabilities of occurrence should be assigned to them. Assuming that all the $2r$ possible faults are equally probable, it is suggested in this paper that, as a first approximation, the *a priori* probability p_i assigned to the fault f_i should be proportional to the number of elements in the ith equivalence class.

Definition 2.1: The weight ω_i of a fault f_i is equal to the number of elements in the ith equivalence class.

Let $p_i = \omega_i / W$ designate the *a priori* probability of f_i where

$$W = \sum_{i=1}^{q} \omega_i = 2r.$$

Clearly, $\Sigma_{i=1}^{q} p_i = 1$. It must be emphasized that the procedures developed in this paper are valid if a different weighting system is used and hence different probabilities of faults result. Such weights can be assigned based on manufacturer supplied information, statistical data, and so on.

The set T of all tests is divided into two disjoint subsets T_0 and T_1 where $T_0(T_1)$ is the subset of all tests for which the output of the failure-free network is 0(1). Consequently, the set of possible faults is divided into three subsets F^0, F^1, and $F^{0,1}$ where $F^0(F^1)$ is the subset of faults covered (i.e., detected) by tests from $T_0(T_1)$ only. $F^{0,1}$ is the subset of faults covered by some tests from T_0 and by some other tests from T_1. The partitioning of the faults is determined by the following theorem.

Theorem 2.1: The fault x_j s-a-α is included in $F^{\alpha}(F^{\bar{\alpha}})$ if in all possible paths from the line x_j to the primary output the parity of inversions is odd (even). The fault is included in $F^{0,1}$ only if there are at least two different paths from the faulty line to the primary output with unequal parity of inversions.

The proof follows directly from the concept of sensitized paths [1] and is therefore omitted.

Corollary 2.1: In fan-out-free networks $F^{0,1} = \phi$ (where ϕ is the empty set) since each fault has a single path to the primary output [7].

The SDT

In order to distinguish among the faults we wish to generate a minimal SDT. We define a minimal SDT in the following way.

Definition 2.2: An SDT for diagnosis is minimal if the detection path is minimal and the weighted average number of tests required to locate a fault $C = \Sigma_{i=1}^{q} l_i p_i$ is minimal where l_i is the number of tests required to locate the fault f_i.

Note that assigning a high *a priori* probability of occurrence to the failure-free network f_0 and minimizing the modified cost function $C_m = \Sigma_{i=0}^{q} l_i p_i$ does not ensure the minimality of the detection path in the generated SDT. An appropriate counterexample exists but is omitted here.

TABLE I
Gate Table for the Tree Network in Fig. 1

	1	2	3	4	5	6	7	8	9
G_6	1	1				0			
G_7				1	1		0		
G_8			0			0		0	
G_9							1	1	1
	Primary Inputs					Internal Lines			Primary Output

The general method used in this paper to generate an SDT is summarized in Fig. 2. According to this method a minimal set including m detection tests $t_{\alpha_1}, t_{\alpha_2}, \cdots, t_{\alpha_m}$ is generated first, forming the detection path of the SDT. From node $k (k = 1,2, \cdots, m)$ of this detection path emanates a diagnosis subgraph including tests to distinguish among the n_k faults which are covered by the test t_{α_k} and were not covered by the $k-1$ previous detection tests. Clearly $\Sigma_{k=1}^{m} n_k = q$.

Let F_j denote the subset of faults covered by the test t_j and let $F^{(k)}$ denote the subset of faults at note k which were not covered by the $k-1$ previous detection tests. Thus $F_j^{(k)} = F_j \cap F^{(k)}$ denotes the subset of faults covered by t_j out of the faults in the subset $F^{(k)}$. We will show later that the test t_j which is maximal at node k and whose weight is maximal will be selected as the detection test t_{α_k}. The notions of maximal test and the weight of a test are defined as follows.

Definition 2.3: A test t_j is maximal at node k if there is no other test t_i satisfying $F_i^{(k)} \supset F_j^{(k)}$.

Definition 2.4: The weight $P_j^{(k)}$ of a test t_j at node k is the sum of probabilities of the faults in $F_j^{(k)}$.

III. A Lower Bound for SDT's

In order to determine a lower bound for the average number of tests required to locate a fault, $C = \Sigma_{i=1}^{q} l_i p_i$, we have to determine the detection path, i.e., select one out of all minimal detection sets and order the tests within the selected set so that the cost function is minimized. Once the detection path is determined, we have a lower bound for the diagnosis in each of the m subgraphs. This lower bound is derived as follows.

Denote by $f_1^i, f_2^i, \cdots, f_{n_i}^i$ and $p_1^i, p_2^i, \cdots, p_{n_i}^i$ the n_i faults in the i-subgraph and their corresponding probabilities. Let $P_i = \Sigma_{k=1}^{n_i} p_k^i = P_{\alpha_i}^i$ (see Definition 2.4) denote the probability of the i-subgraph.

Lemma 3.1: A lower bound for the average number of tests required to locate a fault within the i-subgraph is given by

$$Q_i = -\sum_{k=1}^{n_i} \frac{p_k^i}{P_i} \log_2 \frac{p_k^i}{P_i}.$$

Proof: Let the fault f_k^i be represented by a set of m_k equivalent faults so that

$$\frac{p_k^i}{P_i} = \frac{m_k}{M} \qquad \text{where } M = \sum_{k=1}^{n_i} m_k.$$

Substituting in Q_i and rearranging terms yields

$$Q_i = \sum_{k=1}^{n_i} \frac{m_k}{M} \log_2 \frac{m_k}{M} = \log_2 M - \sum_{k=1}^{n_i} \frac{m_k}{M} \log_2 m_k.$$

The first term, $\log_2 M$, is the lower bound for the average number of tests required to locate a fault out of M equally probable faults [2], [10]. Note that if $\log_2 M$ is not an integer $\lceil \log_2 M \rceil$ should be used instead. However, for the sake of simplicity we shall use $\log_2 M$, and consequently the bound derived will not necessarily be the greatest lower bound. Using the same reasoning, $\log_2 m_k$ is a lower bound for the average number of tests required to locate a fault out of m_k faults which are in this case indistinguishable. Hence, the second term

$$\sum_{k=1}^{n_i} \frac{m_k}{M} \log_2 m_k$$

is the average of all n_i lower bounds. Therefore Q_i is the required lower bound of the average number of tests required to locate an equivalence class of faults out of all n_i equivalence classes. Q.E.D.

Assume now that the detection path is given. The average number of tests required to locate a fault out of the n_i faults in the i-subgraph satisfies

$$\sum_{f_j \in i\text{-subgraph}} l_j p_j \geq (i + Q_i) P_i.$$

Hence,

$$C = \sum_{j=1}^{q} l_j p_j \geq \sum_{i=1}^{m} (i + Q_i) P_i.$$

Denoting $C_b = \Sigma_{i=1}^{m} (i + Q_i) P_i$ and substituting Q_i yields

$$C_b = \sum_{i=1}^{m} P_i (i + \log_2 P_i) - \sum_{j=1}^{q} p_j \log_2 p_j. \qquad (3.1)$$

C_b is the lower bound for C if the detection path is given. In order to find the global lower bound we have to minimize C_b over all detection paths. Using Lagrange multipliers C_b is minimized for the detection path satisfying $P_i = A \cdot 2^{-i}$ $(i = 1,2, \cdots, m)$ where $A = 1/1 - 2^{-m}$, i.e., $P_1 = A/2, P_2 = A/4, \cdots, P_m = A/2^m$.

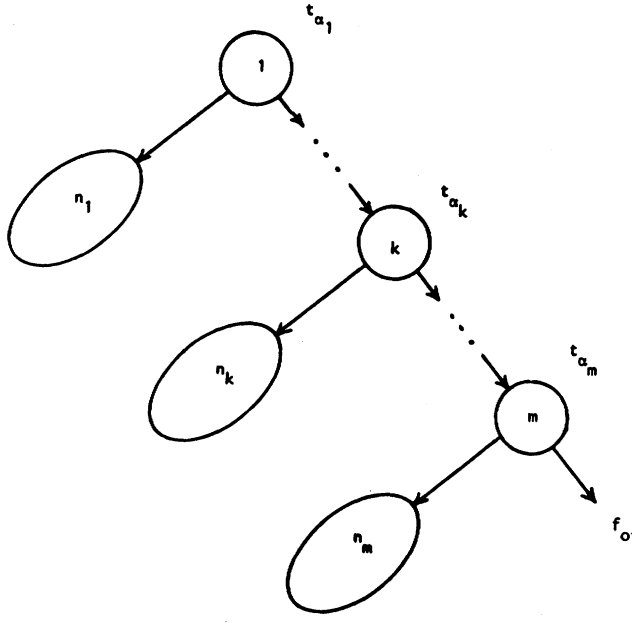


Fig. 2. General form of the SDT.

A better (i.e., higher) lower bound can be obtained by taking into account the fact that the faults in each subgraph cannot be chosen arbitrarily, but have to be covered by a test belonging either to T_0 or T_1. This partitioning of the faults is taken into account in the following minimization of C_b.

Assume first that $F^{0,1} = \phi$ and let $m_0(m_1)$ designate the minimal number of tests from $T_0(T_1)$ required to cover the faults from $F^0(F^1)$. Clearly, $m_0 + m_1 = m$.

We denote by $P^0(P^1)$ the probability of the subset $F^0(F^1)$. The order in which the $m_0 + m_1$ tests are applied can be described by a binary number $B = b_1, b_2, \cdots, b_m$ satisfying $\Sigma_{i=1}^{m} b_i = m_1$, where

$$b_i = \begin{cases} 0, \text{ if } t_{\alpha_i} \in T_0 \\ 1, \text{ if } t_{\alpha_i} \in T_1. \end{cases}$$

There are $m!/m_0!m_1!$ such binary numbers. We have to choose the one which enables the partitioning of P^0 and P^1 to m_0 and m_1 parts, respectively, so that the condition $P_i = A \cdot 2^{-i}$ is satisfied. Fortunately, there is a simple connection between P^1 and the desired number B:

$$P^1 = \sum_{i=1}^{m} b_i P_i.$$

Substituting P_i yields

$$P^1 = \sum_{i=1}^{m} b_i \cdot A \cdot 2^{-i} = A \cdot 2^{-m} \cdot \sum_{i=1}^{m} b_i 2^{m-i} = A \cdot 2^{-m} \cdot N_B$$

where N_B is the decimal value of the binary number B. Hence,

$$N_B = \frac{P^1 \cdot 2^m}{A} = P^1(2^m - 1). \tag{3.2}$$

The binary number B whose decimal value is N_B describes the order in which the tests from T_0 and T_1 will be applied in the detection path. Frequently, as it is shown in the following example, B does not satisfy the condition $\Sigma_{i=1}^{m} b_i = m_1$ and we have to find the two numbers B_1 and B_2 whose values are the closest to N_B and which satisfy $\Sigma_{i=1}^{m} b_i = m_1$. Since C_b is continuous and has a unique minimum, its minimum with the constraint $\Sigma_{i=1}^{m} b_i = m_1$ is achieved at one of these two numbers.

Example: If the *a priori* probabilities are chosen as suggested in this paper, it is clear that $P^0 = P^1 = \frac{1}{2}$ since for each line x_j one fault, say x_j s-a-α, is covered by T_0 and the other fault, x_j s-a-$\bar{\alpha}$, is covered by T_1 [7]. Using equation (3.2) we get $N_B = \frac{1}{2}(2^m - 1)$ which clearly is not an integer. The binary numbers B_1 and B_2 are

$$B_1 = 10^{m_0}1^{m_1-1} \quad B_2 = 01^{m_1}0^{m_0-1}.$$

If different probabilities are assigned to the faults, different binary numbers B_1 and B_2 result. However, it is clear that neither B_1 nor B_2 satisfy (3.2), hence, the condition $P_i = A \cdot 2^{-i}$ is not satisfied. We can overcome this difficulty, which is caused by the constraint on B, by determining the optimal values of the P_i's for a given number B. The function C_b using Lagrange multipliers becomes

$$\tilde{C}_b = \sum_{i=1}^{m} P_i(i + \log_2 P_i) + \lambda_0 \left(P^0 - \sum_{i=1}^{m} \bar{b}_i P_i\right) + \lambda_1 \left(P^1 - \sum_{i=1}^{m} b_i P_i\right).$$

This function is minimized at the point:

$$P_i = \begin{cases} A_0 \cdot 2^{-i}, & \text{if } b_i = 0 \\ A_1 \cdot 2^{-i}, & \text{if } b_i = 1 \end{cases}$$

where

$$A_1 = \frac{P^1 \cdot 2^m}{N_B}, \quad A_0 = \frac{P^0 \cdot 2^m}{(2^m - 1 - N_B)}$$

i.e.,

$$P_i = \left[\frac{P^1}{N_B} \cdot b_i + \frac{P^0}{2^m - 1 - N_B} \cdot \bar{b}_i\right] \cdot 2^{m-i}. \quad (3.3)$$

Substitution in C_b gives

$$C_b = m - \sum_{i=1}^{q} P_i \log_2 P_i + P^0 \log_2 P^0 + P^1 \log_2 P^1 - [P^1 \log_2 N_B + P^0 \log_2 (2^m - 1 - N_B)]. \quad (3.4)$$

The first four terms are constant, therefore we have to compute the value of $-[P^1 \log_2 N_B + P^0 \log_2 (2^m - 1 - N_B)]$ for B_1 and B_2 and choose the one which minimizes it.

Example: Suppose all q nonequivalent faults in the fan-out-free network in Fig. 1 are equally probable. In this case we have

$$F^0 = \{x_7 \text{ s-a-1}, x_8 \text{ s-a-1}, x_9 \text{ s-a-1}\},$$

$$F^1 = \{x_1 \text{ s-a-1}, x_2 \text{ s-a-1}, x_3 \text{ s-a-0}, x_4 \text{ s-a-1}, x_5 \text{ s-a-1}, x_6 \text{ s-a-0}, x_9 \text{ s-a-0}\}.$$

The corresponding probabilities are $P^0 = 0.3$, $P^1 = 0.7$. The minimal numbers of detection tests, which were determined by the subsequent algorithm, are $m_0 = 2$, $m_1 = 3$. Using (3.2) we get $N_B = P^1(2^m - 1) = 21.7$, hence, $B_1 = (10101)_2 = (21)_{10}$ and $B_2 = (10110)_2 = (22)_{10}$. The corresponding values of C_b are $C_b(B_1) = 3.369$ and $C_b(B_2) = 3.368$. The lower bound is therefore 3.368 compared to the lower bound which is computed without taking into account the partitioning of the faults $-\Sigma_{i=1}^{q} P_i \log_2 P_i = \log_2 q = 3.322$.

For general networks, usually $F^{0,1} \neq \phi$ and we therefore have three probabilities, namely, P^0, P^1, and $P^{0,1}$ and three minimal numbers of detection tests m_0, m_1, and $m_{0,1}$ where $m_{0,1} = m - m_0 - m_1$. In this case we have to find the binary number B satisfying $m_1 \leq \Sigma_{i=1}^{m} b_i \leq m_1 + m_{0,1}$ whose decimal value N_B satisfies $P^1(2^m - 1) \leq N_B \leq (P^1 + P^{0,1}) \cdot (2^m - 1)$ and which minimizes C_b.

IV. Generation of Decision Trees for Fan-out-Free Networks

Generating an SDT whose cost function equals the lower bound C_b requires the selection of detection tests with probabilities according to (3.3), hence, it is usually not feasible. The selection of tests to form the minimal feasible SDT requires a complete fault table and involves a great amount of computation. In this section we present a procedure for generating an SDT for fan-out-free networks (tree type networks), directly from the structure of the network. A similar procedure for networks with fan-out lines is now being developed and will be presented in a subsequent paper.

The procedure for fan-out-free networks generates a minimal detection path in the SDT using the following algorithm.

Algorithm 1

Step 1: Set $k = 1$.

Step 2: Choose at node k a maximal test whose probability $P_j^{(k)}$ is maximal at this node.

Step 3: $k = k + 1$.

Step 4: If $F^{(k)} \neq \phi$ go to step 2.

Theorem 4.1: Algorithm 1 generates a minimal detection set for fan-out-free networks.

Proof: This algorithm is a modification of Procedure 2 presented by Berger and Kohavi [7] and proved there to yield a minimal detection set.

According to [7] the detection set is minimal if in Step 2 any maximal test at node k is selected. Step 2 is modified in this paper in order to select tests with probabilities close to those required by (3.3). Such a selection will minimize the generated SDT. Furthermore, it is shown subsequently that Step 2 generates the tests directly without using a fault table and its application requires a small amount of computation.

We proceed now to introduce a method for generating the tests within each diagnosis subgraph. The k-subgraph contains diagnosis tests to distinguish among the n_k faults with total probability P_k. The tests are selected according to a weighting function $V(t)$, which is equal to the sum of the probabilities of the faults covered by the test t out of n_k faults in the k-subgraph. To achieve local optimization, we select as the first test in the k-subgraph the test t_{β_k} for which $V(t_{\beta_k})$ is the closest to $\frac{1}{2}P_K$ [1]. Application of t_{β_k} divides the n_k faults into two subsets—n_k^f faults covered by t_{β_k} and $n_k^p = n_k - n_k^f$ faults not covered by t_{β_k}. These subsets form new diagnosis subgraphs and each of them is treated in the same manner as shown in Fig. 3. For each subset a new weighting function is obtained according to which new tests are selected.

We introduce now the method used to produce the weighting functions and start by examining the conditions under which a fault is detected. A fault f_i is covered by a test t_j if the subnetwork sensitized by t_j contains a sensitized path SP_i from f_i to the primary output. This sensitized path is generated by assigning "enable" values to some lines in the network (not included in the path) [1]. We call these lines control lines and denote them by y_j, $j = 1,2,\cdots,r-1$. The "enable" value for a line x_j (which is a control line y_j for some SP_i) is given by g_j where g_j is the bit in the gate table corresponding to x_j as an input line to some gate. Let y_j equal 1 iff $x_j = g_j$, i.e., $y_j = x_j^{g_j}$. The fault f_i is covered by a test if all control lines along the sensitized path SP_i are equal to 1, i.e., $\Pi_{y_j \in SP_i} y_j = 1$.

Example: In Fig. 1, the sensitized path corresponding to the fault x_1 s-a-α($\alpha = 0,1$) is 1,6,8,9. The control lines for

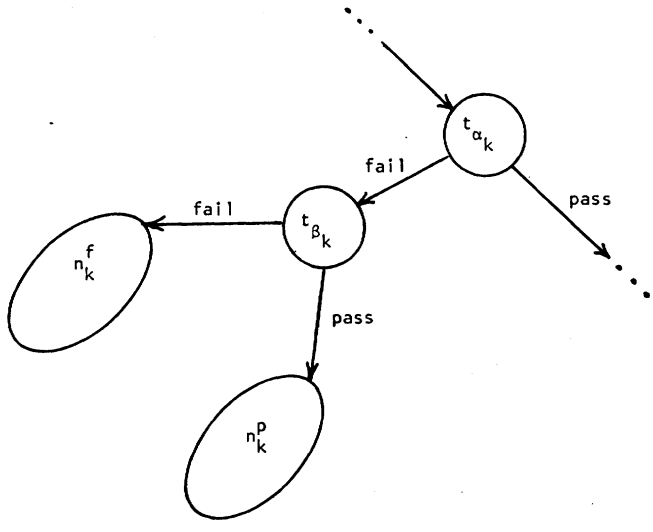


Fig. 3. Diagnosis subgraph in the SDT.

this sensitized path are y_2, y_3, and y_7 where $y_2 = x_2$,$y_3 = \bar{x}_3$ and $y_7 = x_7 = (\overline{x_4 \cdot x_5})$. The fault x_1 s-a-α is detected by any test satisfying $x_1 = \bar{\alpha}$ and $y_2 \cdot y_3 \cdot y_7 = 1$, i.e., $x_2 \cdot \bar{x}_3 \cdot (\overline{x_4 \cdot x_5}) = 1$. The control lines for the fault x_6 s-a-β are y_3 and y_7, i.e., any test satisfying $x_6 = \bar{\beta}$ and $y_3 \cdot y_7 = 1$ detects this fault.

The weighting function $V_k(t)$ evaluating the weight of a test t at node k is defined as follows:

$$V_k(t) = \sum_{i=1}^{n_k} p_i^k \prod_{y_j \in SP_i} y_j,$$

clearly,

$$V_k(t_{\alpha_k}) = \sum_{i=1}^{n_k} p_i^k = P_k.$$

The test t_{β_k} is constructed by assigning values to the control lines corresponding to the n_k faults so that $V_k(t_{\beta_k})$ is as close to $\frac{1}{2}P_k$ as possible. The values of the undetermined lines in the network remain the same as in the detection test t_{α_k}.

The weighting functions $V_k^f(t)$ and $V_k^p(t)$ for the two new subgraphs including n_k^f and n_k^p faults, respectively, are obtained in the following way:

$$V_k^f(t) = \sum_{i=1}^{n_k} p_i^k \prod_{y_j \in SP_i} y_j^f$$

where

$$y_j^f = \begin{cases} 0, & \text{if } y_j = 0 \text{ in } t_{\beta_k} \\ y_j, & \text{otherwise.} \end{cases}$$

This substitution eliminates the probabilities of all the faults not covered by t_{β_k}, leaving the probabilities of the faults covered by t_{β_k}. $V_k^p(t)$ is obtained by $V_k^p(t) = V_k(t) - V_k^f(t)$.

The method described above yields a nearly minimal average number of tests required to locate a fault within the k-subgraph. In the special case where the probabilities of all q faults are equal, a minimal value is achieved as shown in the following theorem.

Theorem 4.2: If all nonequivalent faults in a fan-out-free network are equally probable, the procedure above yields a minimal average number of tests required to locate a fault within the diagnosis subgraph.

Proof: A minimal average number of tests within the diagnosis k-subgraph is achieved if for any subset of n_i faults out of the n_k faults a test can be found which divides these n_i faults into two subsets with $n_i/2$ faults in each for n_i even and $(n_i + 1)/2$, $(n_i - 1)/2$ faults for n_i odd [1], [10]. The existence of such a test is proved in the following way. The n_k faults are included in the subnetwork sensitized by the detection test t_{α_k}. For every gate included in the subnetwork, either all its input lines are sensitized or only one input line is sensitized. In the first case, the faults at the input lines and at the output line are equivalent and cannot be distinguished. In the second case, another input line to that gate can serve as a control input y. By assigning $y = 0$ we delete from the sensitized subnetwork the faults in the subtree feeding this gate. Between two adjacent control inputs along the sensitized subnetwork only equivalent faults can exist. By proper assignment of the control inputs any number of faults out of the n_k faults can be deleted from the subnetwork. This assignment generates a new test which distinguishes between the faults deleted from the subnetwork and the remaining ones. Consequently, the required tests for obtaining the minimal diagnosis within the k-subgraph can be found. Q.E.D.

Example: The sensitized subnetwork by the detection test $t_\alpha = 11001$ includes the lines 1,2,3,6,8, and 9. The five faults $\{x_1$ s-a-0, x_2 s-a-0, x_3 s-a-1, x_6 s-a-1, x_8 s-a-1$\}$ are equivalent and can be distinguished from the fault $\{x_9$ s-a-1$\}$ by proper assignment of the control input $y_7 = x_7^1$, i.e., $x_7 = 0$. The distinguishing test is $t_\beta = 11011$.

V. The Algorithm in Detail

Algorithm 1 for generating the detection path and the procedure introduced previously for generating the diagnosis subgraphs are summarized in the following algorithm.

Algorithm 2

Step 1: Set $k = 1$.

Step 2: Generate the test $t_{\alpha k}$ whose probability at node k of the detection path is maximal, and denote this probability P_k.

Step 3: Specify the n_k faults included in the k-subgraph.

Step 4: Compute the weighting function $V_k(t)$ at node k.

Step 5: Generate the test $t_{\beta k}$ for which $V_k(t_{\beta k})$ is the closest to $\frac{1}{2}P_k$.

Step 6: Compute the weighting functions $V_k^f(t)$ and $V_k^p(t)$.

Step 7: Repeat steps 5 and 6 until the diagnosis subgraph for the n_k faults is complete.

Step 8: Set $k = k + 1$.

Step 9: If $F^{(k)} \neq \phi$ return to Step 2.

The generation of the test $t_{\alpha k}$ in Step 2 is accomplished using accumulating weights which are defined below. To each line x_i we assign a pair of accumulating weights (a_i^0, a_i^1).

Definition 5.1: The accumulating weight a_i^α, $(\alpha = 0,1)$ is the weight of the test for the subnetwork feeding x_i which covers the fault x_i s-a-α and whose weight is maximal.

These accumulating weights are obtained in the following way. Assign initially the pair (d,d) to every input line x_i, $(i = 1,2,\cdots,n)$ where $d = \frac{1}{2r}$. After selecting a detection test which covers the fault x_i s-a-α change a_i^α to 0. Compute the accumulating weights for the remaining lines according to the following rules.

Let x_i and x_j be two inputs to a gate whose output is x_k, $(i < j < k)$ and let $g_i g_j g_k$ be the corresponding row in the gate table.

(i) $a_k^{g_k} = a_i^{g_i} + a_j^{g_j} + d \cdot \delta_{st}$

(ii) $a_k^{g_k} = \max\{a_i^{g_i}, a_j^{g_j}\} + d \cdot \delta_{st}$

where

$$\delta_{st} = \begin{cases} 1, & \text{for the first test covering this fault} \\ 0, & \text{for later tests.} \end{cases}$$

The faults x_k s-a-$\bar{g}_k$, x_i s-a-$\bar{g}_i$ and x_j s-a-$\bar{g}_j$ are equivalent, therefore any test covering one of them covers the others as well. This justifies rule (i) which accumulates the weights of all equivalent faults. Rule (ii) is justified by the following argument: the fault x_k s-a-g_k can be covered by a test which covers also the fault x_i s-a-g_i or the fault x_j s-a-g_j but not both, therefore we select the maximal weight between $a_i^{g_i}$ and $a_j^{g_j}$.

Example: For the tree network in Fig. 1 the initial accumulating weights are

$$a_i^0 = a_i^1 = d \qquad \text{for } i = 1,2,\cdots,5$$

$$a_6^0 = \max\{a_1^1, a_2^1\} + d = 2d; \quad a_6^1 = a_1^0 + a_2^0 + d = 3d$$

$$a_7^0 = \max\{a_4^1, a_5^1\} + d = 2d; \quad a_7^1 = a_4^0 + a_5^0 + d = 3d$$

$$a_8^0 = \max\{a_3^0, a_6^0\} + d = 3d; \quad a_8^1 = a_3^1 + a_6^1 + d = 5d$$

$$a_9^0 = a_7^0 + a_8^0 + d = 6d; \quad a_9^1 = \max\{a_7^1, a_8^1\} + d = 6d.$$

If weights different from those defined in Definition 2.1 are assigned to the faults, the appropriate accumulating weights are obtained as follows. Denote by ω_k^0, ω_k^1 the weights of the faults x_k s-a-0 and x_k s-a-1, respectively, hence,

$$a_i^\alpha = \omega_i^\alpha \ (\alpha = 0,1) \qquad i = 1,2,\cdots,n.$$

(i) $a_k^{g_k} = a_i^{g_i} + a_j^{g_j} + \omega_k^{g_k} \cdot \delta_{st}$.

(ii) $a_k^{g_k} = \max\{a_i^{g_i}, a_j^{g_j}\} + \omega_k^{g_k} \cdot \delta_{st}$.

The following theorem is a straightforward extension of Definition 5.1.

Theorem 5.1: The maximal accumulating weight for the output line x_r at node k, is the weight P_k of the maximal detection test $t_{\alpha k}$.

We denote

$$a_r^\lambda = \max\{a_r^0, a_r^1\}$$

and proceed to generate the maximal test whose weight is a_r^λ using backward tracing from the primary output to the primary inputs. In order to perform this backward tracing we define a diagnosis vector, abbreviated DV, whose r components correspond to the r lines in the network as follows:

$$DV_i = \begin{cases} S^\alpha, & \text{if line } i \text{ is sensitive to the fault s-a-}\alpha \ (\alpha = 0,1). \\ E^\alpha, & \text{if line } i \text{ is a control line with } \alpha \text{ as the "enable" value.} \\ \alpha, & \text{if line } i \text{ has a fixed value } \alpha. \end{cases}$$

This vector is generated in the following way. We set first $DV_r = S^\lambda$ since the primary output is sensitized by each test. The rules of the backward tracing from an output line x_k of a gate to its input lines x_i and x_j are summarized in Table II. Note that in this table

$$X = S \text{ and } Y = E \qquad \text{if } a_i^{g_i} = \max\{a_i^{g_i}, a_j^{g_j}\},$$

$$X = E \text{ and } Y = S \qquad \text{if } a_j^{g_j} = \max\{a_i^{g_i}, a_j^{g_j}\}.$$

The test $t_{\alpha k}$ is determined from DV simply by substituting 0 for every input line x_i for which $DV_i = 0, E^0$ or S^1, and substituting 1 for every input line for which $DV_i = 1, E^1$ or S^0.

TABLE II
Backward Tracing of the Diagnosis Vector

	$\lambda = g_k$	$\lambda = \bar{g}_k$
$DV_k = S^\lambda$	$DV_i = X^{g_i}$, $DV_j = Y^{g_j}$	$DV_i = S^{\bar{g}_i}$, $DV_j = S^{\bar{g}_j}$
$DV_k = E^\lambda$	$DV_i = E^{g_i}$, $DV_j = g_j$	$DV_i = E^{\bar{g}_i}$, $DV_j = g_j$
$DV_k = \lambda$	$DV_i = g_i$, $DV_j = g_j$	$DV_i = \bar{g}_i$, $DV_j = g_j$

In Step 3 of Algorithm 2 we specify the faults in the k-subgraph by eliminating from DV all faults included in the $k-1$ previous subgraphs. These previous faults are accumulated in a fault vector, abbreviated FV, whose r components are defined in the following way:

$$FV_i = \begin{cases} 0, & \text{if no fault in line } x_i \text{ has been covered yet.} \\ F^\alpha, & \text{if the fault } x_i \text{ s-a-}\alpha\ (\alpha = 0,1) \text{ has already been covered.} \\ 1, & \text{if all faults in line } x_i \text{ have already been covered.} \end{cases}$$

The initial value of FV is $FV_i = 0$, $i = 1,2,\cdots,r$. For each subgraph FV is modified by adding the faults covered by this subgraph. The final value of FV indicating the end of the algorithm (Step 9) is $FV_i = 1$, $i = 1,2,\cdots,r$.

In Step 3 we eliminate from DV the faults which have already been covered, by intersecting DV with FV, obtaining a new DV. The rules of this f-intersection are given in Table III.

This intersection eliminates from DV all faults included in the part of the SDT generated previously.

In Step 4 of the algorithm the weighting function $V_k(t)$ is generated using DV. The weights of the equivalent faults are summed up and multiplied by the appropriate control line E^α.

Example: For the network in Fig. 1, the DV corresponding to $a_9^0 = 6d$ is $(E^1,S^1,E^0,E^1,S^1,S^0,S^0,S^0,S^0)$. The first detection test is therefore $t_{\alpha_1} = (10010)$. The weighting function is $V_1(t) = (d \cdot E_1^1 + d)E_3^0 + d \cdot E_4^1 + 3d$, i.e., there are four nonequivalent faults which are covered by t_{α_1}: $\{x_2$ s-a-1$\}$,$\{x_5$ s-a-1$\}$,$\{x_6$ s-a-0$\}$ with weight d each, and the equivalence class $\{x_7$ s-a-0, x_8 s-a-0, x_9 s-a-0$\}$, with weight $3d$.

After computing $V_k(t)$ from DV, FV is modified by union operation with DV, obtaining a new FV. The rules of this f-union are given in Table III.

The test t_{β_k} is generated from $V_k(t)$ by assigning values to the control lines to obtain a weight as close to $\frac{1}{2}a_r^\lambda$ as possible. Once the test t_{β_k} is generated, $V_k^f(t)$ and $V_k^p(t)$ are computed and serve as weighting functions for generating the next tests.

After completing the diagnosis subgraph emanating from node k of the detection path, new accumulating weights are computed for generating a new detection test. The previous accumulating weights are modified in the following way. For every input line x_i whose corresponding DV_i is equal to S^α, modify a_i^α to 0. For the remaining lines use equations (i) and (ii) without the addition of $d \cdot \delta_{st}$ changing only one of each pair of accumulating weights.

Example: For the network in Fig. 1, the weighting function is $V_1(t) = (d \cdot E_1^1 + d)E_3^0 + d \cdot E_4^1 + 3d$. We set $E_3^0 = 0$ and $E_4^1 = 0$ (i.e., $x_3 = 1$ and $x_4 = 0$) in order to get a test $t_{\beta_1} = 10100$ whose weight is $\frac{1}{2}a_9^0 = 3d$. The generated weighting functions are $V_1^f(t) = 3d$ and $V_1^p(t) = (dE_1^1 + d)E_3^0 + dE_4^1$. Using $V_1^p(t)$, the second distinguishing test $t_{\gamma_1} = 10110$ is generated by setting $E_3^0 = 0$ and $E_4^1 = 1$, i.e., $x_3 = 1$ and $x_4 = 1$. After completion of the first diagnosis subgraph the accumulating weights are modified and the second detection test $t_{\alpha_2} = 11001$ is generated. The final SDT for this example is shown in Fig. 4. This SDT is minimal and the weighted average number of tests required to locate a fault in it is 3.27. The lower bound for this example is $C_b = 3.05$.

VI. Summary

The problem considered in this paper is sequential fault diagnosis in combinational networks. Since the possible faults in the network may occur at different frequencies, a model is suggested in which different probabilities of occurrence can be assigned to the different faults. In order to locate these faults, an SDT is generated directly from the structure of the network without using a fault table. The structure of the network is presented in a simple tabular form called the gate table.

Two different objectives in designing an SDT were considered in this paper and hence a new definition of a minimal decision tree was presented. An appropriate lower bound for the cost function of this SDT is derived, thus enabling adequate evaluation of a generated decision tree.

In the last part of the paper an explicit algorithm for generating an SDT is presented. This algorithm is restricted to fan-out-free networks, although the concept of generating distinguishing tests while the backward tracing operation is performed can be used for general networks as well.

For general networks, however, the generation of a minimal SDT directly from the structure of the network

TABLE III
f-Intersection and *f*-Union

$\cap_f$	0	F^α	1
S^α	S^α	$\bar{\alpha}$	$\bar{\alpha}$
$S^{\bar{\alpha}}$	$S^{\bar{\alpha}}$	$S^{\bar{\alpha}}$	α
x	x	x	x

$\cup_f$	S^α	x
0	F^α	0
F^α	F^α	F^α
$F^{\bar{\alpha}}$	1	$F^{\bar{\alpha}}$
1	1	1

$a = 0,1$

$x \neq S^\alpha$

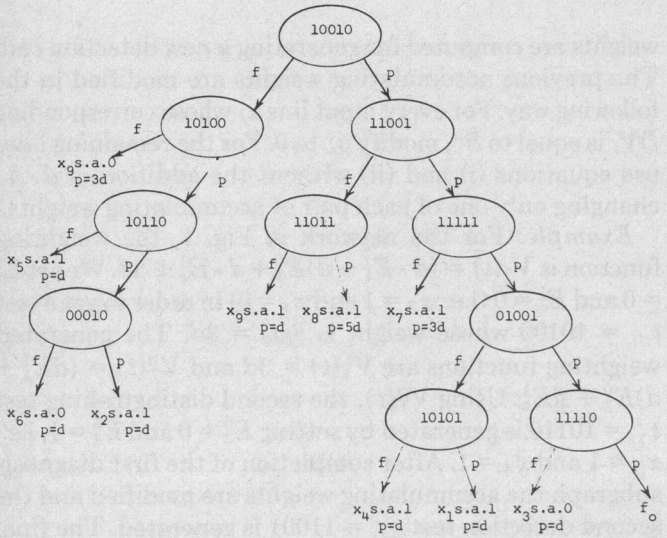


Fig. 4. SDT for the network in Fig. 1.

becomes more complicated especially because the existence of reconverging fan-out lines causes the functional equivalence between faults to be different from the structural equivalence [6]. This in turn results in the fact that the generated SDT by an algorithm employing the method outlined in this paper is not necessarily minimal.

The procedure for networks with fan-out can be described in general terms as follows. Assume part of the SDT has already been generated; the next step is to select a fault which was not covered by the previous tests. A detection test for this fault is next generated. For this detection test a backward tracing is then performed starting from the primary output line towards the primary input lines. There are two objectives for this operation; the first is specifying the faults covered by the detection test and the second is to simultaneously generate additional tests in order to distinguish between these faults.

The use of backward tracing for specifying the faults covered by a given test is preferable to forward simulation techniques (parallel or deductive simulation). While in the forward simulation *all paths* in the network emanating from the primary input lines are checked, in the backward tracing operation *only the sensitive paths* are checked. Hence, the required computation time is considerably smaller.

References

[1] H. Y. Chang, E. G. Manning, and G. Metze, *Fault Diagnosis of Digital Systems.* New York: Wiley, 1970.

[2] W. H. Kautz, "Fault testing and diagnosis in combinational digital circuits," *IEEE Trans. Comput.*, vol. C-17, Apr. 1968.

[3] I. Koren, "Sequential diagnosis of digital systems," presented at the IEEE Proc. 8th Conv., Israel, vol. VIII, 1973.

[4] S. Y. H. Su and Y. C. Cho, "A new approach to the fault location of combinational circuits," *IEEE Trans. Comput.*, vol. C-21, Jan. 1972.

[5] Y. Koga and F. Hirata, "Fault-locating test generation for combinational logic networks," presented at the Int. Symp. on Fault-Tolerant Computing, June 1972.

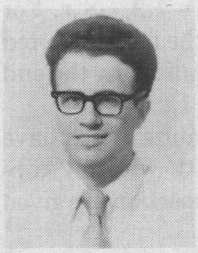
[6] E. J. McCluskey and F. W. Clegg, "Fault equivalence in combinational logic networks," *IEEE Trans. Comput.*, vol. C-20, Nov. 1971.

[7] I. Berger and Z. Kohavi, "Fault detection in fanout-free combinational networks," *IEEE Trans. Comput.*, vol. C-22, pp. 908–914, Oct. 1973.

[8] H. Y. Chang, "A distinguishability criterion for selecting efficient diagnostic tests," in *Spring Joint Computer Conf. Proc., AFIPS Conf. Proc.* Washington, DC: Spartan, 1968.

[9] T. J. Powell, "A procedure for selecting diagnostic tests," *IEEE Trans. Comput.*, vol. C-18, Feb. 1969.

[10] D. E. Knuth, *Fundamental Algorithms.* Reading, MA: Addison–Wesley, 1968.



Israel Koren (S'72–M'76) was born on June 23. 1945. He received the B.Sc. (cum laude), M.Sc., and D.Sc. degrees all in electrical engineering from the Technion–Israel Institute of Technology, Haifa, Israel, in 1967, 1970 and 1975, respectively.

From 1968 to 1971 he was with the Computer Center, Israel Ministry of Defense. In 1972 he joined the Department of Electrical Engineering, Technion–Israel Institute of Technology, where he became a Lecturer in 1975. He was also a Consultant to the local computer industry involved in R & D projects; mainly computer interfacing and data communication terminals. He is now with the Department of Electrical Engineering and Computer Science, University of California, Santa Barbara. His research interests include switching theory, finite automata, fault-tolerant computing, computer architecture, and related subjects.

Zvi Kohavi (S'62–M'71) was born in Haifa, Israel, on December 23, 1936. He received the B.Sc. degree from the Technion–Israel Institute of Technology, Haifa, Israel, in 1960 and the M.Sc. and Ph.D. degrees from the Polytechnic Institute of New York, Brooklyn, NY, in 1962 and 1966, respectively, all in electrical engineering.

From 1966 to 1969 he was an Assistant Professor in the Department of Electrical Engineering and Project MAC, Massachusetts Institute of Technology, Cambridge. In 1969 he joined the Department of Electrical Engineering and Computer Science at the Technion. He spent the 1975 academic year as a Visiting Professor with the Department of Computer Science, University of Utah, Salt Lake City. He has done research on switching theory, finite-state machines, automata theory, and fault detection. He is the author of *Switching and Finite Automata Theory* (New York: McGraw-Hill, 1970), and an Editor (with A. Paz) of *Theory of Machines and Computations* (New York: Academic, 1971).