Time and Space Efficient Method for Accurate Computation of Error Detection Probabilities in VLSI Circuits

Thara Rejimon and Sanjukta Bhanja
University of South Florida, Tampa, FL, USA.
E-mail: (rejimon, bhanja)@eng.usf.edu

Accepted for publication in IEE Journal on Computers and Digital Techniques, 2005

***Abstract*—We propose a novel fault/error model based on a graphical probabilistic framework. We arrive at the Logic Induced Fault Encoded Directed Acyclic Graph (LIFE-DAG) that is proven to be a Bayesian network, capturing all spatial dependencies induced by the circuit logic. Bayesian Networks are the minimal and exact representation of the joint probability distribution of the underlying probabilistic dependencies that not only use conditional independencies in modeling but also exploits them for achieving minimality and smart probabilistic inference. 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 faults/errors on ISCAS'85 benchmarks with high accuracy, linear space requirement complexity, and with an order of magnitude ($\approx$5 times) reduction in estimation time over corresponding BDD based approaches.**

I. 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.

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 reconvergence. State-of-the-art algorithm for computation of *exact* fault detection probabilities, 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.

We model single stuck-at-faults (errors) 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. In a LIFE-DAG, each node describes the state of a line and a directed edge quantifies the conditional probability of the state of a node *given* the states of its parents or its direct causes. Figure 1(a) gives the conditional probability table of an AND gate. 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. We used two stochastic algorithms based on Importance sampling to compute the fault/error detection probability using the

LIFE-DAG model. 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. Specifically, we explore two stochastic inference strategies: Probabilistic Logic Sampling (PLS) [7] and Evidence Pre-propagated Importance Sampling (EPIS) algorithm [6], which are discussed in section V. 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.

We advance the state-of-the-art in fault analysis in terms of space and time requirements and in providing a uniform model. A detailed discussion about time and space complexity is given in section V.

Most of the analysis of FDP was performed in 80's (as proposed by Seth *et al.* [14], Wunderlich [17], etc.) In a later development (1988), Bayesian network was introduced for probabilistic reasoning and belief propagation and it has been applied in artificial intelligence and image analysis. Recently, in [8], [9], switching probabilities and signal arrivals in VLSI circuits have been modeled using a Bayesian Network, however their use in estimation of error detection probabilities in digital logic is new. In this paper, we provide a fresher look into an old problem by adopting a novel and efficient scheme. However, this probabilistic FDP analysis technique can be applied to measure Single Event Transient (SET) sensitivity and soft error susceptibility [1] of logic circuits which are future nanotechnology challenges. We discuss this further in II.

II. Motivation

When high-energy neutrons present in cosmic radiations and alpha particles from impurities in packaging materials hit a semiconductor device, they generate electron-hole pairs and cause a current pulse of very short duration, termed as Single Event Transient (SET). The effect of these SETs may be propagated to an output latch and cause a bit flip in latch, resulting in a *soft error*. As the stored charge in each logical node decreases with decreased dimensions and decreased voltages in nanometer technology, even weak radiations can cause disturbance in the signals, which results in increased soft error failure rate.

SET sensitivity of a node depends on the probability that there is a functionally sensitized path from the node to an output latch (which is the same as the fault detection probability of the node), probability that the generated SET is propagated to the latch and the probability that the latch captures the transitions arriving at its input [1]. If the last two probabilities are assumed to be one, FDP of a node is an accurate measure of the SET sensitivity. In nano-domain circuits, these probabilities are very close to one due to very high operating frequencies, reduced voltage, diminishing dimensions and reduced propagation delays. Hence FDP is a tight upper bound of SET sensitivity in nano-domain circuits. Designers can selectively apply corrective measures to nodes with higher FDP than the ones with lower FDP to reduce soft error failure rates.

III. Bayesian Networks

A Bayesian network is a Directed Acyclic Graph (DAG) in which the nodes represent random variables 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 prior probability table. The CPT quantifies the effect the parents have on the node. Bayesian networks compute the joint probability distribution over all the variables in the network, based on the conditional probabilities and the observed evidence about a set of nodes.

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

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

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 X, X and its descendents are independent of all other nodes in the network. Using the conditional independencies in Eq. 2, we can arrive at the minimal factored representation shown in Eq. 3.

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

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.

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

IV. LIFE-BN: Fault/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 sensitized paths from the fault/error are duplicated. A set of comparator nodes compares the ideal and error-sensitized logic. A logic one in such comparator outputs indicates the occurrence of an error. Fault/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 $< C,F >$ where C represents the combinational circuit an F represents the fault/error set, can be constructed as follows (We illustrate the ideas with the help of a small fault detection circuit and the LIFE-DAG constructed from it, as shown in Figure 2): The model consists of the combinational circuit C (fault/error-free) and a set of fault/error sensitized logic blocks, $S^f \ \forall f \in F$, which propagate the effect of the faults.

- Nodes of LIFE-DAG are random variables with two possible values, 0 or 1. There are 7 types of nodes.

1. $\{Z\}$: Primary inputs.
2. $\{F\}$: Nodes representing injected faults.
3. $\{X\}$: Internal nodes in the fault-free circuit C. Example: X_{21}.
4. $\{X^f \ \forall f \in F\}$: Corresponding internal nodes in the duplicated circuit (descendents of $\{F\}$). Example: $X_{21}^{f_1}$
5. $\{Y\}$: Primary outputs of fault-free circuit C. Example: Y_{22}.
6. $\{Y^f \ \forall f \in F\}$: Corresponding faulty outputs. Example: $Y_{22}^{f_1}$.
7. $\{E_o^f \ \forall o \in \ Output-set, \ \forall f \in F\}$: Set of all the detection nodes (comparator outputs). Cardinality of this set is determined by the number of testable outputs for each fault in the fault set. Example: $E_1^{f_1}$

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

1. $\{Z \rightarrow X\}$: Edges between primary inputs and gates in C, which are directly fed by the primary inputs. These edges are part of the original circuit C.
2. $\{F \rightarrow X^f\}$: Edges from the induced faulty inputs to the duplicate gates in the fault sensitized logic block, S^f.
3. $\{Z \rightarrow X^f\}$: Edges from the primary inputs to the duplicate gates in S^f. Note that this edge indicates that there must be at least one parent of X^f that is in $\{F\}$ or in $\{X^f\}$.
4. $\{X \rightarrow X\}$, $\{X^f \rightarrow X^f\}$: Edges between the nodes representing the internal signals (Edges from the input of a gate to the corresponding output).
5. $\{X \rightarrow X_f\}, \{X \rightarrow Y^f\}$: Edges such that child node v is in S^f(in $\{X^f\}$ or in $\{Y^f\}$) and the parent u is in the original circuit C. These edges are called bridge edges. Note that this edge indicates that there must be at least one parent of v that is in $\{X^f\}$ or in $\{F\}$. Example: $X_{21} \rightarrow Y_{23}^{f_2}$.
6. $\{X \rightarrow Y\}$,$\{X^f \rightarrow Y^f\}$: Edges where parent node u is an internal signal feeding an output gate and the child node v represents the corresponding output signal. Example: $X_{21} \rightarrow Y_{23}$ and $X_{21}^{f_1} \rightarrow Y_{23}^{f_1}$
7. $\{Y \rightarrow E_o^f\}$ and corresponding $\{Y^f \rightarrow E_o^f\}$: Edges where u is the output node in Y (or Y^f) and v is the detection node E_o^f. These edges are quantified by a xor gate. Example: $Y_{22} \rightarrow E_1^{f_1}$ and $Y_{22}^{f_1} \rightarrow E_1^{f_1}$

Theorem: The LIFE-DAG structure, corresponding to the combinational circuit C and a Fault set F is a minimal I-map of the underlying dependency model and hence is a Bayesian network.

Proof: Markov Boundary of a variable v in a probabilistic framework, is the minimal set of variables that make the variable v conditionally independent of all the remaining variables in the network.

Let us order the random variables in the node set, $\{\{Z\},\{F\},\{X\},\{X^f\},\{Y\},\{Y^f\},\{E_o^f\}\}$ such that for every edge (u,v) in LIFE-DAG, u appears before v.

With respect to this ordering, the Markov boundary of any node, $v \in \{\{Z\},\{F\},\{X\},\{X^f\},\{Y\},$
$\{Y^f\},\{E_o^f\}\}$ is given as follows. If v represents an input signal line, then its Markov boundary is the null set. If v is a fault node f in the fault set F, then also its Markov boundary is the null set (since this is treated as a primary input with a particular value). And, since the logic value of an output line is just dependent on the inputs of the corresponding gate (whether v is in $\{X\},\{X^f\},\{Y\},\{Y^f\} or \{E_o^f\}$), the Markov boundary of a variable representing an output line consists of just those that represent the inputs to that gate. Thus, in LIFE structure the parents of each node are its Markov boundary elements hence the LIFE is a boundary DAG. Note that LIFE is a boundary DAG because of the causal relationship between the inputs and the outputs of a gate that is induced by logic. It has been proved in [10] that if a graph structure is a boundary DAG D of a dependency model M, then D is a minimal I map of M. This theorem along with definitions of conditional dependencies in [10] (we omit the details) specifies the structure of the Bayesian network. Thus LIFE is a minimal I-map and thus a Bayesian network (BN).

Quantification of LIFE-BN: LIFE-BN consists of nodes that are random variables of the underlying probabilistic model and edges denote direct dependencies. All the edges are quantified with the conditional probabilities making up the joint probability function over all the random variables of LIFE-DAG. The overall joint probability function that is modeled by a Bayesian network can be expressed as the product of the conditional probabilities. Let us say, $X^{'} = \{X_1^{'}, X_2^{'}, \cdots, X_m^{'}\}$ are the node set in LIFE-DAG, then we can say

$$P(X^{'}) = \prod_{k=1}^{m} P(x_k^{'} | Pa(X_k^{'})) \tag{5}$$

where $Pa(X_k^{'})$ is the set of nodes that has directed edges to $X_k^{'}$. A complete specification of the conditional probability of a two input AND gate output will have 2^3 entries, with 2 states for each variable. These conditional probability specifications are determined by the gate type. By specifying the appropriate conditional probability we ensure that the spatial dependencies among sets of nodes (not only limited to just pair-wise) are effectively modeled.

V. Bayesian Inference

We explore two stochastic sampling algorithms, namely probabilistic Logic Sampling (PLS) and Evidence Pre-propagated Importance Sampling (EPIS). These methods have been proven to converge to the correct probability estimates [6], [7], without the added baggage of high space complexity.

A. Probabilistic Logic Sampling (PLS)

Probabilistic logic sampling is the earliest and the simplest stochastic sampling algorithms proposed for Bayesian Networks [7]. 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.

B. Evidence Pre-propagated Importance Sampling (EPIS)

The evidence pre-propagated importance sampling (EPIS) [5], [6] uses local message passing and stochastic sampling. This method scales well with circuit size and is proven to converge to correct estimates. This is also an anytime-algorithm since it can be stopped at any point of time to produce estimates. Of course, the accuracy of estimates increases with time.

Like PLS, EPIS is also based on importance sampling that generates sample instantiations of the *whole* DAG network, i.e. for all line states in our case. These samples are then used to form the final estimates. The difference is with respect to the importance function used for sampling, which for EPIS takes into account any available evidence. In a Bayesian network, the product of the conditional probability functions at all nodes form the optimal importance function. Let $X = \{X_1, X_2, \cdots, X_m\}$ be the set of variables in a Bayesian network, $Pa(X_k)$ be the parents of X_k, and E be the evidence set. Then, the optimal importance function is given by

$$P(X|E) = \prod_{k=1}^{m} P(x_k|Pa(x_k,E)) \tag{6}$$

This importance function can be approximated as

$$P(X|E) = \prod_{k=1}^{m} \alpha(Pa(X_k))P(x_k|Pa(X_k))\lambda(X_k) \tag{7}$$

where $\alpha(Pa(X_k)) = (P(E^-|Pa(X_k)))^{-1}$ is a normalizing constant dependent on $Pa(X_k)$ and $\lambda(X_k) = P(E^-|x_k)$, with E^+ and E^- being the evidence from parents and children, respectively, as defined by the directed link structure. Calculation of λ is computationally expensive and for this, Loopy Belief Propagation (LBP) [21] over the Markov blanket of the node is used. Yuan *et al.* [6] proved that for a poly-tree, the local loopy belief propagation is optimal. The importance function can be further approximated by replacing small probabilities with a specific cutoff value [5].

This EPIS 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 samples is low. We have found that just thousand samples are sufficient to arrive at good estimates for the ISCAS'85 benchmark circuits. The ability of EPIS to handle diagnostic *and* predictive inference comes at the cost of a somewhat increased time per iterations needed for the calculation of λ messages. We quantify this increase by our experiments.

C. 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.

TABLE I
ISCAS BENCHMARKS AND THE NUMBER OF FAULTS

Circuits	Faults	Red. flt	Hard FLT
c432	524	4	5
c499	758	8	9
c880	942	0	32
c1355	1574	8	32
c1908	1879	9	114
c2670	2747	117	435
c3540	3428	137	218
c5315	5350	59	78
c6288	7744	34	34
c7552	7550	131	586

VI. 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.

The LIFE-BN based model is capable of detecting stuck-at-faults as well as the soft errors caused by single-event-transients. In this work, we present results for a set of stuck-at-faults (*hard* faults), which shows the efficiency of our model, in time and space requirements, compared to the BDD based model. First, we determine the hard faults using 1024 random input vectors [4]. Faults that are not detected by these vectors are hard faults. We tabulate the hard faults in Table I for all the benchmark circuits. Accurate detection probabilities are needed for these hard faults.

We performed experiments using both PLS and EPIS inference schemes. Recall that the Probabilistic Logic-Sampling (PLS) scheme is simple and time-efficient, but cannot handle diagnostic evidence efficiently. However, the Evidence Pre-propagated Importance Sampling Algorithm (EPIS) [6]) can efficiently handle diagnostic evidence, but with increased time for cases when there is evidence. We performed an in-house logic simulation with $500,000$ random vectors to detect the exact fault detection probability of all the faults and used these probabilities to check the accuracy of our model.

The results of detection probabilities computed by Probabilistic Logic Sampling (PLS) [7], and Evidence Pre-propagated Importance Sampling EPIS [6] are shown in Table II. Results are reported for 1000 samples and 3000 samples. 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 EPIS in columns 2 and 3, respectively, and the μ_E and $T(s)$ for PLS 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.

We empirically demonstrate the linear dependence of estimation time on number of samples in Figure 3(a) for benchmark c880. We also present empirical evidence of linear dependence of FDP estimation time on the number of nodes in the fault detection logic for c880 benchmark in Figure 3(b). We considered different subsets of faults in the fault list of the circuit. Figure 4 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 III, 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 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 III.

TABLE II
(A) FAULT DETECTION PROBABILITY ESTIMATION ERRORS AND TIME FOR 1000 SAMPLES. (B) FAULT DETECTION PROBABILITY ESTIMATION ERRORS AND TIME FOR 3000 SAMPLES.

	Bayesian Networks (1000 samples)			
	EPIS		PLS	
	μ_E	T(s)	μ_E	T(s)
c432	0.00012	1.09	0.00032	0.24
c499	0.00064	5.11	0.00131	1.00
c880	0.00042	16.70	0.00070	2.00
c1355	0.00059	27.28	0.00093	3.00
c1908	0.00071	61.39	0.00079	5.00
c2670	0.00003	145.19	0.00029	28.00
c3540	0.00034	807.77	0.00215	34.00
c5315	0.00024	186.88	0.00036	12.00
c6288	0.00000	1779.93	0.00929	65.00
c7552	0.00034	863.73	0.00069	37.24

	Bayesian Netw	
	EPIS	
	μ_E	T(
c432	0.00012	2.1
c499	0.00012	7.5
c880	0.00033	21.0
c1355	0.00032	32.7
c1908	0.00052	71.2
c2670	0.00001	615.8
c3540	0.00020	848.6
c5315	0.00013	205.4
c6288	0.00000	1823.0
c7552	0.00030	902.2

Even though [4] explains BDD based method of FDP computation, the complete algorithm for different types of decomposition techniques could not be obtained from this paper. Hence we could not give a direct comparison of cpu time between the BDD based model and the LIFE-DAG model by reimplementing their algorithm in the same computer we used for experimenting our model.

VII. 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 [12], [17],

TABLE III
COMPARISON WITH THE STATE OF THE ART

	BDD [4]		BN	Ratio	
	PSG	SG	PLS		
	T_{CPU} (s) (t_1) (Y93)		T_{total}(s) (t_2) Y(04)	R (t_1/t_2)	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

[19], [20]. The cutting algorithm [20], computes lower bounds of fault detection probabilities by propagating signal probability values. 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 [19]. This method do not 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 re-convergent fan-out circuits that have multiple path sensitizations.

PREDICT [14] 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 [17], 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) [12] is a compromise between the full range cutting algorithm and the linear time testability analysis, like the controllability and observability program. This is also an approximate scheme.

[18] uses supergate decomposition to compute exact fault detection probabilities of large circuits. PLATO (Probabilistic Logic Analyzing Tool) [4] is a 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 [16] computes exact random pattern detection probabilities. However, this is not scalable for large circuits.

VIII. CONCLUSIONS AND ONGOING WORK

We present a non-simulative probabilistic method for estimating fault/error detection probability for testability and soft error sensitivity analysis. Given a circuit and a fault/error set F, we model the faults by a Logic-Induced-Fault-Encoded (LIFE) DAG, which is a Bayesian Network, exactly capturing all high order dependencies among the signals. We explored two close-to-exact inference schemes for evaluating the detection probabilities. We find that

- The estimates are almost error-free.
- The LIFE-BN approach appears to be 500% more time efficient than a BDD based one.
- The LIFE-BN approach handles all the benchmark circuits without the need for special case handling, unlike BDD based approaches.
- The LIFE-BN approach does not rely on approximations to handle large circuits. However, BDD based methods rely on several decomposition techniques to reduce large circuits into smaller components. Also, this pre-processing time required for the circuit decomposition are usually not reported.
- BDD based methods have been reported to have trouble with some benchmark circuits such as c6288, whereas our model can handle such circuits.

- The space requirement of the LIFE-BN is $O(n)$, whereas the space requirement of exact BDD approach is exponential in the worst case.

We use an exact probabilistic 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. 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 are currently experimenting Bayesian Networks to back-track probabilistically for ATGP and exploring SET sensitivity measures incorporating circuit delays.

REFERENCES

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

[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*, pp. 99–107, 2002.

[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*, pp. 389–398, 2002.

[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, pp. 186–195, June 1993.

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

[6] C. Yuan and M. J. Druzdzel, "An Importance Sampling Algorithm Based on Evidence Pre-propagation," *Proceedings of the 19th Annual Conference on Uncertainty on Artificial Intelligence*, pp. 624-631, 2003.

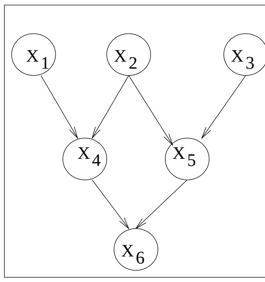
[7] M. Henrion, "Propagation of uncertainty by probabilistic logic sampling in Bayes' networks," *Uncertainty in Artificial Intelligence*, 1988.

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

[9] S. Bhardwaj, S.B.K. Vrudhula and D. Blaauw, "*t*AU: Timing analysis under uncertainty" *International Conference on Computer Aided Design*, Nov. 2003.

INPUT 1 (X1)		INPUT 2 (X2)		OUTPUT (XO)	
P(X1=0)	P(X1=1)	P(X2=0)	P(X2=1)	P(XO=0)	P(XO=1)
1	0	1	0	1	0
1	0	0	1	1	0
0	1	1	0	1	0
0	1	0	1	0	1

(a)



(b)

Fig. 1. (a) Conditional Probability Table of an AND gate (b) A small Bayesian network

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

[11] 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.

[12] 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.

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

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

[15] S. C. Seth, V. D. Agrawal and H. Farhat, "A theory of testability with application to fault coverage analysis," *Proceedings of the 1st Europian test conference*, vol. 12-14, pp. 139–143, April 1989.

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

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

[18] 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, pp. 1369–1377, Nov. 1990.

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

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

[21] K.P. Murphy, Y. Weiss and M.I. Jordan "Loopy belief propagation for approximate inference: an empirical study," *In Proceedings of Uncertainty in AI*, pp. 467–475, 1999.

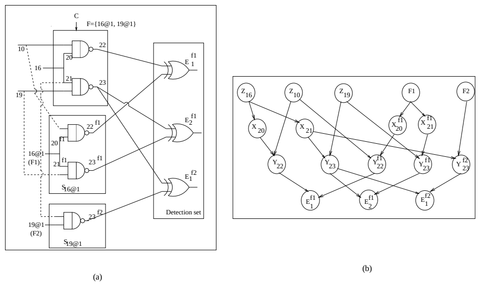


Fig. 2. (a) An illustrative fault detection logic (b) LIFE-DAG model of (a)

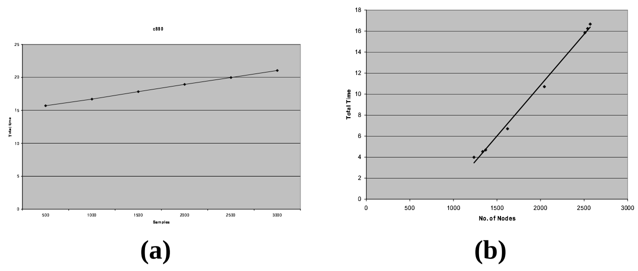


Fig. 3.   (a) Estimation time vs. number of samples in the detection logic for benchmark c880. (b) Estimation time vs. number of nodes in the detection logic for benchmark c880.

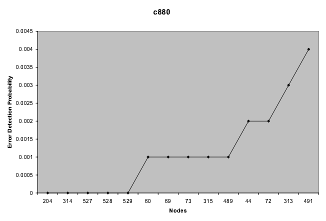


Fig. 4. Detection Probability as SET sensitivity for different nodes in c880.