2016 IEEE 25th North Atlantic Test Workshop

Failures Guide Probabilistic Search for a Hard-to-Find Test

Muralidharan Venkatasubramanian
Ph.D Candidate
Department of Electrical and Computer Engineering,
Auburn University,
Auburn, Alabama 36849
Email: vmn0001@auburn.edu

Vishwani D. Agrawal
James J. Danaher Professor
Department of Electrical and Computer Engineering,
Auburn University,
Auburn, Alabama 36849
Email: vagrawal@eng.auburn.edu

***Abstract*—Previous research has firmly established the NP-complete nature of the fault detection problem. Various algorithms and techniques have been developed to tackle the worst case computation time for test generation due to the increasing complexity of digital circuits. The development of these techniques over the past 50 years has improved a lot of commercial EDA tools. The efficiency of these tools coupled with the considerable long research period has led to claims that ATPG research may have reached itsr maturity. However, whenever traditional algorithms started hitting their limits, studies in non-traditional techniques have improved test coverage. This paper proposes a foray in the realm of quantum computing in order to find tests for hard to detect stuck-at faults. We were encouraged by a reduced complexity quantum search for an element in an unsorted database proposed by Grover and the similarity of that problem to the search for a test in the vector space. Thus, it may be possible to find tests for the last few difficult to detect stuck-at faults much faster than other contemporary algorithms. This paper presents results of various benchmark circuits by comparing the time taken for our algorithm to run with FastScan's ATPG tool. It is shown that for a difficult to detect fault with limited backtracking, FastScan struggles to find a test while our algorithm is able to efficiently find the test in reasonable time.**

***Keywords*: Database search, digital test, quantum computing, test generation, probabilistic correlation.**

I. Introduction

The VLSI Testing problem can be colloquially defined as "Given a fault, find a test." A test to detect a fault present in a circuit with n primary inputs has to be present within 2^n possible combinations. Hence, the problem of VLSI Testing can be rephrased as a database search problem.

The Testing problem has been mathematically proven to be NP-complete [11], [15], [26] leading to an increase in test generation time with an increase in circuit size and complexity. Various test techniques have been designed and implemented to improve the test search time. Some approaches were algorithmic [10], [12], [22]; some were functional tests [2], [21]; and some used various types of weighted random test generators [1], [25]. Different variations of genetic algorithms were also implemented [8], [18], [20], [23], [24]. Alternative techniques like test generation using spectral information [33], anti-random test pattern generation [17] and energy minimization of neural networks [7] amongst others also had their share of successes.

The origin of the present research is our realization that correlations between the bits of a vector might play a significant role in the success of random test generation. This aspect has not been considered before [30].

One of our previous papers [32] drew a comparison between the areas of VLSI Testing and database search. It was shown that the various Automatic Test Pattern Generation (ATPG) algorithms were in essence various interpretations of different types of database search algorithms like depth-first search, breadth-first search, tree traversal, branch and bound etc. Because of the existence of this correlation, it was postulated that an ATPG algorithm based on the fastest database search technique would be very efficient in finding tests for faults, specifically difficult to detect stuck-at faults.

We had previously proposed an unique algorithm attempting to implement Grover's Algorithm for database search in the area of VLSI Testing [31]. Grover's algorithm searches an unordered database of N items to find a specified item. It is a quantum search algorithm which is quadratically faster than any other classical search algorithm [13], [14]. While classical algorithms can have complexity ranging from O(N) to O($\log N$), the complexity of Grover's algorithm is shown to be O($\sqrt{N}$) [4], [6].

The authors in [28] and [29] extended the work of Chakradhar *et al.* [7] and attempted to apply Grover's algorithm to find test vectors for VLSI circuits by reclassifying the testing problem as an energy minimization problem of a neural network. Our method of formulating Grover's algorithm as an ATPG algorithm is quite different.

It is known that there is a strong correlation between the parallel input bits of test vectors applied at the primary inputs (PIs) [2]. Contemporary testing algorithms extract new test vectors based on properties similar to previous successes or try to arrive at the solution in a deterministic manner by trying to propagate the fault to the primary outputs (POs). All these algorithms ignore the failed test vectors and hence are throwing away lots of potentially useful information which can help deduce the solution faster.

We attempt to answer this question: **"How to design a new test algorithm which utilizes the information from failed attempts effectively?"**

Our previous work [31] postulates that by moving away

978-1-4673-8949-5/16 $31.00 © 2016 IEEE
DOI 10.1109/NATW.2016.18

from the test vectors similar in properties to these failed test vectors, we will arrive at the solution in fewer iterations than current algorithms. A growing interest in quantum computing has spurred investigations in the areas of probabilistic computing algorithms [3], [5]. Since a quantum system can have no information loss, utilization of both successful and failed test vector information to generate new vectors can lead us to hypothesize that our algorithm contains the essence of Grover's quantum search algorithm. This differentiates our implementation from all the other algorithms as they do not utilize the failed vector information to generate new tests.

This paper is divided into five more sections. Section II provides a brief overview of the principle concept behind our algorithm, Section III highlights how our simulations were conducted and what sort of software and tools were used, Section IV discusses and elaborates the results obtained from our simulations along with graphs and figures. Sections V and VI highlight our future direction of research which can be undertaken and the conclusion.

II. Algorithm Overview

It is colloquially understood that there are a lot more failed test vectors generated as compared to successful ones when attempting to successfully test a fault (especially a hard to detect fault). Our algorithm deduces the correct test vector by learning the properties of failed test vectors and avoiding their properties in subsequent iterations. The algorithm's search is aided by classifying all the test vectors in the vector space in three broad categories [31]:

- **Activation vectors:** These vectors activate a desired stuck-at fault on the fault line of a circuit. However, not all vectors may propagate the fault to POs. For example, if a line in a circuit is stuck-at-1, these vectors will activate that fault. However, it is possible that the fault may never get propagated to the PO because no path is sensitized.
- **Propagation vectors:** These vectors will sensitize the path to POs and propagate a desired line's fault to the POs. In other words, if any stuck-at fault is placed on a particular line, the vectors in this category will propagate both fault types to the POs.
- **Failed vectors:** These vectors neither activate the fault site to the desired stuck-at fault nor sensitize the path to propagate the vectors to the POs. These vectors only provide information on what to avoid and bound our search in the subsets of "activation and propagation regions" of the vector space.

As Fig. 1 illustrates, the correct test vector lies at the intersection of the activation and propagation vector region. Hence, the ideal test vector will not only activate the desired stuck-at fault but sensitize a path to propagate it to the POs as well. Since hard to detect faults may have only one or two such unique vectors, it might be easier to find vectors which can either activate the fault but do not sensitize a path or vice versa. The partial useful information of these vectors can be used to hone into the correct solution steadily. The failed vectors restrict our search in the region of "partial desirability"

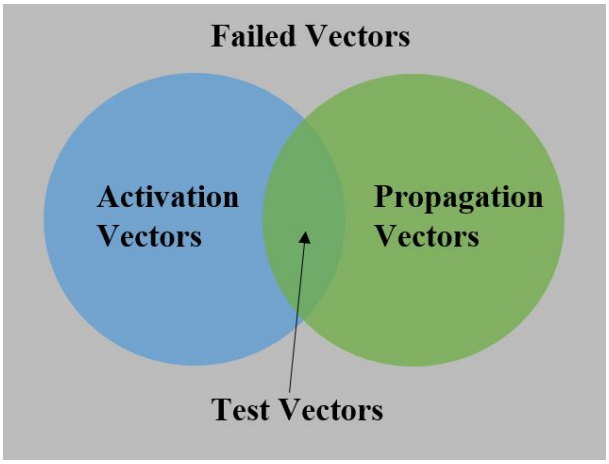

Fig. 1. All test vectors in the vector space classified in appropriate categories for a given stuck-at fault [31].

and hence act as a fence so that we do not search outside of those constraints.

The concept of our algorithm can be implemented in a number of ways. In [31], we skew the independent weighted probabilities of the PIs in such a manner so that the search moves away from the failed test vector region. In other words, if a certain probability weight at the PI generates a logic value (1 or 0) that neither activates the fault nor sensitizes a path to the POs, it is best to invert the probability weight before generating a new value for that line. This method is repeated until the search enters the region of activation vectors and/or propagation vectors (colloquially, region of "partial desirability").

Once the search enters either of these regions, it is in our best interest that the search does not deviate away from this subset of vector space. Henceforth, the weighted probability of the failed vectors is used to modify the weighted probability of the vectors in the partial desirability region. These modifications are made in smaller increments in order to make smaller steps towards the correct test vector.

We have explained the detailed working implementation of our algorithm in our previous paper and hence will not be reproduced again. Readers interested in understanding the nuances and intricacies of our algorithm are encouraged to read [31] to get a thorough understanding of our work.

However, the method explained above is just one of many different implementation techniques. Another technique currently under investigation is to utilize the correlation relationship among the parallel bits of the primary inputs in order for the search to self-prune itself by avoiding vectors with weak correlation and generating more highly correlated vectors.

III. Simulation Setup

The benchmark circuits used for simulations were Register Transistor Level (RTL) models written in Verilog. The faults for each benchmark circuit were chosen such that ideally only one test could detect the fault. The fault simulator used was FastScan [9] from Mentor Graphics. The random pattern

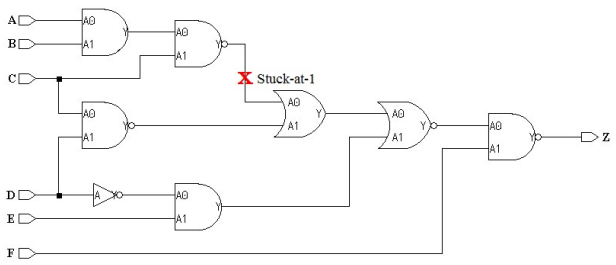

Fig. 2. Gate level design of the test circuit. 'X' marks the stuck-at-1 fault for which the test vector was to be found.

TABLE I. COMPARISON OF ITERATION SPEED FOR DIFFERENT BENCHMARK CIRCUITS USING RANDOM SEARCH, GROVER'S IDEAL QUANTUM SEARCH AND OUR WORK [31].

Circuit	Number of primary inputs $\#PI$	Search space size, $N = 2^{\#PI}$	Average search steps to success		
			Random ATPG $\sim N/2$	Grover's alg. [13], [14] $(\sqrt{N})$	Proposed ATPG [31]
Figure 2	6	64	34	8	14
ALU control	7	128	62	11	15
Decoder	8	256	133	16	24
Ones counter	9	512	288	23	76
CAVL coder	10	1024	500	32	132

generator built into FastScan was used to emulate a random search of test vectors for testing the desired stuck-at fault in the circuits.

The proposed algorithm was written and coded in MATLAB [19] provided by MathWorks. The test vectors extracted from MATLAB was sent to FastScan for verification i.e. to check if the vector can test the fault in the circuit. This was done using the fault simulator mode in FastScan.

IV. RESULTS AND DISCUSSION

Table I compares the average iteration time to find a test for a difficult to detect stuck-at fault in different benchmark circuits. Specifically, the comparison is between Grover's quantum search algorithm (ideal implementation), our algorithm's approach [31] and random search. It can be gleaned from Table I that quantum search can provide much needed gains with an increase in circuit size.

Figure 2 shows a circuit with a stuck-at-1 fault on a line. This circuit was used to test the algorithm's working. Out of the 64 possible test vectors, only one ("111111" on the PIs) can successfully test the stuck-at-1 fault on that line. Also, the fault has eight activation vectors ("111000, 111001, 111010, 111011, 111100, 111101, 111110, 111111") and four propagation vectors ("001111, 011111, 101111, 111111").

Figures 3-6 illustrate in detail the distribution of iterative search time needed to find a test vector for a stuck-at fault over a sample of 100 trial simulations. While both the algorithms have a normal distribution fit, a random search algorithm has iteration values over the entire vector space while our proposed

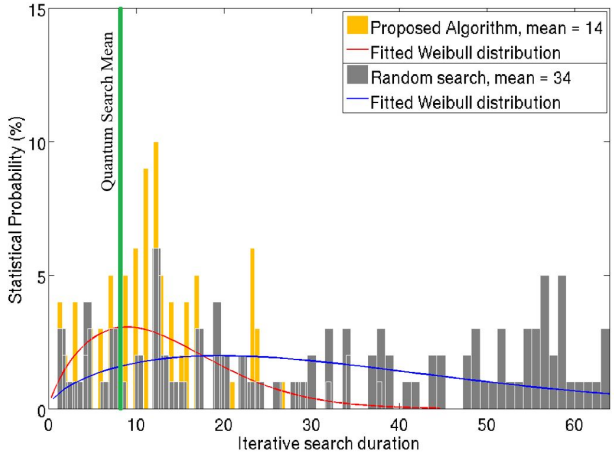

Fig. 3. 100 trials of iterative search for a test vector for a fault in the circuit of Fig. 2. Grey bars indicate percentage of times certain number of iterations were required. The light orange bars indicate similar percentage for the proposed algorithm [31]. The green bar indicates the theoretical mean = 8 of Grover's quantum search.

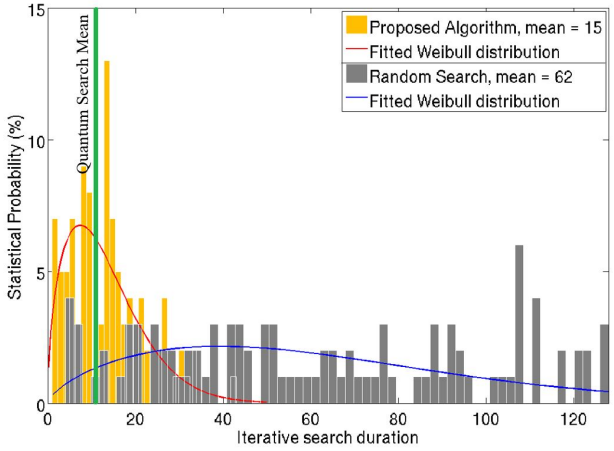

Fig. 4. 100 trials of iterative search for a test vector for a fault in *ALU control benchmark circuit*. Grey bars indicate percentage of times certain number of iterations were required. The light orange bars indicate similar percentage for the proposed algorithm [31]. The green bar indicates the theoretical mean = 11 of Grover's quantum search.

algorithm has a much tighter grouping closer to the ideal value of Grover's search algorithm.

Figure 3 shows the distribution for the test circuit shown in Fig 2. It is seen that on average, our algorithm takes 14 iterations to find a test which is quite close to the Grover's search number (8 iterations). Fastscan's random search on average takes 34 iterations to come to the same conclusion.

Figure 4 depicts the distribution for an ALU control benchmark circuit. This circuit has 7 PIs. On average, our algorithm takes 15 iterations to find a test while the ideal iteration number provided by Grover's algorithm is 11. Fastscan's random search on average takes 62 iterations to come to the same conclusion.

The results for an 8-input Decoder benchmark circuit is

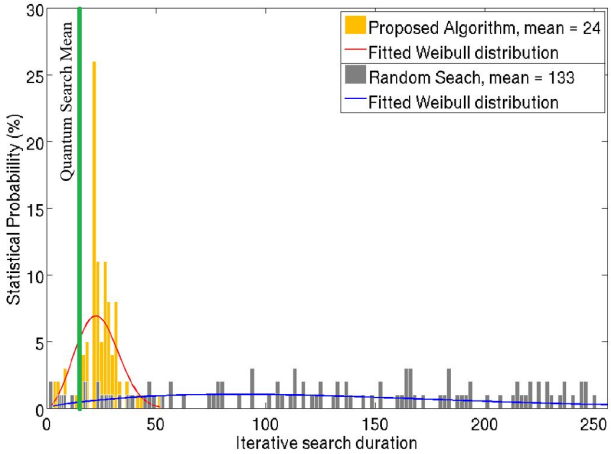

Fig. 5. 100 trials of iterative search for a test vector for a fault in *decoder benchmark circuit*. Grey bars indicate percentage of times certain number of iterations were required. The light orange bars indicate similar percentage for the proposed algorithm [31]. The green bar indicates the theoretical mean = 16 of Grover's quantum search.

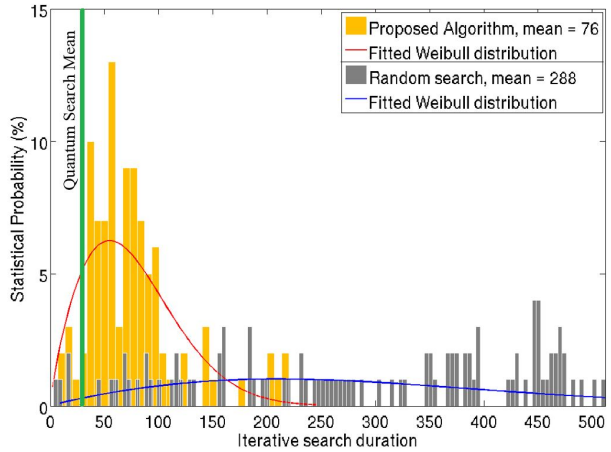

Fig. 6. 100 trials of iterative search for a test vector for a fault in *ones counter benchmark circuit*. Grey bars indicate percentage of times certain number of iterations were required. The light orange bars indicate similar percentage for the proposed algorithm [31]. The green bar indicates the theoretical mean = 23 of Grover's quantum search.

shown in Fig. 5. On average, our algorithm takes 15 iterations to find a test while the ideal iteration number provided by Grover's algorithm is 11. Fastscan's random search on average takes 133 iterations to come to the same conclusion.

In Fig. 6, the results are a bit more interesting. It depicts the iteration time distribution for a 9-input Ones Counter benchmark circuit. While our algorithm is able to find a test in 76 iterations on average, it is quite away from Grover's ideal 23 iterations albeit they are in the same order of magnitude. Fastscan's random search is able to find the test in 288 iterations on average which is one order of magnitude higher than our work. We are still investigating as to why there is such a big difference in the iteration time between our algorithm and Grover's algorithm. Initial examinations show that the stuck-at fault being tested has 87 "activation vectors" but only 4 "propagation vectors" with 1 test vector correctly testing the

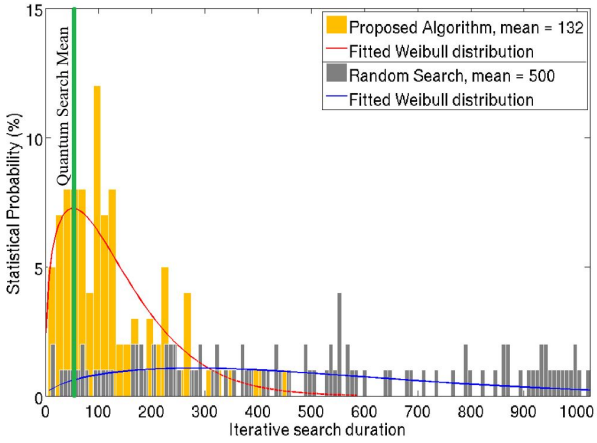

Fig. 7. 100 trials of iterative search for a test vector for a fault in *Context-adaptive variable-length coding (CAVLC) benchmark circuit*. Grey bars indicate percentage of times certain number of iterations were required. The light orange bars indicate similar percentage for the proposed algorithm [31]. The green bar indicates the theoretical mean = 32 of Grover's quantum search.

fault. We hypothesize that the skewed ratio between these two types of vectors added a need to search more comprehensively to find the correct test vector.

A similar trend continues in Fig. 7 as well whic represents the iteration time distribution for a 10-input Context-adaptive variable-length coding benchmark circuit. Our algorithm is able to find a solution in 132 iteratins while Grover's ideal iterations consists of 32 iterations. Further analysis of the vector regions showed that while there are 220 vectors in the "activation region", only 4 vectors exist in the "propagation region".

All in all, from the results shown above, we summarize that once the search enters the "region of partial desirability" (highlighted by activation vectors or propagation vectors), the search for the correct test accelerates very quickly and the algorithm is able to hone into the right test within a few iterations. We can further postulate that our self-learning algorithm iteration time would be have the same order of magnitude as Grover's quantum search algorithm.

V. Future Work

We are currently investigating the iteration time required for bigger and more complex circuits. As Fig. 8 shows, we expect that our algorithm will have the same order of magnitude for searching for test vectors as compared to Grover's algorithm although it might be a tad slower. However, it will be orders of magnitude faster than a random search with an increase in circuit size. Note that the quantity on x-axis is the number of primary inputs (PI) and therefore the search space is exponentially increasing as $2^{\#PI}$.

We are also trying to formalize our algorithm technique with some mathematical proof. Investigations into statistical mathematics unearthed a concept called *Mahalanobis Distance* [16]. It is a general idea that explains how many standard deviations away is a point P from the center of a multivariate probability distribution D of N random variables.

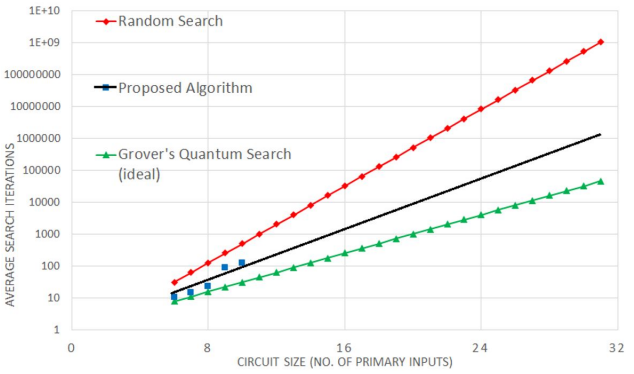

Fig. 8. Test vector search complexity: Average search iterations versus number of primary inputs (n) for random search (2^{n-1}), proposed algorithm, and Grover's quantum search $(2^{n/2})$ treated as a lower bound. Points show experimental data from Table I. Extrapolated proposed algorithm line demonstrates a trend closer to the lower bound.

If we assume the P to be the correct test vector for a fault and the distribution D to be the intersection of activation and propagation vectors, we can theoretically calculate how far is our present iteration from the right solution and the least number of steps needed to attain the result.

Secondly, we are setting up simulations to compare our algorithm with the ATPG in a commercial tool [9]. By targeting difficult to detect faults which either are not detected or take a very long time, we want to show our algorithm's superiority over currently implemented techniques.

The primary challenge in trying to compare our algorithm's efficiency with an ATPG is trying to decide the best way to show the comparison. Since our algorithm is written in MATLAB and lists efficiency as the no. of iterative steps towards success, comparing with Fastscan's ATPG algorithm which lists efficiency in seconds would be an apples to oranges comparison. It is not appropriate to compare the MATLAB simulation time to Fastscan simulation time because they are different software which utilize process resources differently.

We also plan to improve upon our algorithm to bring it close to the mathematical model of Grover's algorithm for random database search. Since the algorithm's core concept of utilizing failed test vector information to search for new vectors is abstract, different methods of implementation can be used to further improve the search speed. One such technique is to use correlations between the PI bits. By utilizing the bit correlation between the primary input lines, it is possible to move away from the undesirable correlation of the failed test vectors and quickly hone into the correct test vector. Finally, our attempt at defining the *oracle* of Grover's search [13], [14] for the ATPG problem will continue.

VI. Conclusion

This paper successfully extends the work described in [31] to more practical benchmark circuits. It has shown the resilience of our algorithm with increasing circuit complexity with promises of providing an even more detailed analysis of bigger benchmarks. The future direction of our work has been clearly highlighted in the previous section and the goal is to tackle them head on to demonstrate our algorithm's versatility. Research in quantum computing is slowly approaching a zenith with previously described algorithms like Shor's algorithm [27] and Grover's algorithm [13], [14] already being used as a test for the working of quantum computers. This algorithm was designed with an end objective to emulate the mathematical model of Grover's algorithm in a more practical manner for VLSI testing. More attempts will be made to tackle the small gap of iteration time so that future implementations align much closer to Grover's algorithm.

References

[1] V. D. Agrawal, "An Information Theoretic Approach to Digital Fault Testing," *IEEE Trans. Computers*, vol. 30, no. 8, pp. 582–587, 1981.

[2] S. B. Akers, "Universal Test Sets for Logic Networks," in *IEEE Conference Record of 13th Annual Symposium on Switching and Automata Theory*, 1972, pp. 177–184.

[3] D. Angluin and L. G. Valiant, "Fast Probabilistic Algorithms for Hamiltonian Circuits and Matchings," *Journal of Computer and System Sciences*, vol. 18, no. 2, pp. 155–193, 1979.

[4] C. H. Bennett, E. Bernstein, G. Brassard, and U. Vazirani, "Strengths and Weaknesses of Quantum Computing," *SIAM Journal on Computing*, vol. 26, no. 5, pp. 1510–1523, 1997.

[5] D. M. Blei, "Probabilistic Topic Models," *Communications of the ACM*, vol. 55, no. 4, pp. 77–84, 2012.

[6] M. Boyer, G. Brassard, P. Høyer, and A. Tapp, "Tight Bounds on Quantum Searching," *Fortschritte der Physik*, vol. 46, no. 4-5, pp. 493–505, 1998.

[7] S. T. Chakradhar, V. D. Agrawal, and S. G. Rothweiler, "A Transitive Closure Algorithm for Test Generation," *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, vol. 12, no. 7, pp. 1015–1028, 1993.

[8] F. Corno, P. Prinetto, M. Rebaudengo, and M. S. Reorda, "GATTO: A Genetic Algorithm for Automatic Test Pattern Generation for Large Synchronous Sequential Circuits," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 15, no. 8, pp. 991–1000, 1996.

[9] "TESSENT FastScan." Mentor Graphics. http://www.mentor.com/products/silicon-yield/products/fastscan/.

[10] H. Fujiwara and T. Shimono, "On the Acceleration of Test Generation Algorithms," *IEEE Trans. Computers*, vol. 32, no. 12, pp. 1137–1144, 1983.

[11] H. Fujiwara and S. Toida, "The Complexity of Fault Detection Problems for Combinational Logic Circuits," *IEEE Trans. Computers*, vol. 31, no. 6, pp. 555–560, 1982.

[12] P. Goel, "An Implicit Enumeration Algorithm to Generate Tests for Combinational Logic Circuits," *IEEE Trans. Computers*, vol. 30, no. 3, pp. 215–222, 1981.

[13] L. K. Grover, "A Fast Quantum Mechanical Algorithm for Database Search," in *Proc. 28th Annual ACM Symp. Theory of Computing*, 1996, pp. 212–219.

[14] L. K. Grover, "Quantum Mechanics Helps in Searching For a Needle in a Haystack," *Physical Review Letters*, vol. 79, no. 2, p. 325, 1997.

[15] O. H. Ibarra and S. Sahni, "Polynomially Complete Fault Detection Problems," *IEEE Trans. Computers*, vol. 24, no. 3, pp. 242–249, 1975.

[16] P. C. Mahalanobis, "On the Generalized Distance in Statistics," *Proceedings of the National Institute of Sciences (Calcutta)*, vol. 2, pp. 49–55, 1936.

[17] Y. K. Malaiya, "Antirandom Testing: Getting the Most Out of Black-Box Testing," in *Proc. Sixth IEEE International Symp. Software Reliability Engineering*, 1995, pp. 86–95.

[18] P. Mazumder and E. M. Rudnick, *Genetic Algorithms for VLSI Design, Layout and Test Generation*. Prentice-Hall, 1999.

[19] "MATLAB." MathWorks. http://www.mathworks.com/products /matlab/.

[20] M. O'Dare and T. Arslan, "Generating Test Patterns for VLSI Circuits Using a Genetic Algorithm," *Electronics Letters*, vol. 30, no. 10, pp. 778–779, 1994.

[21] S. M. Reddy, "Complete Test Sets for Logic Functions," *IEEE Trans. Computers*, vol. 22, no. 11, pp. 1016–1020, 1973.

[22] J. P. Roth, "Diagnosis of Automata Failures: A Calculus and a Method," *IBM Journal of Research and Development*, vol. 10, no. 4, pp. 278–291, 1966.

[23] D. G. Saab, Y. G. Saab, and J. A. Abraham, "CRIS: A Test Cultivation Program for Sequential VLSI Circuits," in *Proc. IEEE/ACM International Conference on Computer-Aided Design*, 1992, pp. 216–219.

[24] D. G. Saab, Y. G. Saab, and J. A. Abraham, "Automatic Test Vector Cultivation for Sequential VLSI Circuits Using Genetic Algorithms," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 15, no. 10, pp. 1278–1285, 1996.

[25] H. D. Schnurmann, E. Lindbloom, and R. G. Carpenter, "The Weighted Random Test-Pattern Generator," *IEEE Trans. Computers*, vol. 24, no. 7, pp. 695–700, 1975.

[26] G. Seroussi and N. H. Bshouty, "Vector Sets for Exhaustive Testing of Logic Circuits," *IEEE Trans. Information Theory*, vol. 34, no. 3, pp. 513–522, 1988.

[27] P. W. Shor, "Polynomial-time Algorithms For Prime Factorization and Discrete Logarithms on a Quantum Computer," *SIAM Jour. Computing*, vol. 26, no. 5, pp. 1484–1509, 1997.

[28] A. Singh, L. M. Bharadwaj, and S. Harpreet, "DNA and Quantum based Algorithms for VLSI Circuits Testing," *Natural Computing*, vol. 4, no. 1, pp. 53–72, 2005.

[29] S. Singh and M. Singh, "Applying Quantum Search to Automated Test Pattern Generation for VLSI circuits," in *Proc. 4th IEEE International Conference on Parallel and Distributed Computing, Applications and Technologies*, 2003, pp. 648–651.

[30] M. Venkatasubramanian and V. D. Agrawal, "A New Test Vector Search Algorithm for a Single Stuck-at Fault using Probabilistic Correlation," in *Proc. 23rd IEEE North Atlantic Test Workshop*, (Johnson City, NY), 2014, pp. 57–60.

[31] M. Venkatasubramanian and V. D. Agrawal, "Quest for a Quantum Search Algorithm for Testing Stuck-at Faults in Digital Circuits," in *Proc. 29th IEEE International Symp. Defect and Fault Tolerance in VLSI and Nanotechnology Systems*, (Amherst, MA), 2015, pp. 128–133.

[32] M. Venkatasubramanian and V. D. Agrawal, "Database Search and ATPG–Interdisciplinary Domains and Algorithms," in *2016 29th International Conference on VLSI Design and 2016 15th International Conference on Embedded Systems (VLSID)*, IEEE, 2016, pp. 38–43.

[33] N. Yogi and V. D. Agrawal, "Spectral RTL Test Generation for Gate-Level Stuck-at Faults," in *Proc. 15th IEEE Asian Test Symposium*, 2006, pp. 83–88.