Reliability Analysis of Logic Circuits

Mihir R. Choudhury, *Student Member, IEEE*, and Kartik Mohanram, *Member, IEEE*

***Abstract*—Reliability of logic circuits is emerging as an important concern in scaled electronic technologies. Reliability analysis of logic circuits is computationally complex because of the exponential number of inputs, combinations, and correlations in gate failures. This paper presents three accurate and scalable algorithms for reliability analysis of logic circuits. The first algorithm, called observability-based reliability analysis, provides a closed-form expression for reliability and is accurate when single gate failures are dominant in a logic circuit. The second algorithm, called single-pass reliability analysis, computes reliability in a single topological walk through the logic circuit. It computes the exact reliability for circuits without reconvergent fan-out, even in the presence of multiple gate failures. The algorithm can also handle circuits with reconvergent fan-out with high accuracy using correlation coefficients as described in this paper. The third algorithm, called maximum-k gate failure reliability analysis, allows a constraint on the maximum number (k) of gates that can fail simultaneously in a logic circuit. Simulation results for several benchmark circuits demonstrate the accuracy, performance, and potential applications of the proposed algorithms.**

***Index Terms*—Gate failures, logic circuits, reliability analysis.**

I. Introduction

IT IS WIDELY acknowledged that there will be a sharp increase in manufacturing defect levels and transient fault rates in future electronic technologies, e.g., [1]–[5]. Defects and faults impact performance and limit the reliability of electronic systems. This has led to considerable interest in practical techniques for reliability analysis that are accurate, robust, and scalable with design complexity. Reliability analysis of logic circuits refers to the problem of evaluating the effects of errors due to noise at individual transistors, gates, or logic blocks on the outputs of the circuit. The models for noise range from highly specific decomposition of the sources, e.g., single-event upsets, to highly abstract models that combine the effects of different failure mechanisms.

Reliability analysis of logic circuits is computationally complex because of the exponential number of inputs, combinations, and correlations in gate failures. Standard techniques for reliability analysis use fault injection and simulation in a Monte Carlo framework. Although parallelizable, they are still not efficient for use on large circuits. Analytical methods for reliability analysis are applicable to very simple structures such as two-input and three-input gates, and regular fabrics [6], [7]. Although they can be applied to large multilevel circuits with simplifying assumptions and compositional rules, there is a significant loss in accuracy. Recent advances in reliability analysis are based on probabilistic transfer matrices (PTMs) [8], Bayesian networks [9], and Markov random fields [10], [11]. However, all approaches require significant runtimes for small benchmark circuits. In the PTM approach, this can be attributed to the storage and manipulation of large algebraic decision diagrams (ADDs) used to represent the probabilistic behavior of the circuit. In the Bayesian network approach, the large runtimes arise from large conditional probability tables that support Bayesian network operations. The use of Markov random fields becomes computationally intensive for arbitrary multilevel logic circuits because it involves minimization of a complex Gibbs distribution function with a large number of variables. Many reliability analysis techniques have been proposed in the context of soft errors [12], [13]. These techniques predict soft error rate in circuits, accounting for electrical masking, and latching-window masking in addition to logical masking. Since these techniques are specific to soft errors, they focus mainly on predicting single gate failure effects.

This paper presents three accurate and scalable algorithms for reliability analysis of logic circuits. The first algorithm, called observability-based reliability analysis, uses observability metrics to quantify the impact of a gate failure on the output of the circuit. The observability-based approach provides a closed-form expression for circuit reliability as a function of the failure probabilities and observabilities of the gates. The closed-form expression is accurate when the probability of a single gate failure is significantly higher than the probability of multiple gate failures.

The second algorithm, called single-pass reliability analysis, leverages insights into the effects of multiple gate failures derived from observability-based reliability analysis. In this algorithm, gates are topologically sorted and processed in a single pass from the inputs to the outputs. Topological sorting ensures that before a gate is processed, the effects of multiple gate failures in the transitive fan-in cone of the gate are computed and stored at the inputs of the gate. Using the joint signal probability distribution of the gate's inputs, the propagated error probabilities from its transitive fan-in stored at its inputs, and the failure probability of the gate, the cumulative effect of failures are computed at the output of the gate. The single-pass reliability analysis algorithm is provably exact for circuits without reconvergent fan-out. Reconvergent fan-out introduces correlations when error probabilities are combined at the point of reconvergence, which is handled using correlation coefficients. The accuracy of the algorithm can be improved by using

Manuscript received April 10, 2008; revised July 23, 2008. Current version published February 19, 2009. This research was supported in part by CAREER Award CCF-0746850 from the National Science Foundation and in part by the A. Richard Newton Graduate Scholarship. This paper was recommended by Associate Editor N. K. Jha.

The authors are with the Department of Electrical and Computer Engineering, Rice University, Houston, TX 77005 USA (e-mail: mihir@rice.edu; kmram@rice.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCAD.2009.2012530

0278-0070/$25.00 © 2009 IEEE

higher order correlation coefficients to capture the correlation effects. However, the computational complexity increases as the number of correlation coefficients increases. This paper explores the tradeoff between accuracy and computational complexity for 0, 4, and 16 correlation coefficients.

The above algorithms for reliability analysis use an independent gate failure model, in which there is a nonzero probability of a large number of gates failing simultaneously. However, in practice, it is reasonable to expect an upper bound, k, on the maximum number of gates that can fail simultaneously in the logic circuit. Termed the maximum-k gate failure model, the gate failures remain independent although the global constraint due to the upper bound k introduces correlation among gate failures. The third algorithm, called maximum-k gate failure reliability analysis, is based on a technique that generates a set of gate failures (of cardinality $\leq k$) according to the independent gate failure model. Given this set of gate failures, single-pass reliability analysis is used to evaluate their effect on the output(s) of the circuit. For a logic circuit with N gates, generating samples of failed gates is challenging because the size of the sample space is $O(N^k)$ and all brute-force solutions are computationally intensive. The sampling algorithm proposed in this paper has a computational complexity of $O(Nk^2)$. Simulation results for several benchmark circuits demonstrate the accuracy, performance, and potential applications of the proposed analysis algorithms to guide logic design for both redundancy-free and redundancy-based reliability enhancement in logic circuits [14]–[16].

This paper is an extended version of [17] and is organized as follows. Section II provides a background in reliability analysis. Section III describes the observability-based algorithm for reliability analysis. Section IV describes the single-pass algorithm for reliability analysis. Section V describes the maximum-k gate failure model and an efficient algorithm for reliability analysis. Section VI presents simulation results. Section VII is a conclusion.

II. Background

The classical model for errors due to noise in a logic circuit was introduced by von Neumann in 1956 [6]. Noise at a gate is modeled as a binary symmetric channel (BSC), with a crossover probability ϵ. In other words, following the computation at the gate, the BSC can cause the gate output to toggle symmetrically (from $0 \rightarrow 1$ or $1 \rightarrow 0$) with the same probability of error, ϵ. Each gate has an $\epsilon \in [0, 0.5]$ associated with it, where ϵ equals zero for an error-free gate, and ϵ equals 0.5 for a perfectly noisy gate (a gate with random output). It is unrealistic for a gate to have $\epsilon > 0.5$ because it would mean that the output of the gate is more likely to be faulty than correct. In such a case, adding a NOT gate at the output of the gate would make the combination of the two gates more reliable. For example, if an AND gate in a circuit has a failure probability 0.7, then adding a NOT gate with a failure probability x at the output of the AND gate means that the failure probability of the combination is $\epsilon' = 0.7x + 0.3(1 - x)$. For any value of x in [0,1], $\epsilon' \leq 0.7$. Hence, in this paper, we only consider gate failure probability in [0, 0.5]. Note that gates are assumed to fail independently of each other. Although this may not be a realistic assumption, since effects of noise are potentially localized and correlated, it helps to simplify reliability analysis while still providing valuable insights into circuit reliability. In this paper, the phrases "gate is in error," "erroneous gate," and "gate has failed" are used to mean that a particular gate is known to produce an incorrect output.

The BSC model allows the effects of different sources of noise such as crosstalk, terrestrial cosmic radiation, electromagnetic interference, etc., to be combined into the failure probability ϵ. At the electrical level, the effects of noise can usually be modeled by a probability distribution about the nominal voltage, instead of a single number. Traditionally, a Gaussian distribution has been shown to be a good approximation for this distribution. Reliability analysis is a problem in the discrete domain (involving Boolean logic) and the noise at the gates, modeled by a Gaussian distribution, is in the continuous domain. As a result, reliability analysis would require Boolean operations on Gaussian distributions to propagate errors from gate inputs to the output. Since the resulting distribution may not be Gaussian, either scalability or accuracy will be lost in this approach. Since there are multiple sources of noise and since it is desirable to work with a higher abstraction for the combined effects of these sources, the Gaussian model for noise is discretized by computing the probability that the nominal voltage exceeds the noise margin for both low and high output voltages. Without loss of generality, the average probability that the low and high voltages exceed the noise margin is used to estimate the gate failure probability ϵ for the rest of this paper. Note that all the algorithms proposed in this paper can be extended to handle $0 \rightarrow 1$ and $1 \rightarrow 0$ gate failure probabilities separately.

Reliability of a logic circuit is defined as the probability of error at the output of the logic circuit δ as a function of failure probabilities $\vec{\epsilon}$ of the gates, where $\vec{\epsilon}$ is the vector containing the failure probabilities $\{\epsilon_1, \epsilon_2, \ldots\}$ of the gates in the circuit. Reliability $\delta(\vec{\epsilon})$ can lie in the interval from zero to one. The expected number of simultaneous gate failures is given by $\sum \epsilon_i$. Thus, the value of ϵ_is determines the distribution of the number of simultaneous gate failures. Note that if $\epsilon_i \neq 0 \forall i$, then the number of simultaneous gate failures can vary from zero to all the gates in the circuit with some nonzero probability (although the probability of a large number of simultaneous gate failures is very low). Such a wide range for the number of gate failures is an artifact of the independent gate failure model. The independent gate failure model can be improved by imposing a limit on the number of simultaneous gate failures. For instance, in the single gate failure model, the limit is one. This paper describes an efficient reliability analysis algorithm when the limit on the number of simultaneous gate failures is $k \leq N$.

In special cases, reliability analysis can leverage existing techniques from switching activity computation [18] and observability computation [19]. If failures at only the primary inputs of the circuit are considered, and gates are assumed to be error free, then the problem of reliability analysis can be solved using switching activity computation algorithms. The $0 \rightarrow 1$ and $1 \rightarrow 0$ error probabilities can be thought of as the switching probability, and the switching activity of every gate can be computed. The resulting $0 \rightarrow 1$ and $1 \rightarrow 0$ switching probabilities at a gate are the error probabilities at that gate. However, in the general case, reliability analysis becomes more complex when gate failures are also considered, because of the exponential number of combinations of gate failures. If only single gate failures are considered, the reliability analysis problem can be solved by simply computing the observability of the gates. The contribution of each gate to the error probability of an output (y) is given by the failure probability of the gate (ϵ) times the observability of the gate at the output y. In the general case, reliability analysis for multiple gate failures becomes more complex because the observabilities of the gates are not independent of each other, and the gate failures can change observability of the gates. These effects are described in greater detail in Section III-A.

The traditional approach to reliability analysis uses fault injection and simulation in a Monte Carlo framework. Recent progress in reliability analysis has seen the use of PTMs [8], Bayesian networks [9], and Markov random fields [10], [11]. Without exception, these approaches suffer from the problem of scalability. Monte Carlo simulations have the added disadvantage of inflexibility, since the entire simulation has to be repeated for any change in circuit structure or $\vec{\epsilon}$. PTM-based reliability analysis uses transfer matrices to represent input–output behavior of noisy circuits. PTMs store the probability of occurrence of every input–output vector pair for each level in the circuit to compute the probability of error at the output of the circuit. This leads to massive matrix storage and manipulation overhead. Even with compaction of the matrices using ADDs, the high runtimes reported for benchmark circuits with 20–50 gates suggest their inapplicability to large circuits. Although this problem is somewhat mitigated in the Bayesian network approach for small circuits, manipulating Bayesian networks for large circuits is potentially intractable. A probabilistic design methodology based on Markov random fields is presented in [10] that uses the Gibbs distribution to characterize reliability in terms of entropy and noise in terms of thermal energy. Evaluating reliability using this technique becomes computationally intensive for arbitrary multilevel logic circuits because it involves minimization of a complex Gibbs distribution function with a large number of variables. This technique is more suitable for evaluating reliability of regular redundancy architectures like triple modular redundancy and NAND multiplexing [11]. Alternatively, analytical approaches developed to study fault-tolerant approaches like NAND multiplexing and majority voting can be used for reliability analysis [6], [7]. However, the simple compositional rules that these approaches use work best on regular structures. When used on irregular multilevel structures such as logic circuits, they suffer significant penalties in accuracy even on small circuits.

III. Observability-Based Reliability Analysis

In this section, an intuitive approach to reliability analysis is described. It is based upon the observation that a failure at a gate close to the primary output has a greater probability of propagating to the primary output than a gate several levels of logic away from the primary outputs. This is because a failure that has to propagate through several levels of logic has a higher probability of being logically masked. This can be quantified by applying the concept of observability [19], which has historically found use in the testing and logic synthesis domains [20].

For reliability analysis, the observability of any wire in the circuit can be defined as the probability that a $0 \rightarrow 1$ or $1 \rightarrow 0$ error at that wire affects the output of the circuit. Note that the observabilities are the noiseless observabilities, i.e., all the gates are assumed noise-free when the observabilities are calculated. Observabilities can be calculated using Boolean differences, symbolic techniques based on binary decision diagrams (BDDs), or simulation. Using the observabilities, a closed-form expression for the reliability, $\delta_y(\vec{\epsilon})$, is derived.

Let us begin with the trivial case of a circuit with a single gate having a failure probability ϵ. Since there is only one gate, the output of the gate is the primary output of the circuit and has an observability o of 1. Hence, the probability of failure of the output is equal to $\epsilon \cdot o = \epsilon$. Now, consider the case of a circuit with two cascaded gates g_1 and g_2 having failure probabilities ϵ_1 and ϵ_2 and observabilities o_1 and o_2. When both gates are error free, the primary output of the circuit is always error free. When at least one gate is in error, the error at the primary output is computed by analyzing two cases when 1) only one gate is in error and 2) both g_1 and g_2 are in error. When exactly one gate has failed, the primary output is in error when the failed gate is observable. Hence, the first error component of the output is $\epsilon_1(1-\epsilon_2)o_1 + (1-\epsilon_1)\epsilon_2 o_2$.

When both g_1 and g_2 are in error, the primary output is in error when g_1 and g_2 are *jointly* observable. Joint observability of two gates g_1 and g_2 is the probability that the primary output toggles when the outputs of both g_1 and g_2 toggle. Note that the joint observability is different from the simultaneous observability of g_1 and g_2, which is the probability that a toggle at g_1 causes a toggle at the primary output and a toggle at g_2 also causes a toggle at the primary output. Computing joint observability for all combinations of multiple gate failures is expensive because joint observability computation is itself computationally demanding and the number of combinations of multiple gate failures is exponential in the number of gates in the circuit. Thus, observability-based reliability analysis makes two simplifying assumptions for estimating the effect of multiple gate failures.

1) The effect of gate failures at the primary output are decoupled from each other, i.e., a failure at each gate g_i is assumed to affect the output with a probability o_i regardless of other gate failures. This assumption allows the joint observability to be replaced by simultaneous observability, which is computationally less demanding, to compute the effect of multiple gate failures at the output.
2) The observability of the gates are assumed to be independent of each other. Using this assumption, the computation of simultaneous observability of two gates can be simplified to the product of the individual gate observabilities. For instance, the probability that g_1 is observable and g_2 is not observable is given by $o_1(1-o_2)$ and the probability that g_1 and g_2 are both not observable is given by $(1-o_1)(1-o_2)$.

Let Ω be the set of all the gates in the circuit and S be the set of all nonempty subsets of Ω. Consider a set $G \in S$ of gates that have failed. The exact effect of these failed gates is given by the joint observability of the gates in G. Using the first assumption, it can be argued that the output is in error whenever an odd number of gates in G are simultaneously observable. Using the same assumption, when an even number of gates in G are simultaneously observable, the effect of these gate failures cancel each other. Thus, the joint observability of gates in G is estimated as the sum of simultaneous observability of an odd number of gates in G. Using the second assumption, the simultaneous observability of the two gates is given by the product of their individual observabilities. Note that in practice, there are cases when an odd number of simultaneously observable gates in G do not cause an error at the primary output and also cases when an even number of simultaneously observable gates in G cause an error at the primary output. These are averaged and absorbed into the probability of an odd number of failed gates being simultaneously observable by the first assumption.

For the two gate example described above, the second error component of the output (both gates in error) is given by $\epsilon_1\epsilon_2(o_1(1-o_2)+o_2(1-o_1))$. The first term $o_1(1-o_2)$ is the probability that g_1 is observable and g_2 is not and vice versa for the second term. The joint observability of g_1 and g_2 is estimated by $o_1(1-o_2)+o_2(1-o_1)$ (odd number of erroneous gates being simultaneously observable). The inaccuracies introduced due to the two simplifying assumptions are illustrated with an example in Section III-A.

With this background, we shall derive the expression for the probability of error at the output for a general circuit with N gates. Without loss of generality, we assume that the circuit has a single output y. Denote the error probability (observability) of the ith gate by $\epsilon_i(o_i)$. Let 2^G denotes the set of all subsets of G. Let $F \in 2^G$ be the set of gates in G that are simultaneously observable. Using the first assumption, the output y will be in error when an *odd number* of gates in G are simultaneously observable, i.e., F contains an odd number of gates. Using the second assumption, the simultaneous observability of a set of gates can be computed by simply multiplying the individual observabilities of the gates. For instance, if G contains three gates (g_1, g_2, and g_3) and $F = \{g_1, g_2\}$, then the probability that g_1 and g_2 are simultaneously observable is given by $o_1o_2(1-o_3)$. In general, the probability that the gates in F are observable is given by $A = \prod_{i\notin F}(1-o_i)\prod_{j\in F} o_j$. The expression $B = \prod_{i\notin F}(1-o_i)\prod_{j\in F} -o_j$ has the same magnitude as A and same sign as A when F has an even number of gates, and opposite sign as A when F has an odd number of gates. Thus, when F has an odd number of gates, the expression $1/2(A-B)$ gives the probability that the gates in F are observable, and when F has an even number of gates $1/2(A-B)$ is equal to zero. Thus, the probability that an odd number of gates in G are observable is given by

$$\sum_{F\in 2^G} \frac{1}{2}\left(\prod_{i\notin F}(1-o_i)\prod_{j\in F} o_j - \prod_{i\notin F}(1-o_i)\prod_{j\in F} -o_j\right). \quad (1)$$

By the first simplifying assumption, the probability of error at the output $y(y_{\text{error}})$ given that the gates in G have failed is also given by (1). Thus

$$\begin{aligned}\Pr(y_{\text{error}}|G) &= \frac{1}{2}\left(\sum_{F\in 2^G}\prod_{i\notin F}(1-o_i)\prod_{j\in F} o_j \right.\\ &\qquad \left. - \sum_{F\in 2^G}\prod_{i\notin F}(1-o_i)\prod_{j\in F} -o_j\right)\\ &= 1/2\left(\prod_{j\in G}(o_j+(1-o_j)) - \prod_{j\in G}((1-o_j)-o_j)\right)\\ &= 1/2\left(1-\prod_{j\in G}(1-2o_j)\right).\end{aligned}$$

The probability that the gates in G are in error and the gates in $G^c(\Omega \setminus G)$ are error free is given by $\prod_{i\in G}\epsilon_i \prod_{j\in G^c}(1-\epsilon_j)$. Thus, the probability of error at the output y is given by

$$\begin{aligned}\Pr(y_{\text{error}}) &= \sum_{G\in S}\prod_{i\in G}\epsilon_i \prod_{j\in G^c}(1-\epsilon_j)\\ &\quad \times \left(\frac{1-\prod_{j\in G}(1-2o_j)}{2}\right)\\ \Rightarrow \Pr(y_{\text{error}}) &= 1/2\sum_{G\in S}\prod_{i\in G}\epsilon_i \prod_{j\in G^c}(1-\epsilon_j)\\ &\quad - 1/2\sum_{G\in S}\prod_{i\in G}\epsilon_i(1-2o_i)\prod_{j\in G^c}(1-\epsilon_j).\end{aligned}$$

Since S contains all nonempty subsets of Ω, the first term $\sum_{G\in S}\prod_{i\in G}\epsilon_i\prod_{j\in G^c}(1-\epsilon_j)$ contains all the terms in $\prod_{i\in\Omega}((1-\epsilon_i)+\epsilon_i)$ (when it is expanded) except $\prod_{i\in\Omega}(1-\epsilon_i)$. Hence, the first term can be replaced by

$$\prod_{i\in\Omega}(1-\epsilon_i+\epsilon_i) - \prod_{i\in\Omega}(1-\epsilon_i) = 1-\prod_{i\in\Omega}(1-\epsilon_i).$$

A similar transformation for the second term yields

$$\begin{aligned}\Pr(y_{\text{error}}) &= 1/2\left(1-\prod_{i\in\Omega}(1-\epsilon_i)\right)\\ &\quad -1/2\left(\prod_{i\in\Omega}(1-\epsilon_i+\epsilon_i(1-2o_i)) - \prod_{i\in\Omega}(1-\epsilon_i)\right)\\ \Rightarrow \Pr(y_{\text{error}}) &= 1/2\left(1-\prod_{i\in\Omega}(1-2\epsilon_i o_i)\right). \quad (2)\end{aligned}$$

Equation (2) is a closed-form expression for the reliability of the output of a circuit as a function of error probabilities at each gate. Since the product of $(1-2\epsilon_i o_i)$ is over all gates in the circuit, it can be computed very efficiently once the observability of each gate is known. It is interesting to note that the closed-form expression computes the exact expression for reliability of an XOR tree circuit. The observabilities of the XOR gates in the tree are independent and always equal to one irrespective of other XOR gate failures in the tree. Thus, the two simplifying assumptions of the observability-based reliability

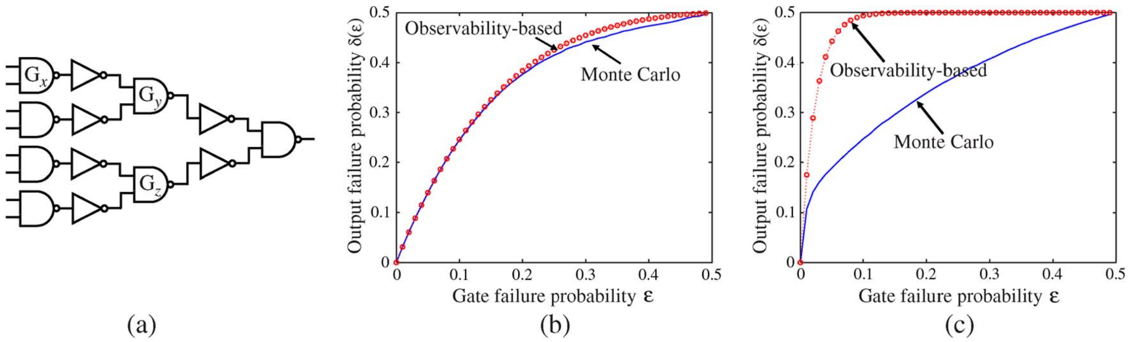


Fig. 1. Circuit for illustrating the effect of noise and correlation on observability.

analysis described previously in this section are valid for an XOR tree.

A. Noise and Correlation Distort Observability

Simulation results indicate that the closed-form expression for $\delta(\vec{\epsilon})$ is highly accurate for small circuits, and deviates by a small margin for ϵ close to 0.5. Note that the same value of gate failure probability has been used for each gate, and hence, $\vec{\epsilon}$ is replaced by ϵ. For example, the Monte Carlo and observability-based curves for $\delta(\epsilon)$ for the circuit in Fig. 1(a) are shown in Fig. 1(b). Simulation results also indicate that the closed-form expression performs well for small values of ϵ in large circuits, and that the accuracy depends on the number of gates in the circuit with $\epsilon > 0$. For example, Fig. 1(c) compares the $\delta(\epsilon)$ curves for a single output of the benchmark circuit b9, where a large error is observed as ϵ increases.

Observability-based reliability analysis is accurate for small ϵ because the probability of single gate failures is significantly higher than the probability of multiple gate failures. Since the effect of an error at a single gate is given by the gate failure probability scaled by its observability, it is exactly accounted for in the closed-form expression for reliability. As ϵ increases, the effect of multiple gate failures starts becoming significant and a deviation of the observability-based curve from the Monte Carlo curve is observed. There are two reasons for the inaccuracy of observability-based analysis in computing the effects of multiple gate failures. Both arise from the simplifying assumptions made in the derivation of the closed-form expression and are related to the fact that observability calculations are done statically.

1) In the Absence of Noise: When the observability calculations are performed in the absence of noise, it is assumed that a path remains sensitized irrespective of failures at gates that contribute to sensitizing that path. However, a failure at one or more of these gates may increase or decrease the observability of the original gate. This is exactly the reason for the inaccuracy arising due to the first simplifying assumption—joint observability replaced by simultaneous observability. For instance, consider gates G_x and G_z in the circuit of Fig. 1(a). Exhaustive analysis indicates that if both G_x and G_z fail, the probability of an output failure is 46/256, i.e., the joint observability of G_x and G_z is 46/256. However, the closed-form expression ignores the effects of how failures at G_z influence the propagation of failures from G_x and estimates this probability to be 19/256. This problem is further exacerbated by the effects of reconvergent fan-out that is common in logic circuits, since observability calculation at the source of reconvergent fan-outs becomes more complex and expensive.

2) On Individual Gates in the Circuit: When observability computation is performed on gates one at a time in the derivation of the closed-form expression for $\delta(\vec{\epsilon})$, the events of two or more gates being simultaneously observable is computed assuming that the events are independent. The second simplifying assumption suffers from this inaccuracy. For instance, consider gates G_x and G_y in the circuit of Fig. 1(a). Assuming independence suggests that G_x is observable even when G_y is not because $o_x(1 - o_y) > 0$. However, since G_x is in the transitive fan-in of G_y, it is clear that G_x is observable *only if* G_y is observable. Assuming independence thus introduces inaccuracies in the closed-form expression.

In conclusion, the observability-based closed-form expression is highly suitable for reliability analysis of small circuits and for small values of gate failure probabilities in large circuits. The algorithm is simple, yet efficient and flexible because a change in the value of noise at any gate(s) just requires recomputation of the closed-form expression (2). Since gate failure rates in current CMOS technologies are of the order of $10^{-8} - 10^{-4}$, it can easily be applied to reliability analysis and optimization.

IV. Single-Pass Reliability Analysis

The efficient single-pass reliability analysis technique described here addresses the accuracy drawbacks of the observability-based algorithm. At the core of this algorithm is the observation that an error at the output of any gate is the cumulative effect of a local error component attributed to the ϵ of the gate, and a propagated error component attributed to the failure of gates in its transitive fan-in cone. When the components are combined, the total error probability at gate g is given by: 1) a $0 \rightarrow 1$ error probability given that its error-free value is zero, $\Pr(g_{0\rightarrow 1})$ and 2) a $1 \rightarrow 0$ error probability given that its error-free value is one, $\Pr(g_{1\rightarrow 0})$.

In general, $\Pr(g_{0\rightarrow 1}) \neq \Pr(g_{1\rightarrow 0})$ for an internal gate in a circuit. Initially, $\Pr(x_{i,0\rightarrow 1})$ and $\Pr(x_{i,1\rightarrow 0})$ are known for the primary inputs x_i of the circuit. In the core computational step of the algorithm, the $0 \rightarrow 1$ and $1 \rightarrow 0$ error components at the inputs to a gate are combined using a weight vector $\mathcal{W}$ to obtain a weighted input error vector $\mathcal{PW}$. The $\mathcal{PW}$ vector is

TABLE I
EXPRESSIONS FOR WEIGHTED INPUT ERROR COMPONENTS

Input vector	Weight	Weighted $0 \rightarrow 1$ input error component
00	$\mathcal{W}_{00}$	$\Pr(i_{0\rightarrow 1})\Pr(j_{0\rightarrow 1})\mathcal{W}_{00}$
01	$\mathcal{W}_{01}$	$\Pr(i_{0\rightarrow 1})(1-\Pr(j_{1\rightarrow 0}))\mathcal{W}_{01}$
10	$\mathcal{W}_{10}$	$(1-\Pr(i_{1\rightarrow 0}))\Pr(j_{0\rightarrow 1})\mathcal{W}_{10}$
Total	$\mathcal{W}(0)$	$\mathcal{PW}(0)$
Input vector	Weight	Weighted $1 \rightarrow 0$ input error component
11	$\mathcal{W}_{11}$	$\big(\Pr(i_{1\rightarrow 0})+\Pr(j_{1\rightarrow 0})-\Pr(i_{1\rightarrow 0})\Pr(j_{1\rightarrow 0})\big)\mathcal{W}_{11}$
Total	$\mathcal{W}(1)$	$\mathcal{PW}(1)$

then combined with the local gate failure probability ϵ to obtain $\Pr(g_{0\rightarrow 1})$ and $\Pr(g_{1\rightarrow 0})$ at the output of the gate. Computation of the: 1) weight vector and 2) weighted input error vector is described below.

Single-pass reliability analysis is performed by applying the core computational step of the algorithm recursively to the gates in a topological order. At the end of the single pass, $\Pr(y_{0\rightarrow 1})$ and $\Pr(y_{1\rightarrow 0})$ is obtained for the output y of the circuit. The reliability δ_y of an output y is then given by the weighted sum of $\Pr(y_{0\rightarrow 1})$ and $\Pr(y_{1\rightarrow 0})$ as follows:

$$\delta_y(\epsilon) = \Pr(y=0)\Pr(y_{0\rightarrow 1}) + \Pr(y=1)\Pr(y_{1\rightarrow 0}).$$

Given the weight vectors at all gates, the time complexity of the algorithm is $O(N)$, where N is the number of gates in the circuit. Note that single-pass reliability analysis gives the exact values of probability of error at the output in the absence of reconvergent fan-out.

1) Weight Vector: The weight vector for a gate stores the probability of occurrence of every combination of inputs at that gate. For instance, the weight vector of a two-input (three-input) gate consists of four (eight) entries. Since the weight vector is just the joint signal probability distribution of the inputs of a gate, it can be computed by random pattern simulation or symbolic techniques based on BDDs. Weight vectors are independent of $\vec{\epsilon}$ and change only if the structure of the logic circuit changes. To improve the efficiency of the algorithm, weight vector computation may be performed once at the beginning and used over several runs of reliability analysis. The BDDs for the gates in the circuit are used to compute the components $\mathcal{W}_{00}, \mathcal{W}_{01}$, etc., of $\mathcal{W}$. For example, if b_1 and b_2 are the inputs to a gate, $\mathcal{W}_{00}$ is given by the number of minterms in $\overline{b}_1\overline{b}_2$ divided by the total number of input vectors to the circuit.

2) Expressions for Weighted Input Error Vector: Expressions for the components of $\mathcal{PW}$, for a two-input AND gate with inputs i and j, are given in Table I. The calculation of $\mathcal{PW}(0)$ to propagate the $0 \rightarrow 1$ error component using the entries in the upper part of Table I is described here. Propagation of the $1 \rightarrow 0$ input error component is similar, using the entries in the lower part of Table I.

Since the probability of a $0 \rightarrow 1$ error is actually the probability of a $0 \rightarrow 1$ error given that the error-free output of the gate is zero, there are only three rows in the upper table, one for each input vector for which the output of the AND gate is zero. The first column in the table is the input vector under consideration. The input vector has been ordered as ij. The second column is the probability of occurrence of the input vector, i.e., the weight vector. The third column is the probability of a $0 \rightarrow 1$ error at g, caused only due to errors at its inputs (when g itself does not fail). The entries in the third column are computed using $\Pr(i_{0\rightarrow 1})$, $\Pr(i_{1\rightarrow 0})$, $\Pr(j_{0\rightarrow 1})$, and $\Pr(j_{1\rightarrow 0})$ as illustrated below with an example.

Consider the input 10, whose error-free output is zero. For g to be in error only due to errors at the inputs, j has to fail, and i has to be error free so that the input to the gate is 11 instead of 10. Thus, the probability of a $0 \rightarrow 1$ error at g due to this input vector is $(1-\Pr(i_{1\rightarrow 0}))\Pr(j_{0\rightarrow 1})$. To compute the effect of the input vector 10, this probability of error is weighted by its probability of occurrence, i.e., by $\mathcal{W}_{10}$. Thus, the value in the third column for the vector 10 is $\mathcal{W}_{10}(1-\Pr(i_{0\rightarrow 1}))\Pr(j_{0\rightarrow 1})$. Similar entries for the inputs 00 and 01 are derived, and summed to obtain an expression for the weighted input error probability $\mathcal{PW}(0)$.

Since we are calculating the weighted $0 \rightarrow 1$ input error probability at g *given* that the error-free output is zero, $\mathcal{PW}(0)$ has to be divided by $\mathcal{W}(0)$ to restrict the inputs to a set for which the error-free output is zero. Thus, the weighted $0 \rightarrow 1$ and $1 \rightarrow 0$ input error probability at g are given by

$$\Pr(g_{0\rightarrow 1}|g \text{ does not fail}) = \mathcal{PW}(0)/\mathcal{W}(0)$$
$$\Pr(g_{1\rightarrow 0}|g \text{ does not fail}) = \mathcal{PW}(1)/\mathcal{W}(1).$$

3) Expressions for $\Pr(g_{0\rightarrow 1})$ *and* $\Pr(g_{1\rightarrow 0})$*:* If g fails with a probability of ϵ, $\Pr(g_{0\rightarrow 1})$ is given by

$$\Pr(g_{0\rightarrow 1}) = (1-\epsilon)\left(\frac{\mathcal{PW}(0)}{\mathcal{W}(0)}\right) + \epsilon\left(1-\frac{\mathcal{PW}(0)}{\mathcal{W}(0)}\right).$$

Similarly, $\Pr(g_{1\rightarrow 0})$ is given by

$$\Pr(g_{1\rightarrow 0}) = (1-\epsilon)\left(\frac{\mathcal{PW}(1)}{\mathcal{W}(1)}\right) + \epsilon\left(1-\frac{\mathcal{PW}(1)}{\mathcal{W}(1)}\right).$$

Note that the two terms $(1-\Pr(i_{1\rightarrow 0}))$ and $\Pr(j_{0\rightarrow 1}))$ are multiplied in the computation of the entries in the third column of Table I. This implies that the events of i being correct and j failing are assumed independent. This assumption is valid if the gate is not a site for reconvergence of fan-out. Since reconvergence causes the two events to be correlated, it is handled separately in Section IV-A.

Although the computation has been illustrated for an AND gate, the computation for an OR gate is symmetric, i.e., there are three rows for the probability of $1 \rightarrow 0$ error table and a single row for the probability of $0 \rightarrow 1$ error table. Inverters, NANDs, NORs, and XORs are all handled in a similar manner and the tables have been excluded for brevity.

Single-pass reliability analysis is illustrated for the circuit shown in Fig. 2. The weight vector, gate failure probability (ϵ), and probability of $0 \rightarrow 1$ and $1 \rightarrow 0$ error are indicated for each gate. The gates are numbered on the order in which they are processed. Since all the gates in the circuit have only two inputs, the weight vector for each gate consists of four entries. All entries of the weight vector for gate 1 are 0.25 because the primary input vectors are equally likely. The fan-out at gate 2 reconverges at gate 6 via gates 4 and 5. Thus, the event

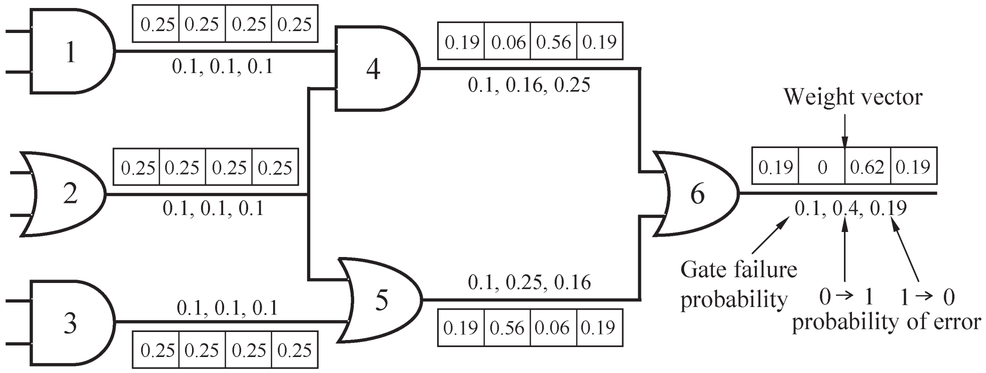


Fig. 2. Illustrative example for single-pass reliability analysis.

of 0 → 1 and 1 → 0 error at the outputs of gates 4 and 5 are correlated. However, independence is assumed and the probability of these events are used in the computation of 0 → 1 and 1 → 0 probability of error values for the output of gate 6.

A. Handling Reconvergent Fan-out

The presence of reconvergent fan-out renders the single-pass reliability analysis approximate because the events of 0 → 1 or 1 → 0 error for the inputs of a gate may not be independent at the point of reconvergence. Handling reconvergent fan-out has been the subject of extensive research in signal probability computation. In this section, the theory of correlation coefficients used in signal probability computation [21], is extended to make single-pass reliability analysis more accurate in the presence of reconvergent fan-out.

This approach relies on the propagation of the correlation coefficients for a pair of wires from the source of fan-out to the point of reconvergence. Note that the word "wire" has been used as opposed to "node" because for a gate with fan-out > 1, each fan-out is treated as a separate wire, but they constitute the same node. The correlation coefficient for events on a pair of wires is defined as the joint probability of the events divided by the product of their marginal probabilities. For signal probability computation, an event on a wire is defined as the value of the wire being one. Thus, for a pair of wires, a single correlation coefficient is sufficient to compute the joint probability of a one on both the wires.

For reliability analysis, however, an event is defined as a 0 → 1 or 1 → 0 error on a wire. Hence, instead of a single correlation coefficient, four correlation coefficients for a pair of wires, one for every combination of events on the pair of wires are used. If v and w are two wires, the four correlation coefficients for this pair are denoted by $\mathcal{C}_{vw}, \mathcal{C}_{v\tilde{w}}, \mathcal{C}_{\tilde{v}w}$, and $\mathcal{C}_{\tilde{v}\tilde{w}}$, where v, w, $\tilde{v}$, and $\tilde{w}$ refers to the event of a 0 → 1, 0 → 1, 1 → 0, and 1 → 0 error at v and w, respectively.

The correlation coefficients come into play at the gates whose inputs are the site of reconvergence of fan-out. At such gates, the events of 0 → 1 or 1 → 0 error at the inputs are not independent. Thus, the entries in the third column of Table I are weighted by the appropriate correlation coefficient, e.g., $\Pr(i_{0\to1})(1-\Pr(j_{1\to0}))$ becomes $\Pr(i_{0\to1})(1-\Pr(j_{1\to0})\mathcal{C}_{i\tilde{j}})$.

1) Correlation Coefficient Computation: The correlation coefficient for a pair of wires can be calculated by first computing the correlation coefficients for the wires in the fan-out source that cause the correlation, and then propagating these correlation coefficients along the appropriate paths leading to the pair of wires. Note that all four correlation coefficients for two independent wires are one. The computation of correlation coefficients for the fan-out source and the propagation of correlation coefficients at a two-input AND gate are described below.

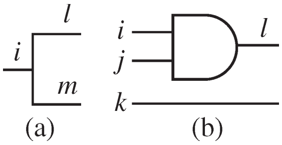


Fig. 3. Computation and propagation of correlation coefficients.

2) Computation at Fan-out Source Node: The fan-out source node i is shown in Fig. 3(a). The correlation coefficient for the pair of wires $\{l, m\}$ is computed as follows:

$$\Pr(l_{0\to1}) = \Pr(l_{0\to1}, m_{0\to1}) = \Pr(l_{0\to1})\Pr(m_{0\to1})\mathcal{C}_{lm}$$

i.e.,

$$\mathcal{C}_{lm} = \frac{1}{\Pr(m_{0\to1})}.$$

$\mathcal{C}_{\tilde{l}\tilde{m}}$ can be computed in a similar manner. $\mathcal{C}_{\tilde{l}m}$ and $\mathcal{C}_{l\tilde{m}}$ are zero because it is not possible to have a 0 → 1 error on m and a 1 → 0 error on l, or vice versa.

3) Propagation at an AND *Gate:* Propagation of correlation coefficients is shown for the AND gate in Fig. 3(b). Let i, j, k be three wires whose pairwise correlation coefficients are known. Computation of the correlation coefficients for the pair $\{l, k\}$ involves propagation of the correlation coefficients through the AND gate, using the correlation coefficients of i, j with k

$$\mathcal{C}_{lk} = \frac{\Pr(l_{0\to1}|k_{0\to1})}{\Pr(l_{0\to1})}.$$

The expression for $\Pr(l_{0\to1}|k_{0\to1})$ in terms of the correlation coefficients of the inputs i, j with k is shown in Fig. 4. The terms in the expression for $\Pr(l_{0\to1}|k_{0\to1})$ are similar to the terms in the third column of the upper part of Table I. The only difference is that the probability of 0 → 1 and 1 → 0 errors have been multiplied by appropriate correlation coefficients. Note that the terms of the weight vector $\mathcal{W}$ include the signal probability of k. The expression for $\mathcal{C}_{\tilde{l}k}$ is derived in a similar manner using the lower part of Table I, and is left out for

$$
\begin{aligned}
\Pr(l_{0\to1}|k_{0\to1}) = \epsilon + \frac{(1-2\epsilon)}{\mathcal{W}(0)} \Big(& \mathcal{W}_{00}\Pr(i_{0\to1}|k_{0\to1})\Pr(j_{0\to1}|k_{0\to1})\mathcal{C}_{ij} + \mathcal{W}_{01}\Pr(i_{0\to1}|k_{0\to1})(1-\Pr(j_{1\to0}|k_{0\to1})\mathcal{C}_{i\tilde{j}}) \\
& +\mathcal{W}_{10}(1-\Pr(i_{1\to0}|k_{0\to1})\mathcal{C}_{\tilde{i}j})\Pr(j_{0\to1}|k_{0\to1})\Big) \\
= \epsilon + \frac{(1-2\epsilon)}{\mathcal{W}(0)} \Big(& \mathcal{W}_{00}\Pr(i_{0\to1})\mathcal{C}_{ik}\Pr(j_{0\to1})\mathcal{C}_{jk}\mathcal{C}_{ij} + \mathcal{W}_{01}\Pr(i_{0\to1})\mathcal{C}_{ik}(1-\Pr(j_{1\to0})\mathcal{C}_{\tilde{j}k}\mathcal{C}_{i\tilde{j}}) \\
& +\mathcal{W}_{10}(1-\Pr(i_{1\to0})\mathcal{C}_{\tilde{i}k}\mathcal{C}_{\tilde{i}j})\Pr(j_{1\to0})\mathcal{C}_{jk}\Big)
\end{aligned}
$$

Fig. 4. Derivation of $\Pr(l_{0\to1}|k_{0\to1})$ in terms of correlation coefficients of its inputs.

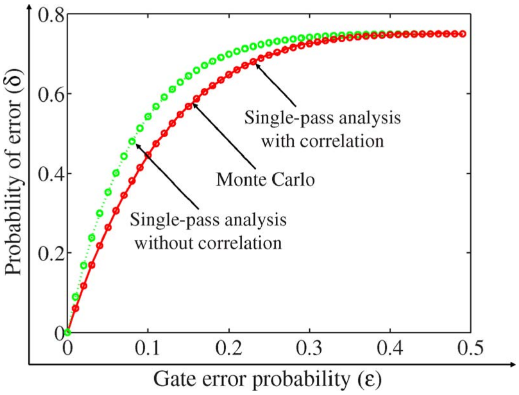


Fig. 5. Handling reconvergent fan-out in single-pass reliability analysis with four correlation coefficients.

brevity. Expressions for $\mathcal{C}_{l\tilde{k}}$ and $\mathcal{C}_{\tilde{l}\tilde{k}}$ are derived by replacing k by $\tilde{k}$ in the expressions for $\mathcal{C}_{lk}$ and $\mathcal{C}_{\tilde{l}k}$, respectively. In Fig. 5, the consolidated probability of error at two correlated primary outputs of benchmark circuit b9 is used to illustrate the accuracy achieved with correlation coefficients.

B. Accuracy Versus Computational Complexity

As shown in Fig. 5, the degree of accuracy obtained using four correlation coefficients for a pair of wires is high and should suffice for most applications. In the example shown in Fig. 3, the correlation coefficients used in the simplest conditional probability terms like $\Pr(l_{0\to1}|k_{0\to1})\mathcal{C}_{lk}$ are termed as first-order correlation coefficients. Second-order correlation coefficients are of the form $\Pr(i_{0\to1}|j_{0\to1}, k_{0\to1})\mathcal{C}_{ijk}$ and so on for higher orders. Correlation coefficients of order greater than one are approximated using the product of first-order correlation coefficients, i.e., $\mathcal{C}_{ijk} = \mathcal{C}_{ij}\mathcal{C}_{ik}$. When only four correlation coefficients are used, there are some first-order correlation coefficients that are not evaluated exactly. For instance, consider the correlation coefficient $(\mathcal{C})$ of a $0 \to 0$ transition on l and a $0 \to 1$ transition on k. Note that $\mathcal{C}$ is different from the correlation coefficients $\mathcal{C}_{lk}$ and $\mathcal{C}_{\tilde{l}k}$. The exact value for $\mathcal{C}$ can be derived as follows:

$$\Pr(l_{0\to0}, k_{0\to1}) + \Pr(l_{0\to1}, k_{0\to1}) = \Pr(k_{0\to1}|l=0).$$

Since $\Pr(l_{0\to1})$ is actually $\Pr(l_{0\to1}|l=0)$, using correlation coefficients the above expression can be rewritten as

$$\Pr(l_{0\to0})\mathcal{C} + \Pr(l_{0\to1})\mathcal{C}_{lk} = \frac{\Pr(k_{0\to1}|l=0)}{\Pr(k_{0\to1})}$$

$$\Pr(l_{0\to0})\mathcal{C} = \frac{\Pr(k_{0\to1}|l=0)}{\Pr(k_{0\to1})} - \Pr(l_{0\to1})\mathcal{C}_{lk}.$$

When only four correlation coefficients are used, $\mathcal{C}$ is approximated to $1 - \Pr(l_{0\to1})\mathcal{C}_{lk}$ because the value of $\Pr(k_{0\to1}|l=0)$ is not known, and cannot be computed easily. This problem can be solved by using 16 correlation coefficients for a pair of wires. These 16 coefficients arise from the correlation among all combinations of $\{0 \to 0, 0 \to 1, 1 \to 0, 1 \to 1\}$ failures for the two wires. The computation of these 16 correlation coefficients is done in the exact same manner as described for four correlation coefficients. Thus, using 16 correlation coefficients, higher accuracy can be obtained at the cost of higher computational complexity in computing and propagating the 16 correlation coefficients.

V. Maximum-k Gate Failure Reliability Analysis

Recall from Section III that $\Omega = \{0, 1, \ldots, N-1\}$ is the set of all gates in a circuit with N gates. The failure probability of the ith gate is denoted as ϵ_i. Both the observability-based and single-pass reliability analysis algorithms assume an independent gate failure model. Under the independent gate failure model, the probability that a subset $(G, G \subset \Omega)$ of gates fail simultaneously is given by the expression

$$\Pr(G) = \prod_{i \in G} \epsilon_i \prod_{i \notin G} (1 - \epsilon_i).$$

Thus, there is a nonzero probability that a large number of gates in the circuit fail simultaneously. For example, in a circuit with 10 000 gates, if each gate has a failure probability of 10^{-3}, then the probability of ten gates failing simultaneously is 0.125. It is not unreasonable to expect such high failure rates for emerging technologies such as carbon nanotube transistors, single electron transistors, or graphene at least with the current fabrication technology for these devices. However, these failure rates are much beyond what has been predicted for future semiconductor technologies. It is possible to adjust the average number of gate failures by varying the gate failure probability. For example, consider a circuit with ten gates and an independent gate failure model. Suppose we are interested in only a single gate failure on average, the gate failure probability must be set to 0.1. By fixing the gate failure probability, we are also fixing the rate at which failures occur in the entire circuit, i.e., the fraction of clock cycles in which at least one gate failure occurs. In this example, the probability of at least one gate failure is $1 - (1 - 0.1)^{10} = 0.65$, i.e., 65 out of 100 clock cycles will result in at least one gate failure. Thus, the independent gate failure model has an inherent disadvantage that the average number of gate failures and the fraction of clock cycles in which gate failures occur are dependent, and cannot be varied independently. This problem is resolved by limiting the maximum number of gate failures.

If we are concerned with only single gate failures, we can set the limit to one. Now, the failure probability of the gates will determine the fraction of clock cycles in which at least one gate failure occurs in the circuit. For instance, if the failure probability is 0.1, the probability that no gate failures occur is equal to $(1-0.1)^{10} = 0.9^{10}$. Probability that exactly one gate fails is $10 \times 0.1 \times (1-0.1)^9 = 0.9^9$. Hence, the fraction of clock cycles in which gate failures occur is $0.9^9/(0.9^{10} + 0.9^9) = 0.53$. Single-event upsets are an excellent example of maximum-k gate failure model. For instance, let us assume that the flux of energetic particle strikes is such that it can cause one upset every N cycles. In the maximum-k gate failure model, this is done by setting the limit on the number of gate failures to one, and adjusting the failure probabilities of the gates such that the error rate of the circuit is $1/N$. To summarize, a more realistic gate failure model is to have an upper bound on the number of simultaneous gate failures.

Under the maximum-k gate failure model, gate failures are still independent of each other. However, an additional constraint, that only a maximum of k gates can fail simultaneously, is introduced. This global constraint causes the gate failures to be correlated with each other, which can be explained mathematically as follows. Note that the maximum number of gate failures that can occur in parallel is an empirical quantity that depends on the source of the noise. For instance, in the case of memories, single/double bit errors are pervasive. A similar characterization of noise sources in logic circuits can be used to determine the maximum simultaneous number of failures in a logic circuit.

Let X_i be a random variable that takes a value one when the ith gate fails and a value zero otherwise. In the maximum-k gate failure model, the X_is are independent of each other, but the additional constraint is that $\sum X_i \le k$. This constraint means that $\Pr(X_{k+1} = 1 | X_1 = 1, X_2 = 1, \ldots, X_k = 1) = 0$. However, $\Pr(X_{k+1} = 1) = \epsilon_{k+1}$. Since the conditional probability is not equal to the marginal probability, the gate failures are correlated with each other.

In the rest of this section, a maximum-k gate failure reliability analysis algorithm that combines a sampling algorithm with single-pass reliability analysis is described. Several techniques to sample k (with given weights) out of a sample space of N have been proposed in literature, e.g., [22]. However, for the maximum-k gate failure model, the size of the sample space is $\binom{N}{0} + \binom{N}{1} + \cdots + \binom{N}{k}$. For N of the order of 10^4, the sample space becomes intractably large even for $k = 3$. However, a special property of the sample space—the independence of gate failures—is exploited in the proposed sampling algorithm to reduce the computational complexity to $O(Nk^2)$.

A. Why Fault Injection and Simulation Does Not Scale

The most straightforward algorithm for reliability analysis with the maximum-k gate failure model is based on fault injection and random pattern simulation in a Monte Carlo framework. By generating a random number uniformly distributed over the interval [0,1] for each gate and comparing it to the failure probability of the gate, it can be determined whether the gate has failed or not. Thus, the set of all failed gates constitutes a sample of failed gates. Since the random numbers for each gate are generated independently, there is no control over the size of the sample. A sample of failed gates has to be discarded if it violates the constraint on the maximum number of gate failures. If the constraint is satisfied, random pattern simulation is performed to evaluate the effect of the failed gates on the outputs. This procedure is repeated for a large number of samples, and their effect on the outputs is averaged.

A major drawback of this algorithm is that for large values of gate failure probabilities and low values of k, a large number of generated samples may not obey the constraint on maximum number of gate failures and have to be discarded. For instance, suppose that the total number of gates in the circuit is 10 000, each gate fails independently with a probability 0.05, and $k = 3$. Since fault injection is performed independently at each gate, the average number of erroneous gates is $10\,000 \times 0.05 = 500 \gg 3$. Hence, a large number of samples will be discarded because they do not satisfy the constraint on the maximum number of gate failures. Thus, a very large number of runs are required to achieve convergence of the solution, and this makes the algorithm computationally intensive.

In contrast, the maximum-k gate failure algorithm, described below, is based on generating only correct samples of gate failures (which satisfy the constraint on the maximum number of gate failures). After generating a sample of failed gates, the single-pass reliability analysis algorithm is used to evaluate the probability of error at the output of the circuit. The probability of failure at each primary output is obtained by averaging over a large number of samples. Since only correct samples are generated and since single-pass reliability analysis is used to compute the effect of each sample of gate failures, the algorithm is orders of magnitude faster the Monte Carlo approach.

B. Sampling Algorithm

When the maximum number of simultaneous gate failures is bounded by k, the sample space of the possible combinations of simultaneous gate failures is reduced to sets G with $|G| \le k$. Let S_ϵ be the sum of the probability of gate failure combinations in this sample space. Thus, S_ϵ is given by the following expression:

$$S_\epsilon = \sum_{\{G \subset \Omega, |G| \le k\}} \prod_{i \in G} \epsilon_i \prod_{i \notin G} (1 - \epsilon_i).$$

Note that $S_\epsilon \ne 1$ because the sample space is reduced to sets G such that $|G| \le k$. Thus, for the maximum-k gate failure model the probability that a subset of gates, G, fail simultaneously is given by the following expression:

$$\Pr(G) = \begin{cases} \prod_{i \in G} \epsilon_i \prod_{i \notin G} (1 - \epsilon_i)/S_\epsilon, & |G| \le k \\ 0, & |G| > k. \end{cases} \tag{3}$$

Let $\alpha_i = \epsilon_i/(1 - \epsilon_i)$. If S_α is defined as

$$S_\alpha = \sum_{\{G \subset \Omega, |G| \le k\}} \prod_{i \in G} \alpha_i$$

TABLE II
SAMPLE SPACE OF POSSIBLE GATE FAILURES

Sample	Probability	In terms of α_i
{}	$(1-\epsilon_1)(1-\epsilon_2)(1-\epsilon_3)(1-\epsilon_4)/S_\epsilon$	$1/S_\alpha$
{1}	$\epsilon_1(1-\epsilon_2)(1-\epsilon_3)(1-\epsilon_4)/S_\epsilon$	α_1/S_α
{2}	$\epsilon_2(1-\epsilon_1)(1-\epsilon_3)(1-\epsilon_4)/S_\epsilon$	α_2/S_α
{3}	$\epsilon_3(1-\epsilon_1)(1-\epsilon_2)(1-\epsilon_4)/S_\epsilon$	α_3/S_α
{4}	$\epsilon_4(1-\epsilon_1)(1-\epsilon_2)(1-\epsilon_3)/S_\epsilon$	α_4/S_α
{1,2}	$\epsilon_1\epsilon_2(1-\epsilon_3)(1-\epsilon_4)/S_\epsilon$	$\alpha_1\alpha_2/S_\alpha$
{1,3}	$\epsilon_1\epsilon_3(1-\epsilon_2)(1-\epsilon_4)/S_\epsilon$	$\alpha_1\alpha_3/S_\alpha$
{1,4}	$\epsilon_1\epsilon_4(1-\epsilon_2)(1-\epsilon_3)/S_\epsilon$	$\alpha_1\alpha_4/S_\alpha$
{2,3}	$\epsilon_2\epsilon_3(1-\epsilon_1)(1-\epsilon_4)/S_\epsilon$	$\alpha_2\alpha_3/S_\alpha$
{2,4}	$\epsilon_1\epsilon_3(1-\epsilon_2)(1-\epsilon_4)/S_\epsilon$	$\alpha_2\alpha_4/S_\alpha$
{3,4}	$\epsilon_3\epsilon_4(1-\epsilon_1)(1-\epsilon_2)/S_\epsilon$	$\alpha_3\alpha_4/S_\alpha$

(3) can be rewritten more succinctly in terms of α_i and S_α as

$$\Pr(G) = \begin{cases} \prod_{i\in G} \alpha_i/S_\alpha, & |G| \le k \\ 0, & |G| > k. \end{cases} \tag{4}$$

As an illustration for the probability distribution expressed by (4), the sample space and associated probability of failure for $N = 4$ and $k = 2$ is shown in Table II.

We start by describing two straightforward techniques for generating samples of erroneous gates according to the probability distribution given in (3). It turns out that the first solution is incorrect because the generated samples have a slightly different probability distribution, and that the second solution is computationally intractable. This motivates the third solution, which is efficient and generates the samples from the correct probability distribution.

The first intuitive approach to generate a sample of $\le k$ gates is to select the gates one at a time over k steps. In each step, the interval [0,1] is divided into subintervals proportional to $\{1, \alpha_1, \alpha_2, \ldots\}$. The subinterval α_i corresponds to failure of gate i. The subinterval proportional to one corresponds to "no gate failure." A uniformly distributed random number is generated in the interval [0,1] and the subinterval in which the generated random number lies indicates the erroneous gate. Note that if the random number lies in the subinterval proportional to one, then no gate is erroneous in that step. If the random number lies in the interval α_i, then the ith gate is designated as failed, and it is removed from the Ω. The reduced Ω is then used for the next step. This procedure is repeated for k steps. For instance, suppose $k = 3$. If the three steps generate $\{1, 1, 1\}$, then no gate has failed (denoted as $\{\}$ in Table II). If $\{1, \alpha_2, \alpha_4\}$ is generated, then gate 2 and gate 4 have failed (denoted as $\{2, 4\}$ in Table II). Although this approach seems correct at first, a closer look reveals that the distribution from which the sample of failed gates is generated by this approach is incorrect. The reason is that the samples of gate failures shown in Table II are unordered sets, i.e., the sample $\{1, \alpha_2, \alpha_4\}$ is the same as $\{\alpha_2, 1, \alpha_4\}$. However, the sample $\{\alpha_1, \alpha_2, \alpha_4\}$ can be generated in six ways in this algorithm (differing only in the order in which the gates are generated in different steps, $\{\alpha_1, \alpha_2, \alpha_4\}$, $\{\alpha_1, \alpha_4, \alpha_2\}, \ldots$). However, the sample $\{1, 1, 1\}$ can be generated only once, and the sample $\{1, 1, \alpha_2\}$ can be generated in only three ways.

A second straightforward but inefficient approach is to divide the interval [0,1] into subintervals of widths equal to the probabilities given in (4). Thus, every subset of possible gate failures has a subinterval in [0,1] with width equal to its probability of occurrence [given by (4)]. Then, a uniform random number in [0,1] is generated, and the subinterval in which it lies indicates the generated sample of erroneous gates. Since, the number of subsets of erroneous gates is $\binom{N}{0} + \binom{N}{1} + \cdots + \binom{N}{k}$, the runtime complexity of this algorithm is $O(N^k)$. For large N, this algorithm is unusable even for small values of k.

```
$G$ – Set of failed gates
$\Omega$ – Set of all gates
$k$ – Maximum number of simultaneous gate failures

$G$ = {}
$r$ = Uniform random number in [0,1]

BuildTable($\Omega$, $k$, $r$) {
   Construct Table III
   NextGateFailure($\Omega$, $k$, $r$)
}

NextGateFailure($\Omega$, $k$, $r$) {
   if ($k = 0$) return
   else
      $i$ = Sampled column in Table III based on $r$
      if ($i \notin$ "Erroneous gate") /* $0 \le i \le k - 1$ */
         $k = i$
         Recompute $r$
         NextGateFailure($\Omega$, $k$, $r$)
      else /* $1 \le i \le N$ */
         $G = G \cup i$
         $\Omega = \Omega \setminus i$
         Recompute $r$
         BuildTable($\Omega$, $k - 1$, $r$)
      end
   end
}
```

Fig. 6. Proposed sampling algorithm.

The proposed algorithm for generating a sample of erroneous gates has a runtime complexity $O(Nk^2)$. The pseudocode for the algorithm is shown in Fig. 6. Table III is used for generating the sample of erroneous gates in the proposed algorithm. The columns of Table III are divided into two parts: 1) erroneous gate, which has N columns, $\{1, 2, \ldots, N\}$ and 2) new sample size, which has k columns, $\{0, 1, \ldots, k-1\}$. The probability distribution of the samples shown in Table II can be rewritten by grouping together samples that share a gate. This grouping can be done systematically as shown in the first part of Table III under the column "Erroneous gate." The jth column in the table is the sum of the probabilities of all samples that contain gate j. The ith row is the probability distribution when the maximum number of gate failures is i. Thus, in the ith row, the probability of every sample of erroneous gates occurs i times, once each under the column corresponding to every gate in the erroneous gate sample.

The `BuildTable` routine in the pseudocode (in Fig. 6) constructs Table III for a given Ω and k. The core routine that generates the sample of erroneous gates is the `NextGateFailure` routine. The possible outcomes of an execution of the `NextGateFailure` routine are the following: 1) if $k = 0$, the algorithm ends; 2) no new erroneous gate is generated and the sample size is reduced by at least one; and 3) a new erroneous gate is generated and the sample size is reduced by one. The random number r is used to decide between

TABLE III
PROPOSED SAMPLING ALGORITHM TABLE

Z	Erroneous gate				New sample size					
	1	2	$\cdots$	N	$k-1$	$k-2$	$\cdots$	1	0	Row total
1	α_1	α_2	$\cdots$	α_N	0	0	$\cdots$	0	1	T_1
2	$\alpha_1(T_1 - Z_{1,1})$	$\alpha_2(T_1 - Z_{1,2})$	$\cdots$	$\alpha_N(T_1 - Z_{1,N})$	0	0	$\cdots$	T_1	1	T_2
3	$\alpha_1(\frac{T_2}{2} - Z_{2,1})$	$\alpha_2(\frac{T_2}{2} - Z_{2,2})$	$\cdots$	$\alpha_N(\frac{T_2}{2} - Z_{2,N})$	0	$\cdots$	$\frac{T_2}{2}$	T_1	1	T_3
$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
k	$\alpha_1(\frac{T_{k-1}}{k-1} - Z_{k-1,1})$	$\alpha_2(\frac{T_{k-1}}{k-1} - Z_{k-1,2})$	$\cdots$	$\alpha_N(\frac{T_{k-1}}{k-1} - Z_{k-1,N})$	$\frac{T_{k-1}}{k-1}$	$\frac{T_{k-2}}{k-2}$	$\cdots$	T_1	1	T_k

outcome 2) and 3) as follows. The entries in the kth row of Table III are used to divide the interval [0, T_k] (T_k is the row total of the kth row) into subintervals. Let $\mathcal{I}$ be the subinterval that contains the random number rT_k.

If $\mathcal{I}$ corresponds to an entry in the "Erroneous gate" part of Table III, then the outcome is 3). In this case, a new erroneous gate g has been generated. The column number corresponding to subinterval $\mathcal{I}$ is used to determine the erroneous gate g. The gate g is added to the sample set G and removed from Ω. The value of k is reduced by one since one erroneous gate has been found. The fraction of overlap that rT_k had with the subinterval I is used as the new value of r. The function BuildTable is recursively called so that a new table (Table III) can be generated for the reduced Ω.

If $\mathcal{I}$ corresponds to an entry in the "New sample size" part of Table III, then the outcome is 2). In this case, no new erroneous gate has been generated, but the sample size has been reduced by at least one. This means that the size of the final sample will be less than k. The column number corresponding to the subinterval $\mathcal{I}$ is used to determine the new value of k, k_{new}. The random number r is also recomputed as described earlier. Since, Ω has not changed, Table III can be reused for the row k_{new}. Hence, in this case, NextGateFailure is directly called instead of BuildTable.

C. Effect of Multiple Gate Failures

After the sample of gate failures has been generated, the effect of these gate failures on the output of the circuit is computed. This problem is solved efficiently using the single-pass reliability analysis algorithm described in Section IV by setting the failure probability of the gates in the sample equal to one and the failure probability of rest of the gates equal to zero. Note that the flexibility of the single-pass reliability analysis is useful here, because for different samples of gate failures, it is only the failure probability of the gates in the circuit that is changing. Since there is no change in the circuit structure, the weight vector $\mathcal{W}$ has to be computed only once. This makes the algorithm for reliability analysis for the maximum-k gate failure model very efficient, and thus scalable to large circuits. Since single-pass reliability analysis is reused as the core reliability analysis engine, the results of this algorithm are also very accurate.

VI. RESULTS

The simulations were run on a 2.4-GHz Opteron-based system with 6 GB of memory. A Monte Carlo framework for reliability analysis based upon fault injection was used for comparison with single-pass reliability analysis. Table IV shows the accuracy of Monte Carlo simulator as the sample size for the simulation is increased from 64 to 6.4 million random patterns. We have chosen sample sizes to be multiples of 64 because a 64-bit parallel pattern simulator was used to implement the Monte Carlo simulator. The percentage error for each sample size is reported with respect to the result for 64 million random patterns. Since the maximum error for the largest benchmark circuit, i10, with 6.4 million random patterns is only 3.6%, we have used this sample size for all Monte Carlo simulations in this paper.

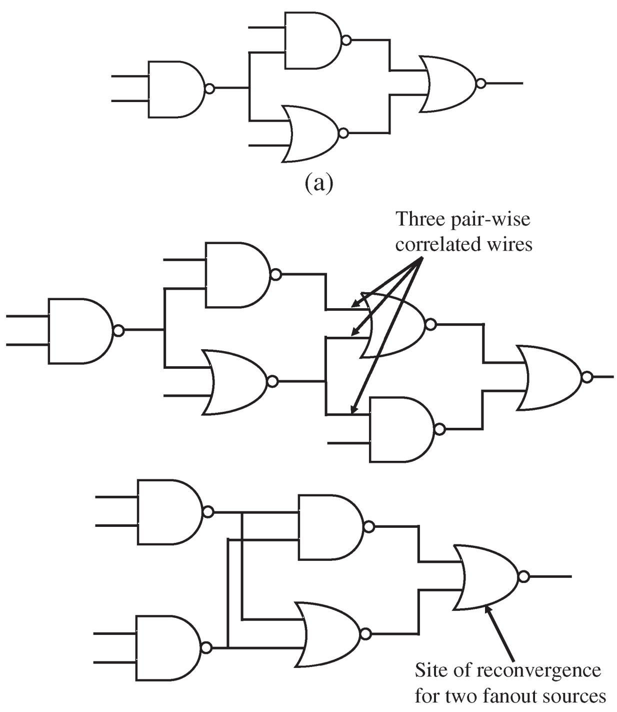
TABLE IV
ACCURACY OF MONTE CARLO-BASED RELIABILITY ANALYSIS FOR DIFFERENT NUMBER OF MONTE CARLO RUNS

Number of patterns	Max. error over all outputs (%)		
	x2 (56 gates)	c1355 (653 gates)	i10 (2643 gates)
64	943	435	3020
640	108	100	320
6400	71	43.4	150
64000	11.6	11.2	21.8
640000	3.7	4.3	7.6
6400000	1.7	1.5	3.6

The benchmark circuits used in [17] were synthesized using gates with a maximum of three inputs. However, the benchmark circuits used in this paper are synthesized using only two input gates. This was done to reduce second-order correlations occurring due to multiple reconvergent paths at a three-input gate. Note that using two-input gates does not completely eliminate second-order correlation coefficients (refer Fig. 7). Moreover, reordering of BDDs using CUDD_REORDER_SIFT was used to lower the runtimes from those reported in [17].

A. Observability-Based Reliability Analysis

A comparison of Monte Carlo and observability-based reliability analysis is presented in Table V. Both Monte Carlo and observability-based reliability analysis were used to compute the reliability of each benchmark circuit for 50 values of $\epsilon \in [0, 0.5]$. Of these, eight values are reported in Table V. Note that the observability-based reliability analysis is accurate for small values of ϵ ($N\epsilon \approx 1$, N is the total number of gates in the circuit). For larger values of ϵ (as shown in Table V), the accuracy of the observability-based reliability analysis is quite low for most of the benchmark circuits. The reasons for this inaccuracy have been discussed in Section III-A. It is interesting to note that the accuracy of the observability-based reliability analysis increases for very large values of $\epsilon (\approx 0.2-0.3)$. This



(a)

(b)

Fig. 7. Figure shows the circuit structures that introduce (a) first-order and (b) higher-order correlation coefficients.

is because for such large values of ϵ, $\delta(\epsilon)$ for the outputs begin to saturate at 0.5 (pure noise). The value of the observability-based closed-form expression also begins to saturate to 0.5 for those values of ϵ. Hence, the percentage error decreases. When ϵ is 0.5, both evaluate to 0.5 and the percentage error is zero. The runtimes for small- and medium-sized circuits are encouraging. However, for the large benchmark circuits (e.g., c3540) the runtime is high. This is because exact observability computation is computationally intensive as reconvergent fan-out increases, and this occurs with the c3540 benchmark.

B. Single-Pass Reliability Analysis

Simulation results comparing single-pass reliability analysis with Monte Carlo simulations are reported in Table VI. In Table VI, columns 1 and 2 give the name and number of gates in the benchmark circuit. Both the Monte Carlo and single-pass reliability frameworks were used to compute $\delta(\vec{\epsilon})$ for ten different values of ϵ over the range 0 to 0.5. Note that the same value of ϵ has been used for all the gates in the circuit, and hence, $\vec{\epsilon}$ is replaced by ϵ. The third column reports the percentage error in single-pass reliability analysis with 0, 4, and 16 correlation coefficients, averaged over all the outputs and over ten values of $\epsilon \in [0, 0.5]$. The cumulative runtime for ten runs is reported in the fourth column. The runtime required for the weight vector computation used in the single-pass reliability analysis is reported under the setup time column in the table.

The maximum percentage error in $\delta(\vec{\epsilon})$ is less than 2% for the largest benchmark circuit, i10. For circuits with significant reconvergent fan-out, e.g., c499, c1355, and c1908, the maximum percentage error in $\delta(\vec{\epsilon})$ is 13.1%, 13.5%, and 6.5%, respectively. The largest percentage error is observed when zero correlation coefficients are used, i.e., when the correlations in failures introduced due to reconvergent fan-out is ignored. The percentage error progressively improves as 4 and 16 correlation coefficients are used. Note that the error correcting benchmark circuits like c499, c1355, and c1908 have large reconvergent fan-out that requires the use of higher-order correlation coefficients. The circuit structure(s) that introduce first-order and higher-order correlation coefficients are shown in Fig. 7(a) and (b) respectively. Since the higher-order correlation coefficients are approximated in all three schemes (0, 4, and 16), only a slight improvement in accuracy is observed when 4 and 16 correlation coefficients are used.

Fig. 8 shows $\delta(\epsilon)$ for two outputs of benchmark i10. The cone sizes of the two outputs are 662 and 1034 gates, respectively. Each graph has two curves: one from Monte Carlo reliability analysis and one from single-pass reliability analysis using zero correlation coefficients. The two curves are indistinguishable, as shown in the figure. The diverse shapes of the curves illustrates not only the complexity of the relation between δ and ϵ, but also the accuracy of single-pass reliability analysis.

Fig. 9 shows the percentage error in $\delta(\vec{\epsilon})$ for each of the 32 outputs of benchmark circuit c499. On each run, the ϵ for each gate was derived from a uniform random distribution over the interval [0,0.5]. Single-pass reliability analysis with zero correlation coefficients was compared to Monte Carlo reliability analysis. The percentage error in $\delta(\vec{\epsilon})$ for each output, averaged over 1000 runs, is 1.5%–3.5%. This illustrates that single-pass reliability analysis is highly accurate even when the ϵ values are allowed to vary independently at every gate.

It is clear from the results that the proposed single-pass reliability analysis technique is highly accurate. Although a head-to-head performance comparison with approaches based on PTMs and Bayesian networks was not possible, it is our belief based on the results reported in [9] that the proposed technique affords at least a 500 X speedup over Bayesian networks on the largest circuit b9: 2.5 seconds for Bayesian networks versus 0.005 seconds per run with single-pass reliability analysis. Note also that results reported in [9] show that Bayesian networks afford a 1000 X speedup over PTMs. In summary, it is reasonable to conclude that the strengths of the proposed single-pass reliability analysis algorithm are its accuracy, scalability to large circuits, and speedup in performance.

C. Maximum-k Gate Failure Model

The results for reliability analysis under the maximum-k gate failure model for different benchmark circuits is shown in Table VII. The results have been presented for three values of k. Single-pass reliability analysis with zero correlation coefficients was used as the core engine to evaluate the reliability for the 50,000 samples generated by the sampling algorithm. A significant fraction of the runtime can be attributed to the weight vector computation $\mathcal{W}$. For the maximum-k gate failure reliability analysis, $\mathcal{W}$ is computed only once and stored for use in all the 50,000 runs. Thus, the runtime reported for maximum-k gate failure reliability analysis shown in Table VII

TABLE V
COMPARISON BETWEEN OBSERVABILITY-BASED AND MONTE CARLO RELIABILITY ANALYSIS FOR TEN BENCHMARK CIRCUITS. EIGHT VALUES OF ϵ HAVE BEEN USED FOR COMPARISON

Benchmark	Size	Average error over all outputs (%)								Runtimes	
		$\epsilon = 0.001$	$\epsilon = 0.01$	$\epsilon = 0.05$	$\epsilon = 0.1$	$\epsilon = 0.15$	$\epsilon = 0.2$	$\epsilon = 0.25$	$\epsilon = 0.3$	Monte Carlo	Observability-based
x2	56	0.13	1.4	21.88	18.98	15.6	12.15	9.03	6.59	8m	0.019s
cu	59	0.23	1.1	16.35	15.61	14.27	13.03	11.23	9.14	9m	0.019s
b9	210	0.2	1.07	30.14	30.19	26.22	21.16	16.23	11.75	37m	0.034s
c499	650	5.3	25	38.29	26.69	17.89	11.70	7.38	4.38	134m	72s
c1355	653	4.5	22.3	37.96	26.5	17.59	11.26	6.90	4.08	135m	95s
c1908	699	1.4	8.6	11.99	9.38	7.02	5.48	4.27	3.15	145m	24.64s
c2670	756	0.89	6	11.45	11.14	9.43	7.62	5.94	4.42	208m	93m
frg2	1024	0.34	2.4	21.36	24.82	25.29	23.29	19.86	15.81	286m	0.657s
c3540	1466	2.2	16.1	24.97	18.97	14.53	11.11	8.50	6.38	431m	16h
i10	2643	2.8	14.2	20.03	18.18	16.30	14.23	12.03	9.75	1668m	> 24h

TABLE VI
COMPARISON OF ACCURACY AND RUNTIMES FOR SINGLE-PASS RELIABILITY ANALYSIS WITH 0, 4, AND 16 CORRELATION COEFFICIENTS

Benchmark	Size	Average error over all outputs and over 10 values of $\epsilon \in [0,0.5]$ (%)			Runtimes (for 10 runs)				
					Monte Carlo	Single-pass			
		0 corr coeffs	4 corr coeffs	16 corr coeffs		Setup time	0 corr coeffs	4 corr coeffs	16 corr coeffs
x2	56	0.86	0.83	0.68	107.6s	0.036s	0.036s	0.029s	0.029s
cu	59	0.32	0.32	0.32	133.2s	0.048s	0.048s	0.028s	0.032s
b9	210	0.42	0.41	0.41	325.9s	0.033s	0.033s	0.048s	0.049s
c499	650	13.1	11.2	11.11	35m	7s	11.2s	74.5s	147s
c1355	653	13.5	11.59	11.45	35m	10s	15.8s	74.9s	137s
c1908	699	6.5	7.85	3.97	29m	0.6s	0.99s	131s	300s
c2670	756	1.34	1.62	0.95	71m	0.7s	1.05s	11.9s	11s
frg2	1024	2.96	2.42	1.49	54m	0.15s	0.22s	1.62s	2.88s
c3540	1466	4.2	5.27	2.89	118m	6.3s	9.68s	17m	39m
i10	2643	1.7	1.14	1.12	687m	13.5s	17.09s	12m	24m

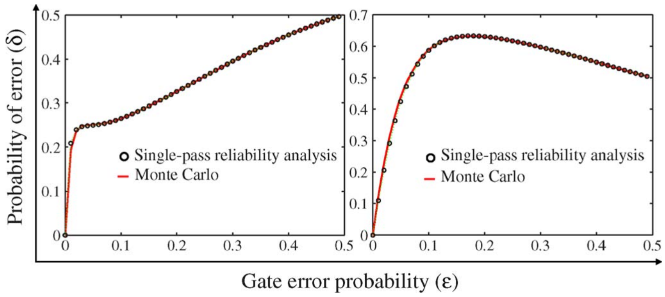


Fig. 8. $\delta(\epsilon)$ curves for two outputs of i10.

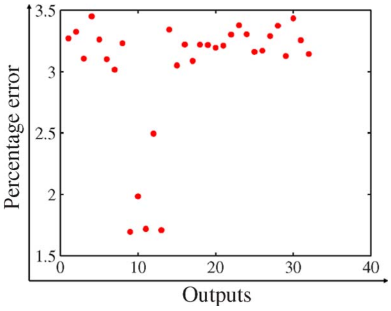


Fig. 9. Average error in $\delta(\vec{\epsilon})$ per output of circuit c499 over 1000 runs. On each run, $\epsilon_i \in$ Uniform (0,0.5) for each gate.

is much less than 50,000 times the runtime reported for single-pass reliability analysis shown in Table VI. Thus, the proposed maximum-k gate failure reliability analysis algorithm is scalable to large circuits. Another important point to be noted is that the runtimes increase very slowly as k increases. This is because the complexity of the sampling algorithm is polynomial in k.

TABLE VII
RUNTIMES FOR DIFFERENT BENCHMARK CIRCUITS UNDER MAXIMUM-k GATE FAILURE MODEL. A TOTAL OF 50-K SAMPLES WERE USED TO EVALUATE RELIABILITY

Benchmark	Size	Runtime for different values of k		
		$k = 1$	$k = 2$	$k = 3$
x2	69	0.29s	0.43s	0.60s
cu	81	0.33s	0.51s	0.71s
b9	167	0.66s	1.02s	1.46s
c499	697	55.15s	57.61s	59.64s
c1355	803	37.39s	49.81s	42.15s
c1908	718	7.99s	10.67s	12.91s
c2670	1073	7.61s	10.27s	13.13s
frg2	1024	5.86s	8.17s	10.75s
c3540	1737	32.68s	37.16s	41.59s
i10	3442	100.27s	108.3s	125.2s

VII. CONCLUSION

Even as reliability gains wide acceptance as a significant design challenge, there is a lack of effective techniques for its analysis and optimization. This paper described three accurate and scalable techniques for reliability analysis of logic circuits for different gate failure models. The observability-based reliability analysis algorithm provides a closed-form expression for reliability using the observabilities of the gates, and is accurate when single gate failures are dominant. The single-pass reliability analysis algorithm is based on a single topological walk through the circuit to compute the reliability of the circuit, and is accurate even for multiple gate failures. The maximum-k gate failure reliability analysis algorithm allows an upper limit, k, on the number of simultaneous gate failures in a circuit. An efficient sampling technique is proposed to generate a set of $\leq k$ failed gates, and single-pass reliability analysis is used to evaluate the effect of these failed gates at the outputs. The three algorithms have potential applications to reliability analysis of failures arising due to a broad range of mechanisms including single-event effects, process variations, and reliability degradation due to aging.

References

[1] R. W. Keyes, "Fundamental limits of silicon technology," *Proc. IEEE*, vol. 89, no. 3, pp. 227–239, Mar. 2001.

[2] J. D. Meindl, Q. Chen, and J. A. Davis, "Limits on silicon nanoelectronics for terascale integration," *Science*, vol. 293, no. 5537, pp. 2044–2049, Sep. 2001.

[3] G. Bourianoff, "The future of nanocomputing," *Computer*, vol. 36, no. 8, pp. 44–53, Aug. 2003.

[4] V. V. Zhirnov, R. K. Cavin, J. A. Hutchby, and G. I. Bourianoff, "Limits to binary logic switch scaling—A Gedanken model," *Proc. IEEE*, vol. 91, no. 11, pp. 1934–1939, Nov. 2003.

[5] M. A. Breuer, S. K. Gupta, and T. M. Mak, "Defect and error tolerance in the presence of massive numbers of defects," *IEEE Des. Test Comput.*, vol. 21, no. 3, pp. 216–227, May/Jun. 2004.

[6] J. von Neumann, "Probabilistic logics and the synthesis of reliable organisms from unreliable components," in *Automata Studies*, C. E. Shannon and J. McCarthy, Eds. Princeton, NJ: Princeton Univ. Press, 1956, pp. 43–98.

[7] A. Sadek, K. Nikoliæ, and M. Forshaw, "Parallel information and computation with restitution for noise-tolerant nanoscale logic networks," *Nanotechnology*, vol. 15, no. 1, pp. 192–210, Jan. 2004.

[8] 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.

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

[10] 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.

[11] D. Bhaduri and S. Shukla, "Nanolab: A tool for evaluating reliability of defect-tolerant nano architectures," in *Proc. Annu. Symp. VLSI*, 2004, pp. 25–31.

[12] B. Zhang, W.-S. Wang, and M. Orshansky, "FASER: Fast analysis of soft error susceptibility for cell-based designs," in *Proc. Int. Symp. Quality Electron. Des.*, 2006, pp. 755–760.

[13] M. Zhang and N. R. Shanbhag, "A soft error rate analysis (SERA) methodology," in *Proc. Int. Conf. Comput.-Aided Des.*, 2004, pp. 111–118.

[14] S. Almukhaizim, Y. Makris, Y.-S. Yang, and A. Veneris, "Seamless integration of SER in rewiring based design space exploration," in *Proc. Int. Test Conf.*, 2006, pp. 1–9.

[15] S. Krishnaswamy, S. Plaza, I. Markov, and J. Hayes, "Enhancing design robustness with reliability-aware resynthesis and logic simulation," in *Proc. Int. Conf. Comput.-Aided Des.*, 2007, pp. 149–154.

[16] M. Choudhury and K. Mohanram, "Approximate logic circuits for low overhead, non-intrusive concurrent error detection," in *Proc. Des. Autom. Test Eur.*, 2008, pp. 903–908.

[17] M. Choudhury and K. Mohanram, "Accurate and scalable reliability analysis of logic circuits," in *Proc. Des. Autom. Test Eur.*, 2007, pp. 1454–1459.

[18] M. Nemani and F. Najm, "Towards a high-level power estimation capability," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 15, no. 6, pp. 588–598, Jun. 1996.

[19] M. Damiani and G. De Micheli, "Observability don't care sets and Boolean relations," in *Proc. Int. Conf. Comput.-Aided Des.*, 1990, pp. 502–505.

[20] T. Larrabee, "Test pattern generation using Boolean satisfiability," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 11, no. 1, pp. 4–15, Jan. 1992.

[21] S. Ercolani, M. Favalli, M. Damiani, P. Olivo, and B. Ricco, "Estimate of signal probability in combinational logic networks," in *Proc. Eur. Test Conf.*, 1989, pp. 132–138.

[22] G. Fishman, "Monte Carlo: Concepts, algorithms and applications," in *Springer Series in Operations Research*. New York: Springer-Verlag, 1995.

Mihir R. Choudhury (S'06) received the B.Tech. degree in computer science and engineering from Indian Institute of Technology, Bombay, India, in 2005 and the M.S. degree in electrical and computer engineering from Rice University, Houston, TX, in 2008, where he is currently working toward the Ph.D. degree in computer engineering.

His research interests include logic synthesis, circuit simulation, and design for reliability in scaled electronic technologies.

Kartik Mohanram (S'00–M'04) received the B.Tech. degree in electrical engineering from Indian Institute of Technology, Bombay, India, in 1998 and the M.S. and Ph.D. degrees in computer engineering from University of Texas, Austin, in 2000 and 2003, respectively.

He is currently an Assistant Professor with the Department of Electrical and Computer Engineering, Rice University, Houston, TX. His primary research interests are in computer engineering and systems, with an emphasis on modeling, simulation, and computer-aided design for scaled electronic technologies.

Dr. Mohanram is a recipient of the National Science Foundation CAREER Award, the Association for Computing Machinery/Special Interest Group Design Automation Technical Leadership Award, and the A. Richard Newton Graduate Scholarship.