On the Size and Generation of Minimal *N*-Detection Tests

Kalyana R. Kantipudi and Vishwani D. Agrawal
Auburn University
Department of Electrical and Computer Engineering
Auburn, AL 36849, USA
kantikr@auburn.edu, *vagrawal@eng.auburn.edu*

Abstract

The main result of this paper, proved as a theorem, is that a lower bound on the number of test vectors that detect each fault at least N times is N times the minimal test set size for $N = 1$. Tests with $N > 1$ have been reported to have a higher defect coverage and hence are of practical interest. We give an integer linear programming (ILP) algorithm for optimally minimizing a given test set for any given N; in general, the value of N can be separately specified for each fault. Results on benchmark circuits show that optimal N-detection tests are easier to find for circuits that are deep and the input cones of primary outputs have large overlap. However, for small depth circuits, where the primary input overlap between output cones is small or nonexistent, the minimization of the N-detection tests requires further investigation.

1. Introduction

Two aspects of testing motivate this work. The first is the usefulness of N-detection vectors and the second is the need to have a small number of combinational vectors. The N-detection vectors contain at least N tests for each modeled fault and are known to have a higher coverage of “real defects” than the conventional single-detection tests that may detect each fault only once. Since the N-detection tests are longer, their use directly translates into higher test time, especially for scan circuits, and therefore higher testing cost.

In this paper, we prove that a lower bound on the size of the N-detection vectors is N times the lower bound on the size of the single-detection vectors. We give an algorithm for finding the minimal or near-minimal N-detection vectors using the integer linear programming (ILP).

In Section 2, we give the background on single and multiple-detection tests. Section 3 gives an important theoretical result of this paper in the form of Theorem 2. Section 4 outlines the new ILP algorithm for the N-detection test minimization. Section 5 discusses how a test generation system can be constructed and illustrates its use. Section 6 gives results and discusses the strengths and weaknesses of the method, while specifying problems for future work.

2. Background

2.1. Minimal Single-Detection Tests

In 1987, Akers *et al.* [2] pointed out that a lower bound on the size of the single-detection test set is the size of the independent fault set (IFS). An IFS is the largest set of faults in which no two faults can be detected by the same vector. In the subsequent years, two methods have evolved for finding test sets whose size approaches the lower bound within practical limits.

2.1.1 Single-Fault ATPG

In the first approach, tests are generated by a conventional ATPG (automatic test pattern generation) program, which sequentially targets single faults followed by fault simulation. The tests so generated are then compacted either statically or dynamically. In static compaction, a pre-generated vector set is reduced without reducing the fault coverage. In dynamic compaction, additional vectors can be generated to provide better compaction of the test set. Much literature exists on these techniques and we cite a few representative references for an interested reader who wishes to explore further [4, 14, 15, 17, 24].

In static compaction the size of the compacted test set largely depends upon the pre-generated test set. Techniques like integer linear programming

(ILP), though expensive, have been used for maximum possible compaction [12, 16, 22]. However, an absolute minimal size according to the lower bound [2] may be guaranteed only when the initial vector set is exhaustive.

In dynamic compaction, it is realized that a fault has multiple tests and a conventional ATPG program arbitrarily selects one. Therefore, replacing a pre-generated test by a new one can provide better compaction. Dynamic compaction is capable of generating the smallest test set [15].

In an alternative approach, single-fault targets for ATPG are taken from an independent fault set (IFS) [2]. A separate test is essential for each fault in the IFS. However, a fault in the IFS has several tests and only a properly selected one will cover all other faults outside the IFS. The techniques for a proper test selection can be quite complex [23] and alternative heuristics generally compromise the minimality of test set [7, 25].

2.1.2 Concurrent ATPG

To detect many single-fault targets, two approaches have been used. In either case, an independence graph is first generated [2]. In this graph, each fault is a node and an edge between a pair of nodes indicates independence of the corresponding faults. In the first approach [26], a largest clique of the independence graph is first found. All tests for each node in that clique are generated and a minimal set is selected to cover all faults. The complexity of finding all tests and then solving the covering problem can be high.

In the other approach, known as independence fault collapsing [8, 9], the independence graph is collapsed into a clique of the largest size. Here two nodes can be collapsed together only if they are not connected by an edge. Next, a minimum set of tests (ideally a single vector) is found for the faults in each node of the collapsed graph. This procedure requires new ATPG algorithms that can simultaneously target a set of single faults, though single-fault ATPG has been used to solve this problem [1, 8].

2.2. *N*-Detection Tests

A significant body of experimental and theoretical work exists to show that stuck-at fault tests that detect each fault multiple times have a higher defect coverage and hence result in reduced defect level [3, 5, 6, 11, 21].

A recent paper [20] gives a method of generating multiple detection tests. A greedy heuristic is used first in random phase and then in a deterministic phase. The authors observed that for c432 circuit their 15-detection test contained 505 vectors, a number greater than 15 times the single-detection test set size. For c6288, the 15-detection test set size was 144, while the single-detection required 14 vectors (minimum reported is 12 vectors [15]).

In this paper, we show that the minimum size of the N-detection test is N times the minimum size of the single-detection test. We also show that the minimum or closer to minimum size of N-detection test may be easier to achieve for larger values of N.

3. Minimality of *N*-Detection Tests

Our discussion is based on the independence graph defined in Subsection 2.1.2. We quote the following result from Akers *et al.* [2].

Theorem 1: *The size of the largest clique in the independence graph is a lower bound on the single-detection test set size.*

Our new result on the N-detection tests is:

Theorem 2: *A lower bound on the size of the N-detection test set is N times the size of the largest clique in the independence graph.*

Proof: Consider the faults that form a maximal clique of the independence graph. This is an independent fault set (IFS). Suppose we generate N test vectors for one fault from the IFS. These vectors cannot detect any other fault in the IFS. For the second fault we therefore generate N new test vectors, which will neither detect the first fault nor any other fault in the IFS. Thus, the N-detection test set for the two faults contains $2N$ vectors. To detect all faults in the IFS N times we must apply this procedure independently to each fault, thus producing N vectors each time. ■

Sometimes, test sets of the sizes equal to the lower bounds of Theorems 1 and 2 can be found. However, in general, these are only lower bounds and the actual test set size can be larger.

4. *N*-Detection Test Minimization by Integer Linear Programming (ILP)

Suppose we have a set of k vectors that detects every fault at least N times. We use diagnostic fault simulation, i.e., fault simulation without fault dropping, to identify the vector subset T_j that detects a fault j, for all j. We assign an integer-valued variable $t_i \epsilon [0, 1]$ to ith vector such that $t_i = 1$ means that ith vector should be included in the minimal vector set and $t_i = 0$ means that ith vector should be discarded. The problem of finding the minimal N-detection set then reduces to assigning values to t_i's so as to

$$minimize \sum_{i=1}^{k} t_i \tag{1}$$

under following constraints:

$$\sum_{t_i \epsilon \{T_j\}} t_i \geq N_j, \forall\ faults\ j \tag{2}$$

where N_j is the multiplicity of detection for the jth fault. In general, N_j can be selected for individual faults based on some criticality criteria or on the capability of the initial vector set. For simplicity, we have assumed all N_j's to be equal. An integer linear program (ILP) solver [13] can then find the [0,1] values of the variables $\{t_i\}$ that define a minimal N-detection vector set. For $N = 1$, the ILP produces the minimal single-detection test set [12, 16, 22]. Integer programming has also been applied to sequential circuit test minimization [10].

Theorem 3: *When the minimization is performed over an exhaustive set of vectors of a combinational circuit, any ILP solution that satisfies expressions (1) and (2) is a minimum N-detection test.*

Example: Table 1 shows the ILP solution for the 74181 four-bit ALU circuit. The circuit has 14 primary inputs and therefore the exhaustive set contains $2^{14} = 16,284$ vectors. Diagnostic fault simulation of these vectors by Hope [19] required 14.3 s on a Sun Ultra 5 with 256MB RAM. Also shown in the table are 2,370 vectors generated by Atalanta [18] to detect every fault 10 times. This circuit has a collapsed set of 237 detectable faults and 10 vectors were generated for each fault. Diagnostic simulation of these 2,370 vectors by Hope required 2.3 s.

Notice that in Table 1 no solution is found for the Atalanta vectors for $N = 40$. This is because the vector set generated by Atalanta had some repeated vectors. When those were removed, some faults were left with less than forty detections. We also observe that the optimized Atalanta vector set starts to diverge from the lower bound, i.e., $12N$, for $N \geq 9$. This is because not all test vectors of each fault are included in the initial set and the ILP may not find the right vector. That is why Theorem 3 does not guarantee optimality in this case.

5. Derivation of *N*-Detection Tests

We generate an unoptimized M-detection test set by an ATPG program, where $M \geq N$. In our case, this was done using Atalanta [18]. This program generates a single-detection test set. Interestingly, repeated runs of the program produce different test sets. This is because different random vectors (due to a different seed in the random number generator) are used each time to cover the easy to detect faults. In most cases, by combining M single-detection test sets we get the required set.

Table 1. N-detection tests for 74181 ALU.

N	16,384 exhaustive vect.		2,370 Atalanta vect.	
	Minimized vectors	ILP CPU s	Minimized vectors	ILP CPU s
1	12	87.47	12	5.19
2	24	63.09	24	8.23
3	36	70.56	36	5.53
4	48	72.12	48	6.53
5	60	65.06	60	5.69
8	96	71.01	96	5.46
9	108	88.01	109	6.24
10	120	68.82	122	5.32
20	240	79.56	262	5.93
40	480	66.08	-	-

A simple analysis of vectors is used to remove any repeated vectors. This is important for a correct result from the ILP when $N > 1$. Next, a fault simulator (Hope [19] in our case) is used for diagnostic fault simulation. The fault simulator determines the vector set $\{T_j\}$ for every fault j. If $|\{T_j\}| < N$ for any fault, then additional vectors are obtained for that fault.

Using the fault simulation data, the ILP constraints (2) are generated and a solver [13] determines the [0,1] integer variables t_i. Results show that for small values of N (i.e., N close to 1), M should be several times N to obtain the minimal or a near-minimal (for very large circuits) test set. However, for large N, $M \approx N$ may some times provide a near minimal N-detection test set.

In general, a suitable value for M is not known. We used an iterative procedure to study the effect this value may have. Tests were generated for the c432 circuit, which is known to have a minimal single-detection test set of 27 vectors [15]. In the first iteration, a single-detection set of 70 vectors generated by Atalanta [18] was used. The ILP [13] reduced this set to 49 vectors for $N = 1$ but no sets were generated for $N \geq 2$ due to insufficient number of detections for some faults. To enhance the vectors, we generated extra tests for 25 faults that were detected last in the 70-vector set. The 25 vectors so generated had several don't care bits, which were enumerated to create 10 vectors from each. Thus, $25 \times 10 + 70 = 320$ vectors were obtained. After removing repeated vectors this set re-

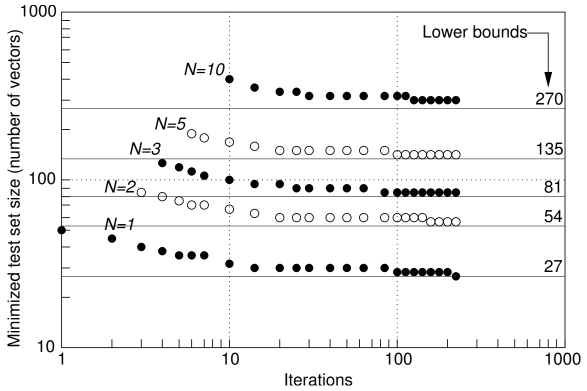


Figure 1. Sizes of N-detection test sets for c432 as a function of iterations.

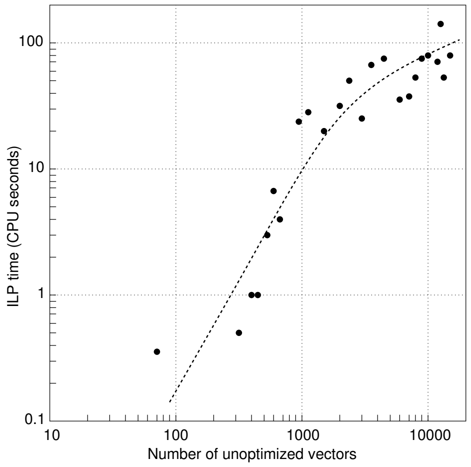


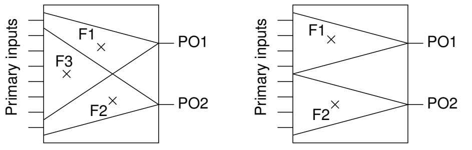
Figure 2. ILP CPU time versus number of unoptimized vectors for c432.

duced to 317 vectors. Subsequent iterations added a new single-detection set generated by Atalanta and then removed any repeated vectors. Figure 1 shows the N-detection test set sizes obtained from ILP as a function of iterations. The 1-detection set converges to the lower bound of 27 only at 225th iteration when the number of unoptimized vectors has reached 14,882. Test set sizes for $N=$ 2, 3, 5 and 10 are 55, 83, 140 and 283, respectively. The corresponding lower bounds are 54, 81, 135 and 270, respectively.

Figure 2 shows the Sun Ultra-5 CPU time for the ILP [13] as a function of iterations. Although the number of vectors increases linearly, a sub-linear increase in the CPU time is observed. The value of N does not seem to affect the run time of ILP.

6. More Results

Table 2 shows results obtained for several ISCAS'85 benchmark circuits. Test sets for 1, 2, 3 and 5 detections were generated. Iterative procedure of the last section was used. Absolute minimal test sets were obtained for c499 (6 iterations), c1355 (7 iterations) and c1908 (12 iterations). The vector sets for c432 (225 iterations) were minimal only for $N = 1$ and 2. All CPU times are for Sun Ultra-5, except two cases marked with asterisk (*) that were run on Sun Ultra-10. The times for $N \geq 2$ were not significantly different from these times, which are for $N = 1$.



(a) Overlapping cones (type I). (b) Non−overlapping cones (type II).

Figure 3. Circuit classification based upon don't cares in test vectors.

Circuits c880 and c6288 are interesting cases. For example, consider the c880 result in Table 2, which is for 30 iterations. The ILP stopped on a CPU time limit and with a non-optimal solution. We observe that the test sets for larger N are closer to optimum. For example, the $N = 1$ set is 92% larger than the optimum but the $N = 5$ set is only 62% larger.

The last three circuits and c880 in Table 2 expose a new problem. In all cases the the time required for ILP to converge to the optimum solution was large. To understand the difficulty, let us examine the illustrations of Figure 3. The first illustration, termed as type I circuit, has a large overlap between the input cones of the primary outputs PO1 and PO2. Note that the ILP minimization assumes that vectors are fully specified, i.e., have no don't care bits. Typically, each fault here has few tests and the vectors have few don't cares. Since a larger subset of the vectors for faults is included in the initial vector set, the ILP can select common vectors for faults if they exist. The four-bit ALU (see Table 1) and three circuits in Table 2 (c499, c1355 and c1908) are type I.

A type II circuit is characterized by non-overlapping output cones. In Figure 3(b), faults F1 and F2 each has a large number of test vectors. We observe two things. First, common test vectors for F1 and F2 always exist. In these vectors the top part is customized to detect F1 and the bottom part, for F2. Second, a test just for F1 essentially has many don't care bits. If we generate only a few vectors by randomly filling those bits, it is quite unlikely that we will create a test that also detects F2. Similar argument holds for tests of F2 being able to detect F1. It is for this reason that ILP is not

Table 2. N-detection test set sizes minimized by ILP.

Circuit name	Unopt. vectors	ILP CPU s	Single-detection		2-detection		3-detection		5-detection	
			Lower bound	Test set size	Lower bound	Test set size	Lower bound	Test set size	Lower bound	Test set size
c432	14822	82.3	27	27	54	55	81	83	135	140
c499	397	5.3	52	52	104	104	156	156	260	260
c880	3042	306.8	13	25	26	44	39	63	65	105
c1355	755	16.7	84	84	168	168	252	252	420	420
c1908	2088	97.0	106	106	212	212	318	318	530	530
c2670	8767	1568.6*	44	71	88	145	132	224	220	391
c6288	243	519.7	6	18	12	27	18	37	30	57
c7552	2156	1530.0*	65	148	130	298	195	468	325	841

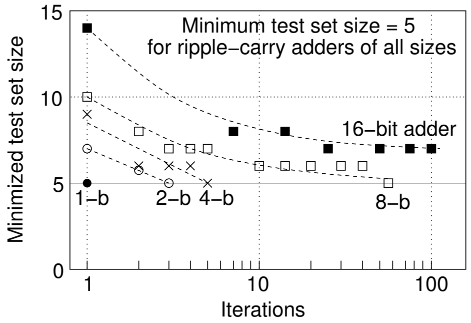


Figure 4. Iterative minimization of single-detection tests for ripple-carry adders.

Table 3. Comparing 15-detection tests.

Circuit name	ILP [this paper]		Heuristic [20]		L.B.
	Vect.	CPU s	Vect.	CPU s	[15]
c432	430	444.8	505	292.1	405
c499	780	24.9	793	153.2	780
c880	321	521.4	338	229.6	195
c1355	1260	52.1	1274	5674.6	1260
c1908	1590	191.0	1648	1563.9	1590
c2670	1248	607.8*	962	9357.6	660
c3540	1411	1223.7	-	-	1200
c5315	924	1368.4*	-	-	555
c6288	134	1206.3	144	1813.8	90
c7552	2370	346.1**	-	-	975

able to select a common test. The circuit c880 and the last three circuits in Table 2 are type II. The circuit c432 may be a borderline case. We were able to get the optimal result for $N = 1$ and 2 by generating a large number (225) of vectors for each fault. Figure 1 shows these 225 iterations amounting to 14,822 vectors shown in Table 2. Such a brute-force method appears unsuitable for c880, c2670, c6288 and c7552.

To investigate further, we generated N-detection tests for a ripple-carry circuit, constructed as an array of identical full-adders. Minimum number of vectors to detect all gate-level single stuck-at faults is 5, irrespective of the size of the adder. Figure 4 shows the minimized vector set size obtained by ILP as a function of iterations. An iteration here means that an additional vector set obtained from Atalanta [18] has been included. The test set size rapidly converges to 5 for 1-bit adder (1 iteration), 2-bit adder (3 iterations) and 4-bit adder (5 iterations). The 8-bit adder required 55 iterations for the optimal test set. For 16-bit adder, the convergence became asymptotic, test set size being 7 even after 100 iterations. Clearly, small adders behave as type I circuits while large ones are type II.

Table 3 compares our ILP result of 15-detection tests with those from a heuristic approach by Lee *et al.* [20] and a lower bound. This lower bound (L.B.) is obtained upon multiplying the theoretical minimum (not always realized [15]) single-detection test set size by 15. CPU times (ILP and [20]) are for Sun Ultra-5, except those with * (Ultra-10) and ** (Sun Fire-280R-900MHz-Dual-Processor). ILP technique does better than the heuristic method though further improvement is possible.

7. Conclusion

The theoretical lower bound on the N-detection tests is useful in assessing the minimality of such tests. Integer linear programming (ILP) is found to be an effective method for minimizing tests. The ILP technique, though used in the past for single-detection tests, has never been applied to N-detection tests. The formulation of the ILP in Section 4, also allows a custom selection of multiplicity of detection for individual faults. This may be useful if certain faults are found to play critical roles in defect detection. Finally, we identify problems specific to minimizing the N-detection tests for deep (type I with large number of primary inputs) and shallow (type II having primary outputs with non-overlapping input cones) circuits.

References

[1] V. D. Agrawal and A. S. Doshi, "Concurrent Test Generation," in *Proc. 14th IEEE Asian Test Symp.*, Dec. 2005.

[2] S. B. Akers, C. Joseph, and B. Krishnamurthy, "On the Role of Independent Fault Sets in the Generation of Minimal Test Sets," in *Proc. International Test Conf.*, 1987, pp. 1100–1107.

[3] M. E. Amyeen, S. Venkataraman, A. Ojha, and S. Lee, "Evaluation of the Quality of N-Detect Scan ATPG Patterns on a Processor," in *Proc. International Test Conf.*, 2004, pp. 669–678.

[4] B. Ayari and B. Kaminska, "A New Dynamic Test Vector Compaction for Automatic Test Pattern Generation," *IEEE Trans. Computer-Aided Design*, vol. 13, no. 3, pp. 353–358, mar 1994.

[5] B. Benware, C. Schuermyer, S. Ranganathan, R. Madge, P. Krishnamurthy, N. Tamarapali, K.-H. Tsai, and J. Rajski, "Impact of Multiple-Detect Test Patterns on Product Quality," in *Proc. International Test Conf.*, 2003, pp. 1031–1040.

[6] R. D. Blanton, K. N. Dwarakanath, and A. B. Shah, "Analyzing the Effectiveness of Multiple-Detect Test Sets," in *Proc. International Test Conf.*, 2003, pp. 876–885.

[7] J.-S. Chang and C.-S. Lin, "Test Set Compaction for Combinational Circuits," *IEEE Trans. on CAD*, vol. 14, no. 11, pp. 1370–1378, Nov. 1995.

[8] A. S. Doshi, "Independence Fault Collapsing and Concurrent Test Generation," Master's thesis, Auburn University, Dept. of ECE, Auburn, Alabama, 2005. *In preparation.*

[9] A. S. Doshi and V. D. Agrawal, "Independence Fault Collapsing," in *Proc. 9th VLSI Design and Test Symp.*, Aug. 2005, pp. 357–364.

[10] P. Drineas and Y. Makris, "Independent Test Sequence Compaction through Integer Programming," in *Proc. International Conf. Computer Design*, 2003, pp. 380–386.

[11] J. Dworak, J. D. Wicker, S. Lee, M. R. Grimaila, M. R. Mercer, K. M. Butler, B. Stewart, and L.-C. Wang, "Defect-Oriented Testing and Defective-Part-Level Prediction," *IEEE Design & Test of Computers*, pp. 31–41, Jan.-Feb. 2001.

[12] P. F. Flores, H. C. Neto, and J. P. Marques-Silva, "An Exact Solution to the Minimum Size Test Pattern Problem," *ACM Trans. Design Automation of Electronic Systems*, vol. 6, no. 4, pp. 629–644, oct 2001.

[13] R. Fourer, D. M. Gay, and B. W. Kernighan, *AMPL: A Modeling Language for Mathematical Programming.* South San Francisco, California: The Scientific Press, 1993.

[14] P. Goel and B. C. Rosales, "Test Generation and Dynamic Compaction of Tests," in *Proc. International Test Conf.*, Oct. 1979, pp. 189–192.

[15] I. Hamzaoglu and J. H. Patel, "Test Set Compaction Algorithms for Combinational Circuits," *IEEE Trans. on CAD*, vol. 19, no. 8, pp. 957–963, Aug. 2000.

[16] D. S. Hochbaum, "An Optimal Test Compression Procedure for Combinational Circuits," *IEEE Trans. Computer-Aided Design*, vol. 15, no. 10, pp. 1294–1299, oct 1996.

[17] S. Kajihara, I. Pomeranz, K. Kinoshita, and S. M. Reddy, "Cost Effective Generation of Minimal Test Sets for Stuck at Faults in Combinational Logic Circuits," *IEEE Trans. on CAD*, vol. 14, no. 12, pp. 1496–1504, Dec. 1995.

[18] H. K. Lee and D. S. Ha, "On the Generation of Test Patterns for Combinational Circuits," Tech. Report 12-93, Dept. of Elec. Eng., Virginia Poly. Inst. and St. Univ., Blacksburg, Virginia, 1993.

[19] H. K. Lee and D. S. Ha, "HOPE: An Efficient Parallel Fault Simulator for Synchronous Sequential Circuits," *IEEE Trans. on CAD*, vol. 15, no. 9, pp. 1048–1058, Sept. 1996.

[20] S. Lee, B. Cobb, J. Dworak, M. R. Grimaila, and M. R. Mercer, "A New ATPG Algorithm to Limit Test Set Size and Achieve Multiple Detections of all Faults," in *Proc. Design Automation and Test in Europe Conf.*, 2002, pp. 268–274.

[21] M. R. Grimaila and S. Lee and J. Dworak and K. M. Butler and B. Stewart and H. Balachandran and B. Houchins and V. Mathur and J. Park and L.-C. Wang and M. R. Mercer, "REDO–Random Excitation and Deterministic Observation – First Commercial Experiment," in *Proc. IEEE VLSI Test Symp.*, 1999, pp. 268–274.

[22] J. P. Marques-Silva, "Integer Programming Models for Optimization Problems in Test Generation," in *Proc. IEEE Asia-South Pacific Design Automation Conf.*, 1998, pp. 481–487.

[23] Y. Matsunaga, "MINT–An Exact Algorithm for Finding Minimum Test Sets," *IEICE Trans. Fundamentals*, vol. E76-A, no. 10, pp. 1652–1658, oct 1993.

[24] I. Pomeranz, L. N. Reddy, and S. M. Reddy, "COMPACTEST: A Method to Generate Compact Test Sets for Combinational Circuits," *IEEE Trans. Computer-Aided Design*, vol. 12, no. 7, pp. 1040–1049, July 1993.

[25] G.-J. Tromp, "Minimal Test Sets for Combinational Circuits," in *Proc. International Test Conf.*, 1991, pp. 204–209.

[26] J. C. Wang and E. P. Stabler, "Collective Test Generation and Test Set Compaction," in *Proc. IEEE Int. Symp. Circuits and Systems*, volume 3, Apr.-May 1995, pp. 2008–2011.