Sequential Diagnosis with Asymmetrical Tests

ANTON BIASIZZO, ALENKA ŽUŽEK AND FRANC NOVAK

Computer Systems Department, Jožef Stefan Institute, Jamova 39, 1000 Ljubljana, Slovenia
Email: anton.biasizzo@ijs.si

In this paper we present the generalization of the test sequencing problem, originally defined for symmetrical tests, that also covers asymmetrical tests. We prove that the same heuristics that has been employed in the traditional solution of the problem (e.g., the AO* algorithm with heuristics based on Huffman’s coding) can be employed also for the generalized case. Examples are given to illustrate the approach.

Received January 27, 1997; revised April 4, 1998

1. INTRODUCTION

Growing demands on system performance and recent advances in very-large-scale integration technology have resulted in an increasing complexity of electronic systems. While such complexity is needed for the system performance, a problem of maintaining and repairing these systems becomes more and more difficult. The primary goal of system maintenance is to keep the system able to perform designated tasks. If the system fails, the system maintenance has to diagnose and repair the detected failures as rapidly as possible to return the system to the correct operation.

The fault-free system operation and the system operation in the presence of a fault can be presented by different system states. The goal of the diagnostic procedure is to identify the actual system state. At the beginning of the diagnostic procedure, the system resides in an unknown system state. The diagnostic procedure isolates the actual system state by evaluating tests which provide information on the system states. The system states are usually termed as stimulus—response pairs that indicate the system’s behavior. In principle, any measurement, signal or other observable event can be viewed as a system test.

It is desirable to develop efficient diagnostic procedures to minimize the expected cost of the diagnosis, while taking into account the failure probabilities and the test costs. Optimization of a diagnostic procedure is known to be an NP-complete problem [1, 2, 3]. To cope with the hardness of optimizing the diagnostic procedure, numerous solutions have been proposed [1, 2, 4, 5]. Most of the proposed solutions use heuristics to direct the search engine towards the most promising sequence of tests.

The test sequencing problem is described by the four-tuple \((S, P, T, E)\), where \(S = \{s_0, s_1, s_2, \ldots, s_M\}\) is a set of possible system states [1]. The fault-free state is denoted by \(s_0\). The states \(s_i (i = 1, 2, \ldots, M)\) correspond to different faulty states of the system. \(P = \{p(s_0), p(s_1), p(s_2), \ldots, p(s_M)\}\) is the \(a\ priori\) probability vector of the system states and determines the probability of the system being in the state \(s_i\) before the diagnostic procedure is started. \(T = \{t_1, t_2, \ldots, t_N\}\) is a set of \(N\) available tests and \(E = \{e_1(t_1), e_2(t_2), \ldots, e_k(t_N)\}\) is a set of test costs. The test cost can be measured in terms of time, equipment requirement, manpower requirement or other economical factors.

In general, tests distinguish among the sets of the system states. In the case of binary tests, two sets of the system states are defined: one corresponding to the \(\text{fail}\) test outcome (set \(A\)) and the other to the \(\text{pass}\) test outcome (set \(B\)). It is obvious that every system state has to be an element of at least one set (i.e. \(A \cup B = S\), the test always has a result). According to the intersection of the sets \(A\) and \(B\), we distinguish between two types of tests [5]:

- a \textbf{symmetrical test} where \(A \cap B = \emptyset\);
- an \textbf{asymmetrical test} which is a more general test form including the cases when \(A \cap B \neq \emptyset\).

The conventional test sequencing problem formulation assumes that tests are symmetrical. For a symmetrical test, the outcome is determined by the state of the system: if the system is in the state \(s_i \in A\) then the test fails if \(s_i \in A \iff\) test fails). Thus a symmetrical test \(t_j\) can be described by a binary column vector \(d_{ij}\) of dimension \((M + 1)\). The \(i\)th element of the test vector \(d_{ij}\) is 1 if test \(t_j\) detects a failure state \(s_i\). In other words, in the case of a failure state \(s_i\) the test \(t_j\) fails, otherwise the test passes.

In the case of an asymmetrical test, there is at least one system state that remains on the candidate list regardless of the test outcome (\(A \cap B \neq \emptyset \Rightarrow \exists s; s \in A \land s \in B\)). Obviously, if a test exhibits asymmetrical behavior for all system states, the test is not useful at all.

Tests are normally performed by applying stimuli at system inputs and observing the response at system outputs. For a complex system composed of several functional blocks it is difficult to design tests that detect failures only in some target functional blocks. In the case of serial interconnection structure they also partially test some other blocks through
which stimuli and response are transmitted. Consequently, in such situations asymmetrical tests are easier to design than symmetrical tests and their costs are lower than the costs of the symmetrical tests.

Because of the symmetrical test assumption in the conventional test sequencing problem formulation, asymmetrical tests are excluded from the consideration and the resulting test sequence may not be optimal. The goal of our paper is to explore possibilities of including available asymmetrical tests to the optimization procedure.

The paper is organized as follows. In Section 2, we propose a generalized test presentation. In Section 3, we describe the inference engine used in the diagnostic procedure. Section 4 describes optimization of the diagnostic procedure using a search algorithm. In Section 5, we present cost estimation needed for the optimization of the diagnostic tree. Here, we also prove that the same cost estimation can be used for both symmetrical and asymmetrical tests. In Section 6, we give an illustrative example of the use of asymmetrical tests. Finally, Section 7 provides a summary and future extensions.

2. PROPOSED GENERALIZED TEST PRESENTATION

Conventional test presentation describes a test by a binary column vector $d_j$. Since the binary variable $d_j$ assigned to the test $t_j$ and the system state $s_l$ does not provide sufficient information to make inferences in the case of the asymmetrical test, the conditional probability vector $d_j$ is used instead. $d_{ij}$ is a conditional probability that test $t_j$ will fail if the system state $s_l$ occurs. It is worth noting that variable $d_{ij}$ for the symmetrical test stays the same as with the traditional presentation. The difference is in the interpretation of its value: in traditional presentations the value of $d_{ij}$ is 1 if test $t_j$ detects (i.e. test fails) the system state $s_l$. This can be redefined: test $t_j$ will fail if the system state $s_l$ occurs (the conditional probability is 1).

EXAMPLE 1. To clarify the proposed test presentation, an example with five states and five tests is presented in Table 1. In this system there are four faulty states $s_1$, $s_2$, $s_3$, $s_4$ and the fault-free state $s_0$. A set of five tests may be used to isolate the failure state. Tests $t_1$, $t_4$ and $t_3$ are symmetrical (binary values). Tests $t_2$ and $t_5$ are asymmetrical. In this example, test $t_1$ exhibits asymmetrical behavior only in the system state $s_2$ and test $t_3$ in the system state $s_3$. The value $d_{21} = 0.8$ means that the probability that the test $t_1$ fails is 0.8, if the system is in the state $s_2$.

The goal of the test sequencing problem is to design a test algorithm that unambiguously identifies any system state in $S$, using tests from the test set $T$, and minimizes the cost function

$$J = p^T Ac = \sum_{j=1}^{N} \sum_{i=1}^{M} a_{ij} p_e c_j$$

where $A = (a_{ij})$ is a binary matrix such that $a_{ij}$ is 1, if the test $t_j$ is used in the path leading to the leaf node $z_i$.

<table>
<thead>
<tr>
<th>Tests</th>
<th>$t_1$</th>
<th>$t_2$</th>
<th>$t_3$</th>
<th>$t_4$</th>
<th>$t_5$</th>
<th>System state probabilities $p_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s_0$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0.32</td>
</tr>
<tr>
<td>$s_1$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0.30</td>
</tr>
<tr>
<td>$s_2$</td>
<td>0.8</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0.16</td>
</tr>
<tr>
<td>$s_3$</td>
<td>1</td>
<td>0</td>
<td>0.5</td>
<td>0</td>
<td>0</td>
<td>0.12</td>
</tr>
<tr>
<td>$s_4$</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0.10</td>
</tr>
</tbody>
</table>

FIGURE 1. Test algorithm for the system of Example 1.

otherwise $a_{ij} = 0$. The leaf node $z_i$ identifies the system state $s_i$. $p_i$, denotes the probability that the test algorithm will stop in the leaf node $z_i$. The relation between the a priori probability of the system states and the leaf node's probability is expressed by

$$p = A \cdot p_c$$

where $A = (\lambda_{ij})$ is a binary matrix such that $\lambda_{ij} = 1$ iff node $z_i$ identifies the system state $s_j$. It is easy to verify that in the case of symmetrical tests $p = p_c$.

The test sequencing problem with asymmetrical tests is a partially observed Markov decision problem [1], where the Markov state $x$ corresponds to the suspected set of the system states (ambiguity set) and the decision corresponds to the test performed in the state $x$.

An optimal test sequence for Example 1 is shown in Figure 1. This test sequence includes the asymmetrical test $t_5$ which gives an additional leaf (the system state $s_3$ resides in two leaves).

Besides the test presentation, the effects of the test evaluation on the system have to be determined when developing the diagnostic procedure. In the following section, the inferences drawn from the test evaluation are investigated.

3. INFERENCE ENGINE

At the beginning of the diagnosis, the system state is unknown and the a priori probabilities of the system states
are given (maximum ambiguity). When test $t_j$ is performed, the probabilities of the system states change. The procedure that tracks the changes of the system states probabilities is called the inference engine [5].

Suppose that at a given node probability distribution $p(x)$ is given and test $t_j$ is performed. One has to determine the a posteriori probabilities of the states for both test outcomes: $p(x|t_j)$ if test $t_j$ passes and $p(x|\bar{t}_j)$ if the test fails. The a posteriori probabilities are derived by Bayes' rule. The probability that test $t_j$ fails is

$$p(t_j = 0) = \sum_{i=0}^{M} p(t_j | s_i) \cdot p(s_i)$$

and the probabilities of the system states under the assumption that the test $t_j$ fails are

$$p(s_i | t_j = 0) = \frac{p(t_j = 0 | s_i) \cdot p(s_i)}{p(t_j = 0)}.$$

Similarly the probabilities under the assumption that the test $t_j$ passes are determined by

$$p(t_j = 1) = \sum_{i=0}^{M} p(t_j = 1 | s_i) \cdot p(s_i)$$

$$p(s_i | t_j = 1) = \frac{p(t_j = 1 | s_i) \cdot p(s_i)}{p(t_j = 1)}.$$

The inference engine determines the probability distribution for each test outcome. Some a posteriori probabilities of the system states become 0 and the corresponding system states are eliminated from the system state identification. Thus new sets of system states for each test outcome are determined ($S_0$ and $S_1$ are the sets if test $t_j$ fails or passes, respectively).

In Figure 2, the inferences at the root node of Example 1 are depicted. If test $t_0$ fails or passes the sets of possible system states are $S_0 = \{0, 2, 3\}$ and $S_1 = \{1, 3, 4\}$, respectively. The a posteriori probability distributions are $p_0 = [0.59, 0.30, 0.11]$ and $p_1 = [0.65, 0.13, 0.22]$, respectively.

In the following section a search algorithm used in the optimization of the diagnostic procedure is presented.

4. SEARCH ALGORITHM

To obtain an optimal solution we formulate the test sequencing problem as a best-first search on an AND/OR graph. The test sequencing problem is known to be NP-complete therefore we use a cost-to-go heuristic estimation to guide the AND/OR graph search.

There exist several AND/OR graph search techniques. Our implementation corresponds to the AO* algorithm [6], an ordered, best-first search algorithm. This algorithm expands only that node $x$ of the search graph that is most promising for reaching the goal nodes on the basis of cost-to-go estimation $h(x)$. The AO* algorithm is guaranteed to find an optimal solution (test algorithm) if $h(x)$ is admissible at every node of ambiguity $x$. Heuristic cost-to-go function is admissible, if it is a lower bound of the cost of the fault tree.

The AO* algorithm consists of the following three basic operations:

- the graph-traversing operation, that follows the best current path;
- node selection and expansion, where the node with the highest cost estimation $h(x)$ is selected. During the expansion, the successor nodes are generated and their cost estimations are performed;
- the cost-revising operation, that updates the cost estimation of the expanded node and propagates the change of the estimation to the predecessor nodes.

As mentioned earlier, the cost estimation of the diagnostic procedure is essential for efficient optimization. In the following section the estimation based on Huffman coding will be described. It will be shown that such estimation is admissible in the case of asymmetrical tests.

5. COST ESTIMATION

The cost-to-go function is derived from Huffman coding [7]. The information theoretic lower bounds ensure that the heuristic estimation is admissible and an optimal solution is found using AO*.

In the following we briefly describe the cost estimation that has been originally developed for symmetrical tests [1]. In the next subsection we give the proof that the same cost estimation is admissible in the case of asymmetrical tests. Consequently, this cost estimation can be used for the optimization of the diagnostic procedure with symmetrical and asymmetrical tests.

5.1. Symmetrical tests

The test sequencing problem with symmetrical tests can be translated to the coding problem in the case when all 2$^M$ (symmetrical) tests are available.

An analogy between the noiseless coding and test sequencing problem with a symmetrical test can be drawn as follows:

- the system states correspond to the source letters;
5.2. Asymmetrical tests

In the case of an asymmetrical test there is no direct analogy with the coding problem, because the asymmetrical test does not divide the set of system states into two disjoint sets.

**Theorem 5.1.** The optimal average code word length \( w^*(x) \) is a lower bound of the average length \( l^*(x) \) of the optimal test algorithm started at \( x \).

**Proof.** Without loss of generality, we can assume that the state system probabilities are in descending order \( \pi = [\pi_0, \pi_1, \ldots, \pi_k, \ldots, \pi_M] \), where \( \pi_0 \geq \pi_1 \geq \ldots \geq \pi_k \geq \ldots \geq \pi_M \).

Suppose that the optimal test algorithm includes some test \( t_i \), which exhibits asymmetrical behavior regarding the system state \( s_k \). When test \( t_i \) is evaluated, it divides the set of system states \( S \) into two sets \( S_0 \) and \( S_1 \), where \( S_0 \cap S_1 = \emptyset \). This results in two leaf nodes of the test algorithm for the system state \( s_k \) with the probabilities \( \pi_{10} \) and \( \pi_{11} \), where \( \pi_{10} + \pi_{11} = \pi_k \).

Let us assume that the leaf nodes \( z_{10} \) and \( z_{11} \) correspond to the different 'virtual' system states \( s_{10} \) and \( s_{11} \), respectively \((S \mapsto S', \pi \mapsto \pi', x \mapsto x')\). By this 'separation' the asymmetrical test \( t_i \) becomes symmetrical and the probability distribution of the extended set of system states is \( \pi' = [\pi_0, \pi_1, \ldots, \pi_{10}, \pi_{11}, \pi_M] \). This does not change the optimal test algorithm, because the probability distributions of the sets \( S_0 \) and \( S_1 \) did not change.

At this point we apply the analogy with the coding theory and get

\[ l^*(x') \geq w^*(x') \]

Suppose that the source letters \( x'_i \) and \( x'_j \) correspond to the system state \( s_i \) and that their code lengths are \( w_i' \) and \( w_j' \), respectively. Without loss of generality we can assume that \( w_i' \leq w_j' \). The average code length is

\[
\sum_{i=0}^{M+1} w_i' \pi_i' + \sum_{i=0}^{M+1} w_i' \pi_i' \pi_i' \geq \sum_{i=0}^{M+1} w_i' \pi_i' + \sum_{i=0}^{M+1} w_i' \pi_i'
\]

where \( w' \) is the average code length with the probability distribution changed to \( p_i' = p_i + p_i' \), \( p_i' = 0 \) and \( p_i' = p_i \) otherwise. The coding scheme remained the same. For this probability distribution an optimal code exists with average code length \( w'^* \):

\[ w'^* \geq w' \geq w'^* \]

This source letter probability distribution is actually the original source letter probability distribution with a source letter added with the probability 0 \((p' = [\pi_0, \pi_1, \ldots, \pi_k, \ldots, \pi_M, 0])\). The smallest probabilities in the probability distributions are \( p_i' = 0 \) and \( p_m' = p_i' \). From the Huffman coding algorithm it follows that the optimal code

---

**Diagram Description:**

- The diagram illustrates a binary tree associated with Huffman code of Example 1.
- The tree consists of nodes labeled with states and branches indicating the path taken during the coding process.
- The states are denoted as \( S_0, S_1, \ldots, S_M \) with associated probabilities \( \pi_0, \pi_1, \ldots, \pi_M \).
- The tree structure is used to determine the optimal code word length for each state.

**Equation:**

\[ w^*(x) \leq w'(x) = l'(x) \]
length \( w'' = w''(S'') = p''_M + w''(S'') \), where \( p''_M = p''_M \). It is easy to see that \( S'' = S \), thus \( w'' = p'' + w'' \) and \( l'' = w'' = p'' + w'' = p'' + w'' > w'' \).

5.3. Test costs

The second difference between the test sequencing problem and the coding problem is that tests may have different costs. For the estimation of the lower bound of the average cost of the test algorithm, the smallest test costs should be considered.

**Theorem 5.2.** Without loss of generality we can assume that the tests are in ascending order by their costs \( 0 \leq c_1 \leq c_2 \leq \ldots \leq c_M \). The lower bound \( h(x) \) of the average cost of the diagnostic tree is given by

\[
h(x) = \sum_{j=1}^{M} c_j + \left[ w^*(x) - w'(x) \right] \cdot c_{w'(x)+1},
\]

where \( w'(x) \) is the integer part of the optimal code length \( w^*(x) \).

**Proof.** The average cost of the optimal tree started at \( x \) is given by

\[
h^*(x) = \sum_{i=0}^{M} p(z_i, x) \left( \sum_{j=1}^{N} a_{ij}^*(x) \cdot c_j \right)
\]

where \( a_{ij}^*(x) = 1 \) if test \( t_j \) is used in the path leading to the leaf node \( z_i \), which identifies system state \( S_k \) in the optimal test algorithm started at \( x \). \( p(z_i) \) is the probability that the test algorithm rooted at \( z_i \) will stop in the leaf node \( z_i \) (the probability that the node \( z_i \) will identify the system state). Let \( l_i^*(x) \) be the number of the test used in the path leading to the leaf node \( z_i \) in the optimal test algorithm started in \( x \),

\[
l_i^*(x) = \sum_{j=1}^{N} a_{ij}^*(x).
\]

Since the test costs are in ascending order, we have

\[
\sum_{j=1}^{N} a_{ij}^*(x) \cdot c_j \geq \sum_{j=1}^{N} c_j \cdot \gamma(l_i^*(x))
\]

and

\[
h^*(x) \geq \sum_{i=0}^{M} p(z_i) \cdot \gamma(l_i^*(x)).
\]

The smallest test costs are chosen for each leaf node identification. Any other choice of test costs gives a rise of the overall tree cost (including test costs in the optimal decision tree).

The \( \gamma(l(x)) \) is a monotone increasing convex function of \( l(x) \). Using Jensen’s inequality, we have

\[
h^*(x) \geq \sum_{i=0}^{M} p(x_i) \cdot \gamma(l_i^*(x)) \geq \gamma(l^*(x))
\]

where \( l^*(x) \) is the average length of the optimal test algorithm started in \( x \). From the monotonicity of \( \gamma(l(x)) \) and \( w^*(x) \leq l^*(x) \) (Theorem 5.1), we derive

\[
h^*(x) \geq \gamma(l^*(x)) \geq \gamma(w^*(x)) = \sum_{j=1}^{M} c_j + \left[ w^*(x) - w'(x) \right] \cdot c_{w'(x)+1}.
\]

6. CASE STUDY

In this section we present some experimental results of the generation of a sequential diagnostic tree.

**Example 2.** The system under test is part of a complex digital circuit and contains three adders and two multiplexers as depicted in Figure 4. Failure of each functional block is presented by a diagnostic symptom state. Since internal test points in such circuits are very hard to reach with test equipment, it is desirable that the test stimuli be applied to the system inputs and their responses are captured at the system outputs. Tests which require access to the internal points might be necessary to provide the required diagnostic resolution. Since the access of the internal points requires special test equipment, their costs are higher compared to the costs of input-output tests.

In this system all functional blocks are mutually interconnected; therefore the test of a particular functional block affects other blocks and may detect some faults in other blocks as well. This information is captured by asymmetrical tests. Tests for the individual functional block were determined and they were justified to the system inputs and their responses propagated from the functional block to the system outputs. The diagnostic capabilities of each developed test were determined with a fault simulation. To increase diagnostic resolution, tests which require access to the circuit internal points were added but their costs are higher. These tests reduce influences between functional blocks and could be described as symmetrical tests. The diagnostic capabilities of all tests are presented in Table 2.

The table includes five symmetrical and five asymmetrical tests. The order of tests in test matrix is not important because the search algorithm finds an optimal solution. However there might be several solutions with equal average cost.

In Example 2 the access to the system internal points eliminates influences between blocks. Tests, which use the access to the internal points, are symmetrical tests. In Table 2, the bold characters are used to represent tests which do not require access to the internal points and exhibit asymmetrical behavior. Other tests are symmetrical tests and their costs are higher.

Symmetrical tests are sufficient for the system diagnosis; however the use of asymmetrical tests could provide a more efficient diagnostic procedure. To illustrate this the diagnostic trees were designed with only symmetrical tests and with all available tests. The optimal diagnostic tree where only symmetrical tests are considered is presented in Figure 5.
This diagnostic tree is small, but its cost is relatively high due to the higher costs of the symmetrical tests. The average cost of the obtained diagnostic tree is 12.4. With inclusion of the asymmetrical tests the overall cost of the diagnostic tree decreases because of the lower costs of the asymmetrical tests. The optimal diagnostic tree is presented in Figure 6.

In Figure 6 the shaded nodes represent asymmetrical tests. Note that an asymmetrical test does not necessarily exhibit asymmetrical behaviour. This diagnostic tree has more nodes than the previous one, but its average cost is 8.19, which is considerably lower than the average cost of the previous diagnostic tree.

**Example 3.** The lower costs of the asymmetrical tests are not the only factors which reduce the average cost of the diagnostic tree. Let us consider an example comparable to those studied by [1, 5]. The complete test matrix, fault probabilities and test costs for this system are given in Table 3. Again the asymmetrical tests are presented by the bold characters in the test matrix.

The system has 16 diagnostic states, 12 symmetrical and 5 asymmetrical tests. All tests have equal cost and for simplicity we can assign all test costs to 1. Tests $t_{13}$, $t_{14}$, $t_{15}$, $t_{16}$ and $t_{17}$ are asymmetrical. The generated optimal diagnostic tree considering only symmetrical tests is presented in Figure 7. Its average cost is 3.7.

The optimal diagnostic tree with inclusion of all available tests is shown in Figure 8. The average cost of the resulting diagnostic tree is 3.6.
TABLE 3. Test matrix, fault probabilities and test costs for Example 3.

<table>
<thead>
<tr>
<th>Tests</th>
<th>t₁</th>
<th>t₂</th>
<th>t₃</th>
<th>t₄</th>
<th>t₅</th>
<th>t₆</th>
<th>t₇</th>
<th>t₈</th>
<th>t₉</th>
<th>t₁₀</th>
<th>t₁₁</th>
<th>t₁₂</th>
<th>t₁₃</th>
<th>t₁₄</th>
<th>t₁₅</th>
<th>t₁₆</th>
<th>t₁₇</th>
<th>p(s₁)</th>
</tr>
</thead>
<tbody>
<tr>
<td>s₀</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0.02</td>
</tr>
<tr>
<td>s₁</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.18</td>
</tr>
<tr>
<td>s₂</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.3</td>
<td>0.06</td>
</tr>
<tr>
<td>s₃</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0.02</td>
</tr>
<tr>
<td>s₄</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0.02</td>
</tr>
<tr>
<td>s₅</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.8</td>
<td>0.06</td>
</tr>
<tr>
<td>s₆</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.5</td>
<td>0.02</td>
</tr>
<tr>
<td>s₇</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.18</td>
</tr>
<tr>
<td>s₈</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.02</td>
</tr>
<tr>
<td>s₉</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.3</td>
<td>0.02</td>
</tr>
<tr>
<td>s₁₀</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.02</td>
</tr>
<tr>
<td>s₁₁</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.3</td>
<td>0.02</td>
</tr>
<tr>
<td>s₁₂</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.6</td>
<td>0.06</td>
</tr>
<tr>
<td>s₁₃</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.3</td>
<td>0.02</td>
</tr>
<tr>
<td>s₁₄</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.6</td>
<td>0.06</td>
</tr>
<tr>
<td>s₁₅</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.2</td>
<td>0.02</td>
</tr>
</tbody>
</table>

Costs  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  |

FIGURE 7. Optimal decision tree for Example 3 with symmetrical tests.

FIGURE 8. Optimal decision tree for Example 3 with all available tests.
This diagnostic tree exhibits asymmetrical behavior of test 17 in state 315; therefore state 315 appears in two leaf nodes. Besides this asymmetrical test, other asymmetrical tests appear in the resulting diagnostic tree but they no longer exhibit asymmetrical behavior since the set of possible diagnostic states has been reduced during the diagnostic procedure. When this occurs, some asymmetrical tests eventually become symmetrical. These tests are equivalent to other symmetrical tests but they reduce the cost of the diagnostic tree.

7. CONCLUSION

In this paper we have described the extension of the test sequencing problem solutions including both symmetrical and asymmetrical tests. The main contribution of the paper is the proof that the same heuristics can be employed in the generalized case as in the traditional solutions. In general, asymmetrical behavior of tests increases the cost of the decision tree; therefore tests in the resulting solution rarely manifest their asymmetrical properties. An asymmetrical behavior of tests tends to increase the cost of the decision tree due to the fact that some system states appear in more than one leaf. However, after studying numerous examples it has been observed that the employed AND/OR graph search algorithm pushes asymmetrical tests towards the leaves of the decision tree where they actually exhibit the symmetrical property. On the other hand, the cost of asymmetrical tests is in general lower than the cost of symmetrical tests. Hence, as shown in the examples, asymmetrical tests do appear in the optimal decision trees which justifies the generalization proposed in this paper.

REFERENCES