Stochastic Computational Models for Accurate Reliability Evaluation of Logic Circuits

Hao Chen and Jie Han
Department of Electrical and Computer Engineering, University of Alberta
Edmonton, Alberta, Canada T6G 2V4
{hc5, jhan8}@ualberta.ca

ABSTRACT
As reliability becomes a major concern with the continuous scaling of CMOS technology, several computational methodologies have been developed for the reliability evaluation of logic circuits. Previous accurate analytical approaches, however, have a computational complexity that generally increases exponentially with the size of a circuit, making the evaluation of large circuits intractable. This paper presents novel computational models based on stochastic computation, in which probabilities are encoded in the statistics of random binary bit streams, for the reliability evaluation of logic circuits. A computational approach using the stochastic computational models (SCMs) accurately determines the reliability of a circuit with its precision only limited by the random fluctuations inherent in the representation of random binary bit streams. The SCM approach has a linear computational complexity and is therefore scalable for use for any large circuits. Our simulation results demonstrate the accuracy and scalability of the SCM approach, and suggest its possible applications in VLSI design.

Categories and Subject Descriptors
B.8.1 [**Performance and Reliability**]: Reliability, Testing, and Fault-Tolerance; F.1.2 [**Modes of Computation**]: Probabilistic computation

General Terms
Algorithms, Design, Performance, Reliability, Theory

Keywords
Stochastic computation, Stochastic computational model, Fault tolerance, Reliability evaluation, Logic circuits

1. INTRODUCTION
The design of reliable VLSI circuits becomes more challenging as CMOS feature sizes keep scaling in the nanometer regime. This downscaling makes circuits more susceptible to various factors that lead to manufacturing defects and interferences with noisy environments that lead to transient soft errors. A variety of novel devices such as carbon nanotubes, silicon nanowires, graphene and molecular electronics, have been investigated for future nanoelectronics. However, their non-deterministic characteristics and the uncertainty inherent in their manufacturing processes will inevitably render low circuit reliability. Therefore, the reliability of nanoelectronic circuits has increasingly been a concern, and will become a major design metric as performance and power are for today. This increasing demand on reliability design calls for accurate evaluation tools for the analysis of circuit reliability [1].

Several analytical approaches, such as those using probabilistic transfer matrices (PTMs) [2], probabilistic gate models (PGMs) [3, 4] and probabilistic decision diagrams (PDDs) [5], provide an accurate analysis of circuit reliability, however they suffer from the problem of having an exponential complexity and are therefore practically infeasible to be used for large circuits. While some other techniques also provide highly accurate results [6 - 9], they were aimed at achieving a tradeoff between accuracy and complexity. In [6], a probabilistic model based on Bayesian networks is proposed for the handling of large circuits using an approximate inference scheme. Authors in [7] present models using Boolean difference calculus, which is applied to probabilistic analysis of logic circuits. In [8], circuit transformations are used to calculate the signal probabilities of all internal nodes of logic circuits. In [9], three scalable algorithms are proposed for reliability analysis, but for large circuits, accuracy is obtained with constraints on error conditions.

In this paper, we propose an accurate approach using stochastic computation for the reliability analysis of logic circuits. This stochastic approach uses a random bit stream to encode signal probability [10, 11]. Stochastic computational models (SCMs) are constructed to implement the probabilistic analyses performed by PGMs. An algorithm using PGMs allows for an accurate reliability evaluation by identifying reconvergent fanouts and decomposing a circuit for each reconvergent fanout [4]. In the worst case, however, the PGM algorithm has a computational complexity that increases exponentially with the number of dependent reconvergent fanouts. In this paper, we show that the proposed stochastic approach handles the signal dependencies introduced by reconvergent fanouts gracefully and eliminates the need to deal with the complexity problem encountered in the PGM algorithm. In contrast to methods based on the simple simulation of random vectors, SCMs explicitly denote signal probabilities, so they are generic and versatile for use in both algorithmic development and practical applications. These advantages of the stochastic approach make it potentially suitable for various VLSI applications.

The remainder of this paper is organized as follows. Section 2 reviews the PGM and its analytical approaches. In section 3, we propose stochastic computational models (SCMs), and a computational approach using SCMs. The accuracy and complexity issues, as well as detailed error analyses, are also presented in this section. Simulation results are presented in Section 4. Section 5 gives conclusions and future work.

GLSVLSI'10, May 16–18, 2010, Providence, Rhode Island, USA.

2. RELIABILITY EVALUATION USING PROBABILISTIC GATE MODELS

2.1 Background

Most faults in combinational circuits are either inherently probabilistic or modeled probabilistically. Therefore, the reliability analysis of logic circuits has been based on the probabilistic treatment of signals in combinational logic networks [12]. The signal probability of an input or output of a logic gate is defined as the probability that the signal is a logical 1. A logic function provides a transform from its input to output probabilities. The reliability of an output is obtained as the probability of the output if the output is expected to be a logical 1, or its complement otherwise. Given independent inputs, Boolean functions can be mapped to the arithmetic operations of signal probabilities, according to the following rules [12, 13]:

Rule I: Boolean "NOT," or $B = \bar{A}$, corresponds to

$$b = 1 - a$$

where $b = P(B = 1)$ and $a = P(A = 1)$.

Rule II: Boolean "AND," or $C = AB$, corresponds to

$$c = a \cdot b$$

where $c = P(C = 1)$, $b = P(B = 1)$ and $a = P(A = 1)$.

Rule III: Boolean "OR," or $C = A + B$, corresponds to

$$c = a + b - a \cdot b$$

where $c = P(C = 1)$, $b = P(B = 1)$ and $a = P(A = 1)$.

By applying "AND," "OR" and "NOT," any type of Boolean logic can be mapped to an arithmetic equation of signal probabilities.

Example: Boolean "XOR" expressed as $C = A\bar{B} + \bar{A}B$ corresponds to

$$c = a \cdot (1 - b) + (1 - a) \cdot b$$

where $c = P(C = 1)$, $b = P(B = 1)$ and $a = P(A = 1)$.

2.2 Probabilistic Gate Models

A probabilistic gate model (PGM) relates a gate's output probability to its input and error probabilities, according to the function and malfunction of the gate [3]. In general, the output probability of a gate can be calculated by the following equation,

$$Z = P(\text{output "1"} \mid \text{gate faulty}) \cdot P(\text{gate faulty}) + P(\text{output "1"} \mid \text{gate not faulty}) \cdot P(\text{gate not faulty}) \quad (1)$$

Assume a von Neumann error, which flips a gate's correct output and resembles the behavior of a soft error, with an error rate ε; let $\varepsilon = P(\text{gate faulty})$ and $p = P(\text{output "1"} \mid \text{gate not faulty})$, we obtain the following equation for any logic gate/function,

$$Z_v = (1 - p) \cdot \varepsilon + p \cdot (1 - \varepsilon) \quad (2)$$

For a two-input AND gate, for example, if X_1 and X_2 represent the input signal probabilities, the output signal probability is then $X_1 X_2$ for a fault-free gate. Given a probabilistic von Neumann error, the output probability of an AND gate is then $Z_v = (1 - X_1X_2)\varepsilon + X_1X_2(1 - \varepsilon)$.

2.3 Computational Algorithms Using PGMs

A simple algorithm can be obtained by the iterative execution of a gate's PGM according to the specific structure of a circuit. The executions of PGMs from a circuit's primary inputs to its outputs produce the signal probability of each output. An overall reliability is obtained by multiplying the reliabilities of all outputs. In this simple PGM algorithm, it is assumed that all signals under consideration are mutually independent. However, this is not true when there are reconvergent fanouts and thus the signals are correlated rather than independent. Hence, the simple PGM algorithm, albeit simplistic to implement, results in an approximate evaluation.

An accurate algorithm accounts for the signal dependencies in a circuit. In a circuit without feedbacks, if all the inputs are mutually independent, reconvergent fanouts are the only topological structures that introduce signal dependencies into the circuit. Fig. 1 (a) shows a simple reconvergent fanout. The fanout originates at point B and reconverges at point D.

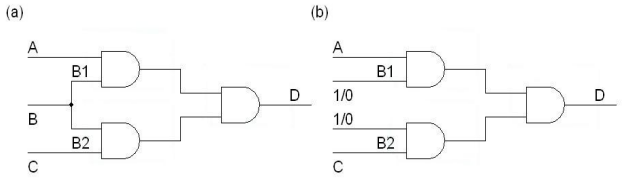

Fig. 1 (a) A reconvergent fanout; (b) A fanout decomposition

If the input to a fanout has a deterministic value, i.e., "1" or "0", the statistical dependence of the two fanout branches is effectively eliminated. Per the definition of statistical independence, we have $P(B_1=1, B_2=1) = P(B_1=1)\,P(B_2=1)$ if and only if $P(B=1)$ equals to 1 or 0. As shown in Fig. 1 (b), the fanout is decomposed into two equivalent circuits with deterministic inputs "1" and "0", virtually containing no fanouts. In this way, signal dependencies are eliminated by the fanout decomposition. Subsequently, the simple PGM algorithm is used to calculate the output probabilities of the two circuits. These obtained output probabilities are then used to evaluate the signal probability at point D, by $Z = Z_1P + Z_0(1 - P)$, where Z_1 and Z_0 are output probabilities when the fanout input is set to "1" and "0", and P is the signal probability at point B.

Given a circuit, signals are traced back from each primary output and all reconvergent fanouts in the paths leading to that output are identified. For each reconvergent fanout, the circuit is decomposed into two sub-circuits as shown above. This process is repeated for every reconvergent fanout until all the reconvergent fanouts are eliminated in the sub-circuits. Finally, the simple PGM algorithm is used to obtain the output probabilities of each sub-circuit and the input probability of each reconvergent fanout, which are then used to obtain the final reliability of the circuit.

As the computation required in the accurate algorithm presented above doubles for each dependant reconvergent fanout, the computational complexity of the algorithm is exponential in the number of dependant reconvergent fanouts, or a complexity of $O(N2^{N_f})$, where N is the total number of gates and N_f is the number of dependent reconvergent fanouts. Like all other accurate algorithms, it makes the analysis of VLSI circuits intractable.

3. RELIABILITY EVALUATION USING STOCHASTIC COMPUTATIONAL MODELS

3.1 Background

3.1.1 Stochastic Computation

Stochastic computation was first introduced in the 1960s for logic circuit design [10, 11], but its origin can be traced back to von

Neumann's seminal work on fault tolerance [15]. In stochastic computation, real numbers are represented by random binary bit streams, which are usually generated by a Bernoulli process. Information is carried in the statistics of the binary streams [16]. Stochastic computation has such advantages as computational simplicity, fault tolerance and high speed [17], which, as will be shown, apply to our approach to reliability evaluation. The essential components of stochastic computation include stochastic processing elements and stochastic sequence generators [16]. As to our application, we use uniformly distributed random bit streams to encode signal probabilities. A certain probability is represented by a proportional number of bits set to a specific value, which is usually the proportion of the mean number of 1's in a bit stream. Basic stochastic processing elements are used to construct stochastic computational models (SCMs). In stochastic computation, an analysis is simplified by using time redundancy.

3.1.2 Computation based on Stochastic Logic

Instead of representing signal probabilities as real numbers in the unit interval $(0,1)$, stochastic logic encodes signal probabilities into binary bit streams serially in the time domain. Fig. 2 illustrates a stochastic encoding [14].

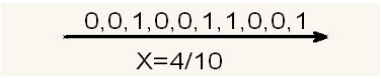

Fig. 2 A stochastic encoding

Stochastic computation transforms Boolean logic operations into probabilistic computations in the real domain. Although each bit is processed by a logic gate, signal operations are no longer Boolean but are probabilistic computations by stochastic logic. For instance, multiplication can be implemented by an AND gate [16], as shown in Fig. 3 (a). In section 2.1, it is shown that Boolean logic operations are mapped to arithmetic operations. Here we observe that an AND gate implements the multiplication of *Rule II*. Similarly, any logic gate can implement a corresponding probabilistic operation as dictated by a mapping rule or a combination of the rules. Fig. 3 (b) shows the stochastic operation performed by an XOR gate. This computational capacity of stochastic logic elements guides us to pursue using stochastic computational models for the numerical evaluation of circuit reliability. An important feature in stochastic computation is that it propagates probabilistic values rather than deterministic ones, which results in inevitable random fluctuations in the representation of probabilities.

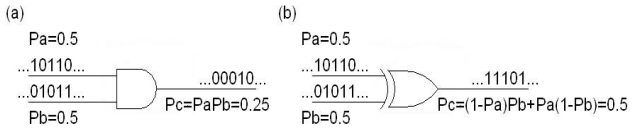

Fig. 3 (a) Stochastic multiplication; (b) Stochastic XOR logic

3.2 Stochastic Computational Models

Stochastic computational models (SCMs) are based on the operations of stochastic logic and the notions of PGMs. As shown in section 2.2, any gate affected by a von Neumann error can be modeled using the PGM equation (2). In fact, equation (2) can be implemented by a stochastic logic of XOR as follows:

$$\mathrm{XOR}_{\mathrm{sto}}(\mathrm{p},\varepsilon) = \mathrm{p}(1-\varepsilon) + (1-\mathrm{p})\varepsilon \qquad (3)$$

where p is the fault-free output probability and ε the gate error rate. Equation (3) shows that the PGM equation (2) can be implemented by a stochastic XOR logic regardless of the type of logic gate modeled by the PGM. Therefore, an SCM can be obtained by adding an XOR gate to an unreliable gate and using the XOR to implement the gate error rate. This is shown in Fig. 4.

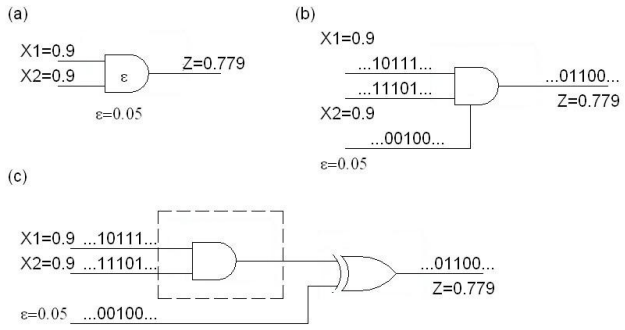

Fig. 4 (a) An unreliable AND gate; (b) A stochastic logic implementation; (c) An SCM implementation with XOR

Fig. 4 shows an unreliable AND gate and its SCM implementation, for which we have

$$\mathrm{p} = \mathrm{X}_1\mathrm{X}_2 \qquad (4)$$

$$\mathrm{XOR}_{\mathrm{sto}}(\mathrm{p},\varepsilon) = \mathrm{p}(1-\varepsilon) + (1-\mathrm{p})\varepsilon \qquad (5)$$

As indicated by equation (5), an SCM is universal in the sense that it can be constructed for an arbitrary logic gate. An SCM significantly reduces the computational complexity of a probabilistic analysis by using redundancy in the time domain and the XOR logic for processing the gate error rate.

3.3 A Computational Approach Using SCMs

A stochastic computational network can be constructed using the SCMs of gates in a circuit for the reliability evaluation of the circuit. The computational network is a nonlinear structure constituted by SCMs. Feeding stochastic input sequences into the network and propagating them from primary inputs to outputs give the output probabilities. A distinguishing feature of the SCM approach is that it handles reconvergent fanouts effortlessly. As shown in section 2.3, the statistical dependence of a fanout's branches is eliminated if and only if the input to the fanout is 1 or 0. When signals are processed in the form of binary bit streams, which consist of 1's and 0's, logic operations do not need to consider the correlations caused by reconvergent fanouts. Instead, signal dependencies are inherently maintained in the distribution patterns of the random binary bit streams. This can be seen as follows. If an AND gate has two independent random bit streams X_1 and X_2 as inputs, its output will be a bit sequence encoding $Z = X_1X_2 = 0.81$ for $X_1 = X_2 = 0.9$. If the inputs are not independent, the output will depend on the correlation of the two input signals. Fig. 5 shows an example where the two inputs are totally dependent, which results in $Z = X_1 = X_2 = 0.9$. In stochastic processing, signal dependencies are well maintained and propagated to the next logic level.

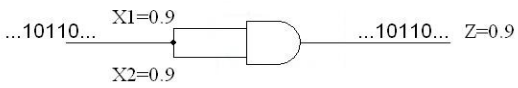

Fig. 5 Signal correlations maintained in stochastic logic processing

We demonstrate the SCM approach by taking the benchmark circuit C17 as an example. Fig. 6 shows the schematic of C17.

First, a stochastic computational circuit is obtained by adding an XOR gate to each of the gates in C17, as shown in Fig. 7. Then input signals, as well as the gate error rate, which is now an input to the XOR gates, are initialized by generating random bit streams. The streams are then propagated through the circuit. In the end, the output bit sequences are read out and decoded into probabilities and reliability.

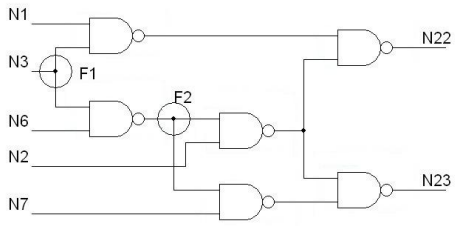

Fig. 6 Schematic of C17

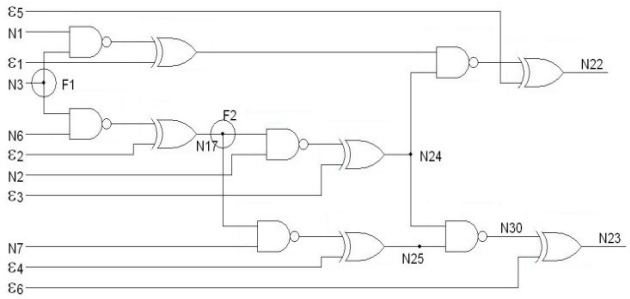

Fig. 7 Stochastic C17 using SCMs

As shown in Fig. 7, a fanout F2 originates at node N17 and reconverges at node N30. In one of our experiments, we observed two sequences representing the same probability 0.095 at nodes N24 and N25, as well as an output sequence at N30 standing for a probability 0.9064. However, if the two signals at N24 and N25 are independent, we should have got an output probability of 0.991 at Node 30. This confirms that the signal correlations due to reconvergent fanouts are well maintained in the stochastic computational network.

An evaluation procedure using the SCM approach is as follows:

1. Construct the stochastic computational circuit by adding an XOR gate to each of the logic gates in the circuit;
2. Generate initial random bit streams encoding signal probabilities of primary inputs and the error rate of each gate in the circuit;
3. Propagate the binary streams from the primary inputs to the outputs and obtain a random bit stream for each output;
4. Decode the signal probability and calculate the reliability of each output from the obtained random bit stream.

3.4 Performance

3.4.1 Accuracy

In SCMs, signal probabilities are carried in the random binary bit streams and signal dependencies are accounted for in the stochastic logic processing network. Hence, the reliability obtained using the SCM approach is accurate and its precision is only limited by such factors as the resolution in the representation of the bit streams, the random permutation and random fluctuation of the stochastic sequences. A detailed error analysis is given in section 3.5.

3.4.2 Complexity

There are two major steps in the stochastic computation for circuit reliability: one is the generation of input random bit streams, which consumes over 90% of the total run time, and the other is the propagation of the bit sequences through the circuit. For the random sequence generation, the SCM approach has a linear complexity with the number of 1's in the sequence needed to be generated, which is proportional to the sequence length for a fixed error rate. Thus it often has a linear complexity with the sequence length. For sequence propagations, the SCM approach has a complexity that increases linearly with the number of gates in a circuit, as well as the length of the random bit sequences.

3.5 Error Analysis

3.5.1 Resolution

The sequence length is an important parameter since it determines the resolution of the results. If we choose a sequence length L of 10, for example, the resolution is 0.1. This means that any probability less than 0.1 cannot be represented. An error due to a limited resolution is illustrated in Fig. 8. Fig. 8 (a) shows a case where $X_1=0.2$ and $X_2=1$. Since $Z= X_1 X_2$, it can be obtained that $Z=0.2$ from the output binary stream. This is an accurate result. Fig. 8 (b) shows a scenario where $X_1=0.8$ and $X_2=0.8$, for which we obtain an output $Z=0.6$, while the correct output should be 0.64. In our experiments, we used a sequence of 1000, which gives a resolution of 0.001. Since the result is rounded to its nearest available representation, the maximum error due to this resolution is 0.0005. This indicates that the results we obtained in a single experiment will have a precision error of up to 0.0005.

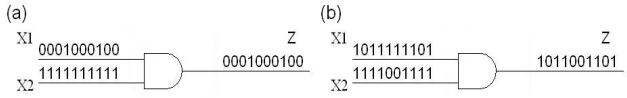

Fig. 8 Resolutions in stochastic computation (a) A desired output; (b) An output imprecision due to a limited resolution

3.5.2 Random Permutation

Errors can also be caused by the random permutation of bits in a sequence. Fig. 9 illustrates an example of a randomized permutation. The logic operation in Fig. 9 (a) gives the desired output value, while the operation in Fig. 9 (b) gives what we consider as an error. In general, longer sequences tend to be better randomized. However, random permutations are probabilistic in nature and therefore they do not always provide the desired results. The error due to random permutation is considered as "noise" and contributes to the facts that the output values are probabilistic rather than deterministic in a stochastic network. This is explained in the following subsection.

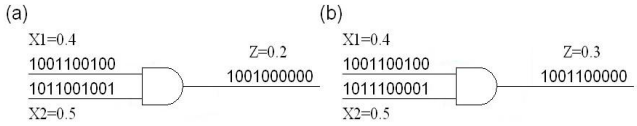

Fig. 9 Random permutations in stochastic computation (a) A desired permutation; (b) A permutation resulting in what is considered as an error

3.5.3 Inherent Random Fluctuation

Random fluctuation is an inherent feature in stochastic computation [14]. Simulations of C17 show that the result of every experiment fluctuates around the expected mean value, as shown in Fig. 10. The result of an experiment is an output sequence obtained for a given input combination. This type of error can be analyzed quantitatively by investigating the mean and variance. The law of large numbers states that the average result obtained from a large number of experiments is close to their mean value. Assuming each experiment X_k is a random variable with the same mean μ and variance σ^2, we have

$$E\left[\frac{1}{n}\sum_{k=1}^{n} X_k\right] = \frac{1}{n}\sum_{k=1}^{n} E[X_k] = \frac{1}{n} n\mu = \mu \quad (6)$$

$$E[(x-\mu)^2] = Var\left(\frac{1}{n}\sum_{k=1}^{n} X_k\right) = \frac{\sigma^2}{n} \quad (7)$$

The error can be measured by the standard deviation,

$$e = |x-\mu| \approx \frac{\sigma}{\sqrt{n}} \quad (8)$$

where *n* is the number of experiments performed. From equation (8) we obtain that the error is proportional to $\frac{1}{\sqrt{n}}$. Therefore, a higher accuracy can be obtained by increasing the number of experiments. Note that, as the precision is limited by the resolution, increasing the number of experiments does not always give a better accuracy.

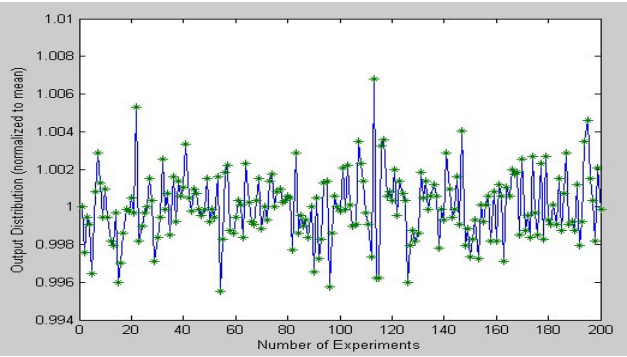

Fig. 10 Random fluctuations in stochastic computation

Fig. 11 shows the distribution of the results from 1000 experiments. According to the Central Limit Theorem, the distribution of a large number of samples approaches a Gaussian distribution as the sample size increases. Here we show that a Gaussian distribution with the mean and variance calculated from the experimental data fits the distribution of the data very well. The deterministic input signals have been affected by noise and become probabilistic in the stochastic processing network.

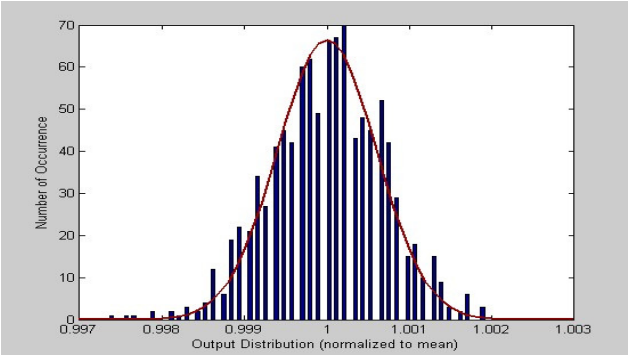

Fig.11 Output distributions are approximately Gaussian

4. SIMULATION RESULTS

A number of LGSynth91 and ISCAS-85 benchmark circuits [18] are used to evaluate the performance of our proposed approach. The simulations were run on a 2.66-GHz Pentium microprocessor with 2 GB memory. The accurate PGM algorithm is used for comparison with the SCM approach and the results are shown in Table 1. Large benchmarks in ISCAS-85 are simulated to demonstrate the efficiency of the SCM approach. The runtime indicates that the SCM approach has significant advantages in dealing with large circuits.

4.1 Accuracy Comparison

An accuracy comparison is performed between the accurate PGM algorithm and the SGM approach. Table 1 shows overall reliabilities for *ε=0.05.* A maximum of 1000 inputs are used. As can be seen, the SCM approach shows highly accurate results - the maximum relative error is around 0.1% for using a sequence length of 1000. The accuracy can be further improved by either increasing the sequence length, or increasing the number of experiments for each input combination. However, these methods significantly increase the runtime, while they only improve the accuracy slightly, as is dictated by equation (8). Since the runtimes of PGM do not include the longer time needed for identifying and decomposing reconvergent fanouts, they appear to be less in Table 1 than those of SCM for circuits of such sizes.

Table 1 Accuracy comparisons between the PGM and SCM approaches

Circuits	Characteristics			PGM ε=0.05		SCM ε=0.05					
						Sequence length 1000			Sequence length 10000		
	gates	inputs	outputs	Reliability	Runtime	Reliability	Relative error	Runtime	Reliability	Relative error	Runtime
C17	6	5	2	0.7582	0.00073s	0.7574	0.106%	0.109s	0.7580	0.0264%	4.016s
cu	43	14	11	0.3865	0.10063s	0.3862	0.0776%	14.154s	0.3866	0.0259%	374.818s
z4ml	45	7	4	0.2546	0.05183s	0.2548	0.0786%	1.906s	0.2543	0.118%	51.322s
pcle	61	19	9	0.2342	0.06808s	0.2343	0.0427%	21.531s	0.2342	0%	559.093s
decod	22	5	16	0.3426	0.001s	0.3430	0.117%	0.586s	0.3423	0.088%	6.893s
parity	15	16	1	0.6029	0.01251s	0.6027	0.0332%	5.186s	0.6028	0.017%	143.649s
majority	10	5	1	0.8623	0.00049s	0.8633	0.116%	0.206s	0.8627	0.046%	5.223s
pm1	41	16	13	0.3886	0.02685s	0.3886	0%	14.773s	0.3887	0.026%	390.948s
x2	38	10	7	0.3822	0.21819s	0.3821	0.0262%	12.568s	0.3820	0.052%	364.524s
mux	50	21	1	0.7920	0.09827s	0.7915	0.0631%	16.382s	0.7920	0%	440.123s

Table 2 Reliability evaluation of ISCAS85 circuits using the SCM approach

Circuits	Characteristics			ε=0.005			ε=0.01			ε=0.05		
	gates	inputs	outputs	Average Reliability	Overall Reliability	Time	Average Reliability	Overall Reliability	Time	Average Reliability	Overall Reliability	Time
C880	383	60	26	0.9219	9.17×10^{-2}	25.1s	0.8914	3.94×10^{-2}	33.7s	0.7440	1.10×10^{-3}	119.6s
C1355	546	41	32	0.9245	5.10×10^{-2}	32.6s	0.9011	2.29×10^{-2}	47.6s	0.7509	8.35×10^{-5}	169.5s
C1908	880	33	25	0.8624	1.50×10^{-2}	55.6s	0.8175	5.40×10^{-2}	76.3s	0.6956	1.69×10^{-4}	271.2s
C2670	1193	157	64	0.9336	6.70×10^{-3}	77.7s	0.9047	7.94×10^{-4}	105.8s	0.7875	8.52×10^{-8}	370.0s
C3540	1669	50	22	0.7978	3.90×10^{-3}	99.8s	0.7466	9.44×10^{-4}	144.3s	0.6687	1.10×10^{-4}	517.6s
C5315	2307	178	123	0.9419	7.67×10^{-4}	157.2s	0.9102	2.92×10^{-5}	209.7s	0.7773	9.44×10^{-13}	745.7s
C6288	2416	32	32	0.6887	3.68×10^{-6}	147.9s	0.6168	1.00×10^{-7}	214.9s	0.5479	3.09×10^{-9}	756.5s
C7552	3512	207	108	0.9203	5.44×10^{-5}	216.9s	0.8797	2.76×10^{-7}	314.1s	0.7544	2.08×10^{-15}	1108.5s

An observation is that the runtime is dominated by the procedure of generating random bit sequences. If we choose fault-free deterministic inputs, the only sequences need to be randomized as initial inputs are the gate error rates. As they are independent processes, it would be possible to further reduce the run time on random bit generations by using parallel processings whenever possible.

4.2 Implementations on Large Benchmarks

The SCM approach is readily applicable to large scale logic circuits. We chose to use a sequence length of 1000 since it gives an average relative error less than 0.1%, and at the same time requires a relatively short execution time. The simulation results are shown in Table 2. We show the outputs' overall reliability, defined as the product of all output reliabilities, and the average reliability for $\varepsilon=0.005$, $\varepsilon=0.01$ and $\varepsilon=0.05$. The obtained overall reliabilities are very low since many circuits have a large number of outputs. The least runtime is obtained for $\varepsilon=0.005$ as it requires the least time for initiating random bit streams.

5. CONCLUSION AND FUTURE WORK

There has increasingly been a need for accurate and efficient reliability evaluation tools for the design and test of VLSI circuits. In this paper, a novel computational approach using stochastic computation is proposed for the reliability evaluation of logic circuits. This approach uses stochastic computational models (SCMs) and accurately evaluates a circuit's reliability with a precision limit imposed by factors such as the resolution, random permutation and random fluctuation of the random binary bit streams used in stochastic computation. Simulation results indicate an average error rate less than 0.1% for the LGSynth91 benchmark circuits.

The proposed SCM approach has a computational complexity that increases linearly with the length of the random bit sequences and the number of gates in a circuit. Compared to the simple simulation of random vectors, the stochastic approach is advantageous in terms of efficiency and scalability. It is therefore potentially useful in the design and test of VLSI circuits and systems. The SCM also provides a general framework for the computational modeling of complex systems. Our ongoing work includes an extension of the current reliability evaluation approach to include more fault models and an exploration of the use of stochastic computational elements in other applications such as the testing and diagnosis of VLSI circuits and systems.

6. REFERENCES

[1] J. Han and P. Jonker, "A system architecture solution for unreliable nanoelectronic devices," IEEE Trans. on Nanotechnology, vol. 1, no. 4, pp. 201–208, December 2002.

[2] S. Krishnaswamy, G. Viamontes, I. Markov, and J. Hayes, "Accurate reliability evaluation and enhancement via probabilistic transfer matrices," in Proc. Des. Autom. Test Eur., 2005, pp. 282–287.

[3] J. Han, E.R. Taylor, J.B. Gao, and J.A.B. Fortes, "Faults, error bounds and reliability of nanoelectronic circuits," Proc. Intl. Conf Appl.-Specific Sys., Arch. & Proc. ASAP'05, Samos, Greece, Jul. 2005, pp. 247-253.

[4] E.R. Taylor, J. Han, J.A.B. Fortes, "Towards accurate and efficient reliability modeling of nanoelectronic circuits," In: IEEE Intl. Conf. on Nanotechnology, Cincinnati, OH, USA, pp. 395–398 (2006)

[5] A. Abdollahi, "Probabilistic Decision Diagrams for Exact Probabilistic Analysis," Proc. Int'l Conference on Computer Aided Design, 2007.

[6] T. Rejimon and S. Bhanja, "Scalable probabilistic computing models using Bayesian networks," in Proc. Int. Midwest Symp. Circuits Syst., 2005, pp. 712–715.

[7] N. Mohyuddin, E. Pakbaznia, M. Pedram, "Probabilistic Error Propagation in Logic Circuits Using the Boolean Difference Calculus," in IEEE Intl. Conf. on Comp. Des., Lake Tahoe, CA, USA, pp. 7-13 (2008)

[8] S. Sivaswamy, K. Bazargan, M. Riedel, "Estimation and Optimization of Reliability of Noisy Digital Circuits," in International Symposium on Quality Electronic Design, San Jose, CA, USA, pp. 213-219 (2009)

[9] M. R. Choudhury and K. Mohanram, "Reliability analysis of logic circuits," IEEE TCAD, vol. 28, no. 3, pp. 392–405, March 2009.

[10] B. R. Gaines, "Stochastic Computing", Spring Joint Computer Conf., 1967, Vol. 30, pp. 149-156.

[11] W. J. Poppelbaump, C. Afuso and J. W. Esch, "Stochastic Computing Elements and Systems", Proceedings of the Fall Joint Computing Conf. 1967, pp. 631-644.

[12] K. P. Parker, and E. J. McCluskey, "Probabilistic Treatment of General Combinational Networks", IEEE Trans. on Computers, vol. C, no. 24, pp. 668–670, June 1975.

[13] I. Bahar, J. L. Mundy, and J. Chen, "A probabilistic-based design methodology for nanoscale computation," in Proc. Int. Conf. Comput.-Aided Des., 2003, pp. 480–486.

[14] X. Li, W.K. Qian, M. Riedel, K. Bazargan, D. Lilja, "A Reconfigurable Stochastic Architecture for Highly Reliable Computing," Proc. Great Lakes Symp. VLSI (GLVLSI), Boston, MA, USA, pp. 315-320 (2009)

[15] J. von Neumann, "Probabilistic logics and the synthesis of reliable organisms from unreliable components," Automata Studies, Shannon C.E. & McCarthy J., eds., Princeton University Press, pp. 43-98, 1956.

[16] B. Brown and H. Card, "Stochastic neural computation I: Computational elements," IEEE Tran. Computers, vol. 50, pp. 891–905, Sept. 2001.

[17] S. Sharifi Tehrani, S. Mannor, and W. J. Gross, "Survey of stochastic computation on factor graphs," in Proc. 37th IEEE Int. Symp. Multiple-Valued Logic, Oslo, Norway, May 2007, pp. 54–59.

[18] E. M. Sentovich, K. J. Singh, L. Lavagno, C. Moon, R. Muigai, A. Saldanha, H. Savoj, P. R. Stephan, R. K. Brayton, and A. L. Sangiovanni-Vincentelli, SIS: A System for Sequential Circuit Synthesis. Technical Report UCB/ERL M92/41, Electronics Research Lab, Univ. of California, Berkeley, CA 94720, May 1992.