Digital Processes, 2 (1976) 209-223

ADAPTIVE FAULT-LOCATING TESTS FOR DIGITAL SYSTEMS*

I. Koren and Z. Kohavi
Department of Electrical Engineering
Technion – Israel Institute of Technology, Haifa, Israel.

(Received February, 26, 1976; in revised form, June 8, 1976)

Abstract

The problem of generating decision trees for adaptive fault diagnosis of combinational digital networks is considered in this paper. The generated decision trees are based on a probabilistic model in which every fault is assigned an *a priori* probability of occurrence. A procedure for generating minimal decision trees is developed, using dynamic programming. For large networks, however, this procedure requires a large amount of computation. To overcome this limitation an alternative approach for generating decision trees is developed, using the information gain weighting function. The number of computations required in the latter procedure is significantly smaller. However, the generated sets of tests are not always minimal.

In the last section, the second procedure is extended to cover the case of module-level diagnosis in which we have only to distinguish between faults in different modules.

Keywords

Fault locating, adaptive decision trees, *a priori* probability of occurrence, dynamic programming, weighting function, module-level diagnosis.

1. Introduction

High reliability is required for many present day uses of digital systems. One important way of maintaining high reliability is through verifying the proper operation of the system and locating any occurring fault. This paper is concerned with the problem of adaptive fault diagnosis in combinational systems. The system under discussion is a combinational logic network with n primary inputs $x_1, x_2, \ldots, x_n$ and ν primary outputs $y_1, y_2, \ldots, y_\nu$, where $y_i = y_i(x_1, x_2, \ldots, x_n)$, $i = 1, 2, \ldots, \nu$. There are r lines in the network $x_1, x_2, \ldots, x_r$. The first n lines are the primary input lines, the last ν lines are the primary output lines. The rest are internal lines to which we have no access when checking the network.

We assume that the possible faults in the network are permanent logical faults of the stuck-at type, i.e., any line x_i in the network ($i = 1, 2, \ldots, r$) can be stuck-at-one (abbreviated x_i s-a-1), or stuck-at-zero (x_i s-a-0). Another assumption is that at any time only one fault will exist in the network. This single-fault assumption is reasonable when the time between network checks is short relative to the interarrival time of faults, thus making the probability of multiple faults negligible. Under these

*A preliminary version of this paper was presented at the 8th Convention of IEEE in Israel in 1973 [2].

assumptions $2r$ different faults are possible. Some of these faults are indistinguishable, i.e., they are equivalent [3]. We say that two faults f_1 and f_2 are equivalent if the output function realized by the network in the presence of fault f_1 is identically the same function as that realized by the network in the presence of f_2. This definition defines an equivalence relation which 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 fault-free network.

In most previous papers on the subject no *a priori* probabilities of occurrence were assigned to the various faults. This is in fact equivalent to an assignment of equal probabilities of occurrence to all $q + 1$ faults. However, since in practice different faults occur at different frequencies, we shall assign a different *a priori* probability to each of the $q + 1$ distinguishable faults. The assignment of such probabilities can be based on manufacturer supplied information, statistical data and so on. A possible assignment which can serve as a first approximation has already been suggested [4]. According to this suggestion the *a priori* probability p_i assigned to the fault f_i $(i = 1, 2, \ldots, q)$ is proportional to the number of elements in the ith equivalence class. The probability $\sum_{i=1}^{q} p_i = 1 - p_0$ is the *a priori* probability of the occurrence of any possible fault in the network. Once p_0 is determined using statistical considerations, the q *a priori* probabilities can be determined.

Our objective is to generate a minimal set of tests to detect the presence of a fault in the network and to locate and identify such a fault; that is, to determine which fault among the q possibilities $f_1, f_2, \ldots, f_q$ in fact occured.

The required set of tests $T = \{t_1, t_2, \ldots, t_N\}$ where t_i is an input combination, can be either fixed (preset) or sequential (adaptive). In the first case (fixed diagnosis), the test applied to the network at the ith stage of the diagnosis does not depend in any way upon the outcome of the $i - 1$ previous tests. In the second case (sequential diagnosis), a sequential decision tree is constructed in which the selection of the test to be applied at the ith vertex of the decision tree depends upon the outcome of the previous tests.

For fixed diagnosis the cost function is the number of tests required (N). For sequential diagnosis, the cost function is the weighted average number of tests required to locate a fault $C = \sum_{i=0}^{q} \ell_i p_i$, where ℓ_i is the number of tests required to locate the fault f_i in the decision tree. The average number of tests, C, for sequential diagnosis is always smaller than, or equal to, the number of tests, N, for fixed diagnosis [1, 5].

Since we need at most one test per possible fault in preset diagnosis and at least $\lceil \log_a(q + 1) \rceil$ tests for adaptive diagnosis [5, 10], we arrive at the following inequality:

$$\lceil \log_a(q + 1) \rceil \leq C \leq N \leq q,$$

where $a = 2^v$ (v is the number of primary outputs) and $\lceil x \rceil$ denotes the least integer which is greater than, or equal to, x.

Since sequential diagnosis usually requires fewer tests to locate those faults with higher probabilities of occurrence, we shall concentrate on generating these types of tests.

2. Minimal sequential diagnosis

Most existing methods for generating decision trees for sequential diagnosis start with a fault table containing m rows and q columns [1, 5-8]. Row i of the table corresponds to a possible test t_i ($i = 1, 2, \ldots, m$) and column j corresponds to a possible fault f_j ($j = 1, 2, \ldots, q$). The (i,j) entry in the table is the binary vector $(z_1, z_2, \ldots, z_v)$, where $z_k = 1$, iff when t_i is applied the output y_k of the network with fault f_j is different from the output y_k of the fault-free network. In such a case, when $z_k = 1$, the fault f_j is said to be *covered* (or *detected*) by test t_i. If all the elements of this vector equal zero, the fault f_j is not covered by the test t_i.

Our objective is to use the fault table for the selection of a set of tests that constitute an adaptive decision tree while minimizing the cost function C. Such a set of tests can be determined by exhaustive comparison of all possible selections with repetitions of the q tests which are required for generating a decision tree out of the m given tests in the fault table [8]. However, such a procedure requires a very long computation time even for fairly small tables.

In order to reduce the required number of computations for generating the decision tree, we shall employ the dynamic programming method whose computational efficiency is greater compared to the direct enumeration method. To this end we now define an objective function and obtain recurrence relations which will be used to compute recursively this objective function. We shall restrict ourselves to networks with only one output, i.e., $v = 1$. The extension to multiple output is straightforward, but the appropriate expressions are lengthy and are therefore omitted.

Consider a decision tree of the type shown in Figure 2. Suppose that a number of tests have already been applied, and consequently a number of possible faults have been distinguished. Let $f_{\alpha_1}, f_{\alpha_2}, \ldots, f_{\alpha_k}$ be any subset of k faults out of the $q+1$ non equivalent faults which remained to be distinguished at vertex α. Let ℓ_{α_i} be the number of tests required to locate fault f_{α_i} out of the k faults. We define the following objective function :

$$g_k(f_{\alpha_1}, f_{\alpha_2}, \ldots, f_{\alpha_k}) = \min \sum_{i=1}^{k} \ell_{\alpha_i} p_{\alpha_i} .$$

The minimization is made over all m tests given in the fault table. This objective function is computed recursively in the following way :

$$g_1(f_{\alpha_1}) = 0,$$

$$g_2(f_{\alpha_1}, f_{\alpha_2}) = p_{\alpha_1} + p_{\alpha_2},$$

$$g_3(f_{\alpha_1}, f_{\alpha_2}, f_{\alpha_3}) = p_{\alpha_1} + p_{\alpha_2} + p_{\alpha_3} + \min_{t} \; g_2(f_{\beta_1}, f_{\beta_2}) \quad ,$$

where f_{β_1}, f_{β_2} are any two faults out of $f_{\alpha_1}, f_{\alpha_2}, f_{\alpha_3}$ and t is a test in the given fault table which distinguishes between f_{β_1}, f_{β_2} and the third fault.

In general

$$g_k\left(f_{\alpha_1}, f_{\alpha_2}, \ldots, f_{\alpha_k}\right) = \sum_{i=1}^{k} p_{\alpha_i} + \min_t \left\{ g_{n_t}\left(f_{\beta_1}, f_{\beta_2}, \ldots, f_{\beta_{n_t}}\right) + g_{k-n_t}\left(f_{\beta_{n_t+1}}, \ldots, f_{\beta_k}\right) \right\},$$

where $f_{\beta_1}, f_{\beta_2}, \ldots, f_{\beta_{n_t}}$ are the n_t faults out of the k given faults which are detected by test t, while $f_{\beta_{n_t+1}}, \ldots, f_{\beta_k}$ are the $k - n_t$ faults which are not detected by t. The test t (out of the m given tests) which minimizes the function g_k is denoted by $t_k(f_{\alpha_1}, f_{\alpha_2}, \ldots, f_{\alpha_k})$. The function g_k is computed for all $\binom{q+1}{k}$ sub-sets of k faults out of the $q+1$ faults and the corresponding optimal test t_k is found. This process is performed for $k = 2, 3, \ldots, q+1$ yielding finally:

$$g_{q+1}(f_0, f_1, \ldots, f_q) = \min_t \sum_{i=0}^{q} \ell_i p_i .$$

The optimal test t_{q+1} is the first test in the decision tree. Application of t_{q+1} partitions the $q+1$ faults into two sub-sets for each of which we have previously found the optimal tests. Each of the two sub-sets is divided again into two new sub-sets until the final minimal decision tree is constructed. Note that by using dynamic programming we do not have to compare all possible decision trees which can be constructed out of the m given tests, in contrast to the method suggested by Brulé *et al.* [8].

To evaluate the efficiency of this method we estimate the number of computations and the size of computer memory required. The computation of each g_k requires comparison of the m given tests; hence the estimated number of computations is:

$$N_{\text{D.P.}} = \sum_{k=2}^{q+1} \binom{q+1}{k} \cdot m = m(2^{q+1} - q - 2).$$

The size of the computer memory required for g_k and t_k is:

$$M_{\text{D.P.}} = 2 \sum_{k=2}^{q+1} \binom{q+1}{k} = 2(2^{q+1} - q - 2).$$

For large networks in which the number of possible faults q is great, $N_{\text{D.P.}}$ and $M_{\text{D.P.}}$ increase rapidly and the dynamic programming method, although more efficient than other known methods, becomes also inefficient. It is important, therefore, to find procedures for generating decision trees for sequential diagnosis which require a smaller number of computations and a smaller size of computer memory. As we shall show in the following section, we can find such procedures which, unfortunately, do not always yield minimal sets of tests, although they are nearly minimal in most cases.

3. Nearly minimal sequential diagnosis

We shall now present a procedure for generating a decision tree by performing local optimization. The decision tree will be constructed starting at its first vertex. At each vertex of the tree one test out of the m given tests will be selected according to a weighting function measuring the ability of the test to distinguish among the faults. The test which maximizes the weighting function is selected as the "best" test at that vertex.

Three different weighting functions were suggested for generating nearly minimal decision trees:

1. The distinguishability criterion due to Chang [6]. This weighting function enumerates the number of pairs of faults distinguished by a test.

2. Powell's weighting function [7], which calculates the probability that a single succeeding test would complete the diagnosis.

3. The information gain function which was first proposed by Brulé *et al.* [8]. This weighting function measures the amount of uncertainty removed by a test when the initial uncertainty is:

$$H_0 = -\sum_{i=0}^{q} p_i \log_2 p_i .$$

Since only the last weighting function enables assignment of different probabilities of occurrence to the different faults, it is the only one suitable for our probabilistic model. One of the main advantages for using the former two weighting functions [6, 7] was their applicability to module-level diagnosis, i.e., diagnosis of a faulty module rather than the identification of the particular fault. We shall, however, show in the next section that the information gain function can also be easily generalized to include module-level diagnosis. Moreover, it is important to note that the information gain weighting function is closely related to the fault diagnosis problem. Specifically, we shall show that the uncertainty at vertex α of the decision tree equals the lower bound of the average number of additional tests required to locate a fault from this vertex until the complete identification of the fault is obtained. Hence, minimization of this uncertainty implies the minimization of the lower bound and as a consequence the actual average number of tests required will in general be reduced.

This will now be proved for the initial vertex as follows:

Let the fault f_i be represented by a set of m_i equivalent faults so that:

$$p_i = \frac{m_i}{M} \quad \text{where} \quad M = \sum_{i=0}^{q} m_i .$$

Substituting in H_0 and rearranging terms yields:

$$H_0 = -\sum_{i=0}^{q} \frac{m_i}{M} \log_2 \frac{m_i}{M} = \log_2 M - \sum_{i=0}^{q} \frac{m_i}{M} \log_2 m_i .$$

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 [5]. 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 will use $\log_2 M$ and consequently the bound is not necessarily the greatest lower bound. Using the same reasoning, $\log_2 m_i$ is a lower bound for the average number of tests required to locate a fault out of m_i faults which are in this case indistinguishable. Hence the second term,

$$\sum_{i=0}^{q} \frac{m_i}{M} \log_2 m_i,$$

is the average of all $q+1$ lower bounds. Therefore, H_0 is the required lower bound of the average number of tests required to locate an equivalence class of faults out of all $q+1$ equivalence classes. This completes the proof for the first vertex. The extension to the following vertices is now obvious.

We present now an algorithm to generate a decision tree using the information gain weighting function. Suppose that k faults, $f_{\alpha_1}, f_{\alpha_2}, \ldots, f_{\alpha_k}$, remain to be distinguished at vertex α of the decision tree. The uncertainty at this vertex is:

$$H_\alpha = -\sum_{i=1}^{k} p_{\alpha_i} \log_2 p_{\alpha_i},$$

where p_{α_i} is the *a priori* probability that fault f_{α_i} will occur. This probability is normalized to satisfy:

$$\sum_{i=1}^{k} p_{\alpha_i} = 1.$$

Application of a test at vertex α divides the k faults into at most $a = 2^v$ sub-sets. The ith sub-set ($i = 0, 1, \ldots, a-1$) contains all the faults for which the vector $(z_1, z_2, \ldots, z_v)$ that appears in row t of the fault table equals the binary number i.

Let $P_i^{(t)}$ ($i = 0, 1, \ldots, a-1$) be the sum of probabilities of the faults included in the ith sub-set. Clearly,

$$\sum_{i=0}^{a-1} P_i^{(t)} = \sum_{j=1}^{k} p_{\alpha_j} = 1.$$

The uncertainty of the ith sub-set after test t was applied is

$$H_i^{(t)} = \sum_{f_j \in i\text{th sub-set}} \frac{p_j}{P_i^{(t)}} \log_2 \frac{p_j}{P_i^{(t)}},$$

where $p_j / P_i^{(t)}$ is the normalized probability of the fault f_j in the ith sub-set.

Let H_t denote the average amount of remaining uncertainty after test t was applied, then:

$$H_t = -\sum_{i=0}^{a-1} P_i^{(t)} H_i^{(t)}.$$

Simple algebraic manipulations yield :

$$H_t = H_\alpha + \sum_{i=0}^{a-1} P_i^{(t)} \log_2 P_i^{(t)}.$$

Consequently, the amount of uncertainty removed (or, equivalently, information gained) by the test t is :

$$G_t = H_\alpha - H_t = - \sum_{i=0}^{a-1} P_i^{(t)} \log_2 P_i^{(t)}.$$

The test t_α selected at vertex α is the test out of the m given tests for which G_t is maximal. The weighting function G_t is maximal when :

$$P_0^{(t)} = P_1^{(t)} = \ldots = P_{a-1}^{(t)} = \frac{1}{a}.$$

3.1. Example

The fault table for the network in Figure 1 is given in Table 1. The *a priori* probabilities of the 11 non-equivalent faults appearing in this table were assigned according to the number of equivalent faults that each such fault represents [4]. In order to select a test for the first vertex of the decision tree, we compute the value of G_t for each test in the fault table. For example, for the test

$t = 00000$ we obtain $P_1^{(t)} = \frac{4}{36} = \frac{1}{9}$ and $P_0^{(t)} = \frac{8}{9}$;

hence

$$G_t = - \sum_{i=0}^{1} P_i^{(t)} \log_2 P_i^{(t)} = 0.501.$$

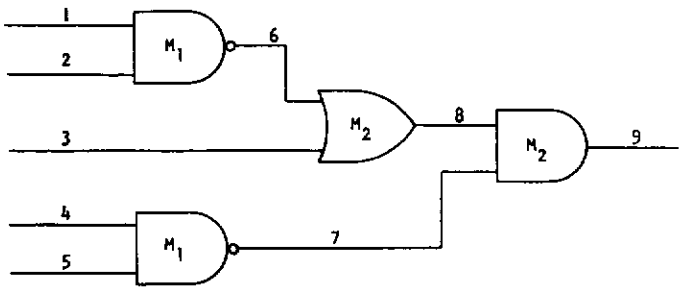

Fig. 1. Example network.

For the test $t = 01001$ we obtain $P_1^{(t)} = \frac{6}{36} = \frac{1}{6}$ and $P_0^{(t)} = \frac{5}{6}$; hence $G_t = 0.656$. This value is in fact the greatest one, and is the same for other tests, e.g. $t = 01010$. We shall arbitrarily choose the test 01001 as the first test.

Applications of the test $t_1 = 01001$ partitions the initial set of 11 faults into

Table 1. Fault matrix for the network in Figure 1.

	Faults and probabilities										
	f_0	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}
	fault-free	x_1 s-a-1	x_2 s-a-1	x_3 s-a-0	x_4 s-a-1	x_5 s-a-1	x_6 s-a-0	x_7 s-a-1	x_8 s-a-1	x_9 s-a-0	x_9 s-a-1
Tests \ Probs.	1/2	1/36	1/36	1/36	1/36	1/36	1/36	3/36	5/36	3/36	1/36
00000							1			1	
00001					1		1			1	
00010						1	1			1	
00100										1	
00110						1				1	
01000		1					1			1	
01001		1			1		1			1	
01010		1				1	1			1	
10000			1				1			1	
10001			1		1		1			1	
10010			1			1	1			1	
10101					1					1	
11001									1		1
11011											1
11100				1						1	
11101				1	1					1	
11110				1		1				1	
11111								1			1

two sub-sets : the first contains the four faults covered by t_1, i.e., $\{f_1, f_4, f_6, f_9\}$, the second contains the seven faults which are not covered by t_1, i.e., $\{f_0, f_2, f_3, f_5, f_7, f_8, f_{10}\}$.

We proceed now to select a distinguishing test for the first sub-set. For $t = 00000$ we get

$$P_1^{(t)} = \frac{4}{6} = \frac{2}{3} \text{ and } P_0^{(t)} = \frac{1}{3};$$

hence $G_t = 0.918$. For $t = 00110$ we get

$$P_1^{(t)} = P_0^{(t)} = \frac{3}{6} = \frac{1}{2};$$

hence $G_t = 1.000$. This value is the maximal one and the test $t_2 = 00110$ is chosen as the second test.

The complete decision tree generated for this example is shown in Figure 2. This decision tree is minimal and the value of the cost function is $C = 4.14$.

It should be noted that the information gain weighting function can also be employed, if, instead of the fault table, a Boolean function $T_i(x_1, \ldots, x_n)$ is given for each fault f_i, such that $T_i(a_1, a_2, \ldots, a_n) = 1$ iff the test $a_1, a_2, \ldots, a_n$ covers the fault f_i. Such Boolean functions can be derived for example by using the Boolean differences method [9]. According to this method, the required function for the fault x_i s-a-α ($\alpha = 0,1$) is

$$T_i(x_1, x_2, \ldots, x_n) = x_i^{\bar{\alpha}} \frac{dy}{dx_i},$$

where $x_i^1 = x_i$, $x_i^0 = \bar{x}_i$, and y is the primary output line. This subject is presented in detail in Appendix 1.

We estimate now the number of computations required to generate a nearly-minimal decision tree. The tree contains q vertices and at each vertex we compare m possible tests and select the one maximizing the weighting function. Hence the estimated number of computations is :

$$N_{\text{W.F.}} = q \cdot m$$

This number is significantly smaller than the number of computations $N_{\text{D.P.}}$ required in the dynamic programming method and their ratio increases as q increases. Similarly, it can be shown that the size of computer memory required for such a nearly minimal solution is significantly smaller than that required for the minimal solution.

4. Module-level sequential diagnosis

Present-day digital systems are constructed of large-scale integrated circuits. In these systems the significant problem is the identification of the smallest replaceable faulty module while the identification of the specific fault within the module is less important. A module can be an integrated circuit, a card or even a sub-system.

Module-level diagnosis requires in most cases a smaller number of tests because

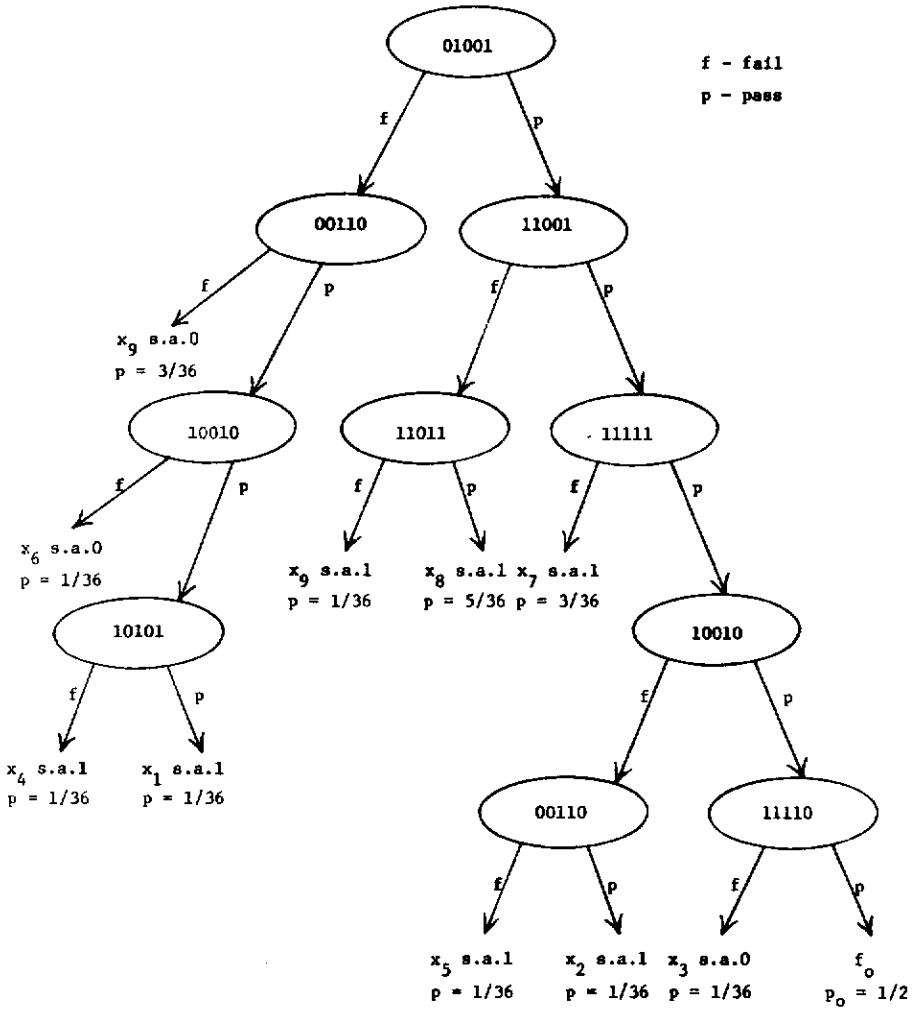

Fig. 2. Decision tree for the network in Figure 1.

only faults in different modules have to be distinguished. Assume that the network is constructed of s modules $M_1, M_2, \ldots, M_s$. Module M_j has n_j faults $f_1^j, f_2^j, \ldots, f_{n_j}^j$ with probabilities of occurrence $p_1^j, p_2^j, \ldots, p_{n_j}^j$. We assume that there are no equivalent faults in two or more different modules. Since, if there are such faults, for example f_α^i and f_β^j in modules M_i and M_j, respectively, we can define a new module M_{ij} containing one fault with probability $p_\alpha^i + p_\beta^j$ instead of f_α^i and f_β^j. The faults f_α^i and f_β^j can now be erased from the lists of faults in M_i and M_j, respectively. Such a new module enables us to distinguish between faults which occur only in M_i, and

faults in M_i which have equivalent faults in M_j. Locating the first type of faults causes replacement of M_i only, while locating the second type of faults may cause serial replacements of both modules M_i and M_j.

For completeness we define the fault-free module M_0 containing only the fault f_0. Let $P_{M_j} = \sum_{i=1}^{n_j} p_j^i$ be the *a priori* probability that a fault will occur in module M_j. Clearly, $P_{M_0} = p_0$ and $\sum_{j=0}^{s} = P_{M_j} = 1$.

We wish to distinguish only among faults in different modules; hence the initial uncertainty is:

$$H_0 = - \sum_{j=0}^{s} P_{M_j} \log_2 P_{M_j}.$$

Clearly, H_0 is the lower bound of the average number of tests required for module-level fault identification. Application of a test t divides the n_j faults in M_j into at most $a = 2^v$ sub-sets. The probability P_{M_j} is divided therefore into the following probabilities:

$$P_{M_j}^{(0)}, P_{M_j}^{(1)}, \ldots, P_{M_j}^{(a-1)}$$

which satisfy:

$$\sum_{i=0}^{a-1} P_{M_j}^{(i)} = P_{M_j}.$$

The average uncertainty remaining after the application of test t is given by

$$H_t = - \sum_{i=0}^{a-1} P_i^{(t)} \sum_{j=0}^{s} \frac{P_{M_j}^{(i)}}{P_i^{(t)}} \log_2 \frac{P_{M_j}^{(i)}}{P_i^{(t)}},$$

where $P_i^{(t)} = \sum_{j=0}^{s} P_{M_j}^{(i)}$ is the probability that test t will produce the vector $(z_1, z_2, \ldots, z_v) = i$.

Simple manipulations yield:

$$H_t = - \sum_{j=0}^{s} \sum_{i=0}^{a-1} P_{M_j}^{(i)} \log_2 \frac{P_{M_j}^{(i)}}{P_i^{(t)}}.$$

Hence the average information gain due to test t is:

$$G_t = H_0 - H_t = - \sum_{j=0}^{s} \left[P_{M_j} \log_2 P_{M_j} - \sum_{i=0}^{a-1} P_{M_j}^{(i)} \log_2 \frac{P_{M_j}^{(i)}}{P_i^{(t)}} \right],$$

which becomes :

$$G_t = \sum_{i=0}^{a-1} P_i^{(t)} \log_2 P_i^{(t)} + \left[\sum_{i=0}^{a-1} \sum_{j=0}^{s} P_{M_j}^{(i)} \log_2 \frac{P_{M_j}^{(i)}}{P_{M_j}} \right]$$

The first term is equal to G_t for single-fault diagnosis. The second term is a correction term which vanishes when each module has only one possible fault. It can be shown that G_t has no local maximum in the probability range $0 \leq P_{M_j}^{(i)} \leq P_{M_j}$, $(i = 0, 1, \ldots, a-1)$. Its maximal value is achieved at an extremal point where the correction term equals zero, i.e. when $P_{M_j}^{(i)}$ is equal to either 0 or P_{M_j} for each i. Hence, G_t is maximal when faults in the same module are not divided into sub-sets and the applied test distinguishes among modules only.

4.1. Example

The network in Figure 1 is constructed of two modules M_1 and M_2. Let $M_3 = M_{1,2}$ be the pseudo-module containing the equivalent faults of M_1 and M_2, e.g., the fault x_6 s-a-1 in M_1 is equivalent to the fault x_8 s-a-1 in M_2; hence x_8 s-a-1 $\in M_3$. The sub-sets of faults in these three modules are :

$$M_1 = \{x_1 \text{ s-a-1}, x_2 \text{ s-a-1}, x_4 \text{ s-a-1}, x_5 \text{ s-a-1}\}$$
$$M_2 = \{x_3 \text{ s-a-0}, x_9 \text{ s-a-1}\}$$
$$M_3 = \{x_6 \text{ s-a-0}, x_7 \text{ s-a-1}, x_8 \text{ s-a-1}, x_9 \text{ s-a-0}\}.$$

The corresponding probabilities are : $P_{M_1} = \frac{4}{36} = \frac{1}{9}$; $P_{M_2} = \frac{1}{18}$; $P_{M_3} = \frac{1}{3}$ and $P_{M_0} = \frac{1}{2}$.

To select a test for the first vertex, we compute the information gain G_t for each possible test. For the test $t = 00000$ the first term in G_t equals 0.501 as computed in the previous example. The second term is computed as follows : we have

$$P_{M_1}^{(0)} = P_{M_1}, \; P_{M_1}^{(1)} = 0, \; P_{M_2}^{(0)} = P_{M_2}, \; P_{M_2}^{(1)} = 0,$$
$$P_{M_3}^{(0)} = \frac{2}{9}, \; P_{M_3}^{(1)} = \frac{1}{9}, \; P_{M_0}^{(0)} = P_{M_0}, \; P_{M_0}^{(1)} = 0;$$

hence the second term equals

$$\sum_{i=0}^{1} \sum_{j=0}^{3} P_{M_j}^{(i)} \log_2 \frac{P_{M_j}^{(i)}}{P_{M_j}} = -0.306.$$

Therefore $G_t = 0.501 - 0.306 = 0.195$.

For the test $t = 11001$ we get

$$P_1^{(t)} = \frac{1}{6}, \; P_0^{(t)} = \frac{5}{6}, \; P_{M_0}^{(0)} = P_{M_0}, \; P_{M_0}^{(1)} = 0, \; P_{M_1}^{(0)} = P_{M_1}, \; P_{M_1}^{(1)} = 0;$$

$$P_{M_2}^{(0)} = P_{M_2}^{(1)} = \frac{1}{36}; \; P_{M_3}^{(0)} = \frac{7}{36}, \; P_{M_3}^{(1)} = \frac{5}{36};$$

hence $G_t = 0.276$.

This value is the greatest one and the test $t_1 = 11001$ is therefore chosen for the first vertex. The final decision tree constructed for this network is shown in Figure 3. The average number of tests required to locate a faulty module in this decision tree is $C = 3.94$.

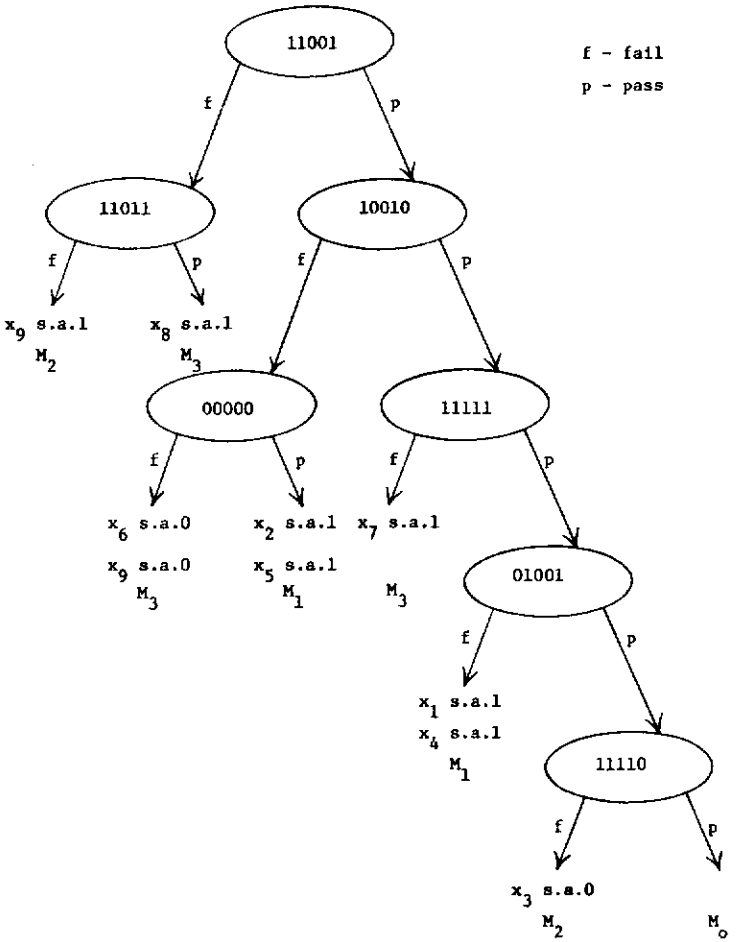

Fig. 3. Decision tree for module level diagnosis.

Appendix 1

In this appendix we show that the information gain weighting function can be employed for generating decision trees for fault diagnosis without a fault table. Instead of a fault table, for each fault f_i we are given a Boolean function $T_i(x_1, x_2, \ldots, x_n)$ satisfying $T_i(a_1, a_2, \ldots, a_n) = 1$ iff the test $(a_1, a_2, \ldots, a_n)$ covers the fault f_i.

These functions can be used in order to determine equivalence between faults and detectability of faults. If $T_i(x_1, x_2, \ldots, x_n) \equiv T_j(x_1, \ldots, x_n)$, then f_i and f_j are indistinguishable, i.e., they are equivalent. If $T_i(x_1, x_2, \ldots, x_n) \equiv 0$, then the fault f_i is undetectable.

In the following, we present a method for using these functions for generating a decision tree. Let $f_{\alpha_1}, f_{\alpha_2}, \ldots, f_{\alpha_k}$ be the k faults with probabilities $p_{\alpha_1}, p_{\alpha_2}, \ldots, p_{\alpha_k}$ which are left for diagnosis at vertex α of the decision tree. If the network has a single primary output, we compute for each test the following probabilities:

$$P_1^{(t)} = \sum_{i=1}^{k} P_{\alpha_i} T_{\alpha_i}(t),$$

$$P_0^{(t)} = 1 - P_1^{(t)}.$$

$P_1^{(t)}$ $(P_0^{(t)})$ is the sum of probabilities of all faults covered (not covered) by the test t out of the k faults at vertex α. The test t for which we get

$$\max_t \left\{ -P_1^{(t)} \log_2 P_1^{(t)} - P_0^{(t)} \log_2 P_0^{(t)} \right\}$$

will be selected to be the distinguishing test at vertex α.

If the network has v primary output lines, the function $T_i(x_1, x_2, \ldots, x_n)$ is a vectorial function:

$$T_i = (T_i^{(1)}, T_i^{(2)}, \ldots, T_i^{(v)}).$$

$T_i^{(j)}(a_1, a_2, \ldots, a_n) = 1$ iff the value of the jth output line in the fault-free network is different from its value in the network containing the fault f_i when the test $(a_1, a_2, \ldots, a_n)$ is applied. Application of a distinguishing test t at vertex α divides the set of k faults into at most $a = 2^v$ sub-sets. Denote by $P_m^{(t)}$ the sum of probabilities of faults which are included in the mth sub-set $(m = 0, 1, \ldots, a-1)$, i.e. the sum of probabilities of faults f_{α_i} satisfying

$$\sum_{j=1}^{v} T_{\alpha_i}^{(j)}(t)\, 2^{v-j} = m.$$

The test t for which we get

$$\max_t \left\{ -\sum_{m=0}^{a-1} P_m^{(t)} \log_2 P_m^{(t)} \right\}$$

will be selected to be the distinguishing test at vertex α.

References

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

[2] Koren, I., Sequential diagnosis of digital systems, *Proc. 8th IEEE Convention, Israel 1973*, Vol. VIII, pp. 35-44.

[3] McCluskey, E.J.,and Clegg, F.W., Fault equivalence in combinational logic networks, *IEEE Trans. Comput.*, C-20 (1971) 1286-1293.

[4] Koren, I. and Kohavi, Z., Sequential fault diagnosis in combinational networks, *IEEE Trans. Comput.*, in press.

[5] Kautz, W.H., Fault testing and diagnosis in combinational digital circuits, *IEEE Trans. Comput.*, C-17 (1968) 352-366.

[6] Chang, H.Y., A distinguishability criterion for selecting efficient diagnostic tests, *AFIPS Proc. Spring Joint Comput. Conf.*, (1968) 529-534.

[7] Powell, T.J., A procedure for selecting diagnostic tests, *IEEE Trans. Comput.*, C-18 (1969) 168-175.

[8] Brulé, J.D., Johnson, R.A. and Kletsky, E., Diagnosis of equipment failures, *IRE Trans. Reliability Quality Control*, RQC-9 (1960) 23-34.

[9] Yau, S.S. and Tang, Y., An efficient algorithm for generating complete test sets for combinational logic circuits, *IEEE Trans. Comput.*, C-20 (1971) 1245-1251.

[10] Knuth, D.E., "Fundamental Algorithms", Vol. I (Addison-Wesley, New York 1968).

Résumé

On étudie dans cet article le problème de la génération d'arbres de décision pour le diagnostic adaptatif des fautes des systèmes logiques combinatoires. Les arbres de décision engendrés sont basés sur un modèle probabiliste dans lequel chaque faute est assortie d'une probabilité d'occurrence a priori. On développe un procédé de génération d'arbres de décision minimaux utilisant la programmation dynamique. Pour les grands systèmes cependant, ce procédé nécessite une grande quantité de calcul. Afin de s'affranchir de cette limitation, on développe une autre méthode, utilisant la fonction de pondération du gain d'information. Ce dernier procédé réduit considérablement la quantité de calcul nécessaire, mais les ensembles de tests engendrés ne sont pas toujours minimaux.

Dans la dernière section, le deuxième procédé est étendu au cas du diagnostic au niveau module, dans lequel on ne distingue des fautes que si elles concernent des modules différents.

Zusammenfassung

In dieser Arbeit wird das Problem der Erzeugung von Entscheidungsbäumen für adaptive Fehlerdiagnose bei digitalen Schaltkreisen behandelt. Der Begriff der erzeugten Entscheidungsbäume stützt sich auf ein wahrscheinlichkeitstheoretisches Modell, in dem jeder Fehler durch eine a priori Wahrscheinlichkeit des Auftretens gegeben ist. Unter Verwendung dynamischer Programmierung wird ein Verfahren zur Erzeugung minimaler Entscheidungsbäume entwickelt. Für grosse Netzwerke erfordert das Verfahren einen grossen Rechenaufwand. Um diesen Nachteil zu umgehen, wird ein alternativer Ansatz zur Erzeugung von Entscheidungsbäumen entwickelt, in dem eine Gewichtsfunktion für das Informationsmass benutzt wird. Die Anzahl der erforderlichen Berechnungen wird dadurch wesentlich geringer, aber die dann erzeugten Testmengen sind nicht immer minimal. Im letzten Teil wird das zweite Verfahren erweitert, um den Fall der Diagnose auf Modul-Ebene zu erfassen, in dem man nur Fehler in verschiedenen Moduln zu unterscheiden hat.