An Accurate Probabilistic Model for Error Detection

Thara Rejimon and Sanjukta Bhanja
Electrical Engineering
University of South Florida
Tampa, Florida, USA
(rejimon, bhanja)@eng.usf.edu

Abstract

*We propose a novel single event fault/error model based on Logic Induced Fault Encoded Directed Acyclic Graph (LIFE-DAG) structured probabilistic Bayesian networks, capturing all spatial dependencies induced by the circuit logic. The detection probabilities also act as a measure of soft error susceptibility (an increased threat in nano-domain logic block) that depends on the structural correlations of the internal nodes and also on input patterns. Based on this model, we show that we are able to estimate detection probabilities of single-event faults/errors on ISCAS'85 benchmarks with high accuracy (zero-error), linear space requirement complexity, and with an order of magnitude (≈***5 times***) reduction in estimation time over corresponding BDD based approaches.*

1 Introduction

Fault Detection Probability (FDP) is an important testability measure that is useful for not only generating test patterns, but also to vet designs for random-input testability. Traditionally, FDP has been used for test point insertions, however, it can also be used as random single-event-transient (SET) sensitivity measure, which is important for characterization of impact of soft errors in logic blocks. Designers can selectively apply corrective measures to nodes with higher FDP than the ones with low FDP. Due to shrinking geometry, low supply voltage and high frequency, soft errors or single-event-upsets in logic circuits would be one of the hard challenges in nano domain. The major cause of soft error is the natural cosmic radiation in the atmosphere, which is dominantly neutron radiation. The factors that influences SET are the rate of high-energy neutron hits on a node, input pattern dependence on whether the SET at the node can reach an output latch and the probability of the latch capturing the transitions [1]. In this work, we focus on the probabilistic modeling of SET that allows an SET generated at a node captured by a latch and measure it by the fault detection probability of that node. Existing estimation techniques for SET [2, 3] rely on simulation and hence are modeling the pattern dependence of SET by estimation methods that are in itself pattern-sensitive. We use a probabilistic zero-error model for detection probability of errors that can be uniformly applied to permanent stuck-at faults as well as soft transient errors predominant in nano-domain. Note that the probabilistic modeling does not require any assumption in the input patterns and can be extended to biased target workload patterns.

Fault Detection Probability (FDP) of a stuck-at fault $f \in F$, where F denotes the set of all faults, in a combinational circuit, C, is the probability that f is detected by a randomly chosen equally-likely input patterns. Signal probabilities, as well as FDP, are affected by spatial correlations induced by re-convergence. Existing algorithms for computation of *exact* fault detection probabilities, mostly based on Binary Decision Diagrams [4] do not scale well, in terms of time and space requirements, with circuit size. They usually require various heuristic-based approximations of the pure model. All the modeling issues pertaining to FDP for single stuck-at faults are relevant for single-event transient soft errors also.

None of the existing algorithms for estimation of FDP, (as proposed by Seth *et al.* [12], Wunderlich [14], etc.) use Bayesian networks as data structure. These works were done in mid 80's. Bayesian network was developed in 1988 as a tool for probabilistic reasoning and it has been applied in artificial intelligence and image analysis. Recently, in [7], switching probabilities in VLSI circuits have been modeled using a Bayesian Network, however their use in estimation of error detection probabilities in digital logic is new.

We model single stuck-at-faults in large combinational circuits using a Logic Induced Fault Encoded Direct Acyclic Graph (LIFE-DAG) graph structure. We prove that such a DAG is a Bayesian Network. Bayesian Networks are graphical probabilistic models representing the joint probability function over a set of random variables.

A Bayesian Network is a directed acyclic graphical structure (DAG), whose nodes describe random variables and node to node arcs denote direct causal dependencies. A directed link captures the direct cause and effect relationship between two random variables. Each node is quantified by the conditional probability of the states of that node *given* the states of its parents, or its direct causes. The attractive feature of this graphical representation of the joint probability distribution is that not only does it make conditional dependency relationships among the nodes explicit but it also serves as a computational mechanism for efficient probabilistic updating. Probabilistic Bayesian Networks can be used not only to infer effects due to known causes (predictive problem) but also to infer possible causes for known effects (the backtracking or diagnosis problem). The diagnostic aspects of BNs makes it suitable for further use in test-pattern generators. To infer using these built BN models, we use stochastic inference schemes based on importance sampling. An importance sampling algorithm generates sample instantiations of the *whole* DAG network, i.e. for all lines in our case. These samples are then used to form the final estimates. At each node, this sampling is done according to an importance function that is chosen to be close to the actual joint probability function. The stochastic sampling strategy works because in a Bayesian Network the product of the conditional probability functions for all nodes is the optimal importance function. Because of this optimality, the demand on the number of samples is low.

We use the stochastic inference scheme, Probabilistic Logic Sampling (PLS) [6]. In Probabilistic Logic Sampling, a full instantiation of the probabilistic network is collected based on an simplified importance function that does not account for any evidence that may be present. The sampling is stopped when the probabilities of the nodes converges. It is worth pointing out that unlike simulative approaches that sample the inputs, importance sampling based procedures generate instantiations for the whole network, not just for the inputs. These samples can be looked upon as Markov Chain sampling of the circuit state space.

We advance the state of the art in fault analysis in terms of space and time requirements and in providing an uniform model. The worst case space requirement for the Bayesian network approach is linear in the number of nodes, specifically, it is $O(nf_{max}|F|)$ where n is the number of nodes, f_{max} is the maximum fan-in and $|F|$ is the cardinality of the fault set. The time complexity, based on the stochastic inference scheme, is also linear in n, specifically, it is $O(n|F|N)$, where N is the number of samples, which we have found to be in the order of 1000's for circuits with 10,000's of signals and 50,000's nodes. This is unlike traditional Binary Decision Diagram (BDD) based approaches, which have large demands on space and time, necessitating time-consuming decompositions and pre-processing to the pure BDD approach. Moreover, our approach works for all the circuits uniformly and requires no preprocessing judgments, such as establishing the proper ordering of variables. BDD based approaches are unsuitable for certain circuits, such as c6288, which a fairly common data path block (16 bit multiplier).

2 Related Work

Due to the high computational complexity involved in computing signal and fault detection probabilities, several approximation strategies have been developed in the past [10, 14, 16, 17]. The cutting algorithm proposed in [17], computes lower bounds of fault detection probabilities by propagating signal probability values. However, this algorithm delivers loose bounds, which may lead to unacceptable test lengths. Also, computing complexity of this algorithm is $O(n^2)$. Lower bounds of fault detection probability were also derived from controllability and observability measures [16]. This latter method gave poor lower bounds due to the fact that they cannot account for the component of fault detection probability due to multiple path sensitizations. The above mentioned methods are satisfactory only for faults that have single sensitizing path for fault propagation to an output and hence will not give good results for highly reconvergent fan-out circuits that have multiple path sensitizations. PREDICT [12] is a probabilistic graphical method to estimate circuit testability by computing node controlabilities and observabilities using shannon's expansion. The time complexity of exact analysis by this method is exponential in the circuit size. PROTEST [14], which is a tool for probabilistic testability analysis, calculates fault detection probabilities and optimum input signal probabilities for random test pattern, by modeling the signal flow. Fault detection probabilities, which are computed from signal probability values, are underestimated due to the fact that the algorithm does not take into account multiple path sensitization. Another method (CACOP) [10] is a compromise between the full range cutting algorithm and the linear time testability analysis, like the controllability and observability program. However, this method does not give exact fault detection probability.

The algorithm proposed in [15] uses supergate decomposition to compute exact fault detection probabilities of large circuits. PLATO (Probabilistic Logic Analyzing Tool) [4] is another tool to compute exact fault detection probabilities using reduced ordered binary decision diagrams (ROBDD)s. Space requirement for constructing the ROBDD of large circuits is very large. Shannon decomposition and divide-and-conquer strategies are used to reduce large circuits into small sub-circuits. Computing complexity of these decomposition methods are quite high. Another BDD based algorithm is proposed in [13] to compute exact random pattern detection probabilities. However, this algorithm could not be used for large circuits because of large space requirements.

In this paper, we propose a novel stuck-at-fault model to compute accurate detection probabilities using Bayesian networks as data structure. Bayesian networks pushes the state-of-the-art significantly in terms of reduced space and time requirements. Using Bayesian networks, we can calculate the optimum input signal probabilities of the random test pattern for a desired fault coverage. We compute the exact fault detection probabilities of a number of ISCAS benchmark circuits.

3 Bayesian Network

A Bayseian network is a Directed Acyclic Graph (DAG) in which a set of random variables make up the nodes of the network and a set of directed links connect pairs of nodes. The links represent causal dependencies among the variables. Each node has a conditional probability table (CPT) except the root nodes. Each root node has a priopr probability table. The CPT quantifies the effect the parents have on the node. Bayesian netwoks compute the joint probability distribution over all the varibles in the network, based on the conditional probabilities and the observed evidence about a set of nodes.

Fig. 1 illustrates a small Bayesian network. The exact joint probability distribution over the variables in this network is given by Eq. 1.

$$P(x_6, x_5, x_4, x_3, x_2, x_1) = P(x_6|x_5, x_4, x_3, x_2, x_1) \\ P(x_5|x_4, x_3, x_2, x_1)P(x_4|x_3, x_2, x_1)P(x_3)P(x_2)P(x_1). \tag{1}$$

In this BN, the random variable, X_6 is independent of X_1, X_2 and X_3 given the states of its parent nodes, X_4 and X_5. This *conditional independence* can be expressed by Eq. 2.

$$P(x_6|x_5, x_4, x_3, x_2, x_1) = P(x_6|x_5, x_4) \tag{2}$$

Mathematically, this is denoted as $I(X_6, \{X_4, X_5\}, \{X_1, X_2, X_3\})$. In general, in a Bayesian network, given the parents of a node n, n and its descendents are independent of all other nodes in the network. Let U be the set of all random variables in a network. **A Markov blanket** of element $X_i \in U$ is a subset S of U for which $I(X_i,\ S,\ U - S - X_i)$ and $X_i \notin S$. A set is called a **Markov boundary**, B_i of X_i if it is a minimal Markov blanket of X_i, i.e. none of its proper subsets satisfy the triplet independence relation. In the above example, the set $\{X_4, X_5\}$ is the Markov bounay of the node X_6.
Using the conditional independencies in Eq. 2, we can arrive at the minimal factored representation shown in Eq. 3.

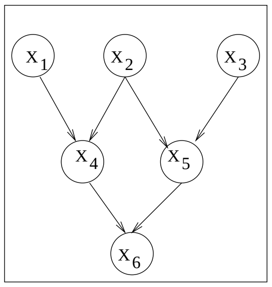

Figure 1. A small bayesian network

$$P(x_6, x_5, x_4, x_3, x_2, x_1) = P(x_6|x_5, x_4)P(x_5|x_3, x_2) \\ P(x_4|x_2, x_1)P(x_3)P(x_2)P(x_1). \tag{3}$$

In general, if x_i denotes some value of the variable X_i and $pa(x_i)$ denotes some set of values for X_i's parents, the minimal factored representation of exact joint probability distribution over m random variables can be expressed as in Eq. 4.

$$P(X) = \prod_{k=1}^{m} P(x_k|pa(x_k) \tag{4}$$

4 LIFE-BN: Single Event Error Model

We first discuss the basics of fault/error detection probabilities for random-pattern testability analysis. Note that the probabilistic modeling does not require any assumption in the input patterns and can be extended to biased target workload patterns. Next, we sketch the concept of Logic-induced-Fault-Encoded (LIFE) Directed Acyclic Graph that represents the underlying fault/error model. In this probabilistic framework, we use partial duplication of the original circuit for the fault detection logic. Only the sensitization paths from the fault/error are duplicated. A set of comparator nodes compares the error-free and error-sensitized logic. A logic one in such comparator outputs indicates the occurrence of an error. Error detection probability of faults/errors that affect multiple outputs, is the maximum probability of the sensitized comparator outputs. These detection probabilities depend on the circuit structural dependence, the inputs and the dependencies amongst the inputs. In this work, however, we assume random inputs for validation and experimentation of our model.

We follow our discussion by proving that such a Logic-induced-Fault-Encoded (LIFE) Directed Acyclic graph is indeed the minimal I-map of the fault model for the $< C, F >$ and, hence, is a Bayesian Network.

Definition: The Logic Induced Fault Encoded Direct Acyclic Graph (LIFE-DAG) corresponding to the fault detection model $< C, F >$ can be constructed as follows:

- Nodes of LIFE-DAG are random variables $\{X_i^{'}\}$, which are of three types

  1. $\{X\}$: Set of nodes representing signals in the circuit C
  2. $\{S_f \forall f \in F\}$: where S_f is the set of all the descendant nodes of fault node f including the fault node f. Note that these nodes are in addition to their counterparts in fault-free X.
  3. $\{X_{To} \forall o \in Output - set\}$: Set of all the detection nodes (comparator outputs). Cardinality of this set is determined by the number of testable outputs for each faults in the fault set.

- Edges of LIFE-DAG are directed and denoted as ordered pair (u, v), denoting u causes v. The edge set can be classified as follows:

  1. $\{E_{(X,X)}\}$: Edges in C such that both vertices u and v are in X. These edges are part of the original circuit C.
  2. $\{E_{(X,S_f)} | f \in F\}$: Edges such that child node v is in fault reachable extension and the parent u is in the original circuit. These edges are called bridge edges. Note that this edge indicates that there must be at least another parent of v that is in S_f.
  3. $\{E_{(S_f,S_f)} | f \in F\}$: Edges where both u and v are in fault reachable extension S_f. These edges indicate that both u and v are descendants of f.
  4. $\{E_{S_f,(X_{To})} | f \in F\}$: Edges where u is the output node in S_f and v is the detection node. These edges are quantified by an xor gate. Moreover, the detection node v must have another parent in X that is the fault free circuit.
  5. $\{E_{(X,X_{To})}\}$: Edges where u is the output node in X and v is the detection node. These edges are quantified by an xor gate.

We illustrate the ideas using a two NAND node circuit with three input lines and two output lines, $\{x_{22} = x_{10}\bar{x}_{16}; x_{23} = x_{16}\bar{x}_{19}\}$. Figure 2 shows the fault detection logic corresponding to 16@1 and 19@1. Block C is the combinational circuit under consideration. Block $S_{16@1}$ represents all the descendant nodes of the fault node 16 (example: $22^{'}$) which are different from their counterpart (22) in C. Block $S_{19@1}$ represents the same for 19@1 fault. The detection set contains the comparator nodes.

Figure 2(b) is the LIFE-DAG model of the above fault detection logic. Nodes $\{10, 16, 19, 20, 21, 22, 23\}$ belong to the set $\{X\}$. Nodes $\{10, 16@1, 20^{'}, 21^{'}, 22^{'}, 23^{'}\}$ belong to the set $\{S_{f1}\}$ where f1 is 16@1 fault, and nodes $\{T1\_19\_1, T1\_16\_1, \cdots\}$ belong to the set $\{X_{To}\}$.

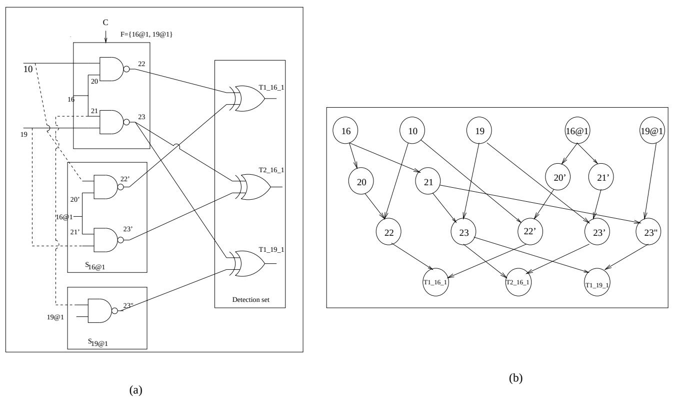

Figure 2. (a) An illustrative fault detection logic (b) LIFE-DAG model of (a)

Similarly, in Figure 2(b), we have edge $\{E_{(16,21)}\}$, which is an element of the edge set $\{E_{(X,X)}\}$ where both 16 and 21 are in $\{X\}$. The edge $\{E_{(21,23")}\}$ is an element of the edge set $\{E_{(X,S_{19@1})}\}$ where 21 $\in$ $\{X\}$ and 23" $\in$ $\{S_{19@1}\}$. The edge between nodes 19@1 and 23", $\{E_{(19@1,23")}\}$ is an element of the edge set $\{E_{(S_{19@1},S_{19@1})}\}$, and the edge $\{E_{(23,T1\_19\_1)}\}$ is an element of the edge set $\{E_{(X,X_{T0})}\}$ where 23 $\in$ $\{X\}$ and $T1\_19\_1$ $\in$ $\{X_{T0}\}$.

5 Bayesian Inference

We explore the stochastic sampling algorithm, namely probabilistic Logic Sampling (PLS). These method have been proven to converge to the correct probability estimates [6], without the added baggage of high space complexity.

Probabilistic Logic Sampling (PLS): Probabilistic logic sampling is the earliest and the simplest stochastic sampling algorithms proposed for Bayesian Networks [6]. Probabilities are inferred by a complete set of samples or instantiations that are generated for each node in the network according to local conditional probabilities stored at each node. The advantages of this inference are that: (1) its complexity scales linearly with network size, (2) it is an any-time algorithm, providing adequate accuracy-time trade-off, and (3) the samples are not based on inputs and the approach is input pattern insensitive. The salient aspects of the algorithm are as follows.

1. Each sampling iteration stochastically instantiates all the nodes, guided by the link structure, to create a network instantiation.

2. At each node, x_k, generate a random sample of its state based on the conditional probability, $P(x_k|Pa(x_k))$, where $Pa(x_k)$ represent the states of the parent nodes. This is the local, importance sampling function.
3. The probability of all the query nodes are estimated by the relative frequencies of the states in the stochastic sampling trace.
4. If states of some of the nodes are known (evidence), such as in diagnostic backtracking, network instantiations that are incompatible with the evidence set are disregarded.
5. Repeat steps 1, 2, 3 and 4, until the probabilities converge.

The above scheme is efficient for predictive inference, when there is no evidence for any node, but is not efficient for diagnostic reasoning due to the need to generate, but disregard samples that do not satisfy the given evidence. It would be more efficient not to generate such samples. We discuss such a method next.

Time and Space Complexity: The space requirement of the Bayesian network representation is determined by the space required to store the conditional probability tables at each node. For a node with n_p parents, the size of the table is 2^{n_p+1}. The number of parents of a node is equal to the fan-in at the node. In the LIFE-BN model we break all fan-ins greater than 2 into a logically equivalent, hierarchical structure of 2 fan-in gates. For nodes with fan-in f, we would need $\lceil f/2 \rceil$ extra Bayesian network nodes, each requiring only 2^3 sized conditional probability table. For the worst case, let all nodes in the original circuit have the maximum fan-in, f_{max}. Then the total number of nodes in the LIFE-BN structure for each fault model is $n + f_{max}n/2$. Thus, the worst case space requirement is linear, i.e. $O(nf_{max}|F|)$ where n is the number of nodes, f_{max} is the maximum fan-in, and $|F|$ is the cardinality of the fault set.

The time complexity, based on the stochastic inference scheme, is also linear in n, specifically, it is $O(n|F|N)$, where N is the number of samples, which, from our experience with tested circuits, is in the order of 1000's for circuits with 10,000 of signals and 50,000 nodes in the fault detecting logic.

6 Experimental Results

We demonstrate the ideas using ISCAS benchmark circuits. The logical relationship between the inputs and the output of a gate determines the conditional probability of a child node, given the states of its parents, in the LIFE-BN. Gates with more than two inputs are reduced to two-input gates by introducing additional dummy nodes, without changing the logic structure and accuracy.

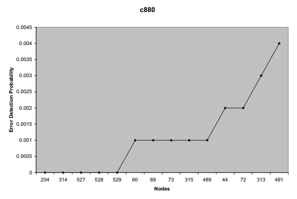

Figure 3. Detection probability as SET sensitivity for different nodes in c880.

First, we determine the hard faults using 1024 random input vectors [4]. Faults that are not detected by these vectors are hard faults. Accurate detection probabilities are needed for these hard faults. The probabilistic modeling however works uniformly for all the single-event faults/errors in the fault set. We performed an in-house logic simulation with $500,000$ random vectors to detect the exact fault detection probability of all the hard faults. We use these probabilities as ground truth. We performed experiments using Probabilistic Logic Sampling. This sampling scheme is simple and time-efficient.

The results of detection probabilities computed by PLS [6] for 1000 samples and 3000 samples are shown in Table 1. In this table, μ_E is the accuracy of modeling in terms of average error in FDP over all the non-redundant hard faults compared to simulation results. $T(s)$ is the total elapsed time, *including memory and I/O access*. This time is obtained by the ftime command in the WINDOWS environment on a Pentium-4 2.0 GHz PC.. We report the μ_E and $T(s)$ for 1000 samples in columns 2 and 3, respectively, and the μ_E and $T(s)$ for 3000 samples in columns 4 and 5, respectively. We partitioned the faults in circuits c3540, c6288 and c7552 into three subsets and determined the detection probabilities in each set by parallelly running the circuits for all the fault sets.

Figure 3 shows the FDP, which can also be used to characterize single-event-transient (SET) error sensitivity, for the nodes in c880. Note that some nodes that have high detection probabilities which should have been captured by the initial simulation, however was undetected. Hence, characterization based on simulation suffers from pattern-dependence of the simulative methods.

In Table 2, we compare LIFE-BN fault modeling with the performance of approaches based on the Binary Decision Diagram (BDD) model, as reported by Krieger *et al.* [4] for these same circuits. They reported results using four type of fault partitioning. We compare our time (column 4) with the time taken by their two best methods, namely and PSG (column 2) and Supergate SG (column 3). In column 5, we report the ratio between the minimum time taken by the BDD based method and the time taken by

Table 1. Fault detection probability estimation errors and time for 1000 samples and 3000 samples.

	Bayesian Networks (PLS)			
	1000 samples		3000 samples	
	μ_E	T(s)	μ_E	T(s)
c432	0.00032	0.24	0.00039	1.26
c499	0.00131	1.00	0.00094	3.48
c880	0.00070	2.00	0.00051	6.58
c1355	0.00093	3.00	0.00083	8.50
c1908	0.00079	5.00	0.00048	13.60
c2670	0.00029	28.00	0.00026	52.12
c3540	0.00215	34.00	0.00163	64.69
c5315	0.00036	12.00	0.00031	27.00
c6288	0.00929	65.00	0.00874	109.87
c7552	0.00069	37.24	0.00081	69.14

Table 2. Comparison with the state-of-the-art

	BDD [4]		BN	Ratio	
	PSG	SG	PLS		
	T_{CPU} (s) (Y93)		T_{total}(s) Y(04)	R	R/16
c432	139	71	0.24	295.83	18.49
c499	80	44	1.00	44.00	2.75
c880	328	1132	2.00	164.00	10.25
c1355	157	79	3.00	26.33	1.65
c1908	686	288	5.00	57.60	3.6
c2670	2051	–	28.00	73.25	4.58
c3540	23630	1732	34.00	50.94	3.18
c5315	82	31	12.00	2.58	0.16
c6288	–	–	65.00	–	–
c7552	1281	–	37.24	34.34	2.15
Average				83.22	5.20

our Bayesian network based approach. The average improvement seems to be 83 times. However, it is not fair to directly compare the times since the times reported by Krieger *et al.* is based on a early 90's computer not specified exactly in the paper, but probably $\approx$ 125 MHz, whereas ours is a 2.0GHz computer. Also, the time reported by Krieger *et al.* is just the CPU time, whereas ours include CPU, I/O, and memory access. Considering all these factors, even if we assume a 16 times speed "handicap" the LIFE-BN with PLS inference scheme appears to be 5 times more efficient than a BDD based one. We provide this scaled time performance ratio (Actual Ratio/16) in column 6 of Table 2.

We present a non-simulative probabilistic method for estimating fault/error detection probability for testability and soft error sensitivity analysis. The simulation method we used to obtain the ground truth is pattern-sensitive whereas our model is pattern-insensitive and time efficient. The estimated probabilities are found to be almost error-free. We are currently experimenting Bayesian Networks to backtrack probabilistically for ATGP and exploring soft error detectability measures.

References

[1] K. Mohanram and N. A. Touba, "Cost-Effective Approach for Reducing Soft Error Failure Rate in Logic Circuits," *International Test Conference*, 2003, pp. 893–901.

[2] D. Alexandrescu, L. Anghel and M. Nicolaidis, New Methods for Evaluating the Impact of Single Event Transients in VDSM ICs, *Proc. Defect and Fault Tolerance Symposium*, 2002, pp. 99–107.

[3] P. Shivakumar, et al., Modeling the Effect of Technology Trends on the Soft Error Rate of Combinational Logic, *Proc. International Conference on Dependable Systems and Networks*, 2002, pp. 389–398.

[4] R. Krieger, B. Becker and R. Sinkovic, "A BDD-based Algorithm for computation of exact fault detection probabilities," *Digest of Papers, Fault-Tolerant Computing*, vol. 22-24, June 1993, pp. 186–195.

[5] J. Cheng, "Efficient Stochastic Sampling Algorithms for Bayesian Networks," *Ph.D Dissertation, University of Pittsburgh*, 2001.

[6] M. Henrion, "Propagation of uncertainty by probabilistic logic sampling in Bayes' networks," *Uncertainty in Artificial Intelligence*, 1988.

[7] S. Bhanja and N. Ranganathan, "Dependency preserving probabilistic modeling of switching activity using Bayesian networks" *Design Automation Conference*, Jun. 2001.

[8] J. Pearl, "Probabilistic Reasoning in Intelligent Systems: Network of Plausible Inference," Morgan Kaufmann Publishers, Inc., 1988.

[9] Y. Fang and A. Albicki, "Efficient testability enhancement for combinational circuits," *Proceedings of the International Conference on Computer Design*, vol. 2-4, pp. 168–172, 1995.

[10] W-B. Jone and S. R. Das, " CACOP-a random pattern testability analyzer," *IEEE Transactions on Man and Cybernetics*, vol. 25-5 pp. 865–871, 1995.

[11] B. P. Philips, "On computing the detection probability of stuck-at faults in a combinational circuit," *IEEE system readiness Technology Conference*, vol. 24-26, Sep. 1991, pp. 301–305.

[12] S. C. Seth, L. Pan, and V. D. Agrawal, "Predict - probabilistic estimation of digital circuit testability," *IEEE International Symposium on Fault-Tolerant Computing*, Jun. 1985, pp. 220–225.

[13] H. Farhat, A. Lioy and M. Pocino, "Computation of exact random pattern detection probability," *Proceedings ofIEEE Custom Integrated Circuits conference*, vol. 9-12, May 1993, pp. 26.7.1–26.7.4.

[14] H. J. Wunderlich , "PROTEST: A Tool for probabilistic testability Analysis," *Proceedings of the 22nd IEEE Design Automation conference*, vol. 14-3, 1985, pp. 204–211.

[15] S. Chakravarty and H. B. hunt, III, "On computing signal probability and detection probability of stuck-at faults," *IEEE Transactions on Computers*, vol. 39-11, Nov. 1990 pp. 1369–1377.

[16] R. Pathak, "A generalized algorithm for bounding fault detection probabilities in combinational circuits," *AUTOTESTCON, Proceedings of IEEE Systems Readiness Technology Conference*, vol. 20-23, Sep. 1993, pp. 683–689.

[17] J. Savir and G. S.Ditlow, and P. H. Bardell, "Random pattern testability," *IEEE Transactions on Computers*, vol. C-33, Mar. 1984, pp. 79–90.