CHEMICAL ENGINEERING TRANSACTIONS
VOL. 33, 2013
A publication of AIDIC
The Italian Association of Chemical Engineering
Online at: www.aidic.it/cet
Guest Editors: Enrico Zio, Piero Baraldi

ISBN 978-88-95608-24-2; **ISSN** 1974-9791
DOI: 10.3303/CET1333043

Test Sequencing Problem Considering Life Cycle Cost Based on the Tests with Non-Independent Cost

Zheng Hu*, Shigang Zhang, Yongmin Yang, Lijun Song, Ying Liu

Laboratory of Science and Technology on Integrated Logistics Support, College of Mechatronics and Automation, National University of Defense Technology, Changsha, Hunan, China
zhenghu@nudt.edu.cn

The previous diagnostic strategy considering life cycle cost is generated based on the assumption that the test placement costs are independent with each other. In this paper, the problem based on non-independent test cost is studied. A general algorithm is proposed to solve this problem. For the special scenario that the value of a placement function is binary, an algorithm with better computational efficiency is proposed. Computational experiment shows that our algorithms are capable of producing better results compared with the previous algorithms.

1. Introduction

Test sequencing problem is to construct a test sequence that achieves high fault isolation with low expected test cost. It is mostly formulated as an AND/OR graph search problem (Pattipati and Alexandridis, 1990; Tu and Pattipati, 2003; Kundakcioglu and Unluyurt, 2007), and only the execution cost is considered. In our previous research, the test sequencing problem considering life cycle cost was addressed and studied (Zhang et al., 2013), where the execution cost at the application stage and the placement cost at the design stage were considered. Here, the execution cost means the cost to use a test (e.g., power and time consumed to perform a test), and the placement cost means the once off cost to design the test system (e.g., cost to buy a sensor, volume and weight occupation).

However, in the previous research, the dependency relationships between the test placement costs are not taken into account (Zhang et al., 2013), which may not hold in the real-world systems. Indeed, it is possible that several tests compete or share a sensor or other test resources, which means that their costs are non-independent (Zhang and Hu, 2012). Taking a sensor detecting the voltage of alternating current for example, we can extract two features from the signal, i.e., amplitude and frequency. They are usually considered as different tests because the faults related to them are different, and their costs are set independently to generate a diagnostic strategy or the optimal sensor placement in the existing algorithms, which is unreasonable as they share the same sensor. Previous test sequencing methods either assume that the placement costs are independent or only consider the execution cost, which means that the results obtained may not be optimal as we wish. Consequently, we study this problem and proposed solution algorithms in this paper. It is formulated based on AND/OR graph searching. A general algorithm and an algorithm for the special scenario are proposed.

2. Problem formulation

We assume that the following information is available to formulate the problem:

1) a set of m+1 system states $S=\{s_1,\cdots,s_m,s_{m+1}\}$ associated with the system, and their corresponding priori probabilities, $P=\{p_1,\cdots,p_m,p_{m+1}\}$, where s_{m+1} denotes the fault-free state and $s_i(1\le i\le m)$ denotes a fault state of the system and $\sum_{i=1}^{m+1} p_i=1$.

2) a finite set of candidate tests $T=\{t_1,t_2,\cdots,t_n\}$, and the execution cost $CE=\{CE_1,CE_2,\cdots,CE_n\}$ corresponding to each test.

Please cite this article as: Hu Z., Zhang S., Yang Y., Song L., Liu Y., 2013, Test sequencing problem considering life cycle cost based on the tests with non-independent cost, Chemical Engineering Transactions, 33, 253-258 DOI: 10.3303/CET1333043

3) a finite set of test subsets $G=\{g_1,g_2,\cdots,g_u\}$, where each subset consists of a number of tests with non-independent cost. Let t_j^k denote the test t_j that belongs to g_k, viz., $g_k=\{t_j^k\}$. G should satisfy $\bigcup_{k=1}^{u} g_k=T$ and $g_k \cap g_r=\varnothing$ for $k\neq r$. Actually, G is a partition of the test set T.

4) a set of placement cost functions $CP=\{cp_1,cp_2,\cdots,cp_u\}$ corresponding to each element in G, which are the functions of the test selection state in each test subset g_k, $k=1,\cdots,u$.

5) diagnostic dictionary matrix (D-matrix) $D=[d_{ij}]_{(m+1)\times n}$, where d_{ij} is 1 if test t_j can detect fault s_i, and 0 otherwise. For the fault-free state, $d_{(m+1)j}=0$, $j=1,2,\cdots n$.

6) execution times N of the sequential fault diagnosis strategy in the life cycle period, which can be determined by the historical data or be calculated from the reliability data (Zhang et al., 2013), such as

$$N=\frac{T_m}{MTBF} \tag{1}$$

where T_m denotes the service life and *MTBF* is the *Mean Time Between Failure* of the equipment.

The sequential fault diagnosis problem is generally formulated as an AND/OR graph search problem. For the test sequencing problem considering life cycle cost, the total cost of a diagnostic strategy is (Zhang et al., 2013)

$$J=N\cdot J_e+J_p \tag{2}$$

where J_e is the execution cost of the diagnostic strategy, which can be found in (Raghavan et al., 1999; Tu and Pattipati, 2003). Let the binary variable set $Y=\{y_1,y_2,\cdots,y_n\}^T$ denote whether the tests are selected or not (1 or 0). y_j^k denotes whether t_j^k is selected and Y satisfies $Y=\bigcup_{k=1}^{u} Y_k$, $Y_k=\{y_j^k\}$. J_p can be written as

$$J_p=\sum_{k=1}^{u} cp_k(Y_k) \tag{3}$$

Our problem is to find a test sequence that minimizes the cost J shown in (2).

3. General solution algorithm

Let x denote an ambiguity node in the AND/OR graph, as shown in Figure 1, where t_j is a test and $\{x_{jp},x_{jf}\}$ are the successive nodes corresponding to the *pass* and *fail* outcomes.

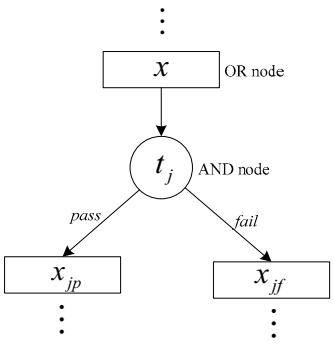

Figure 1: Ambiguity node x in the AND/OR graph

The traditional test sequencing problem only considering execution cost can be solved using the AO* algorithm. For the problem considering life cycle cost, both the execution cost and placement cost should be taken into account. According to our previous research (Zhang et al., 2013), the optimal test at node x in searching process is determined by

$$j^* = \arg\min_j \{N \cdot \hat{P}(x) \cdot h_e(x, j) + h_p(x, j)\} \quad (4)$$

where $\hat{P}(x) = \sum_{s_i \in x} p_i$. As the non-independent placement cost assumption does not impact the execution cost, $h_e(x, j)$ can be estimated using the same strategy proposed in (Zhang et al., 2013). For example, the heuristic function proposed in (Pattipati and Alexandridis, 1990) based on Huffman code can be applied. Specifically,

$$h_e(x, j) = CE_j + P(x_{jp})h_e(x_{jp}) + P(x_{jf})h_e(x_{jf}) \quad (5)$$

where $\{P(x_{jp}), P(x_{jf})\}$ are the probabilities that t_j passes and fails, $\{x_{jp}, x_{jf}\}$ are the successive nodes corresponding to the *pass* and *fail* outcomes as shown in Figure 1, $\{h_e(x_{jp}), h_e(x_{jf})\}$ are the values of evaluation function.

Details of diagnostic strategy generating algorithm (AOL algorithm) can be found in (Zhang et al., 2013). A short summary is given in Figure 2.

(i) Initialize a graph G consisting of a root node, which is the set of all the fault states.
(ii) Select a leaf node in G and expand it according to (4).
(iii) Backtrack based on the newly expanded nodes, update the heuristic function value, reselect the best branch via (4) and mark the optimal tree.
(iv) Go to step ii until the stop criteria is satisfied. Exit with the marked tree as the diagnostic strategy.

Figure 2: Summary of the AOL algorithm

Equation (4) will be applied in the expanding process and backtracking process. As $h_e(x, j)$ is calculated according to (5), calculation of $h_p(x, j)$ is the problem that needs to be studied in this paper. Let $T(x, j)$ denote the tests that have been applied on the path leading to node x and the path from x to the leaf nodes on the condition that t_j is chosen as the next optimal test. We propose to calculate $h_p(x, j)$ via

$$\begin{aligned} & h_p(x, j) = \min_Y \sum_{k=1}^{u} cp_k(Y_k) \\ s.t. \quad & DY \geq (1, \cdots, 1, 0)^T, \\ & D_i Y \geq (1, \cdots, 1)^T, \\ & D_i = [d_{zv}^i = d_{iv} \oplus d_{zv}], \\ & y_v \in \{0,1\}, \forall t_v \in T(x, j),\ y_v = 1, \\ & i = 1, \cdots, m, \\ & v = 1, \cdots, n, \\ & z = 1, \cdots, i-1, i+1, \cdots, m. \end{aligned} \quad (6)$$

Note that we assume that all the faults can be detected and isolated by the available tests. If this assumption is not valid, the faults that have the same test signature (the rows in the D-matrix are identical) can be treated as an equivalent single fault state, and equation (6) remains valid. The problem shown in (6) can be seen as a generalized set-covering problem. As this sub-problem will be solved for many times in the AND/OR graph searching process (expanding and backtracking process), we propose a greedy algorithm here. Similar idea has been applied in the process to generate the feasible solution in (Zhang and Hu, 2012).

Define a similarity matrix based on $T(x, j)$ to denote whether a fault can be detected and isolated,

$$L = [l_{iz}]_{m \times (m+1)} \quad (7)$$

For $i \neq z$, if the test signatures of s_i and s_z are identical based on $T(x, j)$, $l_{iz} = 1$; otherwise, $l_{iz} = 0$. For $i = z$, $l_{iz} = l_{i(m+1)}$.

Define a matrix $A_j = [a_{iz}^j]_{m \times (m+1)}$ to denote the resolving ability of test t_j, where

$$a_{iz}^{j}=\begin{cases} d_{ij}\oplus d_{zj} & i\neq z \\ d_{ij} & i=z \end{cases} \tag{8}$$

The symbol $\oplus$ denotes Exclusive OR operation. Then, the algorithm we propose to solve (6) is shown in Figure 3. In each iteration, the test that reduces the similarity most with the smallest cost is added to the solution. The algorithm stops once the feasible solution for (6) is generated.

(i) Define a variable $hp=0$;

(ii) Select test $t_{j'}$ via $j'=\arg\max\limits_{j\in\{1,2,\cdots,n\}\atop y_j=0}\dfrac{\sum\limits_{i=1}^{m}\sum\limits_{z=1}^{m+1}a_{iz}^{j}l_{iz}}{cp_k(Y_k)|_{y_j^k=1}-cp_k(Y_k)|_{y_j^k=0}}$, where k is the index that statisfies $y_j\in Y_k$;

(iii) Set $T(x,j)=T(x,j)\bigcup t_{j'}$, $y_{j'}=1$, $hp=hp+cp_k(Y_k)|_{y_{j'}^k=1}-cp_k(Y_k)|_{y_{j'}^k=0}$ and update the similarity matrix L defined by (7);

(iv) If $L\neq\mathbf{0}$, go to step (ii);

(v) Exit with $h_p(x,j)=hp$ as the result.

Figure 3: General algorithm

4. Solution algorithm for a special scenario

In the real world systems, there is a common scenario that the signal from a sensor can be transformed into several different tests, such as the tests extracted from the voltage sensor mentioned in section 1. It means that the placement cost function can be written as

$$cp_k(g_k)=\begin{cases} 0 & \text{all the tests in } g_k \text{ are not selected} \\ \overline{cp_k} & \text{otherwise} \end{cases} \tag{9}$$

where $\overline{cp_k}$ is the placement cost independent of the test selection state of g_k. We call the function shown in (9) a binary function. In this scenario, a subset g_k is termed as selected if any test in $g_k=\{t_j^k\}$ belongs to $T(x,j)$. In order to improve the computational efficiency, we treat the tests in the set g_k as a whole when designing the algorithms. Specifically, for $i\neq z$, if the test signatures of s_i and s_z are identical on tests $\{\bigcup g_k \mid g_k \text{is selected},\ k=1,\cdots,u\}$, $l_{iz}=1$; otherwise, $l_{iz}=0$. For $i=z$, $l_{iz}=l_{i(m+1)}$.

Similarly, the resolving ability matrix is defined for each test subset g_k:

$$R_k=[r_{iz}^{k}]_{m\times(m+1)} \tag{10}$$

In (10), for $i\neq z$, $r_{iz}^{k}=1$ if the test signatures of the tests in g_k between s_i and s_z are different, and $r_{iz}^{k}=0$ otherwise. For $i=z$, $r_{iz}^{k}=r_{i(m+1)}^{k}$. The algorithm for the special scenario we proposed is shown in Figure 4.

(i) Define a variable $hp=0$;

(ii) Select $g_{k'}$ via $k'=\arg\max\limits_{k\in\{1,2,\cdots,u\}\atop g_k\text{is not selected}}\dfrac{\sum\limits_{i=1}^{m}\sum\limits_{z=1}^{m+1}r_{iz}^{k}l_{iz}}{\overline{cp_k}}$;

(iii) Set $g_{k'}$ as selected, $hp=hp+\overline{cp_{k'}}$ and update the matrix L;

(iv) If $L\neq\mathbf{0}$, go to step (ii);

(v) Exit with $h_p(x,j)=hp$ as the result.

Figure 4: Algorithm for the scenario with binary placement cost function

5. Computational experiment

In this section, the proposed algorithms are simulated and compared on the systems of $\{m=10, n=15\}$. For comparison, simulations are also carried out using the AOL algorithm proposed in (Zhang et al., 2013) and the AO* algorithm proposed in (Pattipati and Alexandridis, 1990) based on the HEF_3 heuristic evaluation function (Pattipati and Alexandridis, 1990), where the placement costs are assumed to be independent and neglected, respectively. All the simulations were carried out in MATLAB on a PC with 2.53 GHz CPU, 2GB RAM. Results shown in Table 1-Table 4 are averaged over 50 Monte Carlo runs. We assume that the placement costs of at most 3 tests are non-Independent. The D-matrixes are generated randomly with an average density of 40% (percentage of ones in a D-matrix), with the assumption that all the faults can be detected and isolated. The execution cost of each test is generated randomly in [0, 1]. Let $f_{\mathbb{N}}$ denote the function value that $\mathbb{N}$ tests related to a placement cost function (element in CP) are selected. The values of placement cost function are generated by

$$f_{\mathbb{N}} = \max(\{f_{\mathbb{N}-1}\}) + h \cdot r \tag{11}$$

where r denotes a randomly generated fraction between 0 and 1, and h denotes a specified positive integer. Equation (11) will guarantee that the test cost will increase when more tests are added to the solution, which is practical in the real-world. The variable h is set to denote the strength of the cost dependences. A smaller value means that the tests share more resources, and a larger value means a stronger competition relationship between the tests. It is evident that $f_0 = 0$, which means that the placement cost should be zero if no test is applied. In our simulations, h is set to 0, 2 and 4 respectively. N has the values 0.1, 1, 10, and 100. Note that it should be larger than 1 in the real world systems. The reason that N can take the value less than 1 in the simulation is that N is set to test the influence of the weight of placement cost on the performance of different algorithms. The results are shown in Table 1-Table 4. In the special scenario, the value of a placement cost function is either 0 or in [0, 1], i.e., $\overline{cp_k}$ shown in (9) satisfies $\overline{cp_k} \in [0,1]$, $k = 1, \cdots, u$. In order to compare the results in a more clearly way, the costs obtained by different algorithms are divided by those obtained by AO* Algorithm.

Table 1: Simulation results for the general cases (h=0)

N	AO*		AOL		Proposed algorithm	
	time(sec)	*cost*	*time(sec)*	*cost*	*time(sec)*	*cost*
0.1	0.028	1	0.203	0.490	2.416	0.437
1	0.030	1	0.600	0.704	5.230	0.672
10	0.030	1	0.338	0.999	3.537	0.995
100	0.028	1	0.133	1.012	1.761	1.014

Table 2: Simulation results for the general cases (h=2)

N	AO*		AOL		Proposed algorithm	
	time(sec)	*cost*	*time(sec)*	*cost*	*time(sec)*	*cost*
0.1	0.029	1	0.206	0.638	1.480	0.422
1	0.031	1	0.556	0.718	3.978	0.566
10	0.032	1	0.335	0.956	3.915	0.942
100	0.031	1	0.174	1.005	2.113	1.010

Table 3: Simulation results for the general cases (h=4)

N	AO*		AOL		Proposed algorithm	
	time(sec)	*cost*	*time(sec)*	*cost*	*time(sec)*	*cost*
0.1	0.031	1	0.243	0.601	1.636	0.286
1	0.032	1	0.511	0.618	3.421	0.416
10	0.031	1	0.332	1.024	4.094	0.869
100	0.029	1	0.154	1.007	2.128	1.004

Table 4: Simulation results for the cases with binary placement function

N	AO*		AOL		Proposed algorithm		Algorithm for the special scenario	
	time(sec)	*cost*	*time(sec)*	*cost*	*time(sec)*	*cost*	*time(sec)*	*cost*
0.1	0.032	1	0.225	0.500	3.102	0.398	1.043	0.397
1	0.032	1	0.398	0.773	4.198	0.691	1.283	0.688
10	0.032	1	0.361	1.011	4.348	1.001	1.172	0.999
100	0.036	1	0.181	1.008	2.602	1.006	0.685	1.005

Form the simulation results shown in Table 1-Table 3 we can see that the proposed algorithms are effective to solve the test sequencing problem based on the non-independent test placement cost. Better results (the solutions with less cost) are obtained compared with the previous algorithms, especially when *N* is small and h is large. When *N* is large (*N*=100), the results obtained by all the algorithms are almost the same. This is because the placement cost only constitutes a tiny part of the total cost. For the cases with binary placement cost function, the computational time of the special algorithm is shorter than that of the general algorithm, with a result that has a slightly less life cycle cost, as shown in Table 4.

6. Conclusions

The test sequencing problem considering life cycle cost based on the tests with non-independent placement costs was studied and formulated in this paper. A general algorithm and an improved algorithm for the special scenario were proposed. Computational experiments were carried out to test the algorithms, which showed that they can produce better results than the previous algorithms. In the scenario that the placement cost function is binary, the improved algorithm has a better computational efficiency than the one for the general cases.

Acknowledgments

This work was financially supported by the China Civil Space Foundation with grant agreement no. C1320063131.

References

Kundakcioglu O.E., Unluyurt T., 2007, Bottom-Up Construction of Minimum-Cost AND/OR Trees for Sequential Fault Diagnosis. IEEE Trans. Syst., Man, Cybern. A, Syst., Humans, 37(5), 621-629.

Pattipati K.R., Alexandridis M.G., 1990, Application of Heuristic Search and Information Theory to Sequential Fault Diagnosis. IEEE Trans. Syst., Man, Cybern. A, Syst., Humans, 20(4), 872-886.

Raghavan V., Shakeri M., Pattipati K.R., 1999, Test sequencing problems arising in test planning and design for testability. IEEE Trans. Syst., Man, Cybern. A, Syst., Humans, 29(2), 153-163.

Tu F., Pattipati K.R., 2003, Rollout Strategies for Sequential Fault Diagnosis. IEEE Trans. Syst., Man, Cybern. A, Syst., Humans, 33(1), 86-99.

Zhang S., Hu Z., 2012, Optimal Selection of the Tests with Non-independent Cost, 2012 IEEE The 3rd International Conference on Mechanic Automation and Control Engineering (MACE 2012), Baotou, China, 3278-3281.

Zhang S., Hu Z., Wen X., 2013, Test Sequencing Problem Arising at the Design Stage for Reducing Life Cycle Cost. Chinese Journal of Aeronautics.