Black-Box Test-Cost Reduction Based on Bayesian Network Models

Renjian Pan, *Student Member, IEEE*, Zhaobo Zhang, Xin Li, *Fellow, IEEE*, Krishnendu Chakrabarty, *Fellow, IEEE*, and Xinli Gu

***Abstract*—The growing complexity of circuit boards makes manufacturing test increasingly expensive. In order to reduce test cost, a number of test selection methods have been proposed in the literature. However, only few of these methods can be applied to black-box test-cost reduction. In this article, we propose a novel black-box test selection method based on Bayesian networks (BNs), which extract the strong relationship among tests. First, the problem of reducing the black-box test cost is formulated as a constrained optimization problem. Next, multiple structure learning and transfer learning algorithms are implemented to construct BN models. Based on these BN models, we propose an iterative test selection method with a new metric, Bayesian index, for test-cost reduction. In addition, averaging strategies are applied to enhance the reduction performance. Finally, a robust model selection framework is proposed to select the optimal BN model for test-cost reduction. Two case studies with production test data demonstrate that when no prior information is provided, our proposed approach effectively reduces the test cost by up to 14.7%, compared to the state-of-the-art greedy algorithm. Moreover, our proposed approach further reduces the test cost by up to 7.1% when prior information is provided from similar products.**

***Index Terms*—Bayesian network (BN), black-box testing, test-cost reduction, transfer learning.**

I. Introduction

With the aggressive technology scaling in very large-scale integration (VLSI) manufacturing, the manufacturing cost per transistor has decreased steadily over the last few decades [2], whereas the test cost per transistor has not been reduced at the same rate. According to the international technology roadmap for semiconductors (ITRSs), as circuit boards become increasingly complex, test cost becomes a large, and sometimes even dominant, portion of the overall manufacturing cost [3]. Therefore, it is important to reduce test cost and, consequently, improve profit margin for semiconductor manufacturing.

Manuscript received August 9, 2019; revised January 21, 2020; accepted April 16, 2020. Date of publication May 12, 2020; date of current version January 20, 2021. This work was supported in part by the Futurewei Technologies, Inc. A preliminary version of this article was published in the Proc. IEEE VLSI Test Symposium in 2019 [1]. This article was recommended by Associate Editor A. Orailoglu. *(Corresponding author: Renjian Pan.)*

Renjian Pan, Xin Li, and Krishnendu Chakrabarty are with the Department of Electrical and Computer Engineering, Duke University, Durham, NC 27708 USA (e-mail: renjian.pan@duke.edu; xinli.ece@duke.edu; krish@duke.edu).

Zhaobo Zhang and Xinli Gu are with the Department of Network Technology, Futurewei Technologies, Inc., Santa Clara, CA 95050 USA (e-mail: zzhang1@futurewei.com; xgu@futurewei.com).

Digital Object Identifier 10.1109/TCAD.2020.2994257

In board testing, white-box testing and black-box testing are two major parts, and are both required to ensure a low defective parts per million (DPPM) rate for downstream customers [4]. White-box testing (i.e., structural testing) is based on fault models and depends on the specific board structure. It has been widely used over the past few decades due to its low cost [5]. On the other hand, black-box testing (i.e., functional testing) is highly customized and it verifies whether the functional specifications are met by the given board. Although black-box testing is often expensive, it can detect a number of manufacturing defects that can hardly be covered by white-box testing [5]. Moreover, with the increasing complexity of circuit boards, it is extremely difficult or even impossible to ensure a low DPPM with white-box testing only. Hence, black-box testing is also necessary and has become one of the major contributors to the overall test cost [6].

Because of the high cost of black-box testing, efficient methods need to be developed to reduce the test cost. There are two major categories of methods for test-cost reduction in general [7]: 1) test ordering and 2) test selection. For test ordering, different tests are ordered based on their effectiveness for detecting defects [8]. As such, most defects can be detected in the early stage of testing, and only a small set of tests is necessary for most defective boards. Nevertheless, test ordering only reduces the test cost for defective boards. For manufacturing processes with high yield, the overall test-cost reduction is limited.

For test selection, a subset of most effective tests is selected for detecting defects [10]. By applying this subset of tests, most (or even all) defective boards can be detected. Owing to its capability to reduce test cost for both defective and non-defective boards, test selection is usually preferred over test ordering [7]. A number of test selection methods have been successfully proposed for the parametric test of analog and mixed-signal circuits [11], [12], the wafer-probing and final test of digital circuits [13]–[16] and the structural test of digital circuits [10]. However, only few of these methods are applicable to black-box test-cost reduction, and a greedy algorithm has thus far been shown to be the most practical method to ensure high defect coverage [10]. Nonetheless, in practice, this greedy algorithm may suffer from overfitting because the number of defective boards is often limited for production with high yield.

To address the aforementioned issues, we propose a new and "customized" method for black-box test-cost reduction based on the theory of Bayesian network (BN). Our proposed

TABLE I
SUMMARY OF THE PROPOSED ALGORITHMS

Algorithm	Description
Algorithm 1	Constraint-based algorithm for BN structural learning
Algorithm 2	Score-based algorithm for BN structural learning
Algorithm 3	Hybrid algorithm for BN structural learning
Algorithm 4	Model averaging for BNs
Algorithm 5	Constraint-based algorithm for BN transfer learning
Algorithm 6	Score-based algorithm for BN transfer learning
Algorithm 7	Hybrid algorithm for BN transfer learning
Algorithm 8	Black-box test selection
Algorithm 9	Test-set averaging

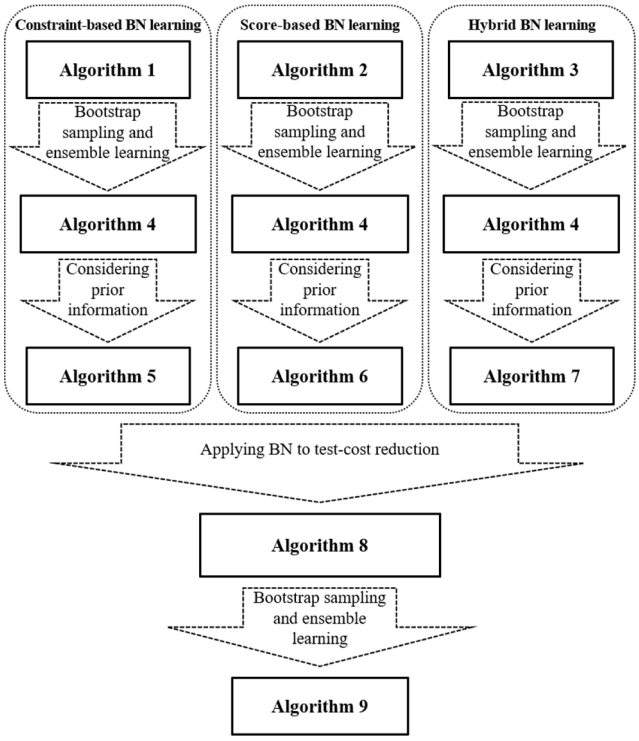


Fig. 1. Relationships between proposed algorithms.

method carries two important contributions: 1) our problem formulation addressing the overfitting issue is new and 2) it is the first attempt in the testing community to develop a BN-based method with adaptive model selection. The proposed method is composed of a number of algorithms, as summarized in Table I. Fig. 1 illustrates the relationship between these algorithms and how our proposed method is organized.

Specifically, we propose a relationship analysis tool based on BN models for black-box tests. To identify the relationship among tests, the BN models are constructed for black-box tests with historical test data based on multiple structure learning methods (e.g., constraint-based methods such as Algorithm 1 [23]–[26], score-based methods such as Algorithm 2 [27], [28], and hybrid methods such as Algorithm 3 [29]). Model averaging techniques in Algorithm 4 are adopted to facilitate robust learning for the proposed BN models. In addition, if prior information (i.e., historical test data) from similar products is available, transfer learning methods, such as Algorithms 5–7 are further applied to improve the reliability of BN models.

Based on our proposed BN models, a black-box test-cost reduction method is proposed in Algorithm 8. For each BN model, a new metric, referred to as the Bayesian index, is developed in this article to evaluate the efficiency of defect coverage of each test, and a subset of K black-box tests with the highest Bayesian index values is selected to meet the given constraints on test cost. Next, to select the best BN model with the most effective subset of tests, a robust model selection algorithm in Algorithm 9 is proposed among all BN models learned by multiple structure learning methods.

Our proposed test-cost reduction approach effectively alleviates the overfitting problem posed by the conventional greedy algorithm, because the BN models can be accurately learned from a limited data set. As demonstrated by our experimental results based on production test data in Section VIII, the proposed approach reduces test cost by up to 14.7%, compared to the conventional greedy algorithm with no prior information provided from similar products. When the transfer learning methods are applied with test data from similar products, test cost can be further reduced by up to 7.1%. For high-volume production, even a 10% test-cost reduction is significant [16].

The remainder of this article is organized as follows. In Section II, we review the background on board-level testing and test-cost reduction. In Section III, we present the problem formulation for black-box test-cost reduction. In Section IV, we introduce the concept of BNs, and explain several basic methods and the model averaging strategy to learn a BN model from the given data set. In Section V, the transfer learning methods for BN models are introduced to utilize the prior information from similar products. In Section VI, we propose the test-cost reduction method based on a novel test-set averaging strategy, and summarize our detailed implementation on model selection in Section VII. The experimental results are presented in Section VIII. Finally, we conclude in Section IX.

II. BACKGROUND

A. Board-Level Testing

In board-leveling testing, a thorough test strategy must be conducted to ensure the reliability of products. After board assembling, process testing is first applied to check if there exists any process flaw or missing component. Next, white-box testing is applied to detect the manufacturing defects caused by soldering, board assembling, etc. After that, black-box testing is applied to detect other manufacturing defects that cannot be covered by white-box testing. Finally, burn-in testing is sometimes applied for products with high-reliability requirements. Among all these steps, white-box testing and black-box testing are the two most important ones and consume most test cost [17].

White-box testing, also known as structural testing, has been widely used over the past few decades [5]. It is usually based on fault models and is often generated by an automatic test pattern generation (ATPG) system. Therefore, its test process is usually rapid and cost-effective. In addition, it often facilitates precise and rapid fault diagnosis when sufficient test points exist [9]. However, with the increasing complexity of circuit

boards, it is extremely difficult or even impossible to ensure a low DPPM with only white-box testing. Therefore, black-box testing has become necessary for modern board-level testing [5].

Black-box testing, also known as functional testing, verifies whether the board under test meets the functional specifications. Compared to white-box testing, black-box testing checks the input-to-output performance of the board instead of individual elements (e.g., CMOS transistors, interconnects, etc.) [17]. It is highly customized and often requires careful design by senior test engineers owing to the controllability and observability issues. In addition, black-box tests are often not reusable when the packaging is changed. Thus, black-box testing is far more expensive than white-box testing [17].

Despite its expensive cost, black-box testing is essential when circuit boards become increasingly complex. If only applying white-box testing, we may encounter from either the under-testing problem with many undetected defects and a high DPPM, or the over-testing problem with high yield loss [18]. Therefore, to ensure a low DPPM and simultaneously a high yield, black-box testing should be applied to detect the manufacturing defects that can hardly be covered by white-box testing. Due to its necessity and expensive cost, black-box testing has become one of the major contributors to the overall test cost [6], and there is a pressing need to develop an efficient method to reduce the black-box test cost.

B. Test-Cost Reduction

In general, the test-cost reduction methods can be classified into two broad categories [7]: 1) test ordering and 2) test selection. The test ordering is a traditional test-cost reduction method. In specific, multiple tests are ordered based on their effectiveness for detecting defects [8]. As such, most defective boards can be detected in the early test stage, and remaining tests are unnecessary for these boards. Therefore, the overall test time and test cost can be reduced.

Over the past few decades, several test ordering methods have been proposed for both analog and digital integrated circuits (ICs) [8], [19]. However, these methods only reduce the test cost for defective boards/circuits. In modern IC manufacturing with high yield, if the test cost were only reduced for defective parts, the overall test-cost reduction would be limited. Moreover, test ordering can only be applied for production test, where diagnosis is not required [9]. Therefore, test ordering is barely used alone for the test-cost reduction in recent years.

To effectively reduce the test cost for products with high manufacturing yield, various test selection methods are widely used in recent years. The key idea is to select a subset of tests that can detect defects most efficiently [10]. By applying this subset, most (or even all) defective boards can be detected. As this subset of tests is applied to all boards including non-defective boards, the test cost reduction method is often more effective than the test ordering approach. Owing to this fact, test selection is usually preferred over test ordering [7].

A number of test selection methods have been proposed in recent years. Chen and Orailoglu [7] proposed an adaptive test strategy for parametric test selection of mixed-signal circuits. A probabilistic model is built for each test, and in the light of correlation analysis, the optimal test subset is selected. Ahmadi *et al.* [12] proposed an adaptive test selection flow for parametric tests by considering the wafer-level process variations in analog/RF ICs. Wang *et al.* [13] proposed a cost model for dynamic selection of the test configurations and procedures for 3-D ICs. Agrawal and Chakrabarty [14] proposed a generic test-cost model for 3-D-stacked ICs, and based on this model, an optimization method is proposed to effectively select tests. Grady *et al.* [15] defined a score function to evaluate each wafer-level test, and complete the wafer-level test selection with this function. Liu *et al.* [16] developed a fine-grained adaptive test strategy for chip-level final test. By evaluating a machine-learning-based quality index (QI), chips are classified into multiple quality levels, and then a probabilistic model is built for adaptive test selection in each level. Xue and Blanton [10] proposed a conventional greedy method for structural test selection.

Although many test selection methods have been proposed, few of these methods can be applied to black-box test-cost reduction, since most of the aforementioned methods are developed for parametric tests or for ICs with specific properties (e.g., specific packaging, 3-D structure, etc.) For black-box testing, we need to treat the board/circuit as a black box with only information from the pass/fail result of each test. To the best of our knowledge, the greedy algorithm has thus far been the most practical method to ensure high defect coverage [10]. This method formulates the test-cost reduction problem as a budgeted maximum coverage problem (BMCP) [20], and solves it by a greedy method. It first selects the test that is most effective to detect the undetected defects and removes the defective boards covered by this test. The aforementioned process is repeated until the test-cost limit is reached. The greedy method is fast and easy to implement. However, it may suffer from overfitting, provided that the number of defective boards is often limited for production with high yield. Therefore, an alternative method is needed for robust black-box test-cost reduction.

III. Problem Formulation

As shown in Fig. 2, to reduce the black-box test cost by test selection, a complete set of black-box tests must be applied to a set of sample boards. As such, a test selection model can be trained with the sample test data and the prior information from similar products (if available). Using this model, a subset of important tests is selected. By only applying this subset of tests to other boards, we can reduce the black-box test cost.

Traditionally, this test selection problem can be formulated as a BMCP [20]. Let $T = \{t_1, t_2, \ldots, t_M\}$ denote a complete set of black-box tests, and $B_{\text{Train}} = \{b_1, b_2, \ldots, b_{N_{\text{Train}}}\}$ denote the set of defective sample boards. Each black-box test t_m is capable of detecting a subset of defective boards $B_{\text{Train},m} \subseteq B_{\text{Train}}$ with the test cost (i.e., average test time per board) c_m.

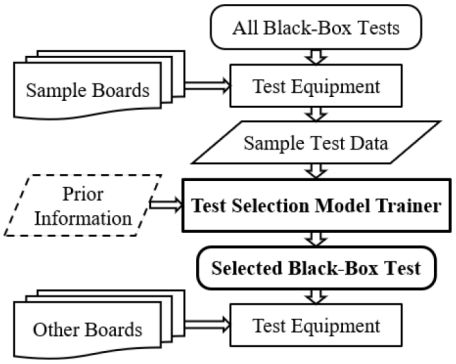


Fig. 2. General flow of black-box test-cost reduction by test selection.

The problem of test selection can be formulated as

$$\begin{aligned} \arg\max_{T_{\text{Opt}}} \quad & \left| \bigcup_{t_m \in T_{\text{opt}}} B_{\text{Train},m} \right| \\ \text{s.t.} \quad & c\big(T_{\text{Opt}}\big) \le c_{\text{Th}} \\ & T_{\text{Opt}} \subseteq T \end{aligned} \tag{1}$$

where $|\bullet|$ denotes the number of elements in a set

$$c\big(T_{\text{Opt}}\big) = \sum_{t_m \in T_{\text{Opt}}} c_m \tag{2}$$

denotes the total test cost for testing T_{Opt}, and c_{Th} is the maximum threshold for the total black-box test cost. To solve BMCP, a conventional greedy algorithm is often utilized [10], [20].

The BMCP formulation aims at maximizing the coverage of the defective boards used for training the test selection model. However, this formulation does not guarantee that the maximum coverage will also be achieved when applying the selected tests T_{Opt} to other defective boards. The number of defective boards is often limited, especially for production with high yield, and the number of boards for training is even less. Therefore, formulating the black-box test selection as BMCP may cause the selected tests to overfit the training boards and exhibit less coverage on other defective boards.

To address this overfitting problem in BMCP, we revise the traditional formulation. Let $B_{\text{Test}} = \{b_1, b_2, \ldots, b_{N_{\text{Test}}}\}$ denote the set of defective boards to which only the selected test set T_{Opt} will be applied. $B_{\text{Test},m} \subseteq B_{\text{Test}}$ denotes the subset of defective boards in B_{Test} that can be detected by the black-box test t_m. The actual goal of test selection is to select a subset T_{Opt} of black-box tests to maximize the defect coverage on B_{Test}. To address this goal, the original BMCP formulation can be revised as

$$\begin{aligned} \arg\max_{T_{\text{Opt}}} \quad & \left| \bigcup_{t_m \in T_{\text{Opt}}} B_{\text{Test},m} \right| \\ \text{s.t.} \quad & c\big(T_{\text{Opt}}\big) \le c_{\text{Th}} \\ & T_{\text{Opt}} \subseteq T. \end{aligned} \tag{3}$$

Nevertheless, solving the optimization problem in (3) is nontrivial, because we cannot directly obtain any information/data from B_{Test}. Fortunately, B_{Train} and B_{Test} share common information even when the number of defective boards is limited. In addition, similar products (if available) can further provide valuable prior information to guide test section. If we properly use the information/data from B_{Train} and other similar products, we should obtain an approximate solution to the problem in (3). In the sections that follow, we will propose a novel methodology to extract the key information from B_{Train} and other similar products to solve the black-box test selection problem.

It is important to note that our problem formulation does not consider the following scenarios.

1) *Order Constraints:* We do not consider any testing tasks with order constraints, as it increases the number of constraints for the resulting optimization problem and, consequently, increases the complexity. We plan to further attack such a multiconstraint optimization problem in our future research.
2) *Granularity of Black-Box Tests:* We consider each black-box test as an atomic component that cannot be further divided into multiple subtests. In other words, each black-box test is either tested with its all contents or not tested. We will further explore the flexibility of forming subtests (e.g., applying only 20% of each black-box test) in our future research.
3) *Streaming/Online Data:* Our problem formulation is batch-based, in which we use all historical data to learn the optimal test set for future testing. If we need to handle streaming/online data, we should reformulate the problem with the complete data set by merging the new data with the historical data. Such an approach is computationally expensive. In our future work, we will develop efficient incremental/online learning methods [21] to handle streaming/online data.

Despite these limitations, we are capable of handling multiple different tests (e.g., a short black-box test and a long one) that share similar contents. Our proposed method treats them as different black-box test items. Based on our formulation, we will select the optimal test items among them. Considering memory test as an example, there may be many different-yet-similar March-based algorithms (e.g., March X, March A, March Y, etc. [4]) associated with different costs. We consider them as different black-box test items and use the proposed algorithm to select the optimal subset of these tests.

IV. Bayesian Network

To solve the problem described by (3), we should first extract useful information from the set B_{Train} of defective boards. As discussed in [13], the relationships among tests contain important information for test-cost reduction. Although the relationships among tests for B_{Train} are not identical to those for B_{Test}, most strong relationships are shared between B_{Train} and B_{Test}. Hence, we need to train a model that can identify the strong relationships among black-box tests.

In recent years, the BN model has been shown to be a practical and effective way to identify strong relationships among random variables (i.e., black-box tests in our application) [22].

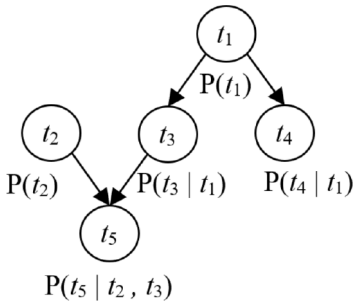


Fig. 3. Example of a BN for five black-box tests.

If BN models can be trained with test data, the strong relationships among black-box tests can be extracted accurately. Therefore, in this section, we introduce the definitions of BNs on black-box tests and illustrate how to learn the BN structure.

A. Definitions

From the perspective of graph theory, the definition of a BN can be given based on a directed acyclic graph (DAG). The DAG is a finite graph $G = (V, E)$, where all edges are directed, and there is no directed cycle. The BN model is defined as a network structure, a DAG $G = (V, E)$, in which each vertex $v_m \in V$ denotes a black-box test t_m, and each edge $e_{m,n} \in E$ denotes the relationship between two tests t_m and t_n. When there is an edge between two tests (e.g., t_m and t_n), it means that they are highly correlated with each other. Namely, when t_m fails, t_n may also fail with high probability.

From the perspective of probability theory, the BN model can also be defined as a probability distribution $P(T)$ over a set of black-box tests $T = \{t_1, t_2, \ldots, t_M\}$. Based on the edges $e_{m,n} \in E$ in the DAG, $P(T)$ can be decomposed into a number of conditional probability distributions (CPDs) on every test t_m

$$P(T) = \prod_{m=1}^{M} P(t_m | T_{\mathrm{Pr}}(t_m); \Theta(t_m)) \tag{4}$$

where $T_{\mathrm{Pr}}(t_m)$ denotes the set of parents of the test t_m in the DAG, and $\Theta(t_m)$ denotes the set of parameters in the CPD of the test t_m. Next, we use an example to intuitively describe how to calculate the CPD and $\Theta(t_m)$.

Fig. 3 shows a simple example of BN for five black-box tests: t_1, t_2, t_3, t_4, and t_5. In this BN, t_1 is the parent of both t_3 and t_4. Both t_2 and t_3 are the parents of t_5. t_1 and t_2 are independent because either one is not an ancestor of the other one, and they do not share any common ancestor

$$P(t_1, t_2) = P(t_1) \cdot P(t_2). \tag{5}$$

Similarly, t_2 is independent of t_3 and t_4

$$P(t_3, t_4 | t_1, t_2) = P(t_3, t_4 | t_1). \tag{6}$$

t_3 and t_4 are conditionally independent given t_1 because t_1 is the only common ancestor of t_3 and t_4

$$P(t_3, t_4 | t_1) = P(t_3 | t_1) \cdot P(t_4 | t_1). \tag{7}$$

t_5 and t_1 are conditionally independent given t_3 because t_1 is no longer an ancestor of t_5 when removing t_3. In addition, t_5 and t_4 are conditionally independent given t_3 because t_5 and t_4 do not share any common ancestor when removing t_3. Thus, we have

$$P(t_5 | t_1, t_2, t_3, t_4) = P(t_5 | t_2, t_3). \tag{8}$$

Therefore, P(T) can be decomposed as

$$\begin{aligned} P(T) &= P(t_1, t_2, t_3, t_4, t_5) \\ &= P(t_1, t_2) \cdot P(t_3, t_4 | t_1, t_2) \cdot P(t_5 | t_1, t_2, t_3, t_4) \\ &= P(t_1, t_2) \cdot P(t_3, t_4 | t_1) \cdot P(t_5 | t_2, t_3) \\ &= P(t_1) \cdot P(t_2) \cdot P(t_3 | t_1) \cdot P(t_4 | t_1) \cdot P(t_5 | t_2, t_3). \end{aligned} \tag{9}$$

For each t_m, the CPD can be described as a conditional probability table (CPT). For example, Table II is the CPT of t_5, describing the CPD $P(t_5 | t_2, t_3)$ by four parameters are $\Theta(t_5) = \{p_a, p_b, p_c, p_d\}$. To calculate these parameters, we only need to count the frequencies of the corresponding cases observed from the data set [22]. Consider a simple example

$$p_b = P(t_5 = 0 | t_2 = 0, t_3 = 1) = \frac{N_{t_5,t_2,t_3}(0, 0, 1)}{N_{t_2,t_3}(0, 1)} \tag{10}$$

where 0/1 denotes the pass/fail result of the black-box test, and $N_{t_2,t_3}(0, 1)$ denotes the number of boards in B_{Train} that pass the test t_2 but fail the test t_3.

TABLE II
CPT CORRESPONDING TO t_5

Conditions	$t_5 = 0$	$t_5 = 1$
$t_2 = 0; t_3 = 0$	p_a	$1 - p_a$
$t_2 = 0; t_3 = 1$	p_b	$1 - p_b$
$t_2 = 1; t_3 = 0$	p_c	$1 - p_c$
$t_2 = 1; t_3 = 1$	p_d	$1 - p_d$

B. Structure Learning

To extract the relationships among black-box tests, we need to learn the BN structure. There are three popular categories of algorithms for structure learning: 1) constraint-based algorithms [23]–[26]; 2) score-based algorithms [27], [28], and 3) hybrid algorithms [29]. As the efficacy of these algorithms is application-dependent [22], [30], we will adopt all these algorithms in this article. In Section VII, we will further introduce a model selection algorithm to automatically select the best algorithm for a given application case.

1) Constraint-Based Structure Learning: Conditional independence relationships (i.e., constraints) are first learned among black-box tests from the training data. Next, the most proper BN structure is obtained to meet these constraints. The aforementioned constraints are usually learned by hypothesis testing based on mutual information to check the conditional independency. Specifically, to check the conditional independency between two black-box tests t_m and t_n given a test set T_S, the null hypothesis is proposed as $P(t_m, t_n | T_S) = P(t_m | T_S).P(t_n | T_S)$. The test set T_S is referred to as the separation set in BNs because if the hypothesis holds, t_m and t_n are separated by T_S in the BN structure [26].

To decide whether to accept or reject this hypothesis, we need to calculate the conditional mutual information

$$I(t_m, t_n|T_S) = \sum_{\substack{i\in R_m \\ j\in R_n \\ k\in R_S}} \frac{N_{t_m,t_n,T_s}(i,j,k)}{N_{\text{Train}}} \log\left(\frac{N_{t_m,t_n,T_s}(i,j,k)\cdot N_{T_s}(k)}{N_{t_m,T_s}(i,k)\cdot N_{t_n,T_s}(j,k)}\right) \tag{11}$$

where R_m, R_n, and R_S are the sets containing all possible test results for t_m, t_n, and T_S, respectively, $N_{\text{tm},tn,\text{TS}}(i, j, k)$ denotes the number of defective boards whose test results satisfy $t_m = i$, $t_n = j$ and $T_S = k$, and $N_{\text{tm,TS}}(i, k)$ denotes the number of defective boards whose test results satisfy $t_m = i$ and $T_S = k$, etc. The decision $D(t_m, t_n|T_S)$ can be made as

$$D(t_m, t_n|T_S) = \begin{cases} \text{Accept}, & I(t_m, t_n|T_S) \le I_{\text{Th}} \\ \text{Reject}, & I(t_m, t_n|T_S) > I_{\text{Th}} \end{cases} \tag{12}$$

where I_{Th} is a threshold determined by the p-value p_α chosen for the hypothesis testing [31], [43]

$$p_\alpha = 1 - \chi^2((|R_m - 1|\cdot|R_m - 1|\cdot|R_S|), I_{\text{Th}}). \tag{13}$$

In (13), $\chi^2(k, x)$ is the cumulative distribution function of x for a chi-square distribution with k degrees of freedom. Once p_α is determined, I_{Th} can be easily found by the chi-square distribution [32]. We will further discuss how to choose an appropriate value for p_α in Section VII.

The decision $D(t_m, t_n|T_S)$ in (12) decides whether the edge $<t_m, t_n>$ should exist or not. If $D(t_m,\ t_n|T_S) = \text{Accept}$, t_m and t_n are conditionally independent given a separation set T_S and, consequently, the edge $<t_m, t_n>$ should not exist. For each pair of (t_m, t_n), we use the symbol $T_{\text{Sep}}(t_m, t_n)$ to represent the smallest set that satisfies $D(t_m, t_n|T_{\text{Sep}}(t_m, t_n)) = \text{Accept}$.

Based on the conditional independence tests, many constraint-based structure learning algorithms have been proposed [23]–[26]. Among them, the PC-stable algorithm has been shown to be a state-of-the-art approach with both great performance and stability [30], [33]. It is named after Peter and Clark who originally proposed the constraint-based learning algorithm [25]. The word "stable" means that the result of the PC algorithm in [26] is independent of the order of black-box tests recorded in the training data. In this article, we will apply it to our constraint-based structure learning.

To illustrate how the PC-stable algorithm works, an example with 4 black-box tests, t_1, t_2, t_3, and t_4, is shown in Fig. 4.

1) *Initialization:* We initialize the skeleton of BN structure as a complete undirected graph as shown in Fig. 4(a).
2) *Learning Skeleton:* We delete the edges based on conditional independence tests in (12). Among all conditional independence tests, we start from those with the empty separation set $T_S = \varnothing$, and then increase the size of separation set until $|T_S| = M - 2$. Following this order, once we observe $D(t_m, t_n|T_S) = \text{Accept}$, we can not only delete the edge $<t_m, t_n>$ but also learn $T_{\text{Sep}}(t_m, t_n) = T_S$. In this example, the decisions of all conditional independence tests are summarized in Table III. According to these decisions, the edge $<t_1, t_2>$ is deleted because $D(t_1, t_2|\varnothing) = \text{Accept}$ and $T_{\text{Sep}}(t_1, t_2) = \varnothing$ as shown in Fig. 4(b). The edges $<t_1, t_4>$ and $<t_2, t_4>$ are deleted because $D(t_1, t_4|t_3) = D(t_2, t_4|t_3) = \text{Accept}$ and $T_{\text{Sep}}(t_1, t_4) = T_{\text{Sep}}(t_2, t_4) = \{t_3\}$ as shown in Fig. 4(c). The other edges $<t_m, t_n>$ remain because t_m and t_n cannot be separated by the set $T\backslash\{t_m, t_n\}$ as shown in Table III.
3) *Determining Edge Directions:* The directions of edges are determined with the information provided by $T_{\text{Sep}}(t_m, t_n)$ [25], [26]. As shown in Fig. 4(d), the directions of $<t_1, t_3>$, and $<t_2, t_3>$ are determined as $t_1 \to t_3 \leftarrow t_2$ because t_3 does not belong to the separation set $T_{\text{Sep}}(t_1, t_2) = \varnothing$. The direction of $<t_3, t_4>$ is determined as $t_3 \to t_4$ because t_3 belongs to the separation set $T_{\text{Sep}}(t_1, t_4) = \{t_3\}$. If the edge $<t_3, t_4>$ is set to the opposite direction $t_4 \to t_3$, the v-structure $t_1 \to t_3 \leftarrow t_4$ occurs, violating the fact that $t_3 \in T_{\text{Sep}}(t_1, t_4)$.

Algorithm 1 PC-Stable Algorithm for BN Learning

```
1. Initialize G as a complete undirected graph.
2. For N_Sep = 0, 1, ..., M − 2:
3.    For each edge <t_m, t_n> in G:
4.       If ∃ T_S ⊆ T \{t_m, t_n} satisfying |T_S| = N_Sep and
         D(t_m, t_n | T_S) = Accept:
5.          Delete <t_m, t_n> from G.
6.          T_Sep(t_m, t_n) = T_S.
7.       End If.
8.    End For.
9. End For.
10. Determine the directions of all edges in G.
```

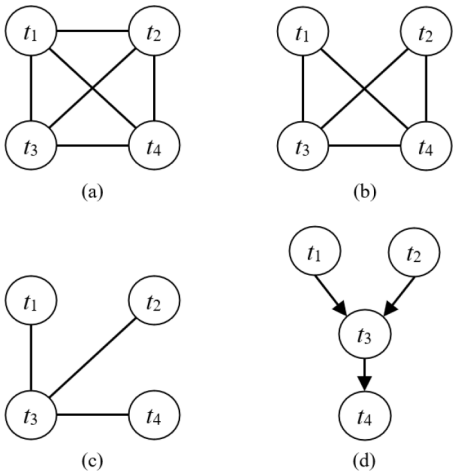


Fig. 4. Example of BN learning. (a) Initial skeleton of G that is a complete undirected graph. (b) Skeleton G after deleting edges based on the conditional independence tests given an empty separation set. (c) Skeleton G after deleting edges based on the conditional independence tests given a separation set containing one black-box test. (d) DAG learned after determining the directions of all edges.

The aforementioned idea of PC-stable algorithm is summarized in Algorithm 1. In most applications, the computational complexity of this algorithm is polynomial in M and N_{Train} [22]. More implementation details can be found in [26].

2) Score-Based Structure Learning: A score function is designed to evaluate the fitting quality, and an optimization is solved to find the optimal BN structure with the highest

TABLE III
CONDITIONAL INDEPENDENCE TESTS FOR THE EXAMPLE IN FIG. 4

$<t_m, t_n>$	T_S	$D(t_m, t_n \mid T_S)$
$<t_1, t_2>$	Ø	Accept
$<t_1, t_4>$	Ø	Reject
$<t_2, t_4>$	Ø	Reject
$<t_1, t_4>$	$\{t_3\}$	Accept
$<t_2, t_4>$	$\{t_3\}$	Accept
$<t_1, t_3>$	$\{t_2, t_4\}$	Reject
$<t_2, t_3>$	$\{t_3, t_4\}$	Reject
$<t_3, t_4>$	$\{t_1, t_2\}$	Reject

score value. One of the widely used score functions is the Bayesian information criterion (BIC) function [22], [34]

$$S_{\mathrm{BIC}}(G) = \sum_{m=1}^{M} \log P(t_m | T_{\mathrm{Pr}}(t_m); \Theta(t_m)) - \frac{D_G}{2} \log N_{\mathrm{Train}} \tag{14}$$

where

$$D_G = \left| \bigcup_{t_m \in T} \Theta(t_m) \right| \tag{15}$$

denotes the total number of parameters in the BN model. A large D_G implies a complex BN model, and vice versa.

There are two terms in the BIC function in (14). The first term is the log-likelihood function, and the second term denotes a penalty for model complexity. By maximizing the BIC function, we achieve a balance between maximizing the likelihood function and simplifying the model structure. As such, the BN model is robust to overfitting, and only the strong relationships between black-box tests are preserved.

Maximizing the BIC function approximates the maximum-a-posteriori (MAP) estimation with uniform prior distribution [34]. However, this MAP approximation is based on the assumption that the number of training boards is much greater than the number of black-box tests (i.e., $N_{\mathrm{Train}} >> M$) [28], [34]. For IC production with high yield, the number of defective boards is often small, and this assumption may not hold. To extend the BIC function to small data set, an extended BIC (EBIC) function has been proposed for BN structure learning [28]

$$S_{\mathrm{EBIC}}(G) = \sum_{m=1}^{M} \log P(t_m | T_{\mathrm{Pr}}(t_m); \Theta(t_m)) - \frac{D_G}{2} \log N_{\mathrm{Train}} - D_G \log M. \tag{16}$$

Compared to the traditional BIC function, the EBIC function adds a penalty term that is proportional to $\log M$. The theoretical foundation of EBIC has been shown [28]. Moreover, if $N_{\mathrm{Train}} >> M$ holds, we have $\log N_{\mathrm{Train}} >> \log M$ so that the additional term in (16) is negligible and the EBIC function is reduced to the BIC function in (14).

Once we confirm the score function $S(\cdot)$, we are capable of learning the BN structure by maximizing the score function. A widely used method is the hill-climbing (HC) algorithm [27]. In Algorithm 2, we summarize the major steps of HC. Starting from an initial structure G, its neighbor set $\mathbf{G}_{\mathrm{Nb}}(G)$ is searched. The set $\mathbf{G}_{\mathrm{Nb}}(G)$ contains all possible structures with only one edge added, deleted, or reversed on G. Next, G is refreshed by any G^* with higher score. The procedure is repeated until no structure G^* with higher score can be found. The computational complexity of Algorithm 2 is $O(K_{\mathrm{Iter}}.M^3.N_{\mathrm{Train}})$, where K_{Iter} is the number of iterations [20]. In our application, the algorithm usually converges within 100 iterations.

Algorithm 2 HC for BN Learning

1. Choose an initial structure G, and set $S_{\mathrm{Max}} = S(G)$.
2. Repeat:
3. For each G^* in $\mathbf{G}_{\mathrm{Nb}}(G)$:
4. If $S(G^*) > S(G)$:
5. Set $S_{\mathrm{Max}} = S(G^*)$ and $G = G^*$.
6. End If.
7. End For.
8. Until S_{Max} does not increase further.

Algorithm 3 Hybrid Learning for BN Structure

1. For each t_m in T:
2. Set $T_{\mathrm{PrC}}(t_m) = \varnothing$.
3. Repeat
4. Select t_{Max} with the maximum $I(t_m, t_{\mathrm{Max}} \mid T_{\mathrm{PrC}}(t_m))$.
5. If $D(t_m, t_{\mathrm{Max}} \mid T_{\mathrm{PrC}}(t_m)) =$ Reject, stop iteration.
6. Set $T_{\mathrm{PrC}}(t_m) = T_{\mathrm{PrC}}(t_m) \cup \{t_{\mathrm{Max}}\}$.
7. End Repeat.
8. End For.
9. Use the HC algorithm to find the optimal G^* with the constraint that all parents of t_m belong to $T_{\mathrm{PrC}}(t_m)$.

3) Hybrid Structure Learning: To take advantage of both constraint-based learning and score-based learning, an efficient methodology has been proposed to appropriately combine them together [29]. It is composed of two major steps: 1) the restriction step that constrains the possible structures of the BN model and 2) the maximization step that searches the optimal structure among all possible structures.

In the restriction step, the mutual information in (11) and the conditional independence test in (12) are used to calculate the parent candidate set $T_{\mathrm{PrC}}(t_m)$ that contains all potential parent nodes for the node $t_m$in the BN model. Starting from an empty set $T_{\mathrm{PrC}}(t_m)$, we iteratively select the test t_{Max} with the maximum mutual information $I(t_m, t_{\mathrm{Max}} | T_{\mathrm{PrC}}(t_m))$. The iteration repeats until $D(t_m, t_{\mathrm{Max}} | T_{\mathrm{PrC}}(t_m)) =$ Reject. Similar to the constraint-based structure learning, the threshold for $D(t_m,\ t_{\mathrm{Max}} | T_{\mathrm{PrC}}(t_m))$ in (12) is determined by appropriately choosing the p-value p_α in (13) [29], as will be discussed in SectionVII. Next, in the maximization step, the score-based HC algorithm is applied to find the optimal BN structure G^* with the constraints that the parent nodes of t_m must belong to the parent candidate set $T_{\mathrm{PrC}}(t_m)$.

In Algorithm 3, we summarize the hybrid algorithm used for structure learning. The computational complexity of Algorithm 3 is identical to that of Algorithm 2. However, Algorithm 3 often takes less computational time than Algorithm 2 in most applications, because Algorithm 3 shrinks the search space by considering the parent candidate sets so that the HC algorithm may converge more quickly [33].

Algorithm 4 Model Averaging for BN Learning

1. Choose a structure learning algorithm.
2. For $k = 1, 2, \ldots, K$:
3. Generate a new board set $B^*{}_k$ from B_{Train} by bootstrap sampling.
4. Apply the given structure learning algorithm to learn the BN structure $G_k = (V, E_k)$ based on $B^*{}_k$.
1) End For.
2) Estimate the confidence of each edge $e_{m,n}$ by:

$$\mathrm{P}(e_{m,n}) = \frac{1}{K}\sum_{k=1}^{K} f_{\text{True}}(e_{m,n} \in E_k), \tag{17}$$

where $f_{\text{True}}(e_{m,n} \in E_k)$ returns 1 if $e_{m,n} \in E_k$, and 0 otherwise.
5. Determine the confidence threshold p_{Th} [37].
6. Select every edge $e_{m,n}$ with $\mathrm{P}(e_{m,n}) > p_{\text{Th}}$. Use these edges to construct $G = (V, E)$.

More implementation details of Algorithm 3 can be found in [29] and [33].

C. Model Averaging

The aforementioned structure learning algorithms are able to accurately learn the BN structure of interest when the amount of training data is large. However, for our application of black-box testing, the number of training boards is often limited compared to the number of black-box tests. In this case, the structure learning algorithms can be unstable and inaccurate [22]. Even adding or deleting a small amount of training data can lead to significant changes in the learned structure (i.e., multiple edges can be added, deleted, or reversed).

To obtain a more accurate and robust BN structure, a model averaging strategy is usually applied to combine multiple BN structures. These multiple BN structures are learned from multiple subsets of the training data by using the same structure learning algorithm (e.g., one of the three learning algorithms in Section IV-B) [33], [35], [37]. In what follows, we will describe this model averaging method in detail.

In Algorithm 4, we summarize the model averaging procedure for BN learning. First, we choose one of the structure learning algorithms described in Section IV-B, and it will be used for all structure learning tasks in the following steps. Next, a data set $B^*{}_k$ is generated by bootstrap sampling, where N_{Train} samples are randomly drawn from B_{Train} with replacement [36]. The given structure learning algorithm is applied to $B^*{}_k$ to learn the BN structure: $G_k = (V, E_k)$. These two steps are repeated for K times to learn K different DAG structures. With these K structures, we estimate the confidence of each edge $e_{m,n}$ defined by $\mathrm{P}(e_{m,n})$ in (17), and determine a confidence threshold p_{Th} by minimizing the difference between the estimated confidence distribution and the ideal distribution [37]. More details on how to determine p_{Th} can be found in [37]. Finally, every edge $e_{m,n}$ with $P(e_{m,n}) > p_{\text{Th}}$ is selected, forming the edge set E for the average DAG structure $G = (V, E)$.

The aforementioned model averaging algorithm combines the DAG structures $\{G_k; k = 1, 2, \ldots, K\}$ learned with K different bootstrap sample sets. Only the edges existing in most DAG structures are used to construct the average structure G. Therefore, when K is sufficiently large, the average structure with fewer noisy edges should be more robust than the structure learned without averaging. In Section VII, we will further discuss how to appropriately choose the structure learning algorithm in step 1 of Algorithm IV.

V. Transfer Learning

In practice, the number of defective boards is often limited (e.g., for production with high yield), and it may be difficult to learn a sufficiently accurate BN model by directly applying the structure learning methods presented in the previous section. Especially for new products, the number of defective boards may be even less than the number of black-box tests. In these cases, additional information must be explored to learn an accurate BN model.

Fortunately, for most new products, there are similar old products (e.g., the previous generation of the same product, the same category of products for different applications, etc.) which share similarities with the new products. Besides, they may be tested by the same black-box tests as the new products. Thus, we can obtain abundant test data from these similar products, and use them to enhance BN structure learning.

The aforementioned idea of utilizing information from similar data set is known as transfer learning in [38]. It has become one of the cutting-edge topics in machine learning [39], and several transfer learning methods for BN models have been proposed in recent years [39]–[42].

In this section, we will focus on transfer learning only from one similar product $\mathbf{P}_{\text{Sim}}$, assuming that $\mathbf{P}_{\text{Sim}}$ has the same black-box test set T as the new product $\mathbf{P}_{\text{New}}$. We will particularly discuss three popular methods: 1) constrained-based transfer learning; 2) score-based transfer learning; and 3) hybrid transfer learning. In the future, we will work on other complex transfer learning problems (e.g., learning from multiple similar products, learning from products with a different black-box test set T, etc.)

A. Constraint-Based Transfer Learning

Constraint-based transfer learning [41] has been particularly developed for the constrain-based structure learning method in Section IV-B. It revises the methodology for conditional independence tests by incorporating valuable information from the similar product P_{Sim}.

Let $D(t_m, t_n|T_S)$ denote the conditional independence test decision for t_m and t_n given T_S in the new product P_{New}, as determined by (12). Let $D_{\text{Sim}}(t_m, t_n|T_S)$ denote the conditional independence test decision in the similar product P_{Sim}. When transfer learning is applied, the final conditional independence test decision $D_{\text{Trans}}(t_m, t_n|T_S)$ should combine the information from both P_{New} and P_{Sim}. Toward this goal, we first introduce two key terminologies: 1) the confidence parameter α to describe the confidence level of the conditional independence test on P_{New} or P_{Sim} and 2) the similarity measurement parameter β to describe the similarity between P_{New} and P_{Sim}.

For the confidence parameter α, it should be large when the amount of data used for independence test is large; otherwise,

it should be small. It has been shown that the error of the independence test based on mutual information is asymptotically proportional to $\log N/2N$, where N is the data set size [43]. Thus, the confidence parameters of the conditional independence test on t_m and t_n given T_S are defined for $\mathrm{P_{New}}$ and $\mathrm{P_{Sim}}$, respectively [41]

$$\alpha(t_m, t_n|T_S) = \max\left\{1 - \frac{\log N_{\mathrm{Train}}}{N_{\mathrm{Train}}} \cdot |T_S|, \quad 0\right\} \tag{18}$$

$$\alpha_{\mathrm{Sim}}(t_m, t_n|T_S) = \max\left\{1 - \frac{\log N_{\mathrm{Sim}}}{N_{\mathrm{Sim}}} \cdot |T_S|, \quad 0\right\} \tag{19}$$

where N_{Sim} is the total number of defective boards in $\mathrm{P_{Sim}}$. The function $\max\{\cdot, 0\}$ is used to keep α non-negative even when $|Ts|$ is extremely large and N_{Train} or N_{Sim} is extremely small.

The similarity measurement parameter β for t_m and t_n given T_S is defined as [41]

$$\beta(t_m, t_n|T_S) = \beta_G \cdot \beta_L(t_m, t_n|T_S) \tag{20}$$

where β_G is the global similarity parameter, and $\beta_L(t_m, t_n \mid T_S)$ is the local similarity parameter. The global similarity β_G is defined as

$$\beta_G = \frac{N_{\mathrm{Common}}}{N_{\mathrm{Ind}}} \tag{21}$$

where N_{Ind} is the number of independent pairs of black-box tests in $\mathrm{P_{New}}$, and N_{Common} is the number of common independent pairs of black-box tests between $\mathrm{P_{New}}$ and $\mathrm{P_{Sim}}$. Here, the independencies are checked by (12) with T_S as an empty set.

On the other hand, the local similarity $\beta_L(t_m, t_n|T_S)$ depends on the conditional independence test decision for t_m and t_n given T_S. If $D(t_m, t_n|T_S)$ for $\mathrm{P_{New}}$ and $D_{\mathrm{Sim}}(t_m, t_n \mid T_S)$ for $\mathrm{P_{Sim}}$ are identical, the local similarity should be high; otherwise, it should be low. $\beta_L(t_m, t_n \mid T_S)$ is defined as

$$\beta_L(t_m, t_n|T_S) = \begin{cases} 1, & D(t_m, t_n|T_S) = D_{\mathrm{Sim}}(t_m, t_n|T_S) \\ 0.5, & D(t_m, t_n|T_S) \neq D_{\mathrm{Sim}}(t_m, t_n|T_S) \end{cases} \tag{22}$$

where the constants 1 and 0.5 are recommended by [41].

Next, we define the weight w for $\mathrm{P_{New}}$ and the weight w_{Sim} for $\mathrm{P_{Sim}}$

$$w(t_m, t_n|T_S) = \alpha(t_m, t_n|T_S) \tag{23}$$

$$w_{\mathrm{Sim}}(t_m, t_n|T_S) = \alpha_{\mathrm{Sim}}(t_m, t_n|T_S) \cdot \beta(t_m, t_n|T_S). \tag{24}$$

By considering these weights, the mutual information in (11) is revised to combine the information from both $\mathrm{P_{New}}$ and $\mathrm{P_{Sim}}$

$$I_{\mathrm{Trans}}(t_m, t_n|T_S) = \sum_{\substack{i\in R_m \\ j\in R_n \\ k\in R_S}} \frac{W_{t_m,t_n,T_S}(i,j,k)}{W} \log\left(\frac{W_{t_m,t_n,T_S}(i,j,k) \cdot W_{T_S}(k)}{W_{t_m,T_S}(i,k) \cdot W_{t_n,T_S}(j,k)}\right) \tag{25}$$

where

$$W = w(t_m, t_n|T_S)N_{\mathrm{Train}} + w_{\mathrm{Sim}}(t_m, t_n|T_S)N_{\mathrm{Sim}} \tag{26}$$

$$W_{t_m,t_n,T_S}(i,j,k) = w(t_m, t_n|T_S) \cdot N_{t_m,t_n,T_S}(i,j,k) + w_{\mathrm{Sim}}(t_m, t_n|T_S) \cdot N_{\mathrm{Sim}:t_m,t_n,T_S}(i,j,k). \tag{27}$$

In (26) and (27), $N_{\mathrm{Sim:tm,tn,TS}}(i, j, k)$ denotes the number of defective boards in $\mathrm{P_{Sim}}$ whose test results satisfy $t_m = i$, $t_n = j$ and $T_S = k$. Similar to (26), we use weighted average to calculate $W_{\mathrm{tm,TS}}(i, k)$, $W_{\mathrm{tn,TS}}(j, k)$, and $W_{\mathrm{TS}}(k)$.

Finally, we revise (12) and make the decision on conditional independence test with the mutual information in (25)

$$D_{\mathrm{Trans}}(t_m, t_n|T_S) = \begin{cases} \text{Accept}, & I_{\mathrm{Trans}}(t_m, t_n|T_S) \leq I_{\mathrm{Th}} \\ \text{Reject}, & I_{\mathrm{Trans}}(t_m, t_n|T_S) > I_{\mathrm{Th}} \end{cases}. \tag{28}$$

Replacing the decision $D(t_m, t_n|T_S)$ by $D_{\mathrm{Trans}}(t_m, t_n|T_S)$ in step 4 of Algorithm 1, the constraint-based transfer learning method is summarized in Algorithm 5.

Algorithm 5 Constraint-Based Transfer Learning for BN

1. Initialize G as a complete undirected graph.
2. For $N_{\mathrm{Sep}} = 0, 1, \ldots, M - 2$:
3. For each edge $<t_m, t_n>$ in G:
4. If $\exists\, T_S \subseteq T \setminus \{t_m, t_n\}$ satisfying $|T_S| = N_{\mathrm{Sep}}$ and $D_{\mathrm{Trans}}(t_m, t_n \mid T_S) =$ Accept:
5. Delete $<t_m, t_n>$ from G.
6. $T_{\mathrm{Sep}}(t_m, t_n) = T_S$.
7. End If.
8. End For.
9. End For.
10. Determine the directions of all edges in G.

Algorithm 6 Score-Based Transfer Learning for BN

1. Choose the initial structure $G_{\mathrm{Trans}} = G_{\mathrm{Sim}}$, and set $\mathrm{S_{Max}} = \mathrm{S}(G_{\mathrm{Sim}})$.
2. Repeat:
3. For every G^* in $T_{\mathrm{Nb}}(G_{\mathrm{Trans}})$:
4. If $\mathrm{S}(G^*) > \mathrm{S}(G_{\mathrm{Trans}})$:
5. Set $\mathrm{S_{Max}} = \mathrm{S}(G^*)$ and $G_{\mathrm{Trans}} = G^*$.
6. End If.
7. End For.
8. Until $\mathrm{S_{Max}}$ does not increase further.

B. Score-Based Transfer Learning

Score-based transfer learning [41] has been particularly developed for the score-based structure learning method in Section IV-B. The basic idea is to utilize the BN structure G_{Sim} learned from $\mathrm{P_{Sim}}$ to initialize the structure learning on $\mathrm{P_{New}}$. In Algorithm 6, we summarize the score-based transfer learning flow. Starting from G_{Sim}, the HC algorithm in Algorithm 2 is applied to find the optimal BN structure with maximum score value based on the training data of $\mathrm{P_{New}}$.

Algorithm 6 is applicable to different choices of score functions. Although other scored-based methods have been proposed for transfer learning of BN structure [40], [42], these methods often rely on a log-likelihood score function without complexity penalty. For our application with small data set, these conventional methods tend to overfit the data and show poor performance [22]. Thus, we apply Algorithm 6 with BIC/EBIC score function for score-based transfer learning in this article.

Algorithm 7 Hybrid Transfer Learning for BN

1. For each t_m in T:
2. Set $T_{\mathrm{PrC}}(t_m) = \varnothing$.
3. Repeat
4. Select t_{Max} with the maximum $\mathrm{I}_{\mathrm{Trans}}(t_m, t_{\mathrm{Max}} \mid T_{\mathrm{PrC}}(t_m))$.
5. If $\mathrm{D}_{\mathrm{Trans}}(t_m, t_{\mathrm{Max}} \mid T_{\mathrm{PrC}}(t_m)) =$ Reject, stop iteration.
6. Set $T_{\mathrm{PrC}}(t_m) = T_{\mathrm{PrC}}(t_m) \cup \{t_{\mathrm{Max}}\}$.
7. End Repeat.
8. End For.
9. Use the HC algorithm to find the optimal G^* with the constraint that all parents of t_m belong to $T_{\mathrm{PrC}}(t_m)$.

C. Hybrid Transfer Learning

To improve the hybrid learning algorithm in Section IV-B, we propose a hybrid transfer learning method, as summarized in Algorithm 7. Compared to Algorithm 3, the hybrid transfer learning method combines the information from $\mathrm{P}_{\mathrm{Sim}}$ and $\mathrm{P}_{\mathrm{New}}$ in the restriction step and calculates the parent candidate set $T_{\mathrm{PrC}}(t_m)$ using the mutual information in (25) and the conditional independence test in (28). In the maximization step, the score-based HC algorithm is applied to find the optimal BN structure G^* with the constraints of $T_{\mathrm{PrC}}(t_m)$. Different from Algorithm 6 that takes G_{Sim} learned from $\mathrm{P}_{\mathrm{Sim}}$ as the initial structure, we set the initial structure G as a null graph because G_{Sim} may violate the constraints defined by $T_{\mathrm{PrC}}(t_m)$ while the null graph does not suffer from this issue.

VI. Test-Cost Reduction

Once the BN models are built with test data from B_{Train}, the strong relationships among different black-box tests are encoded by the BN structure. The remaining problem is how to utilize this information for black-box test-cost reduction. In this section, we propose an iterative test selection method in the light of BN models to solve the optimization problem in (3). To enhance the performance, we further propose a test-set averaging strategy for test selection.

A. Test Selection

To explain the proposed iterative test selection method, we first introduce a key concept: Markov blanket in BNs [22]. For each test t_m, the *Markov blanket* $T_{\mathrm{MB}}(t_m)$ is defined as a set containing the parent tests of t_m, the children tests of t_m, and the tests sharing at least one child with t_m. All tests out of the set $T_{\mathrm{MB}}(t_m)$ are conditionally independent of t_m, as stated by the following theorem [22].

Theorem 1: Given the tests in the Markov blanket $T_{\mathrm{MB}}(t_m)$, the test t_m is conditionally independent of all other tests out of the set $T_{\mathrm{MB}}(t_m)$.

Based on this property of Markov blanket, we define a Bayesian index F_{BN} for test selection

$$F_{\mathrm{BN}}(t_m, T_{\mathrm{Sel}}) = \frac{N_{t_m, T_{\mathrm{MB}}(t_m) \cap T_{\mathrm{Sel}}}(1, \mathbf{0})}{c_m} \tag{29}$$

where T_{Sel} is the set of black-box tests which is already selected, the test cost c_m can be chosen as the average test time (second) of t_m, 1 denotes the "fail" result for t_m, and $\mathbf{0}$ is a vector of 0s representing the "pass" results for all the tests in $T_{\mathrm{MB}}(t_m) \cap T_{\mathrm{Sel}}$. Thus, the Bayesian index denotes the number of defective boards in B_{Train} that can be detected per second by the test t_m while cannot be detected by previously selected tests in $T_{\mathrm{MB}}(t_m)$. It implies the efficiency of applying t_m to detect the uncovered defective boards.

Algorithm 8 Black-Box Test Selection

1. Build a BN model with the test data set B_{Train}.
2. Set $T_{\mathrm{Opt}} = \varnothing$, and $k = 0$.
3. Repeat:
4. Calculate $\mathrm{F}_{\mathrm{BN}}(t_m, T_{\mathrm{Opt}})$ for each test t_m.
5. If $c(T_{\mathrm{Opt}}) + c_m > c_{\mathrm{Th}}$ holds for every unselected test t_m, stop iteration.
6. Select the test t_{Max} with the maximum $\mathrm{F}_{\mathrm{BN}}(t_{\mathrm{Max}}, T_{\mathrm{Opt}})$ and satisfying $c(T_{\mathrm{Opt}} \cup \{t_{\mathrm{Max}}\}) \leq c_{\mathrm{Th}}$.
7. Set $T_{\mathrm{Opt}} = T_{\mathrm{Opt}} \cup \{t_{\mathrm{Max}}\}$, and $k = k + 1$.
8. Until $k > M$.

Algorithm 9 Test-Set Averaging

1. Choose a structure learning algorithm.
2. For $k = 1, 2, \ldots, K$:
3. Generate a new board set $B^*{}_k$ from B_{Train} by bootstrap sampling.
4. Apply the given structure learning algorithm to learn the BN structure $G_k = (V, E_k)$ based on $B^*{}_k$.
5. Apply Algorithm 8 on $B^*{}_k$ with G_k to learn the test set $T_{\mathrm{Opt},k}$.
6. End For.
7. Calculate the frequency f_m of each test t_m appeared in $\{T_{\mathrm{Opt},1}, T_{\mathrm{Opt},2}, \ldots, T_{\mathrm{Opt},K}\}$.
8. Sort $\{f_1, f_2, \ldots, f_M\}$ in descending order, and generate the sorted index list $\{i_1, i_2, \ldots, i_M\}$.
9. Set $T_{\mathrm{Opt}} = \varnothing$.
10. For $m = 1, 2, \ldots, M$:
11. If $c(T_{\mathrm{Opt}}) + \mathrm{c}_{i_m} = c_{\mathrm{Th}}$, set $T_{\mathrm{Opt}} = T_{\mathrm{Opt}} \cup \{\mathrm{c}_{i_m}\}$.
12. End For.

Algorithm 8 describes the steps associated with the proposed black-box test selection method. First, a BN model is built with the test data set B_{Train} using an aforementioned structure learning algorithm in Sections IV or V. The selected test set T_{Opt} is initialized as an empty set. Next, the test t_{Max} with the maximum value of $\mathrm{F}_{\mathrm{BN}}(t_{\mathrm{Max}}, T_{\mathrm{Opt}})$ and satisfying the test-cost constraint is selected and added to T_{Opt}. The selection is repeated until the condition $c(T_{\mathrm{Opt}}) + c_m > c_{\mathrm{Th}}$ holds for every unselected test t_m or all black-box tests are selected.

It is noteworthy that the conventional greedy algorithm is a special case of our proposed approach with a fully connected BN. For a fully connected BN, the Bayesian index in (29) can be rephrased as

$$F_{\mathrm{Greedy}}(t_m, T_{\mathrm{Sel}}) = \frac{N_{t_m, T_{\mathrm{Sel}}}(1, 0)}{c_m} \tag{30}$$

because the Markov blanket $T_{\mathrm{MB}}(t_m)$ equals $T - \{t_m\}$. Now, if we replace $F_{\mathrm{BN}}(t_m, T_{\mathrm{Opt}})$ with $F_{\mathrm{Greedy}}(t_m, T_{\mathrm{Opt}})$ in Algorithm 8, Algorithm 8 will be reduced to the conventional greedy algorithm. Because a fully connected BN keeps both strong and weak relationships among tests, the conventional greedy algorithm is likely to overfit the data.

B. Test-Set Averaging

Algorithm 8 is able to learn a stable and effective test set T_{Opt}. However, as training boards are often limited in our

application, the learning outcome T_{Opt} can be unstable. In order to learn a robust T_{Opt}, we propose a novel test-set averaging strategy. It learns different test sets from different subsets of training data and then combines these test sets to obtain a stable result.

The test-set averaging strategy is summarized in Algorithm 9. We first build K BN models with bootstrapping and learn the selected test sets: $\{T_{\text{Opt},1}, T_{\text{Opt},2}, \ldots, T_{\text{Opt},K}\}$. Next, we identify the tests occurring most frequently in $\{T_{\text{Opt},1}, T_{\text{Opt},2}, \ldots, T_{\text{Opt},K}\}$ and use these "frequent" tests to form T_{Opt}.

VII. Model Selection

In this article, we have introduced several different BN learning algorithms and averaging strategies. None of them would be optimal over all application scenarios. In order to select the optimal method in a specific scenario (i.e., each specific product), a model selection algorithm is needed.

Toward this goal, a cross validation (CV)-based model selection framework is proposed. We first use J-fold CV [44] to validate the performance of each test-cost reduction method and then select the optimal one with the best validation performance. Let $\{\text{m}_1, \text{m}_2, \ldots, \text{m}_L\}$ denote all available test-cost reduction methods. We split the training dataset B_{Train} into J folds, $\{B_{\text{CV},1}, B_{\text{CV},1}, \ldots, B_{\text{CV},J}\}$, and define $p_{\text{DC}}(B_{\text{CV},j}, \text{m}_l)$ as the defect coverage on $B_{\text{CV},j}$ by applying T_{Opt} learned from the other $J-1$ folds with the method m_l. The validation performance $p_{\text{DC}}(\text{m}_l)$ is calculated as

$$p_{\text{DC}}(m_l) = \frac{1}{J}\sum_{j=1}^{J} p_{\text{DC}}\big(B_{\text{CV},j}, m_l\big). \tag{31}$$

We select the optimal method m^* associated with the maximum $p_{\text{DC}}(m^*)$

$$m^* = \underset{\text{m}_l \in \{m_1, m_2, \ldots, m_L\}}{\arg\max} \{p_{\text{DC}}(m_l)\}. \tag{32}$$

Finally, by applying this optimal method m^* on the entire training dataset B_{Train}, we learn the optimal test set T^*_{Opt} as the final outcome.

When applying the aforementioned model selection approach, we consider the following BN learning algorithms: 1) constraint-based learning; 2) constraint-based transfer learning; 3) score-based learning; 4) score-based transfer learning; 5) hybrid learning; and 6) hybrid transfer learning. In 1), 2), 5), and 6), the p-value p_α in (13) corresponding to the threshold I_{Th} in (12) and (28) must be appropriately determined. In this article, we choose the optimal p_α among four possible candidates 0.01, 0.05, 0.1, and 0.2 [32] to maximize the performance metric in (31).

In addition, we consider the following three averaging strategies: 1) model averaging; 2) test-set averaging; and 3) no averaging. Combining them with different BN learning methods, our model selection algorithm selects the best average strategy to learn the optimal test set T^*_{Opt}.

VIII. Experimental Results

In this section, we demonstrate the effectiveness of the proposed test-cost reduction method using two case studies of network products. Both case studies are implemented with production test data. All experiments are performed on a computer with 2.6-GHz CPU and 8-GB memory.

In both cases, a metric must be defined to assess the effectiveness of test-cost reduction. Based on our problem formulation in Section III, all black-box tests are applied to B_{Train} for training the test selection model, and the selected test set T_{Opt} is applied to B_{Test}. Thus, the effectiveness of test-cost reduction is evaluated using the following metric:

$$r_{\text{Red}} = 100\% \cdot \left[1 - \frac{N_{\text{Train}} \cdot c(T) + N_{\text{Test}} \cdot c\big(T_{\text{Opt}}\big)}{(N_{\text{Train}} + N_{\text{Test}}) \cdot c(T)}\right]. \tag{33}$$

It denotes the ratio of test-cost reduction with respect to the total test cost of applying all black-box tests on both B_{Train} and B_{Test}.

For validation purpose, all pass/fail information for our testing samples is known. We first count how many failed boards are not detected by our selected test set. To compute the DPPM, we divide the aforementioned number by the total number of boards under test. Although we cannot disclose this total number because it carries proprietary information from our industry partner, the total number is sufficiently large to estimate the DPPM that is around 100 for both two network products in experiments below.

A. Network Product #1

In this example, 30 functional tests are applied to one network product $\text{P}_{\text{New},1}$, and 147 defective boards are detected. These tests check the memory functionality and take several hours to finish for each board. The same tests are also applied to a similar product $\text{P}_{\text{Sim},1}$, and 206 defective boards are detected. Because this network product is new and more boards will be tested in the future, test reduction is of great importance. For testing and comparison purposes, we include 50% of the defective boards in the set B_{Train} for training and the other defective boards in the set B_{Test}. When calculating test-cost reduction (r_{Red}), we assume that the same number of boards will be manufactured for this network product in the future with the same yield, and we add these new boards to B_{Test}. To remove random fluctuations of the experimental results, our experiments are repeated for 40 times with different boards in B_{Train} and B_{Test} by randomly shuffling the data.

By training our test selection methods with multiple test-cost thresholds (c_{Th}'s) ranging from 0 to $c(T)$ and applying the selected test sets on B_{Test}, we obtain the optimal test sets with different DPPMs for six different methods.

1) *Greedy:* The conventional greedy algorithm in [10] is applied.
2) *Score_BIC:* Algorithm 8 is applied with score-based BN learning using the BIC score function in (14).
3) *Score_EBIC:* Algorithm 8 is applied with score-based BN learning using the EBIC score function in (16).

TABLE IV
TEST-COST REDUCTION RESULTS OF NETWORK PRODUCT #1 FOR GREEDY, SCORE_BIC, SCORE_EBIC, CONSTRAINT, HYBRID, AND MS_SINGLE

Greedy		Score_BIC		Score_EBIC		Constraint		Hybrid		MS_Single	
DPPM	r_{Red}	DPPM	r_{Red}	DPPM	r_{Red}	DPPM	r_{Red}	DPPM	r_{Red}	DPPM	r_{Red}
243.2 (±22.6)	7.6%	152.5 (±24.7)	11.0%	152.5 (±26.1)	19.3%	152.5 (±24.2)	18.4%	152.5 (±26.1)	17.5%	152.5 (±23.8)	22.3%
243.2 (±22.6)	7.6%	152.5 (±24.7)	11.0%	108.1 (±25.7)	18.3%	108.1 (±25.3)	16.8%	108.1 (±28.4)	17.1%	108.1 (±26.5)	20.1%
243.2 (±22.6)	7.6%	152.5 (±24.7)	11.0%	95.3 (±25.2)	16.6%	95.3 (±23.3)	16.3%	95.3 (±20.3)	15.4%	95.3 (±20.4)	17.9%
243.2 (±22.6)	7.6%	152.5 (±24.7)	11.0%	63.6 (±20.3)	14.4%	63.6 (±20.4)	15.7%	63.6 (±24.4)	13.5%	63.6 (±20.3)	17.5%

TABLE V
TEST-COST REDUCTION RESULTS OF NETWORK PRODUCT #1 FOR GREEDY, MS_SINGLE, MS_TRANS, AND MS_MA_TRANS

Greedy		MS_Single		MS_Trans		MS_MA_Trans	
DPPM	r_{Red}	DPPM	r_{Red}	DPPM	r_{Red}	DPPM	r_{Red}
243.2 (±22.6)	7.6%	152.5 (±23.8)	22.3%	152.5 (±22.3)	23.6%	152.5 (±22.1)	27.8%
243.2 (±22.6)	7.6%	108.1 (±26.5)	20.1%	108.1 (±22.4)	22.2%	95.3 (±19.6)	25.0%
243.2 (±22.6)	7.6%	95.3 (±20.4)	17.9%	95.3 (±20.6)	21.7%	95.3 (±19.6)	25.0%
243.2 (±22.6)	7.6%	63.6 (±20.3)	17.5%	63.6 (±18.2)	19.9%	63.6 (±18.2)	21.0%

TABLE VI
TEST-COST REDUCTION RESULTS OF NETWORK PRODUCT #2 FOR GREEDY, SCORE_BIC, SCORE_EBIC, CONSTRAINT, HYBRID, AND MS_SINGLE

Greedy		Score_BIC		Score_EBIC		Constraint		Hybrid		MS_Single	
DPPM	r_{Red}	DPPM	r_{Red}	DPPM	r_{Red}	DPPM	r_{Red}	DPPM	r_{Red}	DPPM	r_{Red}
224.1 (±16.7)	27.3%	150.9 (±15.3)	35.0%	150.9 (±17.1)	35.4%	150.9 (±16.1)	35.3%	150.9 (±16.9)	36.5%	150.9 (±16.5)	36.5%
96.0 (±13.7)	21.1%	96.0 (±17.1)	31.4%	96.0 (±15.5)	33.5%	96.0 (±16.8)	31.6%	96.0 (±16.5)	34.5%	96.0 (±15.4)	34.5%
96.0 (±13.7)	21.1%	64.0 (±16.1)	29.6%	64.0 (±16.1)	30.7%	64.0 (±15.7)	29.7%	64.0 (±16.6)	30.9%	64.0 (±16.6)	31.1%

4) *Constraint:* Algorithm 8 is applied with constrained-based BN learning and $p_\alpha = 0.5$.
5) *Hybrid:* Algorithm 8 is applied with hybrid BN learning and $p_\alpha = 0.5$.
6) *MS_Single:* The model selection algorithm in Section VII is applied based on fivefold CV without considering transfer learning, model averaging, or test-set averaging.

To compare the efficacy among these six methods, we show the test-cost reduction results (r_{Red}) in Table IV. These DPPMs are displayed as x_{Avg} ($\pm x_{\text{Std}}$), where x_{Avg} and x_{Std} denote the mean and standard deviation of DPPM estimated from 40 repeated runs, respectively. As shown in each row of Table IV, Greedy offers the worst performance with the largest DPPM and the least test-cost reduction (r_{Red}). On the other hand, MS_Single achieves the best performance among all methods with the least DPPM and the largest test-cost reduction. In this example, MS_Single achieves up to 22.3%–7.6% = 14.7% test-cost reduction over Greedy.

By further comparing Score_BIC, Score_EBIC, Constraint, Hybrid, and MS_Single in Table IV, we have two important observations. First, there is no single BN-based method that consistently out-performs the others. Second, the proposed model selection strategy (i.e., MS_Single) is able to appropriately choose the optimal BN-based method in each case.

Table V further compares Greedy and MS_Single with the following two test-cost reduction approaches.

1) *MS_Trans:* Model selection is applied with transfer learning, without considering model averaging or test-set averaging.
2) *MS_MA_Trans:* Model selection is applied with transfer learning, model averaging and test-set averaging.

The implementation of model selection is based on fivefold CV. The parameter K is set to 100 for both model averaging and test-set averaging. This value of K is sufficiently large to ensure that the experimental results are stable for our industrial data sets [36].

As shown in each row of Table V, MS_Trans offers larger test-cost reduction than MS_Single with the same DPPM, demonstrating the effectiveness of transfer learning based on the similar product $\text{P}_{\text{Sim},1}$. In addition, MS_MA_Trans offers the best performance among all methods with the least DPPM and the largest test-cost reduction, further demonstrating the effectiveness of our proposed averaging strategy. In this example, MS_MA_Trans achieves up to 25.0%–17.9% = 7.1% test-cost reduction over MS_Single. On the other hand, MS_MA_Trans is most time-consuming among all these approaches, and its runtime is less than 1 h given a fixed c_{Th} in our experiment.

B. Network Product #2

In this example, 21 functional tests are applied to one network product $P_{\text{New},2}$, and 414 defective boards are detected. These tests check activation, query and other related functions. They take several hours to finish for each board. The same tests are also applied to a similar product $\text{P}_{\text{Sim},2}$, and 473 defective boards are detected. Since the number of defective boards in this example is much greater than that in the previous example, we only include 20% of the defective boards in the set B_{Train} for training, and the other 80% defective boards in the set B_{Test}. Our experiments are repeated for 40 times with different boards in B_{Train} and B_{Test} by randomly shuffling the data.

We compare the test-cost reduction results for six different methods: 1) Greedy; 2) Score_BIC; 3) Score_EBIC;

TABLE VII
TEST-COST REDUCTION RESULTS OF NETWORK PRODUCT #2 FOR GREEDY, MS_SINGLE, MS_TRANS, AND MS_MA_TRANS

Greedy		MS_Single		MS_Trans		MS_MA_Trans	
DPPM	r_{Red}	DPPM	r_{Red}	DPPM	r_{Red}	DPPM	r_{Red}
224.1 (±16.7)	27.3%	150.9 (±16.5)	36.5%	150.9 (±8.5)	37.2%	146.3 (±9.4)	37.7%
96.0 (±13.7)	21.1%	96.0 (±15.4)	34.5%	96.0 (±7.8)	35.0%	64.0 (±8.1)	37.1%
96.0 (±13.7)	21.1%	64.0 (±16.6)	31.1%	64.0 (±7.8)	33.7%	64.0 (±7.5)	37.1%

4) Constraint; 5) Hybrid; and 6) MS_Single (with fivefold CV) in Table VI. Similar to the previous example, Greedy offers the worst performance, while MS_Single achieves the best performance. In this example, MS_Single achieves up to 34.5%–21.1% = 13.4% test-cost reduction over Greedy.

In Table VII, we further compare the test-cost reduction for MS_Single, MS_Trans, and MS_MA_Trans, where five-fold CV is applied for model selection. The parameter K is set to 100 for both model averaging and test-set averaging. MS_Trans offers larger test-cost reduction than MS_Single, and MS_MA_Trans achieves the best performance among all methods. In this experiment, MS_MA_Trans achieves up to 37.1%–31.1% = 6.0% test-cost reduction over MS_Single. On the other hand, MS_MA_Trans is most time-consuming, and its runtime is less than 1 h given a fixed c_{Th}.

IX. CONCLUSION

We have presented a test selection method to reduce the cost of black-box testing. First, to extract the strong relationships among black-box tests, BN models have been constructed using multiple structure learning algorithms. Transfer learning algorithms are further adopted if prior information is provided from similar products. Next, based on these BN models, an iterative test selection method-based upon a new metric, Bayesian index, has been proposed for test-cost reduction. Averaging strategies have also been proposed to enhance the performance of test-cost reduction. Finally, a CV-based model selection framework is proposed to select the optimal BN model for test-cost reduction.

As demonstrated by two examples with production test data, the proposed approach achieves up to 14.7% more test-cost reduction than a conventional greedy method when no prior information is provided. When considering prior information from similar products, our proposed method can further reduce the black-box test cost by up to 7.1%.

REFERENCES

[1] R. Pan, Z. Zhang, X. Li, K. Charabarty, and X. Gu, "Black-box test-coverage analysis and test-cost reduction based on a Bayesian network model," in *Proc. 37th VLSI Test Symp.*, Monterey, CA, USA, 2019, pp. 1–6.

[2] C. Mack, "The multiple lives of Moore's law," *IEEE Spectr.*, vol. 52, no. 4, p. 31, Apr. 2015.

[3] *International Technology Roadmap for Semiconductors*, Semicond. Ind. Assoc., San Jose, CA, USA, 2015.

[4] M. Bushnell and V. Agrawal, *Essentials of Electronic Testing for Digital, Memory and Mixed-Signal VLSI Circuits*. Boston, MA, USA: Springer, 2004.

[5] H. Chen, "Beyond structural test, the rising need for system-level test," in *Proc. Int. Symp. VLSI Design Autom. Test*, Hsinchu, Taiwan, 2018, pp. 1–4.

[6] R. Thilak and S. Gayathri, "Fault coverage analysis using fault model and functional testing for DPM reduction," in *Proc. Int. Conf. Emerg. Res. Electron. Comput. Sci. Technol.*, Mandya, India, 2015, pp. 76–81.

[7] M. Chen and A. Orailoglu, "Test cost minimization through adaptive test development," in *Proc. Int. Conf. Comput. Design*, Lake Tahoe, CA, USA, 2008, pp. 234–239.

[8] N. Akkouche, S. Mir, E. Simeu, and M. Slamani, "Analog/RF test ordering in the early stages of production testing," in *Proc. 30th VLSI Test Symp.*, Hyatt Maui, HI, USA, 2012, pp. 25–30.

[9] F. Ye, Z. Zhang, K. Chakrabarty, and X. Gu, "Board-level functional fault diagnosis using artificial neural networks, support-vector machines, and weighted-majority voting," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 32, no. 5, pp. 723–736, May 2013.

[10] C. Xue and R. Blanton, "A one-pass test-selection method for maximizing test coverage," in *Proc. 33rd IEEE Int. Conf. Comput. Design*, 2015, pp. 621–628.

[11] H. Gonçalves, M. Correia, V. Tavares, J. Carulli, and K. Butler, "A fast spatial variation modeling algorithm for efficient test cost reduction of analog/RF circuits," in *Proc. Design Autom. Test Eur. Conf. Exhibit.*, Grenoble, France, 2015, pp. 1042–1047.

[12] A. Ahmadi, A. Nahar, B. Orr, M. Past, and Y. Markis, "Wafer-level process variation-driven probe-test flow selection for test cost reduction in analog/RF ICs," in *Proc. 34th VLSI Test Symp.*, Las Vegas, NV, USA, 2016, pp. 1–6.

[13] K. Wang *et al.*, "Test cost reduction methodology for InFO wafer-level chip-scale package," *IEEE Design Test*, vol. 34, no. 3, pp. 50–58, Jun. 2017.

[14] M. Agrawal and K. Chakrabarty, "Test-cost modeling and optimal test-flow selection of 3-D-stacked ICs," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 34, no. 9, pp. 1523–1536, Sep. 2015.

[15] M. Grady, B. Pepper, J. Patch, M. Degregorio, and P. Nigh, "Adaptive testing—Cost reduction through test pattern sampling," in *Proc. Int. Test Conf.*, Anaheim, CA, USA, 2013, pp. 1–8.

[16] M. Liu, R. Pan, F. Ye, X. Li, K. Chakrabarty, and X. Gu, "Fine-grained adaptive testing based on quality prediction," in *Proc. Int. Test Conf.*, Phoenix, AZ, USA, 2018, pp. 1–10.

[17] L. Wang, C. Wu, and X. Wen, *VLSI Test Principles and Architectures: Design for Testability*. Amsterdam, The Netherlands: Elsevier, 2006.

[18] A. Touati, A. Bosio, P. Girard, A. Virazel, P. Bernardi, and M. S. Reorda, "Microprocessor testing: Functional meets structural test," *J. Circuits Syst. Comput.*, vol. 26, no. 8, 2017, Art. no. 1740007,

[19] X. Lin, J. Rajski, I. Pomeranz, and S. M. Reddy, "On static test compaction and test pattern ordering for scan designs," in *Proc. Int. Test Conf.*, Baltimore, MD, USA, 2001, pp. 1088–1097.

[20] S. Khuller, A. Moss, and J. Naor, "The budgeted maximum coverage problem," *Inf. Process. Lett.*, vol. 70, no. 1, pp. 39–45, 1999.

[21] M. Liu, F. Ye, X. Li, K. Chkrabarty, and X. Gu, "Board-level functional fault identification using streaming data," in *Proc. 37th VLSI Test Sympo.*, Monterey, CA, USA, 2019, pp. 1–6.

[22] D. Koller and N. Friedman, *Probabilistic Graphical Models: Principles and Techniques*. Cambridge, MA, USA: MIT Press, 2009.

[23] I. Tsamardinos, C. F. Aliferis, and A. R. Statnikov, "Algorithms for large scale Markov blanket discovery," in *Proc. Florida Artif. Intell. Res. Soc. Conf.*, 2003, pp. 376–380.

[24] C. Aliferis *et al.*, "Local causal and Markov blanket induction for causal discovery and feature selection for classification part I: Algorithms and empirical evaluation," *J. Mach. Learn. Res.*, vol. 11, no. 7, pp. 171–234, 2010.

[25] P. Spirtes, C. N. Glymour, and R. Scheines, *Causation, Prediction, and Search*. Cambridge, MA, USA: MIT Press, 2000.

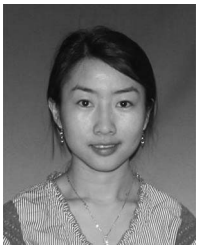
[26] D. Colombo and M. Maathuis, "Order-independent constraint-based causal structure learning," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 3741–3782, 2014.

[27] D. Heckerman, D. Geiger, and D. M. Chickering, "Learning Bayesian networks: The combination of knowledge and statistical data," *Mach. Learn.*, vol. 20, no. 3, pp. 197–243, 1995.

[28] R. Foygel and M. Drton, "Extended Bayesian information criteria for Gaussian graphical models," in *Proc. 23rd Int. Conf. Adv. Neural Inf. Process. Syst.*, 2010, pp. 604–612.
[29] I. Tsamardinos, L. E. Brown, and C. F. Aliferis, "The max–min hill-climbing Bayesian network structure learning algorithm," *Mach. Learn.*, vol. 65, no. 1, pp. 31–78, 2006.
[30] M. Scutari, C. E. Graafland, and J. M. Gutiérrez, "Who learns better Bayesian network structures: constraint-based, score-based or hybrid algorithms?" in *Proc. 9th Int. Conf. Probab. Graph. Model.*, 2018, pp. 416–427.
[31] R. Neapolitan, *Learning Bayesian Networks*, vol. 38. Upper Saddle River, NJ, USA: Pearson Educ, 2004.
[32] P. Greenwood and M. Nikulin, *A Guide to Chi-Squared Testing*, vol. 280. New York, NY, USA: Wiley, 1996.
[33] M. Scutari and J. Denis, *Bayesian Networks: With Examples in R*. Boca Raton, FL, USA: Chapman-Hall, 2014.
[34] G. Schwarz, "Estimating the dimension of a model," *Ann. Stat.*, vol. 6, no. 2, pp. 461–464, 1978.
[35] N. Friedman, M. Goldszmidt, and A. Wyner, "Data analysis with Bayesian networks: A bootstrap approach," in *Proc. 15th Conf. Uncertainty Artif. Intell.*, 1999, pp. 196–205.
[36] B. Efron and R. Tibshirani, *An Introduction to the Bootstrap*. Boca Raton, FL, USA: CRC Press, 1994.
[37] M. Scutari and R. Nagarajan, "Identifying significant edges in graphical models of molecular networks," *Artif. Intell. Med.*, vol. 57, no. 3, pp. 207–217, 2013.
[38] S. Pan and Q. Yang, "A survey on transfer learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, Oct. 2010.
[39] J. Lu, V. Behbood, P. Hao, H. Zuo, S. Xue, and G. Zhang, "Transfer learning using computational intelligence: A survey," *Knowl. Based Syst.*, vol. 80 no. 2, pp. 14–23, 2015.
[40] A. Niculescu-Mizil and R. Caruana, "Inductive transfer for Bayesian network structure learning," in *Proc. 11th Int. Conf. Artif. Intell. Stat.*, 2007, pp. 339–346.
[41] R. Luis and E. Sucar, "Inductive transfer for learning Bayesian networks," *Mach. Learn.*, vol. 79, nos. 1–2, pp. 227–255, 2010.
[42] D. Oyen and T. Lane, "Leveraging domain knowledge in multitask Bayesian network structure learning," in *Proc. AAAI Conf. Artif. Intell.*, 2012, pp. 1091–1097.
[43] N. Friedman and Z. Yakhini, "On the sample complexity of learning Bayesian networks" in *Proc. Conf. Uncertainty Artif. Intell.*, 1999, pp. 274–282.
[44] C. Bishop, *Pattern Recognition and Machine Learning*. New York, NY, USA: Springer, 2006.

Renjian Pan (Student Member, IEEE) received the B.S. degree from the School of Microelectronics, Fudan University, Shanghai, China, in 2017. He is currently pursuing the Ph.D. degree with Duke University, Durham, NC, USA.

He was an Intern with Futurewei Technologies, Santa Clara, CA, USA. His current research interests include machine learning, data analytics, manufacturing test, and diagnosis.



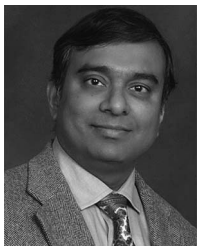
Zhaobo Zhang received the B.Eng. degree from the Department of Electronics Engineering, Tsinghua University, Beijing, China, in 2007, and the M.S.E. and Ph.D. degrees from the Department of Electrical and Computer Engineering, Duke University, Durham, NC, USA, in 2009 and 2011, respectively.

She was an Intern with IBM Research, Yorktown Heights, NY, USA, and Cisco Systems, Inc., San Jose, CA, USA. She is currently with Futurewei Technologies, Santa Clara, CA, USA. As a Staff Engineer, she leads and develops software products to improve system reliability and process automation. Her interests include machine learning, data analytics, real-time monitoring, system reliability, and testing.

Xin Li (Fellow, IEEE) received the Ph.D. degree in electrical and computer engineering from Carnegie Mellon University, Pittsburgh, PA, USA, in 2005, and the M.S. and B.S. degrees in electronics engineering from Fudan University, Shanghai, China, in 2001 and 1998, respectively.

He is currently a Professor with the Department of Electrical and Computer Engineering, Duke University, Durham, NC, USA, and leading the Institute of Applied Physical Sciences and Engineering, Duke Kunshan University, Kunshan, China. His research interests include integrated circuit, signal processing, and data analytics.

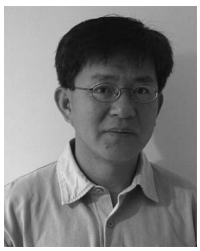
Dr. Li received the NSF CAREER Award in 2012, two IEEE Donald O. Pederson Best Paper Awards in 2013 and 2016, the DAC Best Paper Award in 2010, two ICCAD Best Paper Awards in 2004 and 2011, the ISIC Best Paper Award in 2014, and the Cadence Academic Collaboration Award in 2018, and the Six Best Paper Nominations from DAC, ICCAD, and CICC. He is the Deputy Editor-in-Chief of the IEEE TRANSACTIONS ON COMPUTER-AIDED DESIGN OF INTEGRATED CIRCUITS AND SYSTEMS. He was an Associate Editor of the IEEE TRANSACTIONS ON COMPUTER-AIDED DESIGN OF INTEGRATED CIRCUITS AND SYSTEMS, the IEEE TRANSACTIONS ON BIOMEDICAL ENGINEERING, the *ACM Transactions on Design Automation of Electronic Systems*, IEEE DESIGN&TEST, and *IET Cyber-Physical Systems*. He served on the Executive Committee of DAC, ACM SIGDA, IEEE TCCPS, and IEEE TCVLSI. He was the General Chair of ISVLSI, iNIS, and FAC, and the Technical Program Chair of CAD/Graphics.



Krishnendu Chakrabarty (Fellow, IEEE) received the B.Tech. degree from the Indian Institute of Technology Kharagpur, Kharagpur, India, in 1990, and the M.S.E. and Ph.D. degrees from the University of Michigan at Ann Arbor, Ann Arbor, MI, USA, in 1992 and 1995, respectively.

He is currently the John Cocke Distinguished Professor with the Department Chair of Electrical and Computer Engineering, Duke University, Durham, NC, USA. His current research interests include design-for-test and fault diagnosis of SoCs and 3-D integrated circuits; microfluidic biochips; hardware security; neuromorphic computing systems.

Dr. Chakrabarty was a recipient of the National Science Foundation CAREER Award, the Office of Naval Research Young Investigator Award, the Humboldt Research Award from the Alexander von Humboldt Foundation, Germany, the *ACM Transactions on Design Automation of Electronic Systems* Best Paper Award in 2017, the IEEE TRANSACTIONS ON CAD Donald O. Pederson Best Paper Award in 2015, over a dozen best paper awards at major conferences, the IEEE Computer Society Technical Achievement Award in 2015, the IEEE Circuits and Systems Society Charles A. Desoer Technical Achievement Award in 2017, the Semiconductor Research Corporation Technical Excellence Award in 2018, the IEEE Test Technology Technical Council Bob Madge Innovation Award in 2018, and the Japan Society for the Promotion of Science Invitational Fellowship in the "Short Term S: Nobel Prize Level" category 2018. He is a fellow of ACM and AAAS.



Xinli Gu received the B.Sc. degree in computer science from Shanghai JiaoTong University, Shanghai, China, in 1982, and the M.Sc. and Ph.D. degrees in computer science and information from Linköping University, Linköping, Sweden, in 1992 and 1996, respectively.

He was the Director with Cisco Systems, Inc., San Jose, CA, USA, with 12 years experience, responsible for corporate level product quality, time-to-market, manufacturing test/diagnosis cost reduction, and product yield improvement. He is a Senior Director with Futurewei Technologies, Santa Clara, CA, USA, leading design solution and driving corporate process for complex Telecom product quality and reliability. He also worked with Synopsys, Mountain View, CA, USA, and Ericsson, Stockholm, Sweden. He has published over 30 technical papers and holds several U.S. patents.

Dr. Gu is an active IEEE member and leads system reliability and ASIC DFT test.