New and Improved BIST Diagnosis Methods From Combinatorial Group Testing Theory

Andrew B. Kahng, *Member, IEEE*, and Sherief Reda, *Student Member, IEEE*

***Abstract*—We examine the general problem of built-in-self-test (BIST) diagnosis in digital logic systems. The BIST diagnosis problem has applications that include identification of erroneous test vectors, faulty scan cells, and faulty items. We develop an abstract model of this problem and show a fundamental correspondence to the well-established subject of combinatorial group testing (CGT) (D. Du and F. K. Hwang, *Combinatorial Group Testing and Its Applications*, 1994). We exploit this new perspective to 1) link existing BIST diagnosis techniques to CGT techniques and provide further insights into existing diagnosis algorithms, 2) improve the performance of diagnosis algorithms, and 3) develop new techniques to address the BIST diagnosis problem. Using the ISCAS'89 benchmarks, we empirically demonstrate the effectiveness of our proposed techniques over existing BIST diagnosis techniques. The vastness of the CGT literature suggests that further improvements from existing research in CGT may be obtained.**

***Index Terms*—Fault detection, testing, VLSI BIST diagnosis.**

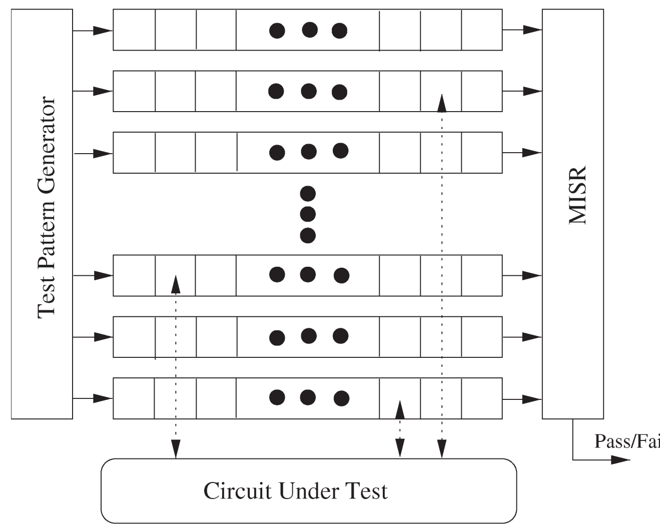

Fig. 1. Scan-based BIST environment.

I. Introduction and Background

ADVANCES in integrated circuit (IC) fabrication technology have enabled integration and electronic devices to reach unprecedented new realms. These advances have also exacerbated the testing problem, since many thousands of test vectors are needed to test newer complex designs. Huge numbers of test vectors can lead to excessive testing time if supplied directly through automatic test equipment (ATE). As a result, built-in-self-test (BIST) has emerged as one of the leading solutions for addressing challenging test problems. BIST offers the promise of low hardware overhead with the clear advantage of at-speed testing.

BIST structures are incorporated into ASIC chips to test their structures. If a chip tests faulty, a diagnosis procedure must be undertaken to identify the source of the fault. In a typical BIST environment, a test pattern generator [e.g., a counter or linear feedback shift register (LFSR)] injects a number of patterns into the scan chain(s). These patterns are applied to the circuit under test, and the circuit's response is captured back into the scan chain(s). The circuit response is then compacted by feeding the output of the scan chain(s) into an LFSR or a multiple-input shift register (MISR) as shown in Fig. 1. The compacted response stored in the LFSR or MISR is called the test signature. The process of test set application, response capture and compaction is usually referred to as a test session. A fault in the circuit manifests itself by changing the response at a number of scan cells and, hence, in the test signature. Scan cell diagnosis is concerned with identifying the set of scan cells that received faulty (or erroneous) response from within the total set of cells [21]. These cells are typically called faulty despite the fact that they are not themselves faulty but rather capture faulty responses. Once these faulty (or erroneous) scan cells are identified, structural analysis and fault simulation are carried out to determine the possible sites of faults and failing test vectors in the circuit under test [11], [17], [19], [22], [24]. Scan cell diagnosis is highly challenging since the signature offered by the LFSR or MISR gives little help toward diagnosing the failure. The MISR output is typically interpreted as pass/fail information with little additional value. Some of the challenges in the BIST diagnosis process are as follows: 1) achieving full diagnostic information, i.e., detecting all scan cells that capture erroneous responses; 2) minimizing diagnosis time, since this translates to reduction in total testing time; and 3) minimizing hardware overhead, i.e., the amount of hardware needed to support BIST diagnosis.

Manuscript received December 27, 2003; revised October 13, 2004 and January 24, 2005. Earlier results of this work have been reported at the Asia and South Pacific Design Automation Conference 2004. New results in this manuscript include extended discussions of various algorithms and a new experiment on the ISCAS'89 benchmarks. This paper was recommended by Associate Editor S. M. Reddy.

A. B. Kahng is with the Department of Computer Science and Engineering and the Department of Electrical and Computer Engineering, University of California San Diego, La Jolla, CA 92093-0114 USA (e-mail: abk@ucsd.edu).

S. Reda is with the Department of Computer Science and Engineering, University of California San Diego, La Jolla, CA 92093-0114 USA (e-mail: sreda@cs.ucsd.edu).

Digital Object Identifier 10.1109/TCAD.2005.854635

Abstractly, the BIST diagnosis process involves a set of items, e.g., scan cells, with some number of these items being faulty. The BIST diagnosis problem seeks to identify all faulty items using the minimum number of tests, i.e., with minimum use of the response compactor (e.g., the MISR in the scan-based BIST). The diagnosis problem can be formally defined as follows.

A. Diagnosis Problem

Given a set $M = \{m_1, m_2, \ldots, m_n\}$ consisting of n items, identify the subset of all faulty items $F \subseteq M$, where $d = |F|$ is unknown, using the minimum number of queries to the tester.

We are now ready to introduce the notion of combinatorial group testing (CGT).

B. Combinatorial Group Testing

CGT is a generic class of algorithms that are applied whenever a large number of individuals or items are subjected to the same test [8]. Instead of testing individual items, CGT groups items and then tests the groups. A group tests positive or faulty when at least one item within the group tests positive; otherwise, it is fault free. A CGT experiment requires defining the groups, and a corresponding diagnosis or decoding procedure to infer the status of the items from the status of the groups. Nonadaptive group testing selects the groups *a priori*, i.e., before the diagnosis procedure, while adaptive group testing uses results from previous tests to guide selection of groups for subsequent tests.

CGT provides a model for the BIST diagnosis problem. While several early results in the field of group testing were obtained at Bell Labs and address the problem of electrical shorts in a set of signal nets in printed circuit boards [6], [10], [23], we are not aware of any other established links between CGT and diagnosis in BIST.

A taxonomy that aligns with the CGT literature classifies BIST diagnosis algorithms into two types: 1) nonadaptive (off-line); and 2) adaptive (on-line). In nonadaptive diagnosis, scan cells are partitioned into a number of partitions before carrying out the diagnosis process. During the diagnosis process, a fixed set of test patterns is repetitively applied to each partition. The test patterns' response, captured in the scan cells, is compacted by an LFSR or MISR and compared to the respective fault-free signature. A number of techniques have been suggested in the fault diagnosis literature to specify the partitions. These partitioning techniques are either pseudorandom [20], [21] or deterministic [3], [25].

In adaptive diagnosis, the partitioning is determined as the diagnosis algorithm unfolds, i.e., future partitions are dynamically determined depending upon results of previously tested partitions [2], [11], [12]. In [11], Ghosh-Dastidar *et al.* suggest partitioning the scan chain into a number of partitions and testing each partition. If a partition tests positive, then each scan cell within the partition is tested. Another approach uses binary search (BS) [12] to adaptively zoom in on the faulty cells within a scan chain by recursively bisecting the scan chain. Other approaches [2], [4] can be considered as hybrids between adaptive and nonadaptive algorithms. In [2], the partitions are determined *a priori* as in nonadaptive diagnosis, but the superposition principle is adaptively used during diagnosis to calculate a partition test response without carrying out the test, potentially reducing the diagnosis time.

In this paper, we show that CGT offers a rich set of techniques to improve solutions to the BIST diagnosis problem. We elucidate how some of the aforementioned BIST diagnosis techniques are reminiscent of CGT techniques. We use this to provide further insights into existing diagnosis techniques and to improve their performance. We also adapt and apply a set of CGT techniques to the BIST diagnosis problem. CGT techniques can further be combined and enhanced by other methods that have been proposed for BIST diagnosis. A number of research directions remain open and offer the prospect of more cost-effective BIST diagnosis solutions as well as new theoretical frameworks. The contributions of this paper can be summarized as follows.

1) We improve the partitioning approach of Ghosh-Dastidar *et al.* [11] by proposing a method for calculating the partition size.
2) We propose a new algorithm called multistage batching, which extends and generalizes the approach of Ghosh-Dastidar *et al.* [11].
3) We extend the BS approach of Ghosh-Dastidar and Touba [12] to benefit from the superposition principle advocated by Bayraktaroglu and Orailoglu [2].
4) We propose the digging procedure, which outperforms BS in the presence of few faults.
5) We propose a number of new algorithms called jumping, doubling, batched BS, and batched digging, which outperform other algorithms in the presence of a large number of faults.
6) We expose some of the underlying principles behind nonadaptive diagnosis.

In the remainder of this paper, Section II gives several new diagnosis techniques as well as a number of improvements to existing methods. Section III assesses the performance of our proposed techniques against existing BIST diagnosis techniques. Finally, Section IV gives our conclusions and a number of directions for future research.

II. New and Improved Algorithms for Diagnosis

In this section, we present new algorithms for the BIST diagnosis problem as well as a number of improvements to existing BIST diagnosis algorithms.

A. Batching-Based Methods

Perhaps one of the oldest CGT techniques is batching [7]. To identify the unknown number of d faulty items, the set M of items under test is divided into disjoint, equal-size batches $b_1, b_2, \ldots, b_k$, where $1 \leq k \leq n$. Each of these batches is then tested. If a batch b_i is identified as faulty, the batching algorithm tests each individual module $m_j \in b_i, \forall 1 \leq j \leq |b_i|$. A crucial parameter affecting the success of this algorithm is the value of k. Without loss of generality, assume that n is

```
Input: M = (ordered) set of items under test.
Output: F ⊆ M = set of faulty items.
-----------------------------------------------
    Set n = |M|, b = √n, k = 0, and F = ∅
    While (true)
        Set l=k × b
        Set h = min((k+1)  b,n)
        If l > |M| then break
        If query(M[l..h]) then
            For u = l to h:
            If query(M[u]) then
            F = F ∪ M[u]
        Set k = k+1
    Return F
```

Fig. 2. One-stage batching diagnosis.

divisible by k. In this case, the number of queries $q(n,k)$ is given by

$$q(n,k) \leq dk + \frac{n}{k}. \tag{1}$$

The batching technique was proposed independently by Ghosh-Dastidar *et al.* [11] for the detection of scan cells that capture faulty responses. While the approach in [11] stops short of specifying the batch size k, Dorfman [7] suggests that if n is viewed as continuous, then $q(n,k)$ is minimized when $k = \sqrt{n/d}$. Since we have no *a priori* knowledge of d, we may assume that $d = 1$, implying that the optimum batch size is $\sqrt{n}$. For one faulty module, this method, which we call one-stage batching, uses $2\sqrt{n}$ queries. A formal description of one-stage batching is given in Fig. 2. In Fig. 2 and in subsequent algorithm descriptions, we use the function $query(\cdot)$, which, given a set of items S, returns true if one or more items of S are faulty; otherwise, it returns false. Calling this function corresponds to execution of a test session. We also use the term "ordered" only to indicate that items in M are indexed to permit reference to individual items or ranges of items.

A possible enhancement to one-stage batching is as follows [7], [16]. First, the set M of items under test is divided into disjoint equal-size batches $b_1, b_2, \ldots, b_k$, where $1 \leq k \leq n$. Each of these batches is then tested. Once a set of faulty batches $b_{f_1}, b_{f_2}, \ldots, b_{f_m}$ (with $m < k$) has been identified, one can remerge all the faulty batches into one set $M_{s_2} = \bigcup_{i=1}^{m} b_{f_i}$, and then reapply the same procedure recursively. This procedure can be repeated for any number of stages until the point where all the batches are identified as faulty. In this last case, we query all the items of the batches. We will refer to this approach by multistage batching. With multistage batching, the number of queries $q(n)$ is given by

$$q(n) \leq \frac{n}{k_1} + \frac{dk_1}{k_2} + \ldots + \frac{dk_{s-1}}{k_s} + dk_s \tag{2}$$

where k_i, $1 \leq i \leq s$, is the number of batches for stage i. The values of k_i, $1 \leq i \leq s$, set the performance of the algorithm.

```
Input: M = (ordered) set of items under test.
Output: F ⊆ M = set of faulty items.
-----------------------------------------------
    Set n = |M|, b = √n, k = 0, U = ∅, and
    F = ∅
    While (true)
        Set l=k × b
        Set h = min((k+1)× b,  |M|)
        If l > |M| then break
        If query(M[l..h]) then
        U = U ∪ M[l..h]
        Set k = k+1
    If |U| ≠ n then F = F∪ MULTISTAGE(U)
    else for u = 1 to |U|: If query(M[u]) then
    F = F ∪ M[u]
    Return F
```

Fig. 3. MULTISTAGE procedure for multistage batching diagnosis.

To estimate a reasonable value for k_i, we investigate the special case of one faulty item, i.e., $d = 1$. If n is viewed as continuous, then from basic calculus, $q(n)$ is minimized when $k = \sqrt{n/d}$. Hence, we may set $k_i = \sqrt{n_{i-1}}$, where n_i is the total number of items in faulty batches at stage i ($n_i \leq dk_i$) and $k_1 = \sqrt{n}$. We will see in Section III that the multistage strategy offers a considerable reduction in the number of queries compared to the approach of Ghosh-Dastidar *et al.* [11]. The multistage batching algorithm, which we refer to as MULTISTAGE, is formally described in Fig. 3.

B. Binary-Search-Based Techniques

Binary Search (BS) may be viewed as the most basic of all adaptive CGT methods [8, p. 128]. The technique was discovered independently by Ghosh-Dastidar and Touba [12] for solving the BIST diagnosis problem. In this basic algorithm, if a faulty set of items $S \subseteq M$ is identified, then S is bisected into two sets S_1 and S_2, and the BS algorithm is recursively applied to each. S_1 and S_2 are typically called the children of S. As analyzed in both [8] and [12], the upper bound on the number of queries is $2d(\log(n/d) + 1) - 1$ for d defective items. BS has good performance for small to moderate values of d (relative to n). However, the algorithm significantly degrades for large values of d, with number of queries approximately $\approx 2n - 1$ (i.e., requiring more than n queries). We sketch the BS algorithm in Fig. 4.

We now improve upon BS in two ways. 1) We apply and extend the superposition principle to be included in BS. Superposition cuts down the number of test sessions required to identify all faulty items. 2) Based on the CGT body of work, we modify the BS algorithm to the *digging* algorithm. Such modification reduces the number of test sessions in the presence of few fault items. We start by introducing the superposition principle.

Input: $M =$ set of items under test.
Output: $F \subseteq M =$ set of faulty items.

Set $n = |M|$ and $F = \emptyset$
If $n = 1$ then return M
Bisect M into M_1 and M_2
If query(M_1) then
  $F = F \cup$ BS(M_1)
  if query(M_2) then $F = F \cup$ BS(M_2)
else $F = F \cup$ BS(M_2)
Return F

Fig. 4. BS procedure.

Input: $M =$ set of items under test, with one or more faulty items.
Output: $m_f =$ faulty item.

Set $B = M$
do
  $B_1 = \lceil \frac{|B|}{2} \rceil$ items from B
  If query(B_1) then $B = B_1$
  else $B = B \setminus B_1$
while $|B| > 1$
Return m_f, where $B = \{m_f\}$

Fig. 5. DIG procedure to identify one item from a set of faulty items.

Input: $M =$ (ordered) set of items under test.
Ouput: $F \subseteq M =$ set of faulty items.

Set the queue $Q = \{M\}$ and $F = \emptyset$.
do
  Pop the head element S of Q
  If query(S) then
    m_f=DIG(S)
    Set $M = M \setminus \{m_f\}$ and $F = F \cup \{m_f\}$
    Bisect M into M_1 and M_2
    If $M_1 \neq \emptyset$ then insert M_1 in Q
    If $M_2 \neq \emptyset$ then insert M_2 in Q
while $Q \neq \emptyset$
Return F

Fig. 6. DIG-BS: BS using a number of DIGs.

The superposition principle states that given an LFSR as a signature analyzer and the signature of two sets of items S_1 and S_2, we can obtain the signature of a new set S_3 by XORing the signatures of S_1 and S_2, i.e., $S_3 = S_1 \oplus S_2$. S_3 is the set of items that are contained in S_1 and S_2 but not in both. Superposition is utilized by Bayraktarolgu and Orailoglu [2], [5] to speed up the pseudorandom and deterministic partitioning techniques of [3], [5], [20], and [21]. We improve BS by noticing that if the tester is an LFSR or MISR, then the superposition principle can be applied to BS. Extending the superposition principle to BS works as follows: If a set S is found to be faulty, then S is bisected into two sets S_1 and S_2. We assume S_1 is tested before S_2. Since S is faulty, there can be three cases.

1) S_1 is fault free but S_2 is faulty: This case is already handled by traditional BS [12]. Since the diagnosis algorithm finds S_1 to be fault free, it must be the case that S_2 is faulty and, thus, the diagnosis proceeds directly to test the children of S_2.
2) S_1 is faulty but S_2 is fault free: After having the signature of the set S_1, the diagnosis algorithm can calculate the signature of S_2 $(S \oplus S_1)$ and compare it to the fault-free response. In this case, the test session associated with testing S_2 is saved by computing its result using the superposition principle.
3) S_1 is faulty and S_2 is faulty: This is similar to case 2), except that the diagnosis technique finds the computed signature of S_2 faulty and, hence, proceeds directly to test its children. Again, the test session associated with testing S_2 is saved by computing it using the superposition principle.

Hence, applying the superposition principle to BS saves a test session in each of cases 2) and 3), leading to an overall reduction in the number of test sessions and, correspondingly, the diagnosis time.

The BS algorithm can also be modified in a different way as follows. The digging algorithm can be considered as an improvement to the BS. Digging reduces the number of queries (test sessions), especially for low values of d (faulty items) [8], [13]. Observe that if there are two defective sets M_1 and M_2, with $M_1 \subset M_2$, then the result of a query on M_1 renders the result of the query on M_2 useless. Hence, with BS, there is the potential for many queries to produce no additional information for the diagnosis process. This suggests that once a faulty set of items M_f is found, a faulty item m_f should be identified from the set. This process is referred to as digging [8]. After the faulty item m_f is identified, m_f is removed from M_f, and digging is resumed on the remaining items. Digging requires $d \log n$ queries. For small values of d, digging improves BS, as we confirm in Section III. We present the DIG and DIG-BS (BS with digging) procedures in Figs. 5 and 6. Extending the superposition principle to the digging procedure can be carried out as with the BS procedure.

C. Combination of Batching and Binary Search

One of the main shortcomings of batching algorithms is their relatively poor performance for small and moderate values of d. This is mainly due to the need to query all faulty items in the faulty batches. On the other hand, BS-based algorithms

Input: $M =$ (ordered) set of items under test.
Ouput: $F \subseteq M =$ set of faulty items.

Set $n = |M|$, $b = \sqrt{n}$, $k = 0$, and $F = \emptyset$
While(true)
    Set $l = k \times b$
    Set $h = \min((k+1)\ \ b, j)$
    If $l > n$ then break
    If query(M, l, h) then
    DIG-BS$(M[l..h])$
    Set $k = k+1$
Return F

Fig. 7. Batched DIG-BS diagnosis.

Input: $M =$ (ordered) set of items under test.
Ouput: $F \subseteq M =$ set of faulty items.

Set $F = \emptyset$
while $M \neq \emptyset$
    if query(M) then $M = \emptyset$
    Set $k = 1$ and $S = \emptyset$
    while (true)
        $S = \min(k, |M|)$ items from M
        if query(S) = false then
        set $k = 2\ \ k$ and $M = M \setminus S$
        else break
    Set $m_f =$DIG(S)
    Set $M = M \setminus \{m_f\}$ and
    $F = F \cup \{m_f\}$
Return F

Fig. 8. DOUBLING algorithm.

excel for small values of d and rapidly deteriorate as d approaches the total number of items n. Hence, one way to improve results for small and moderate values of d is to start by batching; then, once the faulty batching sets are identified, BS or digging (DIG-BS) may be applied to identify the faulty items within these sets. In this hybrid method, batching is used to initially prune a large portion of the search space, clearing the way for BS to identify the faulty items with fewer queries. We will see below that the proposed strategy outperforms other techniques for a significant range of practical values of d. The batched DIG-BS algorithm is described in Fig. 7; a batched BS algorithm based on BS rather than digging may be similarly conceived.

D. Doubling and Jumping

We propose a new BIST diagnosis algorithm based on the works of Bar-Noy *et al.* [1]. Given that the value of d is unknown, the algorithm attempts to estimate the value of d. If d is small, then the algorithm finds large fault-free sets; otherwise, the algorithm finds small faulty sets. To deliver this functionality, the algorithm tests disjoint sets of sizes $1, 2, 4, \ldots, 2^i$ until a faulty set is found. At this point, the algorithm has identified $2^i - 1$ fault-free items and a faulty set of size 2^i, using i tests. The algorithm then identifies a faulty item from the faulty set using BS, which requires i queries. Consequently, the algorithm uses $2i + 1$ queries and detects 2^i items ($2^i - 1$ fault free and 1 faulty). We present this DOUBLING algorithm in Fig. 8.

An interesting modification to doubling, which is called *jumping*, is due to [9]. Instead of testing disjoint sets of sizes $1, \ldots, 2^i$ as in doubling, jumping tests sets having sizes $1 + 2$, $4 + 8, \ldots, 2^i + 2^{i+1}$ until a faulty set is found. Using these "jumps" (i.e., in the ordering of the items), the algorithm identifies fault-free items with $i/2$ tests instead of i tests. However, a faulty set is of size 3×2^i, rather than of size 2^i as in doubling; therefore, it requires more than one query on a subset of size 2^i to reduce the faulty set to either 2^i or to size 2^{i+1} with 2^i fault-free items. We do not describe the details of the jumping algorithm here but assess it empirically in Section III and refer the interested reader to [8, p. 134] or [9].

TABLE I
DETERMINISTIC PARTITIONING OF A SET OF ITEMS ACCORDING TO THE METHOD OF BAYRAKTAROGLU AND ORAILOGLU [3]

Group 0	Test 0 = $\{m_0, m_3, m_6\}$	Test 1 = $\{m_1, m_4, m_7\}$	Test 2 = $\{m_2, m_5, m_8\}$
Group 1	Test 3 = $\{m_0, m_4, m_8\}$	Test 4 = $\{m_1, m_5, m_6\}$	Test 5 = $\{m_2, m_3, m_7\}$
Group 2	Test 6 = $\{m_0, m_5, m_7\}$	Test 7 = $\{m_1, m_3, m_8\}$	Test 8 = $\{m_2, m_4, m_6\}$

TABLE II
CONSTRUCTING THE d-DISJUNCT MATRIX ACCORDING TO THE CGT LITERATURE [8]

	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8
Test 0	1	0	0	1	0	0	1	0	0
Test 1	0	1	0	0	1	0	0	1	0
Test 2	0	0	1	0	0	1	0	0	1
Test 3	1	0	0	0	1	0	0	0	1
Test 4	0	1	0	0	0	1	1	0	0
Test 5	0	0	1	1	0	0	0	1	0
Test 6	1	0	0	0	0	1	0	1	0
Test 7	0	1	0	1	0	0	0	0	1
Test 8	0	0	1	0	1	0	1	0	0

E. Principle of Nonadaptive Diagnosis Methods

In contrast to adaptive diagnosis, nonadaptive diagnosis partitions the set of scan cells into partitions *a priori*, that is, before the start of the diagnosis procedure. A fixed test pattern set is then repetitively applied to each partition, and the response captured in these cells is compacted by an LFSR or MISR and compared to the respective fault-free signature.

TABLE III
NUMBER OF TEST SESSIONS OR QUERIES (PROPORTIONAL TO DIAGNOSIS TIME) FOR THE PROPOSED ALGORITHMS VERSUS METHODS FROM THE DIAGNOSIS LITERATURE

Faults	Scan-chain length, $n = 961$									
	Diagnosis literature				Proposed algorithms					
(d)	Random partitioning [21]	Deterministic partitioning [3], [5]	BS [12]	One-stage batching [11]	DIG-BS	Multi-stage batching	Doubling	Jumping	Batched BS	Batched DIG-BS
1	84	62	15	63	**11**	45	19	21	39	37
2	100	90	28	93	**22**	54	36	33	47	43
3	113	97	39	122	**32**	64	51	45	54	49
4	128	111	50	149	**41**	75	64	56	61	55
5	137	122	61	177	**51**	86	79	67	69	61
6	152	130	70	201	**61**	99	92	78	76	67
7	161	139	80	231	**71**	111	104	88	83	73
8	174	146	89	250	80	124	118	98	90	**78**
9	183	161	97	277	90	138	128	108	96	**84**
10	198	169	107	301	100	152	140	118	104	**90**
11	210	184	116	323	110	168	152	127	111	**95**
12	224	192	123	346	120	184	161	136	117	**102**
13	236	201	132	368	129	198	173	147	124	**107**
14	247	213	139	390	139	216	184	154	131	**113**
15	263	226	147	407	148	234	195	164	138	**118**
16	272	234	157	433	158	249	205	173	143	**125**
17	278	237	162	445	168	265	214	181	150	**130**
18	300	257	170	463	178	287	225	190	157	**136**
19	311	264	178	480	187	302	233	197	164	**141**
20	328	276	185	497	197	316	244	206	170	**147**
21	339	291	192	515	208	331	251	213	176	**152**
22	364	296	199	528	215	346	263	223	183	**158**
23	389	319	206	550	225	364	271	230	188	**164**
24	401	334	212	561	235	385	280	238	193	**169**
25	397	339	221	570	244	401	289	243	200	**174**
26	413	352	226	580	253	410	297	251	208	**179**
27	448	368	233	594	264	432	305	260	212	**186**
28	458	391	241	605	274	455	314	267	219	**192**
29	484	400	246	624	283	475	324	274	226	**196**
30	495	419	251	634	292	483	332	283	230	**201**

Results are the averages of 100 simulations. Smallest numbers of test sessions are in bold.

In the scheme of Rajski and Tyszer [20], [21], scan cells are initially partitioned pseudorandomly. To identify the faulty cells, the diagnosis algorithm initially assumes that all cells are candidates for being faulty. As the algorithm proceeds by repetitively applying the set of test patterns, all cells that belong to subsets that generate fault-free responses are declared fault-free and dropped from the set of candidate faulty cells. After a predetermined number of repetitive test applications, all the cells that remain in the set of candidate faulty cells are declared faulty. The algorithm, however, does not guarantee that all cells remaining in the fault list are indeed faulty. Bayrktaroglu and Orailoglu [3] observe that minimizing the overlap between different partitions in different tests reduces the number of test repetitions and, hence, reduces the diagnosis time. They propose a deterministic partitioning scheme based on their method of quotient uniform partitioning, where partitions in different tests overlap in exactly one scan cell.

For example, using the uniform quotient remainder method [3], we construct a deterministic partitioning on a set $\{m_0, m_1, \ldots, m_8\}$ of nine items as shown in Table I. A group is one set of partitions that cover the nine items, i.e., one row in Table I. This method has the property that partitions in different groups share exactly one item [3] and, hence, the minimum possible overlap. In this example, the same test (a fixed set of test patterns) is applied to each partition, and the partitioning can locate at most two defects. Since there are three partitions

TABLE IV
NUMBER OF TEST SESSIONS OR QUERIES (PROPORTIONAL TO DIAGNOSIS TIME) FOR THE PROPOSED ALGORITHMS VERSUS METHODS FROM THE DIAGNOSIS LITERATURE

Faults	Scan-chain length, n = 10201									
	Diagnosis literature				Proposed algorithms					
(d)	Random partitioning [21]	Deterministic partitioning [3], [5]	BS [12]	One-stage batching [11]	DIG-BS	Multi-stage batching	Doubling	Jumping	Batched BS	Batched DIG-BS
1	314	202	21	203	**15**	121	26	38	112	109
2	366	301	39	300	**28**	134	49	58	122	117
3	396	307	56	400	**42**	147	72	75	132	125
4	413	319	71	499	**55**	161	92	89	143	132
5	445	356	86	598	**69**	176	113	104	152	140
6	468	363	102	698	**82**	193	132	117	163	148
7	497	391	115	787	**94**	209	152	130	173	155
8	513	416	130	886	**108**	226	171	144	183	163
9	535	425	145	973	**120**	243	191	158	193	171
10	552	451	159	1072	**134**	262	208	171	204	178
11	577	461	173	1158	**146**	280	226	185	213	186
12	592	475	186	1257	**160**	299	244	197	224	193
13	593	494	201	1344	**173**	321	261	210	233	202
14	609	502	213	1438	**186**	342	278	224	244	209
15	646	521	228	1516	**199**	359	296	238	254	217
16	646	547	239	1600	**213**	385	314	252	263	224
17	660	541	251	1701	**225**	412	330	265	274	232
18	672	571	265	1781	**239**	438	345	278	284	**239**
19	701	584	277	1869	251	464	363	291	293	**247**
20	712	582	289	1959	265	494	378	304	303	**255**
21	743	610	303	2026	277	512	392	318	314	**263**
22	756	613	314	2123	290	542	411	331	322	**270**
23	766	627	326	2189	302	574	427	344	333	**278**
24	766	645	337	2271	316	593	443	356	341	**285**
25	794	674	351	2354	330	639	458	369	353	**292**
26	810	679	361	2386	342	652	473	382	363	**300**
27	824	691	376	2521	357	686	489	394	371	**308**
28	840	711	385	2597	369	719	502	408	382	**316**
29	846	731	396	2649	381	740	519	421	391	**322**
30	870	740	409	2707	394	774	533	433	401	**331**
31	896	750	422	2831	408	804	547	444	409	**338**
32	894	781	432	2887	420	828	564	457	421	**346**
33	938	785	440	2943	433	863	579	470	428	**353**
34	933	792	452	3011	447	894	593	483	440	**360**
35	949	805	464	3087	459	939	607	494	450	**368**
36	942	830	474	3191	473	961	624	507	459	**375**
37	985	840	486	3219	485	995	638	518	468	**382**
38	981	838	497	3303	498	1043	651	531	478	**391**
39	995	858	510	3419	510	1071	664	543	487	**398**
40	1009	893	520	3450	523	1101	679	555	495	**405**

Results are the averages of 100 simulations. Smallest numbers of test sessions are in bold.

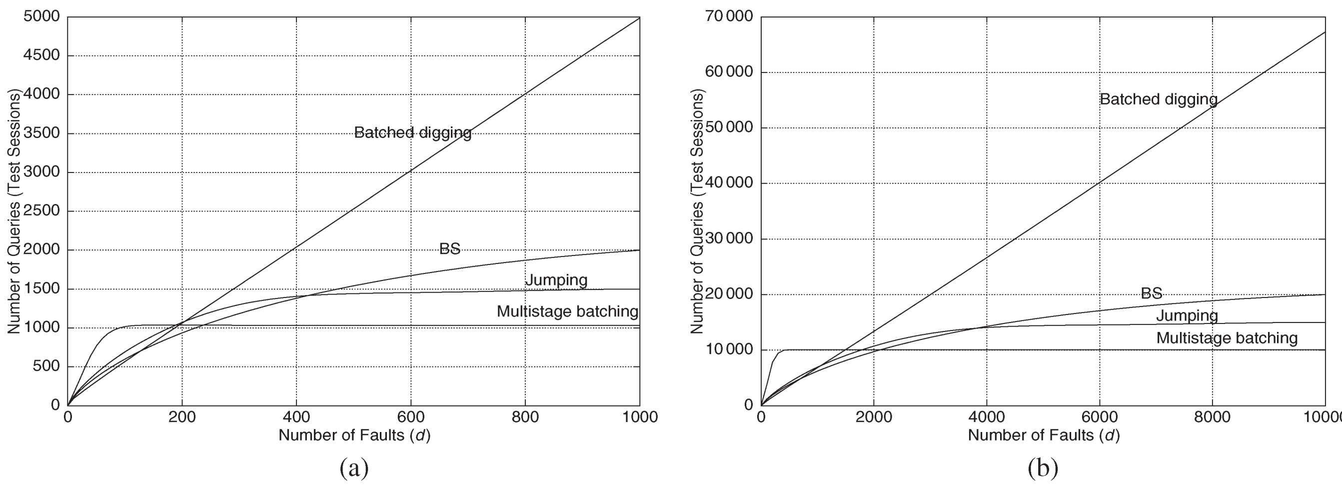

Fig. 9. Effect of n on performance and relative dominance of selected algorithms. (a) Performance of selected algorithms for $n = 1000$. (b) Performance of selected algorithms for $n = 10\,000$.

per group, at least one partition must produce a nonfaulty response [3].[1] The cells of this fault-free partition are removed from the candidate list and the process is repeated until the candidate faulty list contains only the faulty cells. For example, if m_2 and m_7 are faulty, then $\{m_0, m_3, m_6\}$ are declared fault free after testing group 0, and $\{m_4, m_8, m_1, m_5\}$ are declared fault free after testing group 1. Testing group 2 does not remove any additional cells from the set of candidate faulty cells, and, finally, $\{m_2, m_7\}$ are declared faulty.

One impact of CGT on fault diagnosis lies in providing a systematic way to study nonadaptive diagnosis techniques [8] as follows. The general idea is to design a sequence of tests such that for any given set of d faulty items, there is a unique set of tests that detects these faults, i.e., any set of d faults induces a different set of tests with faulty response. By constructing a matching between the set of tests with faulty response and all possible sets of d faulty items, the d faulty items can be identified. Such a sequence of tests is then called d-disjunctive. One method of designing such sequences is called equireplicated pairwise balanced design (EPBD), which is reminiscent of the quotient uniform partitioning technique of Bayraktaroglu and Orailoglu [3].

For the set of nine items previously pointed out, we can construct a 9×9 matrix with the columns representing the possible faulty items and the rows the possible tests. An entry of 1 at the intersection of row i with column j indicates that item j is included in test i. Table II shows such a matrix for the partitioning of Table I. From Table II, we notice the following.

1) There does not exist a pair of items that is included in more than one test.
2) The pairwise of disjunction of any two columns is unique, which implies that no column is a subset of the disjunction of any two other columns, i.e., the matrix is d-disjunct. Here, $d = 2$.
3) From the first two properties, we conclude that for any pair of faulty items, a distinct set of tests will exhibit a faulty response.

The last property leads to faster determination of faulty items, i.e., instead of just iteratively removing items from the candidate fault list until the set of faulty items remain [3], we can directly match the set of tests that exhibit faulty response with the faulty items.

[1] Bayraktaroglu and Orailoglu's [3] idea is that if at least one fault-free partition is identified, then all its scan cells can be declared as fault free, and, consequently, if these fault-free cells participate in any future faulty partitions, the diagnosis algorithm can decide that the fault is not in these fault-free cells but rather in the other cells of the partition.

III. Experimental Results

We empirically assess the aforementioned diagnosis techniques. We compare results of the proposed algorithms against a number of the leading techniques in the diagnosis literature. From the diagnosis literature, we implement most of the recent techniques: the pseudorandom partitioning diagnosis technique of Rajski and Tyszer [20], [21]; the deterministic approach of Bayraktaroglu and Orailoglu [3], [5]; the linear search approach of Ghosh-Dastidar *et al.* [11] (with the partition size calculation we earlier proposed); and the BS approach of Ghosh-Dastidar and Touba [12]. We also implement the digging, multistage batching, doubling, and jumping algorithms, as well as the batched digging and batched BS procedures. Furthermore, we implement the same algorithms (whenever applicable) with superposition [2], [5] to reduce diagnosis time.

In the first set of experiments, we set up an experimental framework similar to that of [2], [3], [5], [12], and [21]. We assume a scan chain of length n scan cells, and we randomly set d scan cells to be faulty and calculate the number of queries (test sessions) needed to detect all faults. For each value of d, we generate 100 random instances and calculate the average number of queries. The number of queries (test sessions) is directly proportional to the amount of time needed to diagnose the BIST system. Modern scan-based designs feature thousands of scan cells [18]. According to Mitra and Kim [18], a typical design features a number of scan chains each comprised of about 10 000 scan cells. Due to the primality constraints of Bayraktaroglu and Orailoglu [3], [5], we select n to be $10\,201\ (= 101 \times 101)$ and also collect results for $n = 961$

TABLE V
EMPIRICAL ASSESSMENT OF DOMINANT ALGORITHMS VERSUS INCIDENCE OF FAULTY SCAN CELLS, FOR $n = 1000$

Percentage of faulty items	Algorithm
$\frac{d}{n}$ < 1:3%	Digging
1:3% < $\frac{d}{n}$ < 10%	Batched digging
10% < $\frac{d}{n}$ < 25%	BS
25% < $\frac{d}{n}$ < 100%	Multistage batching

$(= 31 \times 31)$ to approximate a scan chain of 1000 cells. We compare all diagnosis techniques for these values of scan chain lengths. For the pseudorandom partitioning approach of Rajski and Tyszer [21], we select the number of partitions to be 128 and 32 for $n = 10\,201$ and $n = 961$, respectively. For the deterministic partitioning approach of Bayraktaroglu and Orailoglu [3], [5], the number of partitions is 101 and 31 for $n = 10\,201$ and $n = 961$, respectively. Both partitioning techniques are applied until 100% resolution is attained. For the approach of Ghosh-Dastidar *et al.* [11] (one-stage batching in the CGT literature), we set the partition size to be $\lfloor\sqrt{n}\rfloor$.

We evaluate the aforementioned algorithms without superposition for scan chains of length $n = 961$ and $n = 10\,201$. Results are tabulated in Table III for $n = 961$ and in Table IV for $n = 10\,201$. We say that technique A dominates technique B if technique A results in fewer queries than technique B for all numbers of reported faults. From the tables, we make the following observations, some details of which may be specific to values of scan-chain lengths studied.

1) Deterministic partitioning [3], [5] dominates pseudorandom partitioning [21] with about 15% improvement, which is consistent with results reported in [5]. Both techniques are dominated by BS [12].
2) For a handful values of fault cardinalities ($d \leq 7$ for $n = 961$ and $d \leq 18$ for $n = 10\,201$), digging outperforms other algorithms, while batched digging outperforms other algorithms for the remaining values of faults. At 30 faults, batched digging outperforms the best diagnosis technique (BS) by about 20%.
3) Multistage batching dominates one-stage batching for most values of d and achieves a reduction in the number of test sessions by around 25% at $d = 30$ for $n = 961$ and by around 71% at $d = 30$ for $n = 10\,201$.
4) Jumping dominates doubling for most numbers of faults.

We note that the approaches of pseudorandom and deterministic partitioning in [3], [5], and [21] are helpful in the case of few faults. As the number of faults increases, these algorithms dynamically adjust the number of partitions to be greater than the number of faults [5]. This leads to a dramatic increase in the number of test sessions.

We examine the effect of large number of faults on the performance of selected algorithms (without superposition) by graphically plotting the number of test sessions versus fault cardinality in Fig. 9(a) and (b) for $n = 1000$ and $n = 10\,000$ respectively. We plot the results for the multistage batching, jumping, BS, and batched digging (batched DIG-BS)

TABLE VI
NUMBER OF TEST SESSIONS OR QUERIES (PROPORTIONAL TO DIAGNOSIS TIME) FOR THE PROPOSED ALGORITHMS WHILE APPLYING THE SUPERPOSITION PRINCIPLE

Faults	Scan-chain length, n = 961					
	All algorithms are extended using superposition					
(d)	Random partitioning [21]	Deterministic partitioning [3], [5]	BS [12]	DIG-BS	Batched BS	Batched DIG-BS
1	83	62	**10**	10	37	37
2	91	62	**18**	20	41	41
3	96	62	**26**	29	46	46
4	98	63	**33**	38	51	51
5	98	65	**39**	46	56	56
6	102	67	**45**	55	60	61
7	104	72	**51**	64	65	66
8	104	77	**57**	72	70	71
9	107	81	**63**	82	75	76
10	109	87	**68**	90	79	80
11	115	92	**73**	99	83	85
12	118	94	**78**	107	88	90
13	119	99	**83**	116	92	95
14	120	103	**88**	125	96	99
15	130	109	**94**	133	101	104
16	129	112	**97**	142	105	109
17	135	118	**102**	151	109	113
18	139	120	**107**	159	113	118
19	141	126	**111**	169	117	123
20	144	130	**116**	177	121	127
21	152	133	**120**	186	125	132
22	150	138	**124**	194	129	137
23	156	143	**129**	202	132	141
24	167	148	**134**	213	137	146
25	170	151	**136**	220	141	151
26	178	156	**141**	229	145	155
27	179	163	**145**	239	149	159
28	184	164	**149**	246	153	163
29	191	172	**153**	253	156	168
30	199	179	**158**	263	159	173

Results are the averages of 100 simulations. Smallest numbers of test sessions are in bold.

approaches and summarize these conclusions, for the regime of $n = 1000$, in Table V. These results suggest the following.

1) There is no algorithm that universally, i.e., for all possible fault values, dominates other algorithms.
2) Algorithms that excel for small number of faults perform poorly for large number of faults, and vice versa.
3) An appropriate heuristic may be identifiable *a priori* based on parameters of the diagnosis instance.

To test the effect of superposition, we implement a number of previous diagnosis algorithms using the superposition principle [2], [5] explained earlier, whenever applicable [e.g., the test

TABLE VII
ISCAS'89 BENCHMARKS RESULTS

Benchmark properties				Diagnosis literature			Proposed algorithms					
Benchmark	Scan chain	Av	Max	Random part. [21]	BS [12]	One-stage batching [11]	DIG-BS	Multi-stage batching	Doubling	Jumping	Batched BS	Batched DIG-BS
s208	10	1.14	3	29.81	5.87	7.00	**5.23**	8.65	6.96	7.32	6.51	6.21
s298	20	1.27	5	45.51	8.30	10.60	**7.45**	12.72	11.28	10.49	9.58	9.35
s344	26	1.55	8	44.39	8.35	11.20	**7.63**	12.67	11.45	10.43	8.91	8.94
s838	34	1.26	5	51.68	9.13	13.30	**8.46**	15.11	12.30	11.49	12.14	11.82
s1196	32	1.70	9	58.46	11.50	15.34	**10.84**	17.28	14.00	14.09	13.97	13.25
s1238	32	1.53	8	59.61	10.39	14.19	**9.93**	16.11	12.96	12.94	12.72	12.21
s1423	79	1.80	38	71.46	**13.42**	21.13	13.59	21.30	16.90	16.10	17.60	17.27
s1488	25	1.19	11	54.98	8.29	11.55	**7.33**	13.22	11.46	10.91	10.20	9.67
s9234	250	1.45	16	75.06	16.56	37.14	**13.50**	30.49	20.42	19.63	25.41	23.92
s13207	790	1.76	23	93.01	24.02	76.96	**18.97**	50.23	30.31	29.35	42.61	39.76
s15850	684	2.07	97	92.24	25.32	72.18	**20.97**	49.37	31.93	30.12	41.25	38.65

Scan chain is the length of the scan chain. Max is the maximum number of scan cells capturing faulty responses for any of the possible stuck-at faults. Av is the average number of scan cells capturing faulty responses for all stuck-at faults. HOPE fault simulation terminated incorrectly for benchmarks of larger size.

session (query) result is an LFSR signature]. Specifically, we implement the pseudorandom partitioning diagnosis technique of Rajski and Tyszer [20], [21] and the deterministic approach of Bayraktaroglu and Orailoglu [3], [5] with superposition. We also extend the superposition principle to the BS approach of Ghosh-Dastidar and Touba [12] as explained earlier, and to the proposed approaches of digging and batched digging.[2] Our results are presented in Table VI for $n = 961$. Comparing to the earlier Table III, we find that superposition improves all algorithms. The magnitude of improvement for pseudorandom partitioning and deterministic partitioning is consistent with the results reported in [3] and [5]. However, for the instance parameters that we study, BS seems to benefit the most from applying the superposition principle, with batched BS following closely in quality of results.

The above experiments assess the performance of various algorithms on randomly generated scan chains for different incidences of faults. However, real benchmarks with real faults tend to generate error sequences that are different in nature from random error sequences. In our second set of experiments, we assess the performance of various algorithms using the ISCAS'89 benchmarks. We conduct the following experiment.

1) Given a benchmark, the automatic test pattern generation (ATPG) tool Atalanta [15] is applied to procure a set of test patterns that detects the stuck-at faults of the given benchmark.
2) For each stuck-at fault, we 1) collect the signature of all test patterns using the fault simulator HOPE [14] and 2) use the various diagnosis algorithms to detect all scan cells (outputs) that received faulty responses.

Experimental results on the ISCAS'89 benchmarks are given in Table VII. In the table, we report the average number of queries that each diagnosis algorithm takes to identify all scan cells capturing faulty responses for all stuck-at faults, i.e., results of just one stuck-at fault are not reported but rather the average result of all stuck-at faults. The average and maximum numbers of scan cells affected by a stuck-at fault are also reported. We observe the following.

1) In real benchmarks, very few cells are typically affected by a fault. This is helpful in selecting which diagnosis algorithm should be used, e.g., digging.
2) Our proposed algorithm, digging, outperforms state-of-the-art techniques, in all benchmarks but one, by up to 21%. Furthermore, the improvement tends to increase as the benchmark size increases.[3]

IV. CONCLUSION

Our work has addressed the issue of BIST diagnosis and revealed previously unexplored connections to the field of combinatorial group testing (CGT). We show that the BIST diagnosis problem corresponds precisely to the heart of the well-established field of CGT. We improve a number of BIST diagnosis techniques. For example, we propose a modification, digging, to binary search (BS) that reduces the number of test sessions in the presence of a few faults. We also extend the applicability of the superposition principle to other algorithms. We have also proposed extensions to batching algorithms, e.g., one-stage and multistage batching algorithms. We have also demonstrated additional benefits from the combination of two well-known CGT methods, BS and batching. A number of other

[2] We assume ideal conditions where no aliasing occurs. In general, statistical studies in fault diagnosis show that aliasing has a very low probability, and perhaps one possible solution is to include two different signature analyzers, with different primitive polynomials, in the design. A query is declares faulty if any of the two signatures is faulty.

[3] We had problems with HOPE fault simulation for the remaining ISCAS'89 benchmarks. More specifically, for the remaining benchmarks of the ISCAS'89 suite, HOPE aborted the process of writing out the detected faults for each test pattern.

algorithms, e.g., jumping and doubling, have been proposed and empirically tested. We have conducted an experimental study that compares the various algorithms in an abstract setting and using the practical setting of the ISCAS'89 benchmarks. Our results indicate the competitiveness and effectiveness of the CGT algorithms for BIST diagnosis. The link to CGT may also initiate new methods for BIST diagnosis. We conclude with four possible directions for future work:

1) competitive CGT for theoretical benchmarking of the different diagnosis techniques;
2) nonadaptive diagnosis techniques using binary superimposed codes;
3) special cases of diagnosis wherein the number of faults is known *a priori*;
4) diagnosis in the presence of unreliable tests, where the query response might be erroneous as with aliasing cases of MISRs.

REFERENCES

[1] A. Bar-Noy, F. Hwang, H. Kessler, and S. Kutten, "A new competitive algorithm for group testing," *Discrete Appl. Math.*, vol. 52, no. 1, pp. 29–38, 1994.

[2] I. Bayraktaroglu and A. Orailoglu, "Improved fault diagnosis in scan-based BIST via superposition," in *Proc. ACM/IEEE Design Automation Conf.*, Los Angeles, CA, 2000, pp. 55–58.

[3] ——, "Deterministic partitioning techniques for fault diagnosis in scan-based BIST," in *Proc. Int. Test Conf.*, Atlantic City, NJ, 2000, pp. 273–282.

[4] ——, "Diagnosis for scan-based BIST: Reaching deep into the signatures," in *Proc. Design, Automation and Test Europe*, Munich, Germany, 2001, pp. 102–109.

[5] ——, "Cost-effective deterministic partitioning for rapid diagnosis in scan-based BIST," *IEEE Des. Test. Comput.*, vol. 19, no. 1, pp. 42–53, 2002.

[6] C. C. Chen and F. K. Hwang, "Detecting and locating electrical shorts using group testing," *IEEE Trans. Circuits Syst.*, vol. 36, no. 8, pp. 1113–1116, Aug. 1989.

[7] R. Dorfman, "The detection of defective members of large populations," *Ann. Math. Stat.*, vol. 14, pp. 436–440, 1943.

[8] D. Du and F. K. Hwang, *Combinatorial Group Testing and Its Applications*, 1st ed. Singapore: World Scientific, 1994.

[9] D. Z. Du, G.-L. Xue, S.-Z. Sun, and S.-W. Cheng, "Modifications of competitive group testing," *SIAM J. Comput.*, vol. 23, no. 1, pp. 82–96, 1994.

[10] M. R. Garey, D. S. Johnson, and H. C. So, "An application of graph coloring to printer circuit testing," *IEEE Trans. Circuits Syst.*, vol. CS-23, no. 10, pp. 591–599, Oct. 1976.

[11] J. Ghosh-Dastidar, D. Das, and N. A. Touba, "Fault diagnosis in scan-based BIST using both time and space information," in *Proc. Int. Test Conf.*, Atlantic City, NJ, 1999, pp. 95–102.

[12] J. Ghosh-Dastidar and N. Touba, "A rapid and scalable diagnosis scheme for BIST environments with a large number of scan chains," in *Proc. Very Large Scale Integration (VLSI) Test Symp.*, Montreal, Canada, 2000, pp. 79–85.

[13] F. K. Hwang, "A method for detecting all defective members in a population by group testing," *J. Amer. Stat. Assoc.*, vol. 67, pp. 605–608, 1972.

[14] H. K. Lee and D. S. Ha, "HOPE: An efficient parallel fault simulator by group testing," in *Proc. IEEE Design Automation Conf.*, Anaheim, CA, 1992, pp. 605–608.

[15] ——, "On the generation of test patterns for combinational circuits," Dept. Elect. Eng., Virginia Polytech. Inst. and State Univ., Blacksburg, Tech. Rep. 12-93.

[16] C. H. Li, "A sequential method for screening experimental variables," *J. Amer. Stat. Assoc.*, vol. 57, pp. 455–477, 1962.

[17] C. Liu, K. Chakrabarty, and M. Goessel, "An interval-based diagnosis scheme for identifying failing vectors in a scan-BIST environment," in *Proc. Design, Automation and Test Europe*, Paris, France, 2002, pp. 382–386.

[18] S. Mitra and K. S. Kim, "XMAX: X-tolerant architecture for MAXimal test compression," in *Proc. Int. Conf. Computer Design*, San Jose, CA, 2003, pp. 326–330.

[19] W. H. McAnney and J. Savir, "There is information in faulty signatures," in *Proc. Int. Test Conf.*, Washington, DC, 1987, pp. 630–636.

[20] J. Rajski and J. Tyszer, "Fault diagnosis in scan-based BIST," in *Proc. Int. Test Conf.*, Washington, DC, 1997, pp. 894–902.

[21] ——, "Diagnosis of scan cells in BIST environment," *IEEE Trans. Comput.*, vol. 48, no. 7, pp. 724–731, Jul. 1999.

[22] J. Savir, "Salvaging test windows in BIST diagnostics," *IEEE Trans. Comput.*, vol. 47, no. 4, pp. 486–491, Apr. 1998.

[23] J. K. Skilling, "Method of electrical short testing and the like," U.S. Patent 4 342 959, Aug. 3, 1982.

[24] P. Wohl, J. A. Waicukauski, S. Patel, and G. Matson, "Effective diagnostics through interval unloads in a BIST environment," in *Proc. ACM/IEEE Design Automation Conf.*, New Orleans, LA, 2002, pp. 249–254.

[25] Y. Wu and S. Adham, "BIST fault diagnosis in scan-based VLSI environments," in *Proc. Int. Test Conf.*, Washington, DC, 1996, pp. 48–57.

Andrew B. Kahng (A'89–M'03) received the A.B. degree in applied mathematics from Harvard University, Cambridge, MA, in 1986 and the M.S. and Ph.D. degrees in computer science from the University of California San Diego, La Jolla, CA, in 1989.

He is a Professor of CSE and ECE at University of California San Diego. He has published over 200 papers in the very large scale integrated (VLSI) computer-aided design (CAD) literature. From 1989 to 2000, he was a member of the UCLA computer science faculty. His research is mainly in the physical design and performance analysis of VLSI as well as the VLSI design manufacturing interface. His other research interests include combinatorial and graph algorithms, and large-scale heuristic global optimization. In 1997, he defined the physical design roadmap for the International Technology Roadmap for Semiconductors (ITRS), and since 2001, he has chaired the U.S. and international working groups for design technology for the ITRS. He has been active in the MARCO Gigascale Silicon Research Center since its inception. He was also the founding General Chair of the ACM/IEEE International Symposium on Physical Design and cofounded the ACM Workshop on System-Level Interconnect Planning.

Dr. Kahng is a recipient of three Best Paper awards and an NSF Young Investigator award.

Sherief Reda (S'01) received the B.Sc. and M.Sc. degrees in electrical and computer engineering from Ain Shams University, Cairo, Egypt, in 1998 and 2000, respectively. He is currently working toward the Ph.D. degree at the University of California San Diego, La Jolla, CA.

He has over 20 refereed publications in the areas of physical design, very large scale integrated (VLSI) test and diagnosis, combinatorial optimization, and computer-aided design (CAD) for deoxyribonucleic acid (DNA) chips.

Mr. Reda is the recipient of the Best Paper Award at DATE 2002 and the First Place Award at the ISPD 2005 placement contest.