Multiple Distributions for Biased Random Test Patterns

HANS-JOACHIM WUNDERLICH, MEMBER, IEEE

Abstract—The test of integrated circuits by random patterns is very attractive, since no expensive test pattern generation is necessary and tests can be applied with a self-test technique or externally using linear feedback shift registers. Unfortunately, not all circuits are random testable, because either the fault coverage is too low or the required test length too large. In many cases the random test lengths can be reduced by orders of magnitude using weighted random patterns. However, there are also some circuits for which no single optimal set of weights exists. A set of weights defines a distribution of the random patterns.

In this paper, it is shown that the problem can be solved using several distributions instead of a single one. Furthermore, an efficient procedure for computing the optimized input probabilities is presented. If a sufficient number of distributions is applied, then all combinational circuits can be tested randomly with moderate test lengths. The patterns can be produced by an external chip, and an optimized test schedule for circuits with a scan path can be obtained. Moreover, formulas are derived to determine strong bounds on the probability of detecting all faults.

Fault simulation with weighted pattern shows a nearly complete coverage of all nonredundant faults.

Keywords—Random tests, biased random patterns, multiple weights, low cost test.

I. INTRODUCTION

TESTING by random patterns has many advantages compared to other test strategies, e.g., the self-test capability, a reduction of computing time, and high coverage of parametric faults. During the last few years many papers have been published on problems concerning random tests, such as computing fault detection probabilities and test lengths. Some of these results will be summarized as prerequisites of the later investigations.

In Section II, a theorem is established which provides a strong bound on the probability that all faults of a given set are detected by a given amount of random patterns. Another theorem shows that a real random test and a pseudorandom test by shift register sequences require the same length if the number of primary inputs is sufficiently large.

It is shown that in general the fault coverage increases, and the overall test length decreases if several random pattern sets with different 1-probabilities are applied. The

optimized input probabilities can be computed numerically if a procedure which satisfies certain restrictions is available for estimating fault detection probabilities. These restrictions are discussed at the end of Section II.

In Section III, the complexity of computing an optimized random test schedule is determined. Since this problem is at least *NP*-hard, we avoid the exact calculation by using an efficient heuristic (Section IV). Some implementation details are given in Section V and results are discussed in Section VI. Finally we present a system for generating weighted random test patterns having multiple distributions, which is used for the external test of circuits with integrated scan path.

II. BASIC FACTS

2.1. Fault Detection Probabilities

One of the main tasks in random pattern testing is the computation of fault detection probabilities. Many tools and algorithms have been proposed for estimating these probabilities (e.g., [9], [2], [22], [10], [1]). However, most of them are restricted to the usual stuck-at fault model, but an extension to more complex faults is easy unless a sequential behavior is involved [27]. Unfortunately, algorithms for the exact computation of fault detection probabilities have a very high worst-case complexity.

Observation 1: Computing fault detection probabilities is at least *NP*-hard.

This observation is a simple consequence of the *NP*-completeness of the fault detection problem [13], and hence, all known methods for estimating fault detection probabilities analytically in polynomial time have unbounded relative estimation errors. The conjecture that a stochastic Monte Carlo algorithm would yield a higher precision more efficiently is not true.

Observation 2: Estimating fault detection probabilities is #-complete, i.e., one cannot expect a stochastic algorithm with a sample size bounded by a polynomial in the reciprocal of the relative estimation error.

The proof of Observation 2 is straightforward using methods, for instance, presented in [12]. That means we cannot expect tools for estimating fault detection probabilities with arbitrary high precision in polynomial computing time. As we have to put up with approximated fault de-

tection probabilities, algorithms to compute lengths of random tests based on these values should not be over sophisticated.

2.2. Fault Detection Probabilities and Test Lengths

There are some papers on computing required test lengths for pseudorandom patterns applied by BIST registers or a low-cost test equipment [20]. Taking the pseudorandomness into account, shorter test lengths are obtained but more complex algorithms are required. This is not necessary as random models are sufficiently precise.

Theorem 1: Let $p < 1/4$ be the detection probability of a fault f in a combinational circuit with $i > 4$ primary inputs. Let ϵ be the escape probability that f is neither detected by N_r random patterns nor by N_p pseudorandom patterns. For $2^{i/2} \geq N_p$ we have $N_r \leq N_p + 2(1 - \ln(\epsilon))$, i.e., $N_r \approx N_p$.

Proof: (See Appendix A).
Even if we assume the very low escape probability of $\epsilon := 0.001$, theorem 1 provides $N_r \leq N_p + 16$. Hence, the theorem leads to observation 3.

Observation 3: For not exhaustively testable circuits, the difference between the length of a random test and the length of a pseudorandom test is negligible.

If an exhaustive test of the circuit is possible, then it is preferred anyway. As a consequence, we can use the random model assumption without any loss of generality for those circuits where an exhaustive test is impossible and where $N_r \leq 2^{i/2}$. For instance, if we have to apply less than 8000 patterns to a circuit with more than 25 primary inputs, then random and pseudorandom pattern sets will have the same size.

Now let F be a set of faults of the combinational circuit C with n inputs. Faults in F should not induce sequential behavior. Let $X := \langle x_1, \cdots, x_n \rangle \in [0, 1]^n$ be a tuple of real numbers, denoting the 1-probability for each primary input. For a fault f, let $p_f(X)$ be the probability of detecting that fault by a random pattern generated with distribution X. The probability that all faults of F are detected with N random patterns is estimated by

$$J_N(X) = \prod_{f \in F} \left(1 - \left(1 - p_f(X)\right)^N\right). \tag{1}$$

Of course, formula (1) only holds if we assume that the detection of some faults by N patterns forms completely independent events. Therefore, some authors try to compute an exact value by means of Markov-theory [5], but the next theorem shows that formula (1) is a very precise estimation.

Let $\langle f_i \rangle_{1 \leq i \leq \mu}$ be an enumeration of F, where $i < j$ implies $p_{f_i} \leq p_{f_j}$. In order to simplify the notation we omit the concrete distribution X. The expression $P(A, N)$ denotes the probability to detect all faults of the set $A \subseteq F$

by N random patterns under some distribution X. Then we can show the following theorem.

Theorem 2:
Set

$$J_N := \prod_{f \in F} \left(1 - (1 - p_f)^N\right).$$

Then

$$J_N - (1 - J_N) \sum_{j=2}^{\mu} \left(1 - p_{f_j}\right)^N$$

$$\leq P(F, N)$$

$$\leq J_N + \sum_{j=2}^{\mu} \left(1 - p_{f_j}\right)^N \prod_{k=1}^{j-1} \left(1 - (1 - p_{f_k})^N\right).$$

Proof: (See Appendix B).
Using this theorem we can state the following.

Observation 4: Let $J_N \approx 1$ be the derived probability to detect all faults. Formula (1) underestimates the exact probability by less than $O(|\ln(J_N)|)$ and overestimates it by less than $O(|(1 - J_N) \ln(J_N)|)$.

Observation 4 is derived by

$$\sum_{j=2}^{\mu} \left(1 - p_{f_j}\right)^N \prod_{k=1}^{j-1} \left(1 - (1 - p_{f_k})^N\right) \leq \sum_{j=2}^{\mu} \left(1 - p_{f_j}\right)^N$$

$$\leq \sum_{f \in F} \left(1 - p_f\right)^N$$

$$\leq -\sum_{f \in F} \ln\left(1 - (1 - p_f)^N\right) = -\ln(J_N)$$

using formula (12) in the Appendix.

For instance, if we have three faults with $p_{f1} = 10^{-7}$, $p_{f2} = 5 \cdot 10^{-7}$ and $p_{f3} = 10^{-6}$, then using formula (1), we will need $N = 69 \cdot 10^6$ patterns, in order to detect all faults with probability 0.999. The estimation of Theorem 2 yields:

$$0.999 - 10^{-18} \leq P(\{f_1, f_2, f_3\}, N) \leq 0.999 + 10^{-15}.$$

The following fact has already been observed in [5].

Observation 5: Only the faults with lowest detection probability have impact on the necessary test length.

In [23] it is discussed that those faults having a detection probability which is more than 10 times larger than the minimal detection probability can be neglected.

The next statement has already been established in [18].

Observation 6: The necessary number of random patterns increases linearly with the reciprocal of the minimal fault detection probability.

Thus during a conventional random test, the size of a test can grow exponentially with the number of inputs. For instance, consider an AND32 (Fig. 1) where each of
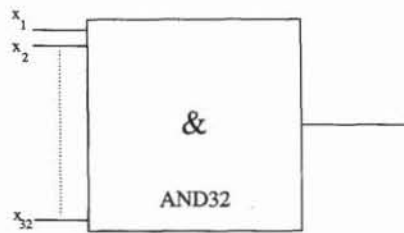
Fig. 1. 32 Input AND.

the 32 inputs is set to "1" with probability x. Then an arbitrary stuck-at-0 fault is detected with probability x^{32}, and each of the 32 stuck-at-1 faults with probability $(1 - x) \cdot x^{31}$. For $x = 0.5$ and a test confidence of 0.999 for detecting all 33 faults, formula (1) yields $0.999 = (1 - (1 - 2^{-32})^N)^{33}$ and $N \approx 4.48 \cdot 10^{10}$.

By using unequiprobable patterns, i.e., $x \neq 0.5$, test lengths can be reduced drastically [22], [15]. For example, by setting

$$x := \sqrt[32]{0.5}$$

we would need approximately $N \approx 6 \cdot 10^2$ patterns.

Observation 7: There are circuits for which using unequiprobable patterns can reduce the test lengths by orders of magnitude.

In [23], [24] an efficient procedure for computing optimized input probabilities was presented. Unfortunately, some circuits are resistant to this optimization when only a single distribution X is used. For the connection of an AND32 and an OR32 in Fig. 2 no better single distribution exists than $\langle x_i = 0.5 \mid i \in I \rangle$.

The problem is solved by first applying 600 patterns with input probability

$$x := \sqrt[32]{0.5}$$

and then 600 patterns with input probability

$$x := 1 - \sqrt[32]{0.5}.$$

This way we obtain a complete fault coverage with confidence 0.999 with $N_r = 1200$ random patterns.

In the rest of this paper, we deal with the problem of computing several distributions X for random patterns in order to minimize the overall test length based on tools estimating fault detection probabilities.

2.3. Efficiency and Accuracy of the Testability Measures

Computing optimized distributions X is essentially done by numerical algorithms maximizing formula (1). Recently several algorithms have been proposed for exactly computing fault detection probabilities using a 4-valued logic [10] or using some graph-theoretic properties of the
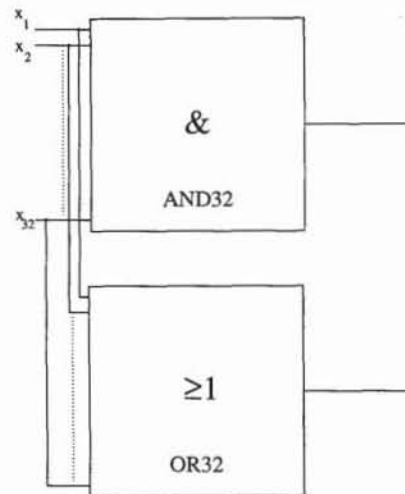

Fig. 2. Not random testable circuit.

circuit [19]. However, until now, no reports about their measured performance are available. Furthermore, fault detection probabilities have to be estimated by a testability measure which meets the following requirements.

a) High Efficiency: During optimization, the detection probabilities $p_f(X)$ have to be evaluated very often for different arguments X. The involved fault detection probabilities $p_f(X)$ cannot be determined exactly due to the high problem complexity. Therefore, we have to use heuristics for estimating fault detection probabilities, and dispense with the exact computation.

b) Unique Results Instead of Bounds: The so-called "cutting algorithm" is a heuristic algorithm to compute bounds on the detection and signal probabilities [9]. If one of the bounds is constant 0 or 1, this information is not sufficient, since these bounds are not sensitive to variations of the input probability distribution X. In order to decide on optimal input probabilities, especially the response to faults with low detection, probabilities are interesting. Thus we need a single real number as an estimate.

c) Handling Weighted Input Probabilities: For each fault, its detection probability will be computed several times for different input probabilities X. If these probabilities differ only in a few input positions, the algorithm should take advantage of this fact.

d) No Random Errors: If the algorithm is run for different input probabilities, the estimation error should not be a random variable. If the error is random, computing optimal weights turns into a stochastic optimizing problem which has a very high complexity. Standard optimizing procedures like the Newton iteration in general do not converge based on stochastic inputs. Only in special cases can these algorithms be modified, for instance, for PLA's as described in [26].

These four requirements are fulfilled by the tool PRO-TEST (probabilistic testability analysis) as described in [22], [23]. Furthermore, this tool has the advantage that the user can control the tradeoff between the precision of the estimation and the required computing time. All the results reported in this paper are obtained with PRO-TEST.

III. OPTIMIZING INPUT PROBABILITIES

Now we can formulate the optimizing problem in a more formal way.

3.1. Problem A

Let G be the desired probability to detect all faults. Find a number k, k distributions X^i, and k numbers N_i, $i = 1, \cdots, k$, such that

$$G \leq \prod_{f \in F} \left(1 - \prod_{i=1}^{k} \left(1 - p_f(X^i) \right)^{N_i} \right)$$
$$\text{and} \quad N := \sum_{i=1}^{k} N_i \text{ is minimal.}$$

The problem is solved if we set k equal to the minimal number of deterministic test patterns, that is the size of the smallest possible test set. Then each $X^i \in [0, 1]^n$ represents a test pattern, we have $N_i = 1$ for each pattern, and $N = k$. However, the problem to find a minimal test set has been proven to be *NP*-complete [3], hence, there is little hope to develop an efficient CAD tool based on a solution for the *NP*-hard Problem A. Even the weaker Problem B turns out to be *NP*-hard.

3.2. Problem B

Let G and k be given. Find k distributions X^i and numbers N_i, $i = 1, \cdots, k$, such that

$$G \leq \prod_{f \in F} \left(1 - \prod_{i=1}^{k} \left(1 - p_f(X^i) \right)^{N_i} \right),$$
$$\text{and} \quad N := \sum_{i=1}^{k} N_i \text{ is minimal.}$$

It is easy to prove that Problem A can be reduced to Problem B, so an efficient algorithm for B cannot be expected either. Therefore, our goal is not an optimal solution of problem A or B, but we satisfy ourselves with finding an efficient procedure. Fig. 2 indicates that the problem arises because different faults of the circuits require different distributions. Hence, we formulate our problem as follows.

Optimizing Problem: Let G and k be given. We are searching for a partition $\langle F_1, \cdots, F_k \rangle$ of $F := F_1 \cup \cdots \cup F_k$, distributions $X^1, \cdots, X^k$ and numbers $N_1, \cdots,$ N_k, such that

$$G \leq \prod_{i=1}^{k} \prod_{f \in F_i} \left(1 - \left(1 - p_f(X^i) \right)^{N_i} \right), \quad \text{and} \quad N := \sum_{i=1}^{k} N_i \tag{2}$$

is sufficiently small.

For $k := 1$ this problem has already been solved in [23], [24], and we now list some basic results presented there. For the input probabilities $X := \langle x_1, \cdots, x_n \rangle \in [0, 1]^n$ we have for all faults f:

$$p_f(X) = p_f(x_1, \cdots, x_{i-1}, 0, x_{i+1}, \cdots, x_n)$$
$$+ x_i \big(p_f(x_1, \cdots, x_{i-1}, 1, x_{i+1}, \cdots, x_n)$$
$$- p_f(x_1, \cdots, x_{i-1}, 0, x_{i+1}, \cdots, x_n) \big). \tag{3}$$

This is a straightforward consequence of Shannon's formula. Now we can compute a fault detection probability and its partial derivative for an arbitrary value of x_i, if we know the values under the conditions that input i is constant "0" and constant "1."

$$\frac{dp_f(X)}{dx_i} = p_f(x_1, \cdots, x_{i-1}, 1, x_{i+1}, \cdots, x_n)$$
$$- p_f(x_1, \ldots, x_{i-1}, 0, x_{i+1}, \cdots, x_n). \tag{4}$$

By some straightforward approximations using formula (12) in the Appendix, formula (2) leads to

$$\ln(G) \approx \sum_{f \in F} - \left(1 - p_f(X) \right)^N \approx - \sum_{f \in F} e^{-p_f(X)N}. \tag{5}$$

We call a tuple $X \in [0, 1]^n$ *optimal*, if the objective function

$$\delta_N^F(X) := \sum_{f \in F} e^{-p_f(X)N} \tag{6}$$

is *minimal*. Obviously this corresponds to the fact that the probability of detecting all faults by N patterns is maximal. Minimizing the objective function would need exponential effort in general. However, a sufficient heuristic is found, since the first partial derivative of the objective function can be computed explicitly:

$$\frac{d\delta_N^F(X)}{dx_i} = - \sum_{f \in F} N \big(p_f(x_1, \cdots, x_{i-1}, 1, x_{i+1}, \cdots, x_n)$$
$$- p_f(x_1, \cdots, x_{i-1}, 0, x_{i+1}, \cdots, x_n) \big)$$
$$\cdot e^{-p_f(X)N}. \tag{7}$$

The next step shows that the second derivative is positive everywhere:

$$\frac{d^2\delta_N^F(X)}{dx_i^2} = \sum_{f\in F} N^2\big(p_f(x_1, \cdots, x_{i-1}, 1, x_{i+1}, x_n)$$

$$- p_f(x_1, \cdots, x_{i-1}, 0, x_{i+1}, \cdots, x_n)\big)^2$$

$$\cdot e^{-p_f(X)N}. \tag{8}$$

Thus the objective function is strictly convex with respect to a single variable, and these explicit formulas can be used to find the optimal value for x_i by the bisection method, the regula falsi or the Newton iteration [7]. The complete optimizing procedure is the following.

> **Procedure** Optimize
> (F: Faultsets, X: Startvector)
> Old $:= 2\delta_N(X)$
> New $:= \delta_N(X)$
> **While** Old $> (1+\lambda)$New$+\epsilon$ **do**
>   Old $:=$ New
>   **For** $i := 1$ to n **do**
>   Search optimal value y using (7), (8) and
>   a standard technique for input i
> $x_i := y$
>   New $:= \delta_N(X)$.

The parameters λ and ϵ are specified by the user, and they determine another tradeoff between accuracy and computing time. In the next section we discuss the extension to multiple distributions.

IV. PARTITIONING A FAULT SET

In order to gain efficiency, the optimizing problem is solved by splitting the fault set into two subsets iteratively. In this section, it is discussed how to find two tuples $V_1, V_2 \in [0, 1]^n$ and a partition $F_1 \cup F_2 = F$, such that the sum of the two corresponding objective functions is minimized:

$$\delta_N^{F_1}(V_1) + \delta_N^{F_2}(V_2) := \sum_{f\in F_1} e^{-p_f(V_1)N} + \sum_{f\in F_2} e^{-p_f(V_2)N}$$

$$< \delta_n^F(X).$$

For each $F^* \subset F$ the objective function $\delta_N^{F^*}$ may be multimodal and its global minimization would need exponential effort. Hence, we do not try to compute a global minimum, but we look for a direction, where starting from a tuple X_0 the decrease of $\delta_N^{F^*}$ is maximal. The next theorem will give a helpful hint.

Theorem 3: Let $U \subset R^n$ be convex, $\zeta: U \to R$, and grad $(\zeta) := (d\zeta/x_i)_{1\le i\le n}$ be the gradient of ζ. For each $X_0 \in U$ the vector $-$grad $(\zeta)(X_0)$ indicates the direction of strongest decrease. If ζ is linear a local minimum is found on the line $X_0 - \alpha$ grad $(\zeta)(X_0)$, $\alpha \ge 0$.

Proof: Mathematical calculus.

Even though $\delta_N^{F^*}$ is not a linear function, Theorem 3 claims that $-$grad $(\delta_N^{F^*})(X_0)$ is the required direction. Thus we define the new function

$$\xi_N^{F^*}: R^+ \cup \{0\} \to R$$

by

$$\xi_N^{F^*}(\alpha) := \delta_N^{F^*}\big(X_0 - \alpha \text{ grad }(\delta_N^{F^*})(X_0)\big).$$

The formula

$$D(F^*, N, X_0, 0) := \frac{-d\xi_N^{F^*}(0)}{d\alpha} \tag{9}$$

exactly measures the decrease of our objective function in its optimal direction. The solution of

$$D(F^*, N, X_0, \gamma) = 0 \tag{10}$$

provides input probabilities $X_0 - \gamma$ grad $(\delta_N^{F^*})(X_0)$, defining a minimum point in this direction. Therefore, our weakened partitioning problem can be solved by F_1 and F_2 such that

$$\sqrt{D(F_1, N, X_0, 0)} + \sqrt{D(F_2, N, X_0, 0)} > 0 \tag{11}$$

is maximal, the square root is used for normalizing. It should be noted that for linear functions this process would indeed be optimal.

The rest of this section discusses the tasks necessary for partitioning. These tasks have to be done only for the small subset of faults with lowest detection probability due to Observation 5. If this set is small enough, the presented method will compute a global optimal solution maximizing formula (11). For large fault sets computing time can be saved, if the method is somewhat simplified.

a) Computing the Gradient: The gradient for $X_0 - \gamma$ grad $(\delta_N^{F^*})(X_0)$ can be computed explicitly using (7). If additionally (3) is used, it is immediately seen that we only have to compute $p_f(X)$ and either $p_f(x_1, \cdots, x_{i-1}, 0, x_{i+1}, \cdots, x_n)$ or $p_f(x_1, \cdots, x_{i-1}, 1, x_{i+1}, \cdots, x_n)$ for this purpose.

b) Sorting the Fault Set: For each fault let

$$d_f(X_0) := \sqrt{\sum_{i=1}^{n} \left(\frac{de^{-p_f(X)N}}{dx_i}\right)^2 (X_0)}$$

$$= \| \text{grad }(e^{-p_f(X)N})(X_0)\|$$

be the Euclidean norm of the gradient of $e^{-p_f(X)N}$ in X_0, and let $\langle f_i \rangle_{i\le k}$ be an enumeration of F with

$$i \le k \Rightarrow d_{f_i}(X_0) \ge d_{f_i}(X_0).$$

Now we select a constant value c and the most important subset of faults $F \subset F$ where $F := [f_i | i \le c]$. The results presented in the next section are provided for $c = 20$. As usual, if the number of faults with low detectability is small enough, then a global optimum can be achieved.
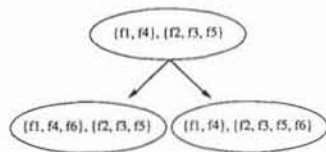
Fig. 3. Node at level 5 with its both successors.

c) Starting Partitioning: Firstly, we are looking for a starting partitioning $F_a, F_b \subset F$:

> Set $F_a, F_b := \varnothing$.
> For $i := 1$ to c do
> if $\sqrt{D(F_a \cup \{f_i\}, N, X_0, 0)} + \sqrt{D(F_b, N, X_0, 0)} >$
> $\sqrt{D(F_a, N, X_0, 0)} + \sqrt{D(F_b \cup \{f_i\}, N, X_0, 0)}$
> then $F_a := F_a \cup \{f_i\}$.
> else $F_b := F_b \cup \{f_i\}$.

This starting partitioning corresponds to an objective value:

$$\nu := \sqrt{D(F_a, N, X_0, 0)} + \sqrt{D(F_b, N, X_0, 0)}.$$

d) Constructing a Search Tree: Now a search tree T can be constructed, where each node represents two disjoint subsets of F. Node A is a direct successor of node B, if one of the subsets of A is equal to one of B, and if the other subset contains exactly one more fault (see Fig. 3). Thus a node of depth k represents a partition of the first k faults into two subsets.

Due to the triangle inequality, at a node A at depth $m \leq c$ with fault sets F_v, F_w the search can be stopped if

$$\nu \geq \sqrt{D(F_v, N, X_0, 0)} + \sqrt{D(F_w, N, X_0, 0)}$$
$$+ (m - c) d_{fm}(X_0)$$

since no leaf succeeding node A will be better than the starting partitioning. If we reach a leaf this way, then a better solution F_a, F_b is found and ν must be updated. The complexity of this process is distinctly lower than 2^c, since most of the branches are aborted at a very early stage of the search.

e) The Complete Partitioning: The remaining faults of F (if some exist) are now added to the sets F_a and F_b in the same way as described in Section IV-c). Finally, for each fault it is checked whether or not the result can be improved by changing its membership. This greedy algorithm is stopped if

$$\nu := \sqrt{D(F_a, N, X_0, 0)} + \sqrt{D(F_b, N, X_0, 0)}$$

is maximized.

f) Computing a Tuple of Optimal Input Probabilities: If the gradient for δ_N^F is already computed, (10) is solved by a bisection method. This provides a γ_a with $D(F_a, N, X_0, \gamma_a) = 0$ and a γ_b with $D(F_b, N, X_0, \gamma_b) =$

0. We set

$$F_1 := F_a, V_1 := X_0 - \gamma_a \text{ grad } (\delta_N^{F_1})(X_0)$$

and

$$F_2 := F_b, V_2 := X_0 - \gamma_b \text{ grad } (\delta_N^{F_2})(X_0).$$

Finally, we improve V_1 and V_2 by the procedure Optimize of Section III. If the gradient is unknown, Optimize is started immediately.

V. MULTIPLE OPTIMAL DISTRIBUTIONS

Of course partitioning is not restricted to two sets. Experience has shown that better results are obtained by a successive procedure as compared to partitioning into m sets at one time.

> **Procedure** Multiple_Optimize
>   (F: Faultsets, X: Startvector,
> m: Number of distributions).
> $F[1] := F$
> $X[1] := X$
> **For** $i :=$ **to** $m - 1$ **do**
> Let $j \leq m - 1$ be such that $F[j]$ requires the largest test set.
> Partition $F[j]$ into F_a, F_b.
> Optimize $(F_a, X[j], X_a)$ and Optimize $(F_b, X[j], X_b)$ as mentioned in Section IV-e).
> $F[j] := F_a, X[j] := X_a, F[i] := F_b$
> $X[i] := X_b$.

This procedure provides m distributions X_i, and m disjoint fault sets F_i. Computing the m test lengths N_i by (2) would overestimate the necessary size, since there is a chance that a pattern according to X_i detects some faults of a set F_j also, $j \neq i$. The optimal numbers N_i can be computed by the formula of Problem B) in Section III.

The test application is simplified, if we assume that we use the same amount N^* of patterns for each distribution. In this case the mentioned formula is simplified into

$$G \leq \prod_{f \in F} \left(1 - \prod_{i=1}^{k} (1 - p_f(X^i))^{N^*} \right)$$

and $N := m^* N^*$.

VI. RESULTS

In Table I optimizing results are shown based on detection possibilities provided by PROTEST. The first example is the ANDOR32-circuit of Fig. 2. For the well-known benchmark circuits [8], $k := 1, \cdots, 8$ optimized input probabilities have been computed. The first column denotes the circuit name, the second one the necessary number of not optimized, equiprobable random patterns, and the following 8 columns contain the necessary number of random patterns assuming that for each distribution

TABLE I
DISTRIBUTIONS AND TEST SIZES

Circuit	equiprobable	Necessary number of random patterns weights:							
		1	2	3	4	5	6	7	8
AndOr	3.8e10	3.8e10	**1.6e3**	1.8e3	2.1e3	2.2e3	2.4e3	-	-
c17	8.5e1	5.8e1	8.0e1	6.0e1	5.3e1	**5.0e1**	5.7e1	6.6e1	7.5e1
c432	2.4e3	1.1e3	9.9e2	9.8e2	**9.3e2**	9.3e2	9.4e2	9.9e2	1.1e3
c499	1.4e3	1.4e3	1.3e3	1.4e3	1.3e3	1.3e3	**1.3e3**	1.3e3	1.3e3
c880	3.7e4	**1.0e3**	2.1e3	1.2e3	1.1e3	1.2e3	1.1e3	1.2e3	1.4e3
c1355	2.7e6	2.7e6	3.1e6	4.0e6	4.1e6	4.8e6	**8.3e5**	5.3e6	5.2e6
c1908	6.0e4	1.9e4	2.9e4	**1.5e4**	1.9e4	1.8e4	2.9e4	3.5e4	3.8e4
c2670	1.1e7	1.1e5	7.3e4	7.4e4	6.4e4	5.6e4	6.5e4	4.8e4	**4.5e4**
c3540	1.1e6	6.4e5	1.3e6	1.9e6	1.7e6	**2.4e5**	2.9e5	3.3e5	3.5e5
c5315	5.3e4	1.0e4	**8.1e4**	9.4e3	9.3e3	9.9e3	1.1e4	1.3e4	1.2e4
c6288	8.2e2	3.0e2	**2.6e2**	2.7e2	3.0e2	4.1e2	2.9e2	2.7e2	2.7e2
c7552	4.8e11	1.2e6	7.1e5	6.4e5	4.5e5	2.6e5	2.3e5	2.3e5	**2.2e5**

TABLE II
FAULT COVERAGE PROVIDED BY SIMULATION OF WEIGHTED PATTERNS

Circuit	Number of simulated patterns	Number of redundant faults by [17]	Number of not detected faults	Fault coverage (%)	Fault coverage with respect to all detectable faults (%)
c17	10	0	0	100.00	100.00
c432	512	4	4	99.32	100.00
c499	539	8	8	98.99	100.00
c880	260	0	0	100.00	100.00
c1355	2244	8	8	99.53	100.00
c1908	2308	9	9	99.60	100.00
c2670	10766	117	117	95.48	100.00
c3540	12200	137	137	96.30	100.00
c5315	1316	59	59	99.07	100.00
c6288	109	34	34	99.59	100.00
c7552	34827	131	139	98.13	99.86
*c7552	57320	131	131	98.26	100.00

the same amount N^* of patterns is applied. For all examples a detection confidence of $G = 0.999$ was required for the probability of detecting all faults. For the set of distributions which results in a minimal overall size, the number of test patterns is printed in bold letters.

For the small circuit C17 some distributions degenerate to deterministic test patterns. This circuit with 5 primary inputs is not appropriate for random pattern testing, and the requirements of Theorem 1 are not fulfilled. The ANDOR violates Theorem 1 in the equiprobable case but not for weighted patterns. Three points are outstanding.

Firstly, all of the benchmark circuits and the counter-example ANDOR32 can be tested by only a few thousand random patterns. From a theoretical point of view, all circuits can be made random testable by the presented procedure.

Secondly, the overall number of necessary patterns does not decrease monotonically with the number of distributions. This is a practical consequence of the discussed problem complexity and the applied heuristics during optimizing and partitioning.

Thirdly, the results differ slightly from the results reported, e.g., in [23], [24], since they depend on parameters of the testability measure to a large extent. The parameters were chosen such that partitioning could be done within less than 1 h of computing time for each circuit. The program is running under the operation system UNIX, and the experiments were performed on a SUN 3/50 workstation.

However, the results of Table I are only estimations, they denote that at most one out of thousand test sets of these lengths will fail to obtain a complete coverage of all irredundant faults. These predictions on fault coverage are validated by fault simulation as shown in Table II.

During fault simulation it is useful to merge the m differently weighted pattern sets, since the fault dropping technique cannot be exploited otherwise. In the first column of Table II, the circuit name is found. The second column contains the numbers of merged weighted patterns which have been applied. For each distribution the same amount of patterns have been generated; the number of the pattern that detected the last fault is always given.

The third column of Table II contains the number of redundant faults found by the deterministic test pattern generator of Schulz and Auth [17], which identifies all redundant faults in the benchmark circuits. The fourth column contains the number of faults not detected by simulation, the fifth column the fault coverage with respect to all faults, and the sixth the fault coverage with respect to all detectable faults. In each case, the optimal number of distributions which is found in Table III has been applied.

It should be noted that the amount of simulated patterns was much smaller than that one required by PROTEST, due to restrictions of the available fault simulator. Presumably this is the reason for the 0.1% undetected faults of the circuit c7552, and thus the weighted patterns will be another application field of the recently proposed fast fault simulators for combinational circuits (e.g., [16], [21]). In the example *c7552, a higher precision of PROTEST is used in order to compute 17 different sets of weights. Then the computing time for optimizing increases from approximately 1 to 7.5 h, but fault simulation shows a higher fault coverage. In both cases c7552 and *c7552, fault simulation was aborted if 20 000 patterns did not detect any new fault.

Table III compares the results of fault simulation using

TABLE III
FAULTS NOT DETECTED USING WEIGHTED AND EQUIPROBABLE PATTERNS
(REDUNDANCIES)

Circuit	Number of distributions	Number of patterns	Number of faults not detected	
			weighted	equiprobable
C17	5	10	0	1
C432	4	512	0	6
C499	6	539	0	6
C880	1	260	0	51
C1355	6	2244	0	0
C1908	3	2308	0	21
C2670	8	10766	0	515
C3540	5	12220	0	2
C5315	2	1316	0	8
C6288	2	109	0	8
C7552	8	34827	8	268

Fig. 4. LSSD-based random test.

equiprobable patterns and weighted patterns. The first column contains the applied number of distributions. These are the numbers where the presented partitioning and optimizing procedures predict the shortest test length. In general, since no global statement about the optimal number of distributions can be obtained sufficiently, it is recommended to stop the partitioning at that point, where the test length is short enough for practical applications. The second column of Table III is the number of weighted patterns simulated. The third column is the number of testable faults not detected during simulations, i.e., redundancies are removed.

The same amount of equiprobable patterns was simulated, and the last column contains the number of testable faults not detected by equiprobable patterns. For the equiprobable case, results of longer simulation runs are found in [29], where 2^{21} random patterns did not lead to a complete fault coverage for some circuits.

VII. APPLICATIONS

The mentioned tools for estimating fault detection probabilities are mainly used to predict the necessary test length of a random test, which can be carried out by a BIST structure like a BILBO [14]. Since a large class of circuits is resistant to such a conventional random test, optimized input probabilities should be computed. A test strategy based on weighted random patterns can also be implemented as self-test using a so called GURT (generator of unequiprobable random tests) [25]. However, even this way, not all circuits can be dealt with adequately.

The presented method of computing multiple distributions is applicable to all combinational circuits, but unfortunately there is no obvious way to implement them by
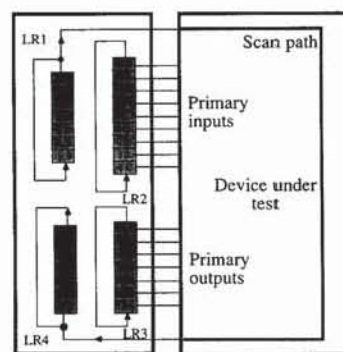
a BIST technique. However, of course, they can be used for a so called LSSD or scan-path random test ([11], [4]). Fig. 4 shows the basic architecture of a scan design in combination with random pattern generation using linear feedback shift registers.

The pattern generator and the signature registers are built on an external chip which sequentially generates random patterns with multiple distributions. Such a test chip has been designed and processed as a gate array [6]. Currently, a circuit which can generate weighted patterns corresponding to multiple distributions has been designed and processed.

Using these chips leads to a weighted random pattern test system at low costs, where a similar or even a better fault coverage is reachable as compared to a conventional deterministic test (see [28]). In addition to the low priced test equipment the test application time will also decrease to the high-speed pattern generation.

VIII. CONCLUSIONS

An efficient method has been presented to compute multiple distributions for random patterns, which can be applied successively. Using multiple distributions, all combinational circuits can be made random testable, and a high fault coverage which is competitive with that reached by most of the deterministic test pattern generators is provided by a few thousands of random patterns.

The differently weighted random test sets can be applied to scan path circuits using an external chip, combining the advantages of a low cost test and high fault coverage. Since the size of the random test set is rather small, the fault coverage can be validated by simulation.

Furthermore several facts about testing using random patterns have been proven. It has been shown, that the number of random patterns required for a certain fault coverage can be computed without regard to the *pseudo-random* property and with the *independence* assumption for fault detection.

APPENDIX A
PROOF OF THEOREM 1

Clearly we have $N_p \leq N_r$. The power series of the logarithm provides

$$\frac{p}{1-p} = \sum_{i=1}^{\infty} p^i \geq \sum_{i=1}^{\infty} \frac{p^i}{i} = -\ln(1-p) \geq p,$$

$$\text{for } |p| < 1. \qquad (12)$$

Fault detection by random patterns follows the binomial distribution, thus we have

$$\epsilon = (1-p)^{N_r} \text{ or}$$

$$-\ln(\epsilon) = -N_r \ln(1-p) \geq pN_r. \qquad (13)$$

Fault detection by pseudorandom patterns follows the hypergeometric distribution, that is

$$\epsilon = \frac{\binom{2^i - 2^i p}{N_p}}{\binom{2^i}{N_p}} = \frac{(2^i(1-p))!}{(2^i(1-p) - N_p)! \, 2^i!}$$

$$= \prod_{k=0}^{N_p-1} \frac{2^i(1-p) - k}{2^i - k}$$

$$= \prod_{k=0}^{N_p-1} \left(1 - \frac{2^i}{2^i - k} p\right).$$

Since $k \leq 2^{i/2}$ and $p < 1/4$, we have $(2^i/2^i k) p < 1$, and by using the left inequation of (12) we get

$$-\ln(\epsilon) = -\sum_{k=0}^{N_p-1} \ln\left(1 - \frac{2^i}{2^i - k} p\right)$$

$$\leq \sum_{k=0}^{N_p-1} \frac{\frac{2^i}{2^i - k} p}{1 - \frac{2^i}{2^i - k} p}$$

$$= \sum_{k=0}^{N_p-1} \frac{2^i}{2^i(1-p) - k} p$$

$$\leq pN_p \left(\frac{2^i}{2^i(1-p) - N_p}\right)$$

$$= p\left(N_p + \frac{N_p^2 + 2^i pN_p}{2^i(1-p) - N_p}\right).$$

Since $N_p^2 \leq 2^i$ and $pN_p \leq pN_r \leq -\ln(\epsilon)$, this yields

$$-\ln(\epsilon) \leq p\left(N_p + \frac{2^i(1 + pN_p)}{2^i(1-p) - N_p}\right)$$

$$\leq p\left(N_p + \frac{1 - \ln(\epsilon)}{1 - p - 2^{-i/2}}\right)$$

$$\leq p\left(N_p + 2(1 - \ln(\epsilon))\right).$$

Hence,

$$N_p \leq N_r \leq \frac{-\ln(\epsilon)}{p} \leq N_p + 2(1 - \ln(\epsilon))$$

and the theorem is proven.

APPENDIX B
PROOF OF THEOREM 2

We set $\delta_{n+1} := P(\{f_i \mid i \leq n+1\}, N) - \prod_{i=1}^{n+1} (1 - (1 - p_{f_i})^N)$.

Now we have $P(F, N) = J_N + \delta_\mu$, and using the Bayesian formula we can estimate

$$\delta_{n+1} = P(\{f_i \mid i \leq n\}, N) - \prod_{i=1}^{n+1} (1 - (1 - p_{f_i})^N)$$

$$- (1 - p_{f_{n+1}})^N P(\{f_i \mid i \leq n\}$$

$$N \mid \text{no pattern detects } f_{n+1})$$

$$= P(\{f_i \mid i \leq n\}, N) - (1 - (1 - p_{f_{n+1}})^N)$$

$$\cdot \prod_{i=1}^{n} (1 - (1 - p_{f_i})^N)$$

$$- (1 - p_{f_{n+1}})^N P(\{f_i \mid i \leq n\}$$

$$N \mid \text{no pattern detects } f_{n+1})$$

$$= \delta_n + (1 - p_{f_{n+1}})^N \left(\prod_{i=1}^{n} (1 - (1 - p_{f_i})^N)\right.$$

$$\left. - P(\{f_i \mid i \leq n\}, N \mid \text{no pattern detects } f_{n+1})\right).$$

Thus $\delta_{n+1} \leq \delta_n + (1 - p_{f_{n+1}})^N \prod_{i=1}^{n} (1 - (1 - p_{f_i})^N)$, and since $\delta_1 = 0$:

$$\delta_{n+1} \leq \sum_{j=2}^{n+1} (1 - p_{f_j})^N \prod_{k=1}^{j-1} (1 - (1 - p_{f_k})^N).$$

On the other hand

$$P(\{f_i \mid i \leq n\}, N \mid \text{no pattern detects } f_{n+1}) \leq 1$$

and we have

$$\delta_{n+1} \geq \delta_n + (1 - p_{f_{n+1}})^N \left(\prod_{i=1}^{n} (1 - (1 - p_{f_i})^N) - 1\right)$$

$$\geq \delta_n - (1 - p_{f_{n+1}})^N (1 - J_N)$$

$$\geq -(1 - J_N) \sum_{j=2}^{n+1} (1 - p_{f_j})^N.$$

REFERENCES

[1] E. J. Aas and M. R. Mercer, "Algebraic and structural computation of signal probability and fault detectability in combinational circuits," in *FTCS-17, Dig. Papers*, 1987, pp. 72–77.

[2] S. K. Jain, V. D. Agrawal, "STAFAN: An alternative to fault simulation," in *Proc. 21st Design Automation Conf.*, 1984, pp. 18–23.

[3] B. Krishnamurthy and S. B. Akers, "On the complexity of estimating the size of a test set," *IEEE Trans. Comput.*, vol. C-33, pp. 750–753, 1984.

[4] P. H. Bardell and W. H. McAnney, "Parallel pseudorandom sequences for built-in test," in *Proc. 1984 Int. Test Conf.*, 1984, pp. 302–308.

[5] J. Savir and P. H. Bardell, "On random pattern test length," in *Proc. 1983 Int. Test Conf.*, 1983, pp. 95–106.

[6] T. Bergsträsser, "Entwurf eines modifizierten Zufallsmustergenerators," Studienarbeit an der Fakultät für Informatik, Universität Karlsruhe, 1985.

[7] E. Blum and W. Oettli, *Mathematische Optimierung*. Berlin, Germany: Springer Verlag, 1975.

[8] F. Brglez *et al.*, "Accelerated ATPG and fault grading via testability analysis," in *Proc. IEEE Int. Symp. on Circuits and Systems*, June 1985.

[9] J. Savir, G. Ditlow, and P. H. Bardell, "Random pattern testability," in *FTCS-13, Dig. Papers*, 1983, pp. 80–89.

[10] S. Chakravarty and H. B. Hunt III, "On the computation of detection probability for multiple faults," in *Proc. 1986 Int. Test Conf.*, 1986, pp. 252–262.

[11] E. B. Eichelberger and E. Lindbloom, "Random-pattern coverage enhancement and diagnosis for LSSD logic self-test," *IBM J. Res. Develop.*, vol. 27, no. 3, pp. 265–272, May 1983.

[12] M. R. Garey and D. S. Johnson, *Computers and Intractability—A Guide to NP-Completeness*. San Francisco, CA: Freeman, 1979.

[13] O. H. Ibarra and S. K. Sahni, "Polynomially complete fault detection problems," *IEEE Trans. Comput.*, vol. C-24, 1975.

[14] B. Koenemann *et al.*, "Built-in logic block observation techniques," in *Proc. Test Conf.*, 1979, pp. 37–41.

[15] R. Lisanke *et al.*, "Testability-driven random pattern generation," *IEEE Trans. Computer-Aided Design*, vol. 6, pp. 1082–1087, 1987.

[16] F. Maamari and J. Rajski, "A reconvergent fanout analysis for efficient exact fault simulation of combinational circuits," in *Proc. FTCS-18*, Tokyo, 1988, pp. 122–127.

[17] M. H. Schulz and E. Auth, "Advanced automatic test pattern generation and redundancy identification techniques," in *Proc. FTCS-18*, Tokyo, 1988.

[18] J. J. Shedletsky, "Random testing: Practically versus verified effectiveness" in *Proc. FTCS-7*, 1977, pp. 175–179.

[19] S. C. Seth *et al.*, "An exact analysis for efficient computation of random-pattern testability in combinational circuits," in *Proc. FTCS-16*, 1986, pp. 318–323.

[20] K. D. Wagner *et al.*, "Pseudorandom testing," *IEEE Trans. Comput.*, vol. C-36, 1987.

[21] J. A. Waicukauski *et al.*, "Fault simulation for structured VLSI," *VLSI Syst. Design*, Dec. 1985.

[22] H.-J. Wunderlich, "PROTEST: A tool for probabilistic testability analysis," in *Proc. 22nd Design Automation Conf.*, 1985.

[23] ——, *Probabilistische Verfahren für den Test hochintegrierter Schaltungen*, Informatik-Fachberichte 140. Berlin, Germany: Springer-Verlag, 1987.

[24] ——, "On computing organized input probabilities for random tests," in *Proc. 24th Design Automation Conf.*, 1987.

[25] ——, "Self test using unequiprobable random patterns," in *Int. Symp. on Fault-Tolerant Computing*, FTCS-17, 1987, pp. 258–263.

[26] ——, "The random pattern testability of programmable logic arrays," in *Proc. IEEE Int. Conf. on Computer Design, ICCD'87*, 1987, pp. 682–685.

[27] H.-J. Wunderlich and W. Rosenstiel, "On fault modeling for dynamic MOS circuits," in *Proc. 23rd Design Automation Conf.*, 1986, pp. 540–546.

[28] J. A. Waicukauski *et al.*, "WRP: A method for generating weighted random patterns," presented at IEEE Design for Testability Workshop, Vail, CO, 1988.

[29] J. L. Carter *et al.*, "ATPG via random pattern simulation," in *Proc. IEEE Int. Symp. on Circuits and Systems*, June 1985.

*

Hans-Joachim Wunderlich (M'86) received the diploma degree in mathematics from the University of Freiburg, F.R. Germany, in 1981, and the Dr. rer. nat. degree from the University of Karlsruhe in 1986.

In 1982, he was a consultant at the Fraunhofer-Institute of Industrial Engineering, Stuttgart, where he worked in the field of operations research. In 1983, he joined the Institute of Computer Design and Fault Tolerance, University of Karlruhe, where he has been the head of a research group on automation of circuit design and testing since 1986. His research interests include computer-aided design for testability, test generation, and digital simulation.