Proceedings of the 14th Asian Control Conference
July 5-8, 2024, Dalian, China

Functional Test-Cost Reduction Based on Optimization Modeling and Congestion Control

1st Peng Bai
Department of Automation, USTC
Hefei, China
baipeng@lenovo.com

2nd Kangcheng Wang
Institute of Artificial Intelligence, Hefei Comprehensive National Science Center
AHU-IAI AI Joint Laboratory, Anhui University
Hefei, China
kcwang@iai.ustc.edu.cn

3rd Yun-Bo Zhao
Department of Automation, USTC
Institute of Artificial Intelligence, Hefei Comprehensive National Science Center
Institute of Advanced Technology, USTC
Hefei, China
ybzhao@ustc.edu.cn

4th Yu Kang
Department of Automation, USTC
Institute of Artificial Intelligence, Hefei Comprehensive National Science Center
Institute of Advanced Technology, USTC
Hefei, China
kangduyu@ustc.edu.cn

5th Wenhao Fang
Institute of Artificial Intelligence, Hefei Comprehensive National Science Center
AHU-IAI AI Joint Laboratory, Anhui University
Hefei, China
wa23201077@stu.ahu.edu.cn

***Abstract*—Functional testing is a crucial process to guarantee the quality of electronic products. In recent years, the cost of functional testing has been rising with the increasing complexity of products, and reducing testing costs is of great significance to the economic efficiency of electronic manufacturing enterprises. Related research has not yet fully considered the issue of the non-uniform distribution of functional testing samples in practical applications, making it challenging to ensure the effectiveness of reducing testing costs. Motivated by the concept of TCP congestion control algorithms, this article presents an enhanced congestion control algorithm tailored for the functional testing process and proposes a method to reduce testing costs accordingly. The proposed method can design dynamically changing testing strategies based on optimal modeling. On the simulation data closely resembles the actual data, the proposed method can significantly reduce the testing costs compared to the pure optimization modeling method.**

***Index Terms*—Intelligent manufacturing, optimization modeling, congestion control, functional testing, cost reduction.**

I. Introduction

Functional testing is an important method for ensuring factory yields by testing the functionality of electronic products, especially their motherboards [1]. The testing process comprises two stages: the motherboard testing stage and the finished product testing stage. With the increasing complexity of products, the cost of functional testing is rising. Therefore, reducing the cost of functional testing is crucial for enhancing the economic efficiency of electronic manufacturing companies [2].

This work was supported by the Key Research and Development Program of Anhui (No. 202104a05020064),(Corresponding authors: Yun-Bo Zhao, Yu Kang).

The cost of the functional testing process mainly consists of the cost of testing functional items and the cost of reworking defects that were not identified. Both types of costs can be converted to time costs and are determined by the testing strategy of the motherboard testing stage. This strategy is represented by a binary vector or matrix that specifies which items and samples are to be tested. Since the two types of costs are tightly coupled, the total cost does not vary monotonically with the testing strategy. An efficient testing strategy can effectively balance the two types of costs and reduce the overall cost. Thus, designing an effective testing strategy is essential for reducing the cost of functional testing [3].

Related studies can be divided into two types: test ordering and test selection [4]. Test ordering methods arrange the order of the testing items and terminate the testing process when identifying defective items. As a result, these methods can reduce the testing cost of defect samples. Test selection methods can reduce the testing cost of all samples by selecting a portion of items for testing [5]. Since the yields of current electronic products are relatively high, test selection methods are often more effective than test ordering methods in reducing the cost of functional testing and have been widely studied in various types of circuits in recent years [6], [7]. Existing methods typically assign the same testing state to all samples. This means that the testing strategy is commonly represented as a binary vector, which limits the potential reduction of testing costs. To solve this problem, Kang et al. [8] set the testing proportion for each testing item through optimization modeling to effectively balance the testing cost and rework cost, thereby significantly reducing the overall testing cost.

However, existing studies typically assume that the defects follow a uniform distribution. In practice, functional defects resulting from manufacturing defects are often concentrated. The strategy based on the assumption of a uniform distribution is challenging to apply to real-world situations: the missed detection rate increases significantly when defects are concentrated, while excessive testing time is required during other periods, making it difficult to achieve cost reduction through testing effectively in both scenarios. Therefore, it is necessary to fully consider the practical issue of non-uniform distribution of samples and design dynamically changing testing strategies.

TCP congestion control algorithm is a network traffic control algorithm that can dynamically adjust the congestion window based on real-time network conditions through phases including slow start and congestion avoidance [9]. This approach effectively prevents network congestion and has been extensively studied and implemented [10]. Utilizing the congestion control algorithm for dynamic adjustment of testing proportion is a feasible way to further decrease testing costs. Considering that the change in testing proportion and the adjustment of congestion window share similarities but also have significant differences, the algorithm design should be integrated with the specific characteristics of the functional testing item to ensure the effectiveness of the algorithm. Up to now, no relevant research has been identified to enhance the congestion control algorithm for addressing the issue of non-uniform distribution of real industrial data samples.

Motivated by the concept of congestion control algorithms, this article introduces an enhanced congestion control algorithm. Based on the results of optimization modeling, the algorithm integrates the features of functional items and allows for the sample distribution. The enhanced algorithm can generate a dynamic testing strategy and significantly reduce testing costs compared to the pure optimization modeling method when applied to simulation data that closely resembles actual data.

The main contents of the rest of the article are as follows: Section II designs a functional testing cost reduction method based on optimal modeling and congestion control algorithm; Section III conducts simulation experiments to verify the effectiveness of the method; Section IV provides a comprehensive summary of the entire article.

II. Method

This section provides a brief review of optimal modeling of the functional testing process, proposes enhanced congestion control algorithms for designing dynamic testing strategies, and lists method evaluation criteria.

A. Optimization Modeling of Functional Testing

In our previous work, we developed the optimization model of the functional testing process by establishing a general framework for the process [8]. This subsection provides a brief review of that modeling process.

Denote the motherboard testing strategy as $\mathcal{S}$. When proportional testing is used, the testing strategy specifies the proportion of each functional testing item to be tested. In this case, the strategy can be expressed as:

$$\mathcal{S} : (p_1, \dots, p_{N_T}) \tag{1}$$

where $p_j \in [0,1] (j = 1, \dots, N_T)$ denotes the proportion of the jth functional testing item tested, and N_T denotes the total number of testing items.

To minimize the cost of testing while satisfying the necessary constraints, the optimization problem for establishing a functional testing process is as follows:

$$\min_{\mathcal{S} \in \mathbb{S}} c(\mathcal{S}) \tag{2}$$

$$\text{s.t. } \boldsymbol{g}(\mathcal{S}) \leq \mathbf{0} \tag{3}$$

where $\mathbb{S}$ denotes the set of possible testing strategies; $c(\mathcal{S})$ denotes the cost that can be optimized when adopting the testing strategy $\mathcal{S}$. This cost is referred to as the average effective cost and is used to reflect the total testing cost of the functional testing process in this article; $\boldsymbol{g}(\mathcal{S}) = [g_1(\mathcal{S}), \dots, g_4(\mathcal{S})]^{\mathrm{T}}$ denotes a vector of constraints, where $g_1(\mathcal{S})$ denotes that the average testing time per motherboard must not exceed the average manufacturing time of the motherboard, $g_2(\mathcal{S})$ denotes that the number of defective motherboards reworked must not be lower than the number of defective motherboards identified by the testing machine, $g_3(\mathcal{S})$ denotes that the number of defective finished products reworked must not be lower than the number of defective motherboards assembled into finished products after the functional testing of the motherboards, and $g_4(\mathcal{S})$ denotes that the total number of rework workers must not be higher than the number of rework workers used by the current unoptimized testing strategy.

The average effective cost in Eq. (2) is calculated as:

$$c(\mathcal{S}) = \sum_{j=1}^{N_T} p_j c_{Tj}^m + \Delta c_R \prod_{j=1}^{N_T} (1 - p_j r_{Dj}) \tag{4}$$

where c_{Tj}^m denotes the average testing cost of the jth testing item of each motherboard; Δc_R denotes the difference between the cost of repairing each defective laptop in the finished product testing stage and the cost of repairing the defective motherboards it contains in the motherboard testing stage; and r_{Dj} denotes the defective rate of the jth item of the test, which is considered the overall inherent defective rate of the testing item and can be determined by analyzing testing results over an extended period.

Solve the aforementioned nonlinear programming problem to derive the optimal solution and denote it as:

$$\mathcal{S}^* : (p_1^*, \dots, p_{N_T}^*) \tag{5}$$

B. Enhanced Congestion Control Algorithm

Although the optimization problem in Section II-A can help achieve an optimal testing strategy, in practice, the distribution of defective items is often highly non-uniform because defects resulting from the production process tend to be concentrated. The aforementioned issue causes the defective rate r_{Dj} to fluctuate significantly. This results in a high testing time cost

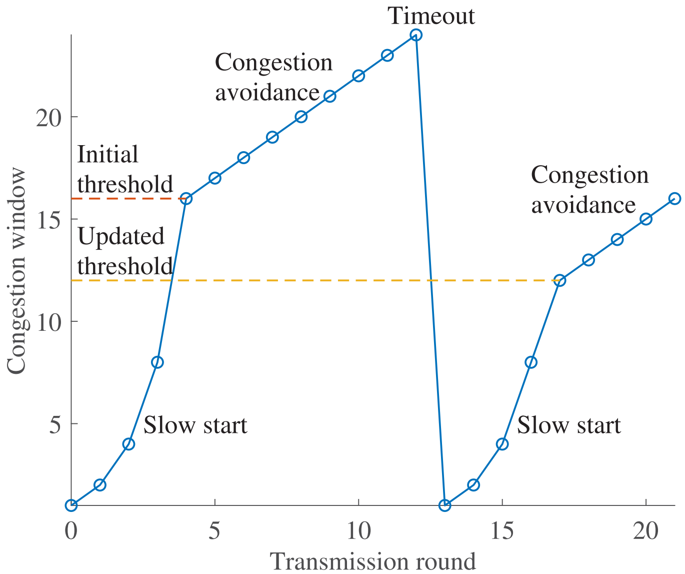

Fig. 1: Schematic diagram of a conventional congestion control algorithm

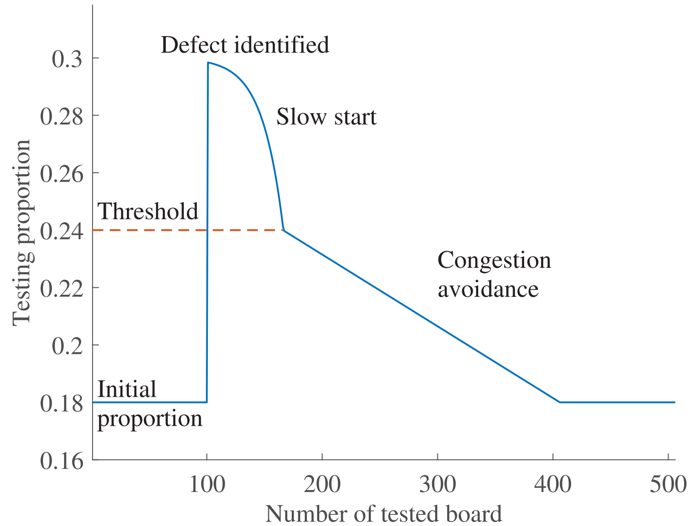

Fig. 2: Schematic diagram of the enhanced congestion control algorithm tailored for functional testing

when the defective rate is low. Conversely, when the defective rate is high, the rate of identified defects at the motherboard testing stage decreases, leading to an increase in the rework rate at the subsequent finished product testing stage and an increase in the rework cost within a short timeframe. This shows that there is still much room for optimizing the total testing cost.

To effectively solve this problem, this subsection proposes a test-cost reduction method to design a testing strategy that allows for sample distribution and can dynamically adjust the testing proportion. This is achieved by enhancing the existing congestion control algorithm to further reduce the total testing cost.

The curve of the congestion window of a conventional congestion control algorithm concerning the number of transmission rounds is shown in Fig. 1. At the beginning of the curve, it usually rises in the form of an exponential function, known as the "slow start" phase. When the initial threshold is reached, the curve transitions to rising in the form of a linear function, referred to as the "congestion avoidance" phase. If a timeout occurs, the curve reverts to the initial value and adjusts the threshold to half the size of the congestion window at the time of the timeout event. Next, the curve restarts slowly, still rising as an exponential function, and changes again to rise as a linear function when the updated threshold is reached, and re-entering the congestion avoidance phase. These procedures are repeated.

Motivated by the dynamic changes in the congestion window, this article dynamically adjusts the testing proportion of functional testing to achieve the goal of identifying as many defects as possible while using the lowest possible testing proportion.

The variation of the testing proportion with the number of testing rounds is illustrated in Fig. 2. At the beginning of the algorithm, the testing proportion is set to a low initial value. When a defect is identified, the testing proportion increases to the maximum value. Due to the high probability of defects reappearing within a short timeframe, the testing proportion is initially decreased following an exponential function. This phase corresponds to the "slow start" phase in conventional congestion control algorithms. When the testing proportion decreases to a pre-set threshold, there is still a relatively high probability of defects reappearing within a relatively short timeframe. Therefore, it is transformed into a linear function to decrease the testing proportion until it returns to its initial value. This phase corresponds to the "congestion avoidance" phase in conventional congestion control algorithms. If a new defect is identified at any phase, the testing proportion is reset to the maximum and then gradually decreases again based on the functions mentioned earlier.

C. *Congestion Control-Based Dynamic Testing*

This subsection outlines the dynamic testing method based on the enhanced congestion control algorithm proposed in the previous subsection.

Denote the initial, maximum, and threshold values of the enhanced algorithm as $\boldsymbol{p}^{\text{init}}$, $\boldsymbol{p}^{\text{max}}$, and $\boldsymbol{p}^{\text{thresh}}$, respectively. Denote the parameters of the exponential function as $\boldsymbol{\alpha}_1$ and $\boldsymbol{\alpha}_2$, and denote the parameters of the linear function as a and $\boldsymbol{b}$; denote the number of the motherboard on which the defect is identified, the number of the last motherboard descending in the form of an exponential function, and the number of the last motherboard descending in the form of a linear function as $\boldsymbol{n}^0$, $(\boldsymbol{n}^{\text{ss}})_{\text{ub}}$, and $(\boldsymbol{n}^{\text{ca}})_{\text{ub}}$, respectively. The dynamic testing strategy for each testing item is indicated as:

$$\mathcal{S}^{\text{dyn}} : (p_1^{\text{dyn}}, \ldots, p_{N_T}^{\text{dyn}}) \tag{6}$$

Based on the description in the previous subsection, the function of the dynamic testing proportion of the jth item is defined as:

$$p_j^{\text{dyn}} = \begin{cases} p_j^{\text{init}}, & \text{if } n \in \left[0, n_j^0 - 1\right] \\ p_j^{\max} - \alpha_{1j} e^{n/\alpha_{2j}}, & \text{if } n \in \left[n_j^0, (n_j^{\text{ss}})_{\text{ub}}\right] \\ an + b_j, & \text{if } n \in \left[(n_j^{\text{ss}})_{\text{ub}} + 1, (n_j^{\text{ca}})_{\text{ub}}\right] \\ p_j^{\text{init}}, & \text{if } n \in \left[(n_j^{\text{ca}})_{\text{ub}} + 1, N_B\right] \end{cases} \tag{7}$$

where n denotes the number of motherboards undergoing testing during the motherboard testing stage.

To ensure the effectiveness of the dynamic testing method, the selection of the testing proportion should be closely related to the characteristics of the testing items. Note that the established optimization model in Section II-A is designed based on the characteristics of testing items, and the optimized testing strategy already contains key information such as the testing time and the defective rate of each testing item. Therefore, this subsection sets the parameters of the enhanced algorithm based on the results of the optimization modeling and implements the dynamic adjustment of the testing proportion accordingly.

In Eq. (7), the parameters $\boldsymbol{\alpha}_1$ and $\boldsymbol{\alpha}_2$ need to be determined by integrating the characteristics of the testing items with practical experience. Here, the benchmark values α_1^{uni} and α_2^{uni} are first set based on the expert's experience. Subsequently, the optimization results are combined with Eq. (5), which considers the values of each item as follows:

$$\alpha_{1j} = p_j^* / \alpha_1^{\text{uni}} \tag{8}$$

$$\alpha_{2j} = p_j^* \alpha_2^{\text{uni}} \tag{9}$$

In addition, in Eq. (7), the parameter a is set according to expert experience; the parameters $(\boldsymbol{n}^{\text{ss}})_{\text{ub}}$, $\boldsymbol{b}$, and $(\boldsymbol{n}^{\text{ca}})_{\text{ub}}$ can be calculated as follows:

$$(n_j^{\text{ss}})_{\text{ub}} = \alpha_{2j} \left(\ln(p_j^{\max} - p_j^{\text{thresh}}) - \ln(\alpha_{1j}) \right) \tag{10}$$

$$b_j = p_j^{\text{thresh}} - a (n_j^{\text{ss}})_{\text{ub}} \tag{11}$$

$$(n_j^{\text{ca}})_{\text{ub}} = (p_j^{\text{init}} - b_j)/a \tag{12}$$

In the enhanced algorithm, the selection of the initial value $\boldsymbol{p}^{\text{init}}$, the maximum value $\boldsymbol{p}^{\max}$, and the threshold value $\boldsymbol{p}^{\text{thresh}}$ are crucial for the algorithm's effectiveness. Here again, by utilizing the optimization results (Eq. (5)) in Section II-A, the aforementioned parameters are set as follows:

$$p_j^{\text{init}} = \beta^{\text{init}} p_j^* \tag{13}$$

$$p_j^{\max} = \beta^{\max} p_j^* \tag{14}$$

$$p_j^{\text{thresh}} = (p_j^{\text{init}} + p_j^{\max})/2 \tag{15}$$

where β^{init} and $\beta^{\max}$ are empirically set parameters, the principle of setting them is: $\beta^{\text{init}} < \beta^{\max}$, and the values of both should be chosen so that the ranges of p_j^{init}, $p_j^{\max}$, and p_j^{thresh} are within [0, 1].

By setting the testing proportion of each testing item using the Eq. (7), the testing proportion can be dynamically adjusted based on real-time testing results. Considering that the testing strategy from the proposed method may require more rework workers to ensure the progress of rework, the constraint $g_4(\mathcal{S})$ related to the total number of rework workers is modified as follows:

$$n_R^{ml}(\mathcal{S}) \le n_R^{ml}(\mathcal{S}_0(\mu_0)) + \Delta n_R^{ml} \tag{16}$$

where $n_R^{ml}(\cdot)$ denotes the total number of rework workers, $\mathcal{S}_0(\mu_0)$ denotes the current testing strategy, and Δn_R^{ml} denotes the number of additional rework workers that can be added.

D. Evaluation Criteria

In this article, the following four criteria are used to evaluate the method's performance:

1) The average effective cost, i.e., the $c(\mathcal{S})$ in Eq. (4);
2) The average testing time $\bar{t}_T(\mathcal{S})$, and the calculation of this criterion is described in detail later in Eq. (17);
3) Number of testing machines required $q(\mathcal{S})$;
4) Total number of rework workers $n_R^{ml}(\mathcal{S})$.

A lower value for each of these indicators indicates a more efficient strategy.

The criterion $\bar{t}_T(\mathcal{S})$ denotes the average time spent by each motherboard on a single testing machine for the items specified in the testing strategy $\mathcal{S}$. Considering that the method proposed in this article dynamically adjusts the testing proportion, the conventional fixed-proportion testing time calculation method cannot be applied. Therefore, the formula for calculating the average testing time in this article is:

$$\bar{t}_T(\mathcal{S}) = \frac{1}{N_B} \sum_{i=1}^{N_B} \sum_{j=1}^{N_T} T_{i,j} | (\mathcal{S}_{i,j} = 1) \tag{17}$$

where N_B denotes the total number of motherboards undergoing testing during the motherboard testing stage, $T_{i,j}$ and $\mathcal{S}_{i,j}$ denote the testing time and testing state of the jth testing item of the ith motherboard, respectively. The initial value of $\mathcal{S}_{i,j}$ is randomly generated based on the proportion in $\mathcal{S}$. When the testing ratios are adjusted using the method proposed in this article, the testing statuses of several subsequent motherboards are randomly regenerated based on the new testing ratios obtained by the enhanced algorithm.

III. Case Study

In this section, the effectiveness of the proposed method is verified using simulation data.

A. Data Preparing

Actual testing data of a certain type of motherboard from an electronics manufacturer is collected, and simulation data that closely resembles the actual data are generated to safeguard trade secrets.

The motherboard has a total of 10 testing items. To simulate the phenomenon of non-uniform sample distribution in real production, two sets of testing data containing testing results of 10^5 and 4.2×10^3 motherboards, respectively, are generated by varying yields. The former set is assigned as the training set, while the latter is assigned as the test set. The yields of 2000 motherboards at the beginning and end of the test set are higher than the yields of the training set, while the yields of

200 motherboards in the middle of the test set are lower than the yields of the training set.

The mean value of testing time for each functional testing item $\bar{t}_{Ti}$ is a random number uniformly distributed within [0.2, 10], and the variance is taken as $\bar{t}_{Ti}/100$. The testing time for each testing item is randomly generated from the mean and variance. According to the progress of the motherboard manufacturing, the average manufacturing time $\bar{t}_{\text{manuf}}$ is around 1.16 seconds. Based on actual production experience, the rework cost difference Δc_R is set to 10 seconds.

Combined with expert experience, the parameters of the enhanced congestion control algorithm are set as follows: $\beta^{\text{init}} = 0.6$, $\beta^{\max} = 1$, $\alpha_1^{\text{uni}} = 200$, $\alpha_2^{\text{uni}} = 60$, and $a = -2.5 \times 10^{-4}$.

B. Simulation Study

To further validate the effectiveness of the method proposed in this article, the approach outlined in Section II-A was employed as a comparative method for ablation experiments.

The hardware configuration for this simulation study includes an Intel Core 3.9GHz CPU and 16GB of RAM, while the software configuration consists of Windows 10 and Matlab R2023a. The modeling time for the comparison method is about 2.04 seconds, while the modeling time for the proposed method is the sum of the modeling time for the comparison method and the modeling time for the enhanced congestion control algorithm, which is about 2.25 seconds.

The evolving trend of the testing proportions for each testing item obtained by the proposed method is illustrated in Fig. 3.

Through a simulation study, it was determined that an additional 5 rework workers ($\Delta n_R^{ml} = 5$) are required to maintain the rework progress. The values of the evaluation criteria of the strategies obtained from each method on the training set are shown in Table I, where Δ^{dyn} represents the relative change in the criteria values between the proposed method and the comparison method (in percent).

TABLE I: Evaluation criteria of functional testing strategies on the training set

Criterion	Unit	$\mathcal{S}^*$	$\mathcal{S}^{\text{dyn}}$	$\Delta^{\text{dyn}}(\%)$
$c(\mathcal{S})$	CNY/board	0.1041	0.0986	-5.2976
$\bar{t}_T(\mathcal{S})$	seconds/board	18.8677	13.8411	-26.6414
$q(\mathcal{S})$	set	17	13	-23.5294
$n_R^{ml}(\mathcal{S})$	person	57	62	8.7719

The effect of each method's strategy on the test set is verified using the same values of $q(\mathcal{S})$ and $n_R^{ml}(\mathcal{S})$ as in the training set. Fig. 4 illustrates the dynamic change in the testing proportions on the test set when the proposed methods are applied. The values of the evaluation criteria of each method on the test set are presented in Table II.

C. Result Analysis

From Fig. 3, it can be seen that the testing proportions of different testing items vary with the number of motherboards tested. In other words, the algorithm enhanced in this article designs the testing proportions based on the characteristics of the testing items. From Fig. 4, it can be seen that most of the changes in the testing proportions occur between the 2000th and the 2200th motherboards (i.e., the 200 samples in the middle of the test set). It is also evident that most cases of identifying defective motherboards, especially those identifying new defects when the testing proportions have not yet returned to their initial values, occur during this period. This is consistent with the distribution of the defects. The testing proportion of part of testing items remains unchanged indicating that the initial value of the testing proportion for these testing items can ensure that the missed defects are kept at a low level and do not have a significant impact on the total cost.

TABLE II: Evaluation criteria of functional testing strategies on the test set

Criterion	Unit	$\mathcal{S}^*$	$\mathcal{S}^{\text{dyn}}$	$\Delta^{\text{dyn}}(\%)$
$c(\mathcal{S})$	CNY/board	0.1041	0.0984	-5.5212
$\bar{t}_T(\mathcal{S})$	seconds/board	18.8677	13.3638	-29.1710

From Table I, it can be seen that the strategy derived from the enhanced algorithm in this article can effectively reduce the average effective cost compared to the strategy obtained by pure optimization modeling on the training set. Simultaneously, it can significantly reduce testing time and the number of machines required for testing. This validates that, in comparison to a fixed testing proportion, a dynamically changing testing proportion can substantially reduce testing costs. From Table II, it can be seen that the criteria of the strategy derived from pure optimization modeling remain consistent between the test set and the training set because the strategy lacks the ability to adapt dynamically to changes in the sample distribution. In contrast, the enhanced algorithm can dynamically adjust the testing strategy by considering the sample distribution, leading to further reductions in the average effective cost and testing time.

IV. Conclusion

In this article, an enhanced congestion control algorithm for addressing the issue of non-uniform distribution of real-world functional testing samples was presented. A method for reducing the cost of functional testing was proposed based on the enhanced algorithm. The proposed method can design dynamically changing testing strategies based on optimization modeling. On simulation data close to actual data, the proposed method is capable of significantly reducing the testing cost compared to the pure optimization modeling method.

References

[1] Z. Li, K. Y.-T. Lai, P.-H. Yu, K. Chakrabarty, T.-Y. Ho, and C.-Y. Lee, "Structural and functional test methods for micro-electrode-dot-array digital microfluidic biochips," *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.*, vol. 37, no. 5, pp. 968–981, May 2018.

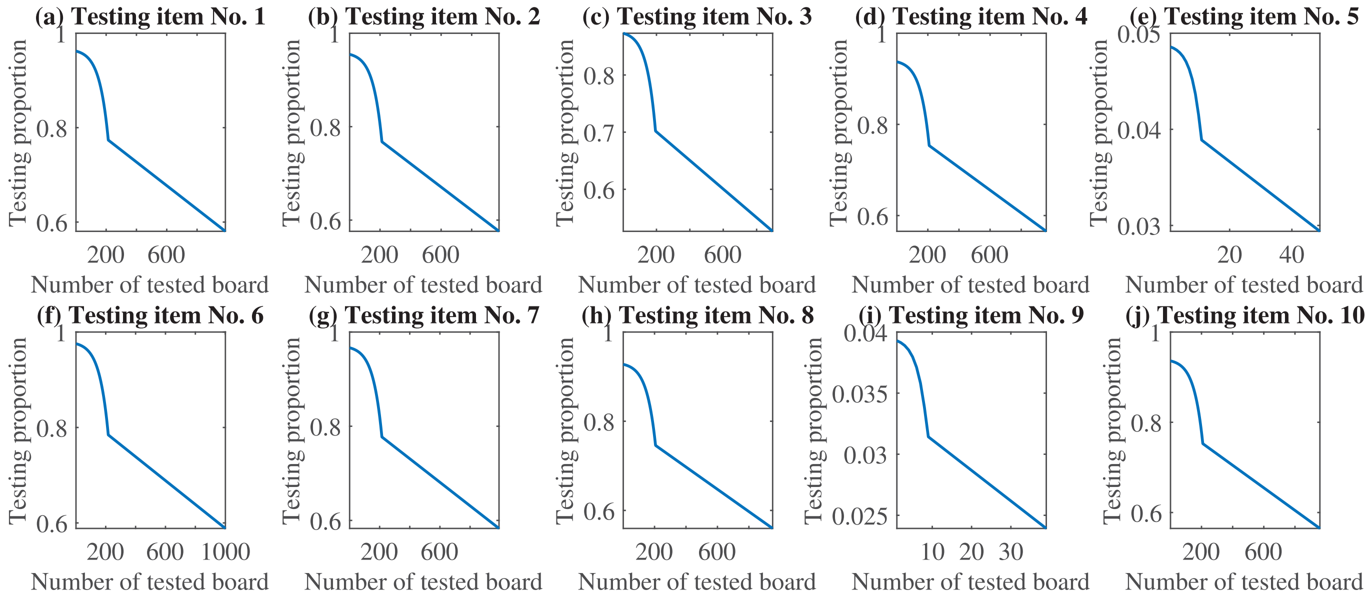

Fig. 3: Variation trend of testing proportions of functional testing items

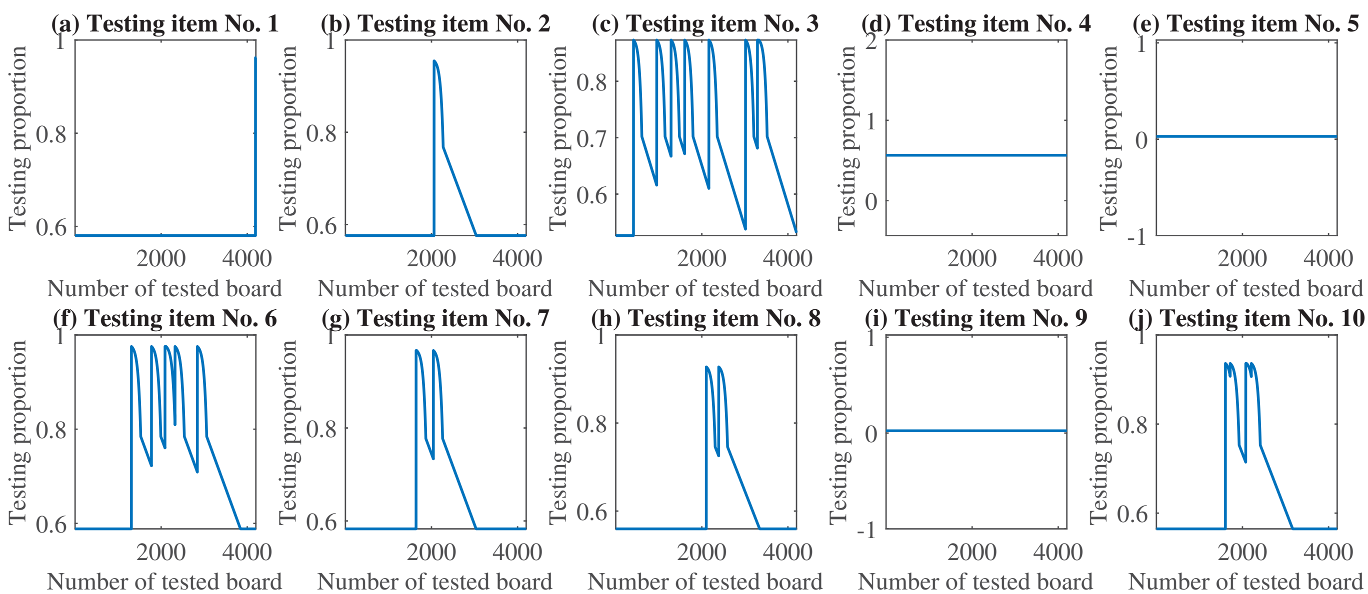

Fig. 4: The dynamic variation of the proportions of functional testing items on the test set

[2] B. Arslan and A. Orailoglu, “Aggressive test cost reductions through continuous test effectiveness assessment,” *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.*, vol. 35, no. 12, pp. 2093–2103, 2016.

[3] I. Pomeranz, “Functional constraints in the selection of two-cycle gate-exhaustive faults for test generation,” *IEEE Trans. Very Large Scale Integr. VLSI Syst.*, vol. 29, no. 7, pp. 1500–1504, Jul. 2021.

[4] R. Pan, Z. Zhang, X. Li, K. Chakrabarty, and X. Gu, “Black-box test-cost reduction based on Bayesian network models,” *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.*, vol. 40, no. 2, pp. 386–399, 2020.

[5] H.-G. Stratigopoulos and C. Streitwieser, “Adaptive test with test escape estimation for mixed-signal ICs,” *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.*, vol. 37, no. 10, pp. 2125–2138, oct 2018.

[6] Y. Li, E. Yilmaz, P. Sarson, and S. Ozev, “Adaptive test for rf/analog circuit using higher order correlations among measurements,” *ACM Transact. Des. Automat. Electron. Syst.*, vol. 24, no. 4, pp. 1–16, Jun. 2019.

[7] M. Pradhan, B. B. Bhattacharya, K. Chakrabarty, and B. B. Bhattacharya, “Predicting x-sensitivity of circuit-inputs on test-coverage: A machine-learning approach,” *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.*, vol. 38, no. 12, pp. 2343–2356, dec 2019.

[8] Y. Kang, P. Bai, K. Wang, and Y. Zhao, “Modelling and optimizing motherboard functional testing in laptop manufacturing,” *J. Syst. Sci. Complex.*, 2024.

[9] V. Jacobson, “Congestion avoidance and control,” *ACM SIGCOMM Comp. Commun. Rev.*, vol. 18, no. 4, pp. 314–329, Aug. 1988.

[10] H. Jiang, Q. Li, Y. Jiang, G. Shen, R. Sinnott, C. Tian, and M. Xu, “When machine learning meets congestion control: A survey and comparison,” *Comput. Netw.*, vol. 192, p. 108033, Jun. 2021.