AO* VARIANT METHODS FOR AUTOMATIC GENERATION OF NEAR-OPTIMAL DIAGNOSIS TREES

Xavier OLIVE [*,**] **Louise TRAVE-MASSUYES** [*]
Hervé POULARD [**]

[*] *LAAS-CNRS, 7 Avenue du Colonel ROCHE*
F-31077 TOULOUSE Cedex 4
[**] *ACTIA, 25 Chemin de Pouvourville - B.P. 4215*
F-31432 TOULOUSE Cedex 4

Abstract: This paper deals with a model-based Automatic GENeration of DiAgnosis trees method (AGENDA) using the AO* algorithm. The inputs of this algorithm are the anticipated faults which may occur on the system to diagnose with their respective occurrence probability, the anticipated tests that can be performed and the cross-table which assigns to each (fault/test) pair the set of modalities which are expected as outcome of the test when the fault occurs. The main drawback of AGENDA is the required high computation time to obtain an optimal diagnosis tree for large complex systems. This paper presents two methods which allow us to generate the diagnosis trees in a more efficient way while reducing the optimality loss.

Keywords: Test sequencing problem, model based, AO*, near optimal diagnosis tree.

1. INTRODUCTION

In the automotive domain, the use of electronic systems to control several functions is widely spread. These functions span diverse automotive areas such as fuel injection, ABS, ... These electronic systems are composed of a voltage supply, sensors and actuators linked to electronic control units (ECUs for short) by a wire harness.

The main task of the ECU is to elaborate and send control signals to the actuators, taking into account the signals received by the sensors. Moreover, an ECU is equipped with a self-diagnosis function that reliably detects the failing electric circuits. However, the ECU is not able to localize precisely the faulty components within the functional circuit. In order to accurately localize the faulty components, diagnosis tree are built.

Currently, diagnosis trees are built by human experts resulting by frequently errors due to system complexity. Hence, it is imperative to reduce human intervention in the generation of diagnosis trees. That's why AGENDA (Faure, 2001), an off-line automatic diagnosis tree generation method was developed. AGENDA is based on an AO* algorithm having as input a "cross-table" (fault signature matrix) and using some heuristic function.

This task of finding an optimal diagnosis tree is well-known to be NP-complete (Moret, 1982). So, to avoid high computation time associated to the generation of an optimal diagnosis tree, methods which allow to obtain near-optimal trees are proposed in the litterature. Different reduced test subsets are used by Faure (Faure, 2001). The notion of ϵ-*admissibility* which allows to control the suboptimality and a *Multistep information Heuristic* approach are presented in (Raghavan *et al.*, 1999). Rollout strategies are described in (Tu and Pattipati, 2002) for dealing with large complex system. Some other variants like MAO* and WAO* are proposed in (Chakrabarti *et al.*, 1989) and (Chakrabarti *et al.*, 1988), respectively. This paper proposes two new methods in order to reduce this too high computation time by building a near-optimal diagnosis tree.

Section 2 sets the test sequencing problem. Section 3 details the AO* algorithm. Section 4 and 5 presents the two proposed new methods. Section 6 gives some performance evaluation.

2. TEST SEQUENCING PROBLEM

In this section, we define the main features of the test sequencing problem (Pattipati and Dontamsetty, 1992) : given a faults set, a tests set and a "crosstable", designing an optimal diagnosis tree by minimizing equation 2.

2.1 Fault set

Let E be the set of the n_E elementary components e_k with $k \in \{1, ..., n_E\}$ which constitute the system to diagnose. Let n_{e_k} be the number of abnormal behavior $AB_l(e_k)$ with $l \in \{1, ..., n_{e_k}\}$ of the elementary component e_k and $\neg AB(e_k)$ its normal behavior.

A fault is a n_E vector which associates to each of the n_E elementary component e_k one of its n_{e_k} abnormal behavior or its normal one. For such a system, $\prod_{k=1}^{n_E}(n_{e_k}+1)$ faults may occur.

For the following, let F be the fault set composed by the n_F faults f_i with $i \in \{1, ..., n_F\}$ and p_i, their respective a priori occurrence probability.

2.2 Test set

A test is defined as a physical variable measurement or as an observable manifestation.

A prediction process allows to find all the possible outcomes of each test w.r.t. to every fault of F. All this outcomes define the domain value of the test. For each test, a set of modalities is computed as a partition of the test domain value.

For the following, let S be the test set composed by n_S tests s_j with $j \in \{1, ..., n_S\}$, n_M^j their respective number of modalities m_k^j with $k \in \{1, ..., n_M^j\}$ and c_j, their respective cost representing the measurement tool configuration cost and the measurement points accessibility cost.

We gives below some definitions which are used in the next sections.

Definition 1. (Binary test). A test is said to be binary if it has a number of modalities exactly equal to two.

Definition 2. (Exclusive test). A test is said to be exclusive if, for any fault, one and only one modality is expected for the concerned test.

Definition 3. (Multi-modal test). A test is said to be multi-modal if it has a number of modalities greater or equal than two.

Definition 4. (Entire test set). A entire test set is the set of all physically available tests on a system.

Definition 5. (Discriminating test set). A test set is said to be discriminating for a fault set F if it is able to discriminate all the faults of F.

Definition 6. (Unit test cost assumption). Under unit test cost assumption, the test cost is equal to 1 for any considered test.

2.3 Cross-table

Given F and S, the corresponding "cross-table" C has n_F rows and n_S columns. Each of its cells $C(i,j)$ contains the set of $n_M^{i,j}$ modalities among the n_M^j possible ones which are expected as outcome of the test s_j in occurrence of the fault f_i.

Moreover, the conditional probabilities $P(s_j = m_k^j|f_i)$ of *"having m_k^j as outcome of the test s_j knowing that the fault f_i has occurred"* for any $k \in \{1, ..., n_M^j\}$ are also available in the $C(i,j)$ cell. These conditional probabilities are normalized as shown on equation (1). For any modality m_k^j which does not belong to the $C(i,j)$ modalities set then $P(s_j = m_k^j|f_i) = 0$.

$$\sum_{k=1}^{n_M^j} P(s_j = m_k^j|f_i) = 1 \tag{1}$$

For the following, C represents the "cross-table" corresponding to the F and S sets and $C(i,j)$, the cell of C relative to the fault f_i and the test s_j.

3. TEST SEQUENCING PROBLEM RESOLUTION

3.1 Diagnosis tree

A diagnosis tree may be viewed as an AND/OR tree (Pattipati and Alexandridis, 1990). An OR node correspond to a faults subset and an AND node to a test. The root node is an OR node composed of the faults sets F. A leaf node is an OR node and represents one possible fault. A not leaf OR node has one and only one AND node child corresponding to the test to apply whereas an AND node has several OR node children corresponding to the modalities of the given test.

For the following, let T be the diagnosis tree which discriminates the fault set F by using the test set S according to the cross-table C. Let n_L be the number of leaves $\{l_1, ..., l_{n_L}\}$ and $P(l_i)$ the occurrence probability of each leaf l_i such that $\sum_{i=1}^{n_L} P(l_i) = 1$. Let d_{ij} be a boolean variable equal to 1 if the test s_j belongs to the path from the root to the leaf l_i and 0 otherwise.

The objective function K of a diagnosis tree T, defined by equation (2), is considered to evaluate the different possible diagnosis trees of a same system.

$$K(T) = \sum_{i=1}^{n_L} P(l_i) \times \left(\sum_{j=1}^{n_S} d_{ij} \times c_j \right) \qquad (2)$$

Under unit test cost assumption, the objective function $K(T)$ is equivalent to the mean depth of the tree T.

3.2 AO algorithm*

Due to length restriction, the used heuristic are not developed here. We use an admissible heuristic which, combined with the AO* algorithm, guaranties to obtain an optimal diagnosis tree (Bagchi and Mahanti, 1983). For details about the used heuristics, see (Pattipati and Alexandridis, 1990) and (Yeung, 1994).

Initialization At the beginning of the AO* algorithm, the implicit AND/OR search graph is composed of only its root R, an OR node composed of entire faults set F.
The current optimal diagnosis tree T^* and L^*, the set of its expandable leaves are also initialized to $\{R\}$. Moreover, since R is a leaf, the cost-to-go value of R, $F(R)$ is initialized to the Heuristic Evaluation Function (HEF) value $h(R)$ of R.

Iterative treatment At each step of the AO* algorithm, the leaf N of the set L^* that has the highest HEF value $h(N)$ is removed from the set L^*.
The n_A AND nodes N_j^A children of the node N, are created. Each node N_j^A refers to one test s_j.
The n_M^j OR nodes $N_{j,k}^O$, children of each node N_j^A, are also created. Each node $N_{j,k}^O$ refers to the fault subset resulting from the fault subset corresponding to the node N knowing that the k^{th} outcome has been observed for the test s_j.

For each created node $N_{j,k}^O$, $F(N_{j,k}^O)$ is computed as $h(N_{j,k}^O)$. Then, for each created node N_j^A, $F(N_j^A)$ is computed as shown in equation 3.

$$F(N_j^A) = c_j + \sum_{k=1}^{n_M^j} F(N_{j,k}^O) \qquad (3)$$

Let $P_N(f_i)$ denote the occurrence probability of the fault f_i at the node N.
For each node N_j^A, the occurrence probability of each fault f_i that appears in the resulting subset corresponding to any created OR node $N_{j,k}^O$, called $P_{N_{j,k}^O}(f_i)$, is computed according the Bayes' rule as shown in equation 4. The first step of this computation corresponds to the evaluation of the probability that the outcome m_k^j is observed for test s_j at the node N_j^A, called $P_{N_j^A}(s_j = m_k^j)$.

$$\begin{cases} P_{N_j^A}(s_j = m_k^j) = \sum_{i=1}^{n_F} P(s_j = m_k^j | f_i) \times P_N(f_i) \\ P_{N_{j,k}^O}(f_i) = \dfrac{P(s_j = m_k^j | f_i) \times P_N(f_i)}{P_{N_j^A}(s_j = m_k^j)} \end{cases} \qquad (4)$$

At last, a recursive treatment is performed on the successive nodes from the node N to the root R, both included. This treatment consists in updating successively the cost-to-go values F of these nodes and the mark of the selected AND nodes which constitute the current optimal diagnosis tree T^*. Let N_c be the current node on which the recursive treatment has to be performed.

- *N_c is an OR node*
  Let n_c be the number of children of N_c, called N_j^c. At most one of these n_c children is marked.
  The mark on the selected AND node, child of N_c, is removed. For the node N_c, the cost-to-go value $F(N_c)$ is then computed as shown in equation 5 and the j^{th} AND node for which this minimum is reached is marked.

$$F(N_c) = \min_{j=1}^{n_c} F(N_j^c) \qquad (5)$$

- *N_c is an AND node*
  Let n_c be the number of children of N_c, called N_k^c. If N_c corresponds to the test s_j, then N_k^c refers to the n_M^j possible outcomes for s_j and $n_c = n_M^j$.
  The probability to reach the node N_k^c from the node N_c, called $P_{N_c}(N_k^c)$, is equivalent to the probability to observe the outcome m_k^j for the test s_j at the node N_c, called $P_{N_c}(s_j = m_k^j)$ and computed as already shown in equation 4.
  For the node N_c, the cost-to-go value $F(N_c)$ is then computed as shown in equation 6.

$$F(N_c) = c_j + \sum_{k=1}^{n_M^j} P(N_k^c) \times F(N_k^c) \quad (6)$$

The set L^* is updated according to the current marked optimal diagnosis tree T^*.

Stop condition The AO* algorithm stops when the set L^* is empty. Then, the current marked optimal diagnosis tree T^* is the definitive optimal diagnosis tree and $J(R) = F(R)$.

4. STATIC TEST SET REDUCTION METHOD

Faure (Faure, 2001) proposes to use two discriminating tests subsets in order to generate near-optimal diagnosis tree. Their definitions are given below.

Definition 7. (First Discriminating test subset). Let us consider a set S of n_S tests s_j ordered by increasing cost. A First Discriminating test subset S_{First} is defined by adding to S_{First} the tests of S one by one, according to the previous cost order, until a discriminating test subset is obtained.

Definition 8. (Minimal Discriminating test subset). A discriminating test subset S' is said to be minimal if and only if, for any of its n'_S tests s'_j with $j \in \{1, ..., n'_S\}$, $S' - \{s'_j\}$ is not able to discriminate the fault set.

We add a third definition concerning the discriminating tests subset of a given optimal diagnosis tree.

Definition 9. (Optimal Discriminating test subset). An optimal discriminating test subset S^* is a discriminating test subset composed of the tests associated to a given optimal diagnosis tree for the considered fault set F.
Moreover, it is important to underline that, for a same initial test set, it exists as many optimal discriminating test subsets as optimal diagnosis trees.

Remark 10. An optimal discriminating test subset is not necessarily a minimal discriminating test subset. Figure 1 shows an example.

Remark 11. An optimal discriminating test subset necessarily includes at least one minimal discriminating test subset.

Faure (Faure, 2001) uses the two test subsets defined in 7 and 8, as input of the test sequencing problem to build a near-optimal diagnosis tree.

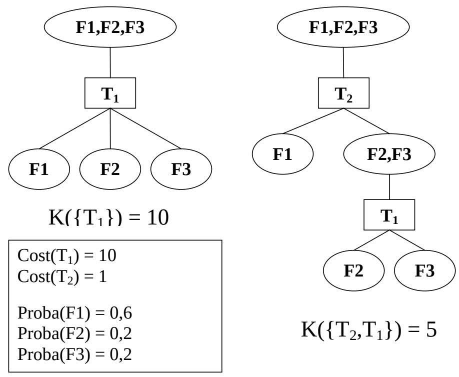

Fig. 1. Minimal and optimal test subset

The entire test set are replaced by S_{Min} or S_{First} ; we call these two near-optimal methods AO^*_{Min} and AO^*_{First}, respectively.

5. TEST SET REDUCTION METHOD

In our application domain, the cardinality of the tests set could be more than one magnitude order higher than the cardinality of the faults set. In this case, as the AO* algorithm evaluates all the remaining (i.e. not yet used) tests for each OR node at each iteration, the step of heuristic computation is very expensive in terms of computation time.

Indeed, given the n_F faults of F, the diagnosis tree needs, in the worst case of binary tests discriminating just one fault from all the others, n_F tests. So, at last iteration of the AO* algorithm, $n_S - (n_F - 1)$ tests would be evaluate by the heuristic function.

So we proposed to select a subset of the remaining tests set before the evaluation of one OR node according to a given criteria. We call this method AO^*_{Dyn}. It is important to notice that its criteria is precomputed one time for all the other, otherwise it is obvious that replacing an heuristic evaluation by another one has no effect.

A similar method called *Limited Search AO** restricted to binary tests is proposed in (Raghavan *et al.*, 1999). A user specified parameter is used to limit the number of considered tests after a heuristic evaluation based on information gain.

5.1 AO^*_{Dyn} Method

The method AO^*_{Dyn} modifies the iterative step of the AO* algorithm. Indeed, instead of considering all the available tests for the chosen OR node, only a subset is selected. Several criteria of selection may be used.

A test s_j is characterized by 3 main features : its constant cost c_j, its number of modalities n_M^j and its efficiency $eff_j = Log(n_M^j)/c_j$ (Yeung, 1994).

These main features allow to order a priori the tests. The most interesting is the third one because it combines the first two. When tests are ordered, the number n_{max} of tests to be chosen is a constant defined by the user.

The method is based on selecting the first n_{max} ordered tests and developing the corresponding AND nodes. We have simply reduce the number of tests to be estimated by the HEF without evaluating any HEF, but by using the precompiled criteria.

Note that for each OR node created from the previous AND nodes, a new reduction is performed from the entire test set. So the reduction at some step has no effect on the next iteration of the AO* algorithm in terms of optimality.

6. ITERATIVE GENERATION OF THE DIAGNOSIS TREE METHOD

The method proposed in this section, is an *Anytime* one : indeed, it gives the best diagnosis tree w.r.t. the available time.

As described in section ??, the tests subsets S_{Min} and S_{First}, defined by definitions ?? and 7, allow to obtain efficient near-optimal diagnosis tree with acceptable computation time. This diagnosis tree may be used as a first reference that can evolve to a better one. So, we propose then to make the used tests subset evolve in order to have a better near-optimal solution. To do this, we describe next the evolution of the test subset and the impact on the AO* algorithm.

We call Any_{Min} the approach when the test set is initialized with S_{Min}, and Any_{First} when it is initialized with S_{First}.

6.1 Test set evolution

The tests subset used to generate the diagnosis tree is initialized with one of the both tests subset : S_{Min} or S_{First}.

Let us consider a discriminating test subset $S_{Current}$ which is used to obtain a global near-optimal solution. As written above, the tests subset $S_{Current}$ may evolve, it is to say some tests must be added to $S_{Current}$ in order to obtain a better global near-optimal diagnosis tree.

To choose the next test to be added, we use the same precomputed criterion as for the $AO*_{Dyn}$ approach : the test efficiency.

Let us consider $S_{Possible} = \{s_{p0}, ..., s_{pn_p}\}$, the set of n_p tests that can potentially be added to $S_{Current}$: $S_{Possible} = S - S_{Current}$. Without loss of generalities, we assume that the tests are ordered by increasing efficiency in $S_{Possible}$. So, the test to add is s_{p0}.

6.2 AO algorithm impact*

Let us consider K_0, the reference cost which is initialized with the cost of the diagnosis tree obtained from the initial tests subset (S_{Min} or S_{First}), and $K_{Current}$, the cost of the current creating diagnosis tree.

This approach doesn't modify the core of the AO* algorithm, but only its stop condition. Indeed, when the current cost $K_{Current}$ becomes greater than the reference cost K_0, it is not necessary to follow the search in the AND/OR tree because the final corresponding cost will be greater than the current one $K_{Current}$, so than K_0 and, finally the associated diagnosis tree will not be a better near-optimal solution.

The update of the reference cost has to be done when the final cost $K_{Current}$ of a diagnosis tree is lower than the reference cost K_0 :

If $K_{Current} \leq K_0$, then $K_0 = K_{Current}$.
If $K_{Current} \geq K_0$, then K_0 keeps its value.

6.3 Iteration of Anytime method

We define an iteration as the addition of a test to the test subset $S_{Current}$ and the performance of the AO* algorithm.

The main drawback of this approach is the necessary increasing of the cardinality of the tests subset $S_{Current}$ which increases the AO* algorithm computation time. But, in the case of too bad solution, the modified stop condition allows to stop the AO* algorithm prematurely without high loss of time, and to go to the next iteration.

To avoid high computation time, the AO* algorithm execution time can be preferred to the complete exploration of the tests subset by bounding the cardinality of $S_{Current}$ with an user parameter n_{Max}. When this bound is reached, $S_{Current}$ is reduced to the last found discriminating optimal test subset S^*.

The main advantage is the computation time control. Indeed, this time may be bounded. So, given a duration, the best near-optimal diagnosis tree is found.

7. PERFORMANCE EVALUATION

The performance of the two proposed methods $AO*_{Dyn}$ and Any_{Min} (and its variant Any_{First}) is evaluated against the three other methods AO* with entire test set (optimal), AO^*_{Min} and AO^*_{First}, on 3 different systems Σ_i with $i \in \{1, 2, 3\}$ ordered by increasing complexity order. These systems are realistic automotive models.

The figure 2 shows the computation time for each system and each method, between brackets, the gap in percent w.r.t. the optimal cost and the number of iterations to obtain the optimal cost for the *Anytime* algorithms, between square brackets. The gap w.r.t. the optimal cost is not given for the *Anytime* algorithms as it is already zero.

	Σ_1	Σ_2	Σ_3
Optimal	$< 1''$(-)	$1'32''$(-)	$3'55''$(-)
AO^*_{Min}	$< 1''$(9%)	$< 1''$(61%)	$< 1''$(35%)
AO^*_{First}	$< 1''$(0%)	$1'05''$(0%)	$7''$(0%)
AO^*_{Dyn}	$< 1''$(0%)	$< 1''$(1%)	$3''$(1%)
Any_{Min}	$2''$ [4]	$47''$ [11]	$8''$ [4]
Any_{First}	$< 1''$ [1]	$1'05''$ [1]	$7''$ [1]

Fig. 2. Evaluation for 3 different systems

Figure 3 shows the number of iteration versus the gap in percent w.r.t the optimal cost for 3 systems $\{\Sigma_1, \Sigma_2, \Sigma_3\}$ when Any_{Min} method is used. Σ_2 converges slowly to the optimal cost because a particular test which discriminates a particular fault has to be included in the test sequence. In general, at least 10 iterations are needed to be at less than 10% from the optimal cost.

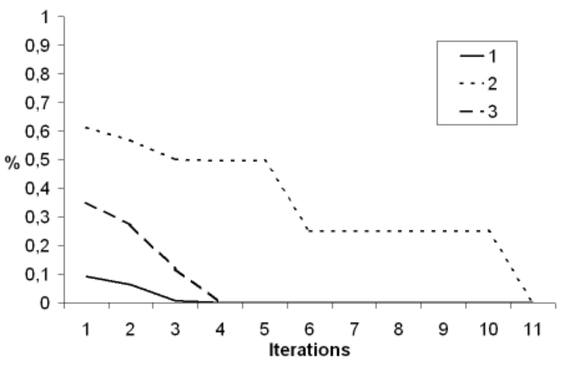

Fig. 3. Gap w.r.t. the optimal versus iterations number for the 3 systems

The AO^*_{First} and the *Anytime* methods are the two best ones for our application domain. The Any_{First} is the preferred method because its first iteration is equivalent to apply the AO^*_{First} method and has the same computational time. But the next iterations can give a better result if more time is available for the computation.

8. CONCLUSION

This paper explains how to generate automatically near-optimal diagnosis trees and how to control the computation time.

The processing time is proportional to the number n_S of tests in the initial test set S. This is why we propose to reduce dynamically, i.e. during the AO* algorithm, S in a subset $S^{'}$ such that $S^{'} \subseteq S$ in order to decrease the computation time. The obtained diagnosis tree is then near-optimal.

REFERENCES

Bagchi, A. and A. Mahanti (1983). Admissible Heuristic Search in AND/OR Graphs. *Theoritical Computer Science* **24**, 207–219.

Chakrabarti, P.P., S. Ghose, A. Acharya and S.C. de Sarkar (1989). Heuristic Search in Restricted Memory. *Artificial Intelligence* **41**, 197–221.

Chakrabarti, P.P., S. Ghose and S.C. de Sarkar (1988). Admissibility of AO* When Heuristics Overestimate. *Artificial Intelligence* **34**, 97–113.

Faure, P. P. (2001). An Interval Model-Based Approach for Optimal Diagnosis Tree Generation : Application to the Automotive Domain. PhD thesis. LAAS.

Moret, B. M. E. (1982). Decision trees and diagrams. *Computer Surveys* **14**(4), 593–623.

Pattipati, K. R. and M. Dontamsetty (1992). On a Generalized Test Sequencing Problem. *IEEE Transactions on Systems, Man and Cybernetics* **22**(2), 392–396.

Pattipati, K. R. and M. G. Alexandridis (1990). Application of Heuristic Search and Information Theory to Sequential Fault Diagnosis. *IEEE System, Man and Cybernetic* **20**(4), 872–887.

Raghavan, V., M. Shakeri and K. R. Pattipati (1999). Optimal and Near-Optimal Test Sequencing Algorithms With Realistic Test Models. *IEEE System, Man and Cybernetic - Part A* **29**(1), 11–26.

Tu, F. and K. R. Pattipati (2002). Rollout Strategies for Sequential Fault Diagnosis. *IEEE System, Man and Cybernetic - Part A.*

Yeung, R. W. (1994). On Noiseless Diagnosis. *IEEE System, Man and Cybernetic* **24**(7), 1074–1082.