Bayesian Network Model Test Configuration Method based on Genetic and Binary Discrete Particle Swarm Combination Algorithm

Lu Han*, Xianjun Shi and Taoyu Wang
Naval Aviation University
*953339581@qq.com

Abstract. Test configuration is an important part in the process of testability design. Most of the existing test configuration methods are based on multi-signal flow diagram model and adopt genetic algorithm or particle swarm optimization (pso). However, there are many problems in this solution. First of all, the multi-signal flow diagram model has poor ability to express uncertain information, low model accuracy, large deviation in the calculation of testability indicators, and the model does not have the ability to learn and update. Secondly, the efficiency of genetic algorithm is low and the computation time is long, while the binary discrete particle swarm optimization algorithm is easy to fall into the local optimal. To solve the above two problems, a test configuration method based on the hybrid algorithm of genetic - binary discrete particle swarm optimization is proposed. This method can combine the global search ability of genetic algorithm with the optimal speed of binary particle swarm optimization, and use the bayesian network model to calculate more accurate testability indexes. It is proved that the algorithm can make full use of the high-precision information provided by the model, and the calculation speed is fast. And it is not easy to fall into the local optimal solution.

1. Introduction
The equipment structure is becoming more and more complex, and higher requirements are put forward for fault detection and isolation. Testability is a design feature that describes the system health status which can be monitored and tested. Testability model is a model established to design, analyze and evaluate testability of products. A good testability model not only helps equipment testability design development, but also has important significance for maintenance and fault testing and isolation during the equipment life cycle.

The Bayesian network testability model uses the Bayesian network to characterize the correlation between faults and tests[1]. With the help of the Bayesian network’s ability to process uncertain information, it can cope with the uncertainty of test transmission and test failure. Meanwhile, with the learning ability of the Bayesian network, the model can be updated with the increase of fault test data, which makes the model applicable to all stages of the equipment life cycle. In addition, the model has strong evidence processing capabilities. The utilization of relevant information is high, which can provide more information for R & D personnel and equipment maintenance personnel to make decisions[2].

Compared with other models, the test configuration based on the Bayesian network model has more benefits. In the test design of equipment, test indicators need to be assigned to various levels of
modules, but there are still problems about how to meet the index requirements at the minimum cost[3]. Bayesian networks have a strong ability to process uncertain information and can calculate tests more accurately[4]. Bayesian network models also provide more information. Through the Bayesian network model, test indicators can be assigned to the smallest test unit, and can be refined to the location of the measurement points. Through the above methods, find the best balance between test cost and index requirements. For example, some measuring points are equipped with the best sensors, but still cannot meet the requirements. The Bayesian network model can calculate how much the indicators of the alternative or other measuring points need to be improved to compensate. For another example, if the measuring point has been selected, there are multiple configuration schemes for changing the sensor of the measuring point, and the scheme with the lowest cost meeting the testability index can be calculated through the Bayesian network[5]. However, it greatly increases the amount of calculation, which will take a long time for complex systems.

Completing the Bayesian network test configuration manually is a very heavy work with heavy workload and many repetitive tasks. The commonly used optimization algorithms include genetic algorithm and particle swarm optimization[6]. Genetic algorithm (ga) can search the global optimal solution but the calculation speed is slow, while particle swarm optimization (pso) is fast but sometimes falls into local extremum. Therefore, in view of the above problems, this paper proposes a test configuration method for combined genetic and binary discrete particle swarm optimization (pso) based on the Bayesian network model, which can perform global search with fast computing speed and make full use of the information provided by the model.

2. The testability model of Bayesian networks

There is no specific guidance method for test configuration. Generally, some optimization algorithms are used to solve this problem, such as genetic algorithm, firefly algorithm, bat algorithm, particle swarm optimization algorithm and so on. A combined genetic and binary discrete particle swarm optimization (pso) algorithm is applied to the testable model of Bayesian networks. So let's talk about these two algorithms.

2.1. Genetic Algorithm

Genetic algorithm (ga), a newly developed optimization algorithm, is a search algorithm based on natural selection and genetic principles. With the iterative operation of the algorithm, the process is like biological evolution[7]. The good features are handed down and the unadaptive features are eliminated. The new generation is better suited to the environment than the previous generation, and the whole group is moving in a better direction. Because genetic algorithm has mutation operator, it can solve the problem of local optimization.

Basic concepts in genetic algorithm:

- Chromosome: a collection of genetic factors. In genetic algorithms, it is one-dimensional string structured data.
- Genetic factor: the basic unit of genetic material that controls biological traits.
- Loci: the positions of genetic factors on a chromosome that determine what information is inherited.
- Phenotype: external expression of a trait determined by the chromosome.
- Individual: An entity with characteristic chromosomes.
- Group: A collection of individuals.
- Fitness: the degree to which individuals adapt to their environment.
- Selection: the operation of selecting several pairs of individuals from a group with a certain probability is called selection.
- Crossover: the interchanging of parts of two chromosomes, also known as recombination.
- Variation: the operation of making genetic factors change with a certain probability is called variation.
- Coding: mapping from phenotype to genotype.
Decoding: mapping from genetic subtypes to phenotypes.
Process:
1. Parameter coding
   In general, genetic algorithms cannot directly deal with the parameters of the problem space, so they need to establish the relationship between the parameters of the problem space and chromosomes, and convert them into chromosomes composed of genes in a certain structure.
2. Generation of initial population:
   The initial population was formed to receive subsequent natural selection.
3. Design of fitness function:
   The main method is to convert the objective function of the problem into an appropriate fitness function.
4. Genetic operation design:
   Includes: selection, crossover, variation. Notice that these three are random operators, and the migration of individuals in the population to the optimal solution is random; The effect of genetic manipulation is closely related to operation probability, population size and fitness function setting. The operation method of genetic operator is related to individual coding mode.
5. Control parameter design:
   Set population size, genetic algebra, crossover probability, mutation probability, etc.

2.2. Binary discrete particle swarm optimization algorithm
Particle swarm optimization (PSO) searches according to the speed of particles and adjusts the speed by the optimal location of the Shared group and its own historical optimal location[8]. Its essence is also based on population and evolution. However, unlike genetic algorithm, it does not select, cross or vary individuals, but updates particle velocity, making each particle move towards its own historical optimal position and the historical optimal position of the population, so as to achieve the optimization of the whole population.

Basic concepts in particle swarm optimization:
- Particle current position: \( X_i = \{x_{i1}, x_{i2}, \ldots, x_{in}\} \);
- The historical optimal position searched by the particle itself, the determination of the optimal position comes from the level of the evaluation function: \( P_{best-i} = \{p_{i1}, p_{i2}, \ldots, p_{in}\} \);
- Historically Optimal Location Searched by Particle Swarm: \( G_{best} = \{g_1, g_2, \ldots, g_n\} \);
- Particle search speed, \( v_{id} = \omega \times v_{id} + c_1 \times rand() \times (p_{id} - x_{id}) + c_2 \times rand() \times (g_d - x_{id}) \), where \( \omega \) is the inertia constant and defaults to 1 in this article; \( c_1, c_2 \) is the learning factor and \( rand() \) is a random number[9].

The entire algorithm is optimized around speed updates:
- Due to the use of binary, there is no position update formula of the particle swarm algorithm, so the speed is converted to a probability of 0/1 in binary. This requires the speed to be mapped to the [0,1] interval. The original mapping method uses the sigmoid function, but there are many problems make the randomness stronger and stronger as the iteration progresses. In the hybrid algorithm, the role of the binary particle swarm is mainly to speed up the optimization calculation. Therefore, the improved sigmoid[10] function is used to make the particles have a faster convergence speed and reduce randomness.

\[
s(v_{id}) = \begin{cases} 
1 - \frac{2}{1+\exp(-v_{id})} & v_{id} \leq 0 \\
\frac{2}{1+\exp(-v_{id})} - 1 & v_{id} \geq 0
\end{cases}
\]  

Determine the position change of the binary particle swarm:
When \( v_{id} < 0 \),
\[
x_{id} = \begin{cases} 
0 & rand() \leq s(v_{id}) \\
x_{id} & otherwise
\end{cases}
\]  

\( 1 \)
When $v_{id} > 0$, 
\[
x_{id} = \begin{cases} 
1 & \text{rand()} \leq s(v_{id}) \\
x_{id} & \text{otherwise}
\end{cases}
\]  
\(3\)

3. Bayesian network model

The Bayesian network testability model is a testability model that uses Bayesian networks to represent faults and test correlations. Compared with the correlation model, the bayesian network model has a strong ability to deal with uncertain information. It has high utilization of fault data, low distortion of model, and more accurate testability index. Moreover, with the learning ability of bayesian network, the model can be constantly updated with the increase of data. Configure the corresponding tests at all locations of the equipment where tests can be deployed, and the test metrics for each test are given by expert knowledge or historical experience or usage data.

Bayesian network model can be used for precise reasoning and evidence processing by cluster tree and connection tree. Therefore, the failure detection rate, false alarm rate, missed detection rate and other information of each test for each module unit can be calculated. Of course, it is also possible to calculate the detection probability of single failure mode $f_i$ under the current test configuration scheme.

\[
\gamma_{FDR} = \sum_{i=1}^{L} \alpha_{f_i} \gamma_{FDR_i} - \sum_{i=1}^{L} \alpha_{f_i} \gamma_{FDR_i} \prod_{x=1}^{L} p_{d_{ix}} - \cdots - (-1)^{L-1} \prod_{x=1}^{L} p_{d_{ix}}
\]  
\(4\)

pd_{ix} represents the detection rate of fault $f_i$ by test $t_i$ inferred by processing the evidence through the Bayesian network.

The fault detection rate of the system is:

\[
\gamma_{FDR} = \frac{\sum_{i=1}^{L} \alpha_{f_i} \gamma_{FDR_i}}{\sum_{i=1}^{M} \alpha_{f_i}}
\]  
\(5\)

$L_0$ is the number of faults that can be detected, and $m$ is the total number of faults.

Here, it is considered that the failure detection rate of the test for a fault is more than 70%, that is, it has the ability to detect the fault.

The fault isolation rate is calculated by Equation (6).

\[
\gamma_{FIR} = \frac{\sum_{i=1}^{L} \alpha_{f_i} \gamma_{FDR_i}}{\sum_{i=1}^{L} \alpha_{f_i}}
\]  
\(6\)

$L_4$ represents the number of faults that can meet the fault isolation requirements.

4. Genetic and binary discrete particle swarm combination algorithm

Although the improved discrete binary particle swarm algorithm improves the convergence speed, it weakens the original algorithm's weak global optimization search capability. To solve this problem, a genetic-discrete binary particle swarm combination algorithm is proposed, which is enhanced by the genetic algorithm.

The principle flow of the algorithm is as follows:

Firstly, the selection, crossover and mutation of genetic algorithm are carried out according to the fitness of each generation. After the prescribed algebra, the optimal test configuration is quickly found through the discrete binary particle swarm optimization algorithm. The process is shown in the figure 1.
Since the object is a bayesian network model, the algorithm has higher utilization of fault information and more detailed test configuration. According to the limitation of sensor performance or cost, at each measuring point, the r&d personnel shall provide a number of testing schemes with different indicators and different costs to establish the structure of the bayesian network model, and the CPT information in the network shall be determined by the selected test scheme.

Establish optimization constraint conditions:

\[
\begin{align*}
\gamma_{FIR} & \geq \text{FIR} \\
\gamma_{FDR} & \geq \text{FDR}
\end{align*}
\]  

(7)

Where FIR, FDR is the index required for the testability design of the system, and \(\gamma_{FIR}, \gamma_{FDR}\) is the testability index calculated through the Bayesian network model.

The selected fitness function is:

\[
f(s) = \frac{\gamma_{FIR} \gamma_{FDR}}{\sum_{i=1}^{n} c_i}
\]  

(8)

In the formula, \(c_i\) is the cost of the arranged tests, and \(n\) is the number of the arranged tests.

5. Example verification

Based on simulation data and expert experience, a test model for the Bayesian network of a radio altimeter transceiver module is established, as shown in the figure 2.

**Figure. 1:** Algorithm flow chart

**Figure. 2:** The bayesian network model of transceiver

Testability design requirements: fault detection rate FDR reaches 97%, fault isolation rate FIR is 55%. According to the above requirements, three algorithms were used to test the configuration.

The genetic algorithm based on multi-signal flow graph model, the genetic algorithm based on bayesian network and the algorithm in this paper were used to carry out the test configuration work for the transceiver module. The results are shown in Table 1.
Table 1: Testability and performance parameters

| Algorithm                  | Fault detection rate | Fault isolation rate | Cost | Operation time |
|----------------------------|----------------------|----------------------|------|----------------|
| The genetic algorithm based on MSFG | 100%                 | 54.3%−62.5%          | 3.89~6.20 | 21s           |
| The genetic algorithm based on BN   | 98.52%               | 56.5%                | 3.89  | 98s            |
| the algorithm in this paper        | 98.52%               | 56.5%                | 3.89  | 32s            |

According to the above table, it can be seen that the genetic and binary particle swarm combination algorithm for Bayesian networks can effectively reduce the test cost, increase the calculation speed, and search for the global optimal solution. The genetic algorithm has a slow calculation speed, especially the genetic algorithm for the multi-signal flow graph model only considers the location of the test, and does not consider what kind of test should be configured at that location to meet the testability index while reducing costs.

6. Conclusion
When there is no specific test configuration method for Bayesian network testability model, this paper proposes a genetic and binary particle swarm optimization algorithm, which can make full use of the information provided by Bayesian network model, and reduce the cost as much as possible while meeting the testability design index. Moreover, this method is fast in calculation and has the ability of global search, so it is not easy to fall into the local optimal solution. This algorithm can give full play to the value of Bayesian network model, improve testability design efficiency and reduce workload.

Reference
[1] Wang Xiaowei et al. Modeling and analysis of system testability based on Bayesian network. China Test, 2011. 37 (05): 90-93.
[2] van Bork Riet, Rhemtulla Mijke, Waldorp Lourens J, Kruijs Joost, Rezvanifar Shirin, Borsboom Denny. Latent Variable Models and Networks: Statistical Equivalence and Testability. [J]. Multivariate behavioral research, 2019.
[3] Xiaofeng Tang, Aiqiang Xu, Ruifeng Li, Min Zhu, Jinling Dai. Simulation-Based Diagnostic Model for Automatic Testability Analysis of Analog Circuits [J]. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2018, 37 (7).
[4] Dai Jing, et al. A new method for modeling and analysis of aeronautical mechatronic systems. Journal of Aeronautics, 2010. 31 (02): 277-284.
[5] A. Gray. Modern Differential Geometry. CRE Press, 1998.
[6] Tian-Mei Li, Cong-Qi Xu, Jing Qiu, Guan-Jun Liu, Qi Zhang, Gang Li. The Assessment and Foundation of Bell-Shaped Testability Growth Effort Functions Dependent System Testability Growth Models Based on NHPP [J]. Mathematical Problems in Engineering, 2015, 2015.
[7] He, Xing; Wang, Hongli; Lu, Jinghui. Testability Analysis of Inertial Measure Unit Based on Multi-signal Model [J]. Sensors & Transducers, 2012, 16.
[8] Wu Xinfeng et al. Test optimization based on improved binary particle swarm genetic algorithm. Journal of Ordnance and Equipment Engineering, 2019. 40 (05): 146-150.
[9] Wei Shengyun et al. Improved node selection algorithm for binary particle swarm optimization. Journal of Xidian University, 2016. 43 (02): 150-156.
[10] Liu Jianhua, Yang Ronghua and Sun Shuihua. Analysis of discrete binary particle swarm algorithm. Journal of Nanjing University (Natural Science Edition), 2011. 47 (05): 504-514.