Probabilistic Error Propagation Modeling of Logic Circuits

S. Gupta, A.J.C. van Gemund
Faculty of Electrical Engineering, Mathematics, and Computer Science
Delft University of Technology
P.O. Box 5031, NL-2600 GA Delft, The Netherlands

Abstract

Recent study has shown that accurate knowledge of the false negative rate (FNR) of tests can significantly improve the diagnostic accuracy of spectrum-based fault localization. To understand the principles behind FNR modeling in this paper we study three error propagation probability (EPP) modeling approaches applied to a number of logic circuits from the 74XXX/ISCAS-85 benchmark suite. Monte Carlo simulations for random injected faults show that a deterministic approach that models gate behavior provides high accuracy ($O(1\%)$), while probabilistic approaches that abstract from gate modeling generate higher prediction errors ($O(10\%)$), which increase with the number of injected faults.

1. Introduction

In software debugging spectrum-based fault localization (SFL) has gained widespread interest (e.g., [1, 8]). Given this development, application of SFL to hardware becomes of interest. While the performance of SFL will fundamentally be less than a traditional, model-based diagnosis (MBD) approach to hardware diagnosis, the fact that SFL does not require specific, behavioral component models may often outweigh the additional cost (and inaccuracy) of a modeling approach.

Recently, a Bayesian approach to SFL has been introduced [2, 3] that outperforms low-cost, statistical SFL approaches at moderate increase of computational cost. An important requirement of Bayesian approaches to SFL, however, is knowledge on the probability that a faulted component will actually generate a system-level failure. The probability that a test that covers a faulted component does not capture the defect is also known as false negative rate (FNR), or coincidental correctness (in the software engineering domain). In [6] it is shown that the performance of Bayesian approaches is quite sensitive to the accuracy of this probability estimation.

Let g_j denote the probability that component $c_j, j = 1, \ldots, M$, when faulted, will not generate a system-level failure ('g' for 'good'). An important reason for g being (much) larger than zero is the fact that (1) a faulted component need not generate an error at its output(s), and (2) even if it does, that error may fail to propagate through the entire system. For example, consider the simple logic circuit comprising an INV gate (c_1) connected to an AND gate (c_2) according to Figure 1. Suppose the INV gate is faulted. For input $x = (\mathrm{X}, 0)$ (X = don't care) an error at the output of c_1 will be masked by the fact

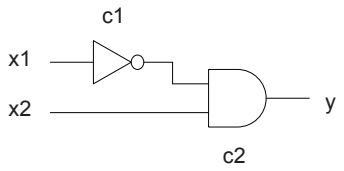

Figure 1. Example logic circuit

that c_2 will always produce $y = 0$. However, for input $x = (X, 1)$ an inverter error will always propagate to y. Assuming a uniform input value probability distribution, the probability of failure at y is 0.5, and consequently, $g_1 = 1 - 0.5 = 0.5$.

Similar to mutation analysis in software, one can inject faults in hardware and observe the FNR of the test suite to derive g_j. Indeed, Monte Carlo simulation has been frequently used. However, similar to the software case, low-complexity static approaches are preferable, provided their accuracy is tolerable.

In this paper we study an analytic approach to error propagation probability (EPP) computation for logic circuits, aimed to provide an understanding of the basic rules underlying the FNR of computational systems. Given a model that predicts the error probability at a circuit output given the error state of its inputs, the problem of modeling g_j can be framed as an EPP problem pertaining to the subcircuit that connects the outputs of the faulted components to the circuit's primary output under consideration. The probability that the output of a faulted component is, in fact, erroneous is a separate, component-specific problem that relates to its fault mode, and intermittency (if any), and will not be considered in this paper (In terms of the PIE approach in software [11] we focus on the propagation part.)

EPP in logic circuits has been studied in the context of reliability studies, primarily motivated by an increasing soft error rate due to the ever decreasing gate sizes [4]. While there exists an approach to compute the EPP through circuits of arbitrary topology given the fault model of each gate involved, this approach is not entirely compositional, i.e., one cannot *numerically* compute the EPP at the next stage in the circuit given the EPP computed at the previous stage. Rather, one must compile a *symbolic* expression for the entire circuit path from primary input to output in order to suppress higher-order terms (variables with exponents) occurring in the compilation due to expression composition [10], followed by numeric evaluation given the inputs and faulted components. Clearly however, for very large circuits (and path combinations) the exponential space complexity of such a symbolic scheme is prohibitive.

In this paper we present a compositional, probabilistic approach to EPP computation. To avoid the above suppression problem (and the associated exponential space complexity of the symbolic scheme) we employ probabilistic gate models instead of the exact gate expressions used in the earlier approach. While our method produces exact estimates of the *mean value* of the EPP, the EPP value found for a particular circuit output may differ from the correct value. However, a certain error is acceptable provided the correct posterior probability ranking in the diagnosis algorithm is not too seriously affected.

The paper is organized as follows. In Section 2 we introduce the EPP problem, and describe why deterministic modeling suffers from exponential complexity problems when an exact solution is required. In Section 3 we present our probabilistic model. In Section 4 we show how the FNR can be computed under a single-fault assumption (so the g_j) using the deterministic model. In Section 5 we discuss the accuracy of the deterministic and probabilistic model for multiple faults by comparing their predictions to Monte Carlo simulations using a subset of the 74XXX/ISCAS benchmark circuits. Section 6 summarizes the paper.

2 Deterministic EPP

In this section we summarize the general results for a number of binary gates, and describe the problems associated with composition when deriving the EPP for circuit ouputs.

2.1 Gate-level EPP

Consider a binary AND gate with inputs x_1, x_2, and output y. If the AND gate is nominal (i.e., not faulted) it holds $y = x_1 x_2$. In the following we will derive the EPP for the AND, i.e., the probability that an error at either or both inputs propagates to the output. Let e_i and v_i denote the error probability and value truth probability (v for value) of input x_i, respectively. We distinguish four input cases:

- If $(x_1, x_2) = (0, 0)$ (with probability $(1 - v_1)(1 - v_2)$), an error in both inputs (with probability $e_1 e_2$) will propagate to y.
- If $(x_1, x_2) = (0, 1)$ (with probability $(1 - v_1)v_2$), an error in x_1 (with probability e_1) will propagate to y.
- If $(x_1, x_2) = (1, 0)$ (with probability $v_1(1 - v_2)$), an error in x_2 (with probability e_2) will propagate to y.
- If $(x_1, x_2) = (1, 1)$ (with probability $v_1 v_2$)), an error in any input (with probability $e_1 + e_2 - e_1 e_2$) will propagate to y.

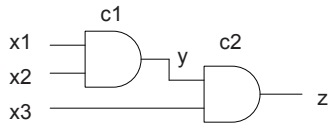

Figure 2. Simple circuit

Summing up the probabilities the error probability of y is given by

$$\begin{aligned} e_y &= (1-v_1)(1-v_2)e_1e_2 + (1-v_1)v_2e_1 + v_1(1-v_2)e_2 + v_1v_2(e_1+e_2-e_1e_2) \\ &= e_1(1-e_2)v_2 + e_2(1-e_1)v_1 + e_1e_2(1-v_1-v_2+2v_1v_2) \end{aligned}$$

In the same way the EPP expressions are derived for binary gates OR, XOR, NXOR. In summary,

- AND gate

$$e_1(1-e_2)v_2 + e_2(1-e_1)v_1 + e_1e_2(1-v_1-v_2+2v_1v_2) \tag{1}$$

- OR gate

$$e_1(1-v_2) + e_2(1-v_1) - e_1e_2(1-2v_1v_2) \tag{2}$$

- X[N]OR gate

$$e_1 + e_2 - e_1e_2 \tag{3}$$

Note that not all gates require knowledge of the input value probabilities v_i, which implies that for certain circuits an exact EPP can be computed numerically without having to resort to symbolic techniques.

2.2 Circuit-level EPP

Consider the ternary AND circuit shown in Figure 2 for which we derive e_z using the above EPP models. It follows

$$\begin{aligned} e_y &= e_1(1-e_2)v_2 + e_2(1-e_1)v_1 + e_1e_2(1-v_1-v_2+2v_1v_2) \\ v_y &= v_1v_2 \\ e_z &= e_y(1-e_3)v_3 + e_3(1-e_y)v_y + e_ye_3(1-v_y-v_3+2v_yv_3) \\ &= (e_1(1-e_2)v_2 + e_2(1-e_1)v_1 + e_1e_2(1-v_1-v_2+2v_1v_2)) \\ &\quad (1-e_3)v_3 + e_3(1-(e_1(1-e_2)v_2 + e_2(1-e_1)v_1 + e_1e_2(1-v_1-v_2+2v_1v_2)))v_1v_2 + \\ &\quad (e_1(1-e_2)v_2 + e_2(1-e_1)v_1 + e_1e_2(1-v_1-v_2+2v_1v_2))e_3(1-v_1v_2-v_3+4v_1v_2v_3) \end{aligned}$$

It can be seen that in e_z higher-order terms for v_1, v_2 occur when multiplying e_y and v_y (e.g., v_1^2). In [10] it is shown that expression correctness is preserved if the exponents are suppressed (i.e., terms such as v_1^k must be reduced to v_1). While suppression in the above, *symbolic* scheme provides analytic correctness, it becomes clear that a *numeric*, compositional scheme where e_y and v_y are first computed and then substituted into the expression for e_z, will produce incorrect results when the $v_i < 1$. Consequently, the need of suppression in the composition of individual gate expressions prohibits a simple, numerical approach to compositional EPP.

3 Probabilistic EPP

In this section we describe our probabilistic approach to EPP computation. As discussed in the previous section, including the signal value probabilities v_i into the EPP models prohibits a numerical approach to compositional EPP circuit analysis. Consequently, we propose an approach where we abstract from the specific gate models such that the individual v_i are no longer required. Consider a binary AND gate. Instead of taking into account the individual v_i we merely take into account that the AND gate function produces $y = 1$ in 1 out of 4 input cases. Let a denote the value truth probability of a gate's output (i.e., the number of '1's in its truth table). Then for an AND gate it follows $a = 1/4$. Note that in this approach we do not distinguish between any gates that have $a = 1/4$ (i.e., 4 binary gates, of which AND is only one example).

In the following we derive the EPP for binary gates based on the single knowledge that $a = 1/4$. Suppose correct input values for x_i would lead to $y = 1$ (probability $a = 1/4$). Any error at either input (with probability $e_1 + e_2 - e_1e_2$) will lead to a different entry for y in the 4-entry truth table. Suppose an error-free input would produce $y = 1$ (probability a). For an $a = 1/4$ type truth table, the probability that a change in the truth table from this $y = 1$ entry will produce $y = 0$ equals $\mathsf{Pr}(0|1) = 1$, as all alternatives in the truth table have $y = 0$ (the possible truth table outputs for $a = 1/4$ are $\{0, 0, 0, 1\}, \{0, 0, 1, 0\}, \{0, 1, 0, 0\}$, and $\{1, 0, 0, 0\}$). Consequently, the probability of this $1 \rightarrow 0$ event occurring is $1/4 \cdot (e_1 + e_2 - e_1e_2) \cdot 1$. Similarly, it follows that the probability of an $0 \rightarrow 1$ event is given by $e_y = 3/4 \cdot (e_1 + e_2 - e_1e_2) \cdot 1/3$. Summing the probabilities it follows that for and $a = 1/4$ type of gate we have

$$e_y = \frac{e_1 + e_2 - e_1e_2}{2} \tag{4}$$

Note that the above probabilistic model equals the deterministic AND EPP model (1) for random inputs (i.e., $v_i = 0.5$). However, this is coincidental, and does not hold for any a. Consequently, our probabilistic approach is not just an instantiation of the deterministic model for random signal values ($v_i = 0.5$). In fact, the above, abstract approach does not assume anything about signal value probabilities.

The above approach is easily generalized to any a value. Again, suppose an error-free input would produce $y = 1$ (probability a). Since there are $(1 - a)2^2$ 0's in a truth table that are distributed over $2^2 - 1$ entries next to the original $y = 1$ entry, it follows that the probability of a change in the truth table from the $y = 1$ entry will produce $y = 0$ equals $\mathsf{Pr}(0|1) = (1 - a)2^2/(2^2 - 1)$. Consequently, the probability of an $1 \rightarrow 0$ event (i.e., an output error) is given by $e_y = a \cdot (e_1 + e_2 - e_1e_2) \cdot (1 - a)2^2/(2^2 - 1)$. Similarly, it follows that the probability of a $0 \rightarrow 1$ event is given by $e_y = (1 - a) \cdot (e_1 + e_2 - e_1e_2) \cdot a2^2/(2^2 - 1)$. Summing the probabilities it follows

$$e_y = \frac{8}{3}(e_1 + e_2 - e_1e_2)a(1 - a) \tag{5}$$

Again, note that for most a values the above probabilistic model does not equal the earlier, deterministic models. Instead, the model represents the average over all possible deterministic gate models with a particular a, as has been verified using MC simulation. Thus the probabilistic model provides an exact estimation of the *mean* EPP over all gate models with particular a but will only provide an approximation for particular gates. For instance, for $a = 1/2$ Eq. (5) yields $e = 2(e_1 + e_2 - e_1e_2)/3$ while the XOR produces $e_{\text{XOR}} = e_1 + e_2 - 2e_1e_2$. For $e_1 = e_2 = 1$, we have $e = 2/3$ and $e_{\text{XOR}} = 0$, respectively, which implies a significant estimation error. Consequently, the probabilistic model should be used with care.

While the probabilistic model circumvents the suppression problem there may be modeling situations where knowledge of a is not available, since in an SFL context no modeling information is required, or available. In that case we need to abstract even further, and simply assume that the outcome of a boolean function is randomly distributed between 0 and 1. Though one might assume Eq. 5 would apply with $a = 1/2$ this is not the case, as in the situation where we do not assume anything about the gate's truth table we cannot compute $\mathsf{Pr}(0|1)$ or $\mathsf{Pr}(1|0)$ as the number of 0's and 1's are not known. Rather, it holds $\mathsf{Pr}(0|1) = \mathsf{Pr}(0) = (1 - a) = 1/2$ (and, similarly, $\mathsf{Pr}(1|0) = a = 1/2$). Consequently, the probability of an $1 \rightarrow 0$ event (i.e., an output error) is given by $e_y = \cdot(e_1 + e_2 - e_1e_2) \cdot 1/2 \cdot 1/2$. Similarly, it follows that the probability of a $0 \rightarrow 1$ event is given by $e_y = \cdot(e_1 + e_2 - e_1e_2) \cdot 1/2 \cdot 1/2$. Summing the probabilities it follows

$$e_y = \frac{1}{2}(e_1 + e_2 - e_1e_2) \tag{6}$$

Note that this result equals the probabilistic gate model for $a = 1/4$, rather than $a = 1/2$.

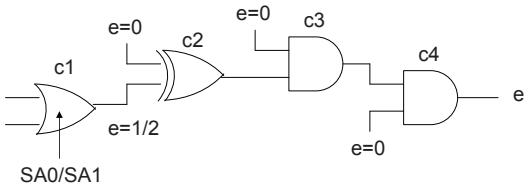

Figure 3. Example logic circuit

4 Coincidental Correctness

As mentioned in the Introduction the fact that the EPP of errors generated by faulted components is (much) less than unity implies that tests that involve faulted components may not capture the defects (i.e., the test may still pass). As this coincidental correctness is an important input to Bayesian diagnosis techniques in this section we derive this parameter g_j where j is the index of the (single-)faulted gate in a logic circuit.

In the following we assume that a gate can be either stuck-at-zero (SA0) or stuck-at-one (SA1) with equal probability. Independent of the actual truth table of the gate this implies that the EPP of the faulted gate's output is given by e=1/2. Let a be the fraction of '1's in the truth table. In the SA0 case this implies that in a fraction a of the cases SA0 leads to an error, while in the SA1 case the error probability is $(1-a)$. Summing the probabilities we have $e = 1/2 \cdot a + 1/2 \cdot (1-a) = 1/2$, regardless of a.

To determine if the gate error will propagate to a primary output we apply the EPP models derived in the earlier sections. In principle, we apply the probabilistic model to avoid suppression problems when we cannot rule out the possibility of AND or OR gates in the circuit. However, computing the g factor for *multiple*-fault candidates is typically not performed in view of the exponential number of possible multiple-fault combinations that have to be taken into account. Rather, one computes the *single-fault* g (only M components), and estimates g for multiple-fault candidates based on an OR-model (the probability a test that involves M_f faulted components will still pass is when each component would yield a pass, i.e., the product of the individual g_j [1]).

The use of the above OR model (or any other model that estimates g for multiple-fault candidates in terms of the individual g_j) implies that we only need to compute the EPP for a circuit with only *one* faulted gate. This has profound consequences for the EPP calculation as shown by the following. Consider the circuit shown in Figure 3. As there is only one faulty gate, all inputs have $e_i = 0$. Consequently, all deterministic gate models (Eq. (1) - (3)) reduce to

- AND gate

$$e_1 v_2 \tag{7}$$

- OR gate

$$e_1(1 - v_2) \tag{8}$$

- X[N]OR gate

$$e_1 \tag{9}$$

where we have chosen e_1 as the gate input that is within the error propagation path from faulted gate to primary output (i.e., $e_2 = 0$). It can be easily seen that composition of the above, reduced EPP models does *not* suffer from the suppression problem (unless there are reconvergent subcircuits). Consequently, for single fault g modeling we need not necessarily resort to the probabilistic EPP models with the associated variance problems.

As the definition of g_j is static the actual signal values v_i are not known. Consequently, in the above EPP models we will assume $v_2 = 1/2$. This implies that for AND and OR gates the EPP is attenuated by a factor 2 while X[N]OR gates pass on the EPP. In the example circuit in Figure 3 it follows that $e = 1/8$, and, consequently, $g_1 = 7/8$.

[1] Although generally accepted practice, the OR model is known to be an approximation, and is currently subject to research by the authors.

Note that the deterministic model applies only if the circuit does not include reconvergent subcircuits, since this (re)introduces the suppression problem. Nevertheless, the deterministic model can be applied at minimal loss of accuracy as long as reconvergence is small, as is the case in the 74XXX/ISCAS85 circuits, as is shown in the next section.

5 Experimental Results

In the above we have presented three EPP models, (1) the deterministic model, that, when implemented in terms of numeric composition, suffers from suppression when considering multiple-faults and/or circuit reconvergence (Section 2), (2) the probabilistic model that has knowledge of the fraction of 1's in a gate's truthtable (Eq. 5), and (3) the probabilistic model that has no gate model, whatsoever (Eq. 6).

In this section we present accuracy data for all three models for a number of circuits from the 74XXX/ISCAS85 benchmark suite [5, 7] [2]. The benchmark circuits are described in Table 1.

Circuit	Description	|gates|	|inputs|	|outputs|
74182	4-bit carry-lookahead generator	19	9	5
74L85	4-bit magnitude comparator	33	11	3
74181	4-bit ALU	65	14	8
74283	4-bit adder	36	9	5
c17	simple test circuit	6	5	2
c432	27-chan interrupt controller	160	36	7
c499	32-bit SEC circuit	202	41	32
c880	8-bit ALU	383	60	26
c1355	32-bit SEC circuit	546	41	32
c1908	16-bit SEC/DED circuit	880	33	25
c2670	12-bit ALU and controller	1193	233	140
c3540	8-bit ALU	1669	50	22
c5315	9-bit ALU	2307	178	123
c6288	16x16-bit multiplier	2416	32	32
c7552	32-bit adder/comparator	3512	207	108

Table 1. 74XXX/ISCAS85 benchmark circuits

The predictions of the deterministic model and the probabilistic models Eq. 5 and Eq. 6, are compared to the results of Monte-Carlo simulation (random inputs and fault injections), yielding relative prediction errors ϵ_1, ϵ_2, and ϵ_3, respectively, according to

$$\epsilon_i = |\frac{e_i - e_{\text{MC}}}{e_{\text{MC}}}|$$

Each circuit output is considered as a separate circuit (cone) within the complete circuit.

Table 2 shows the mean (E), variance (V), lower bound (L), and upper bound (U) of the relative errors ϵ_1 through ϵ_3, respectively, for single-fault injections (rounded to 3 decimals). We show the accuracy results per output for the smaller 74XXX circuits, while for the ISCAS circuits we summarize the error statistics for all outputs combined. As expected, the deterministic model performs very well. The average prediction error, due to reconvergence, is quite small. Outliers within 10%, with the exception of c880. The probabilistic models perform less accurately, with an average error well less than 10%, with outliers up to 45%.

Table 3 and 4 show similar data for double and triple faults, respectively. Contrary to expectation, the average error in the deterministic model due to ignoring suppression (next to reconvergence) is still in the order of percents, although the upper bound increases considerably (up to 28%). However, the error increases with fault cardinality as the need for suppression increases. The average error of the probabilistic models is much higher (in the order of ten percents) with outliers up to 45%, and also increases with cardinality. For all cardinalities the difference between both probabilistic models is relatively small. The significance of the latter result is that knowledge of the pdf of the truth table does not significantly affect prediction performance.

[2] Data on more circuits will be available in the final version.

Circuit	Output	E[ϵ_1]	S[ϵ_1]	L[ϵ_1]	U[ϵ_1]	E[ϵ_2]	S[ϵ_2]	L[ϵ_2]	U[ϵ_2]	E[ϵ_3]	S[ϵ_3]	L[ϵ_3]	U[ϵ_3]
74182	00	0.000	0.000	0.000	0.001	0.000	0.000	0.000	0.001	0.000	0.000	0.000	0.001
74182	01	0.009	0.017	0.000	0.049	0.098	0.193	0.000	0.598	0.098	0.193	0.000	0.598
74182	02	0.010	0.018	0.000	0.049	0.098	0.194	0.000	0.598	0.098	0.194	0.000	0.598
74182	03	0.007	0.015	0.000	0.052	0.059	0.140	0.000	0.474	0.059	0.140	0.000	0.474
74182	04	0.003	0.016	0.000	0.070	0.020	0.060	0.000	0.198	0.020	0.060	0.000	0.198
74L85	00	0.004	0.005	0.000	0.014	0.004	0.004	0.000	0.008	0.004	0.004	0.000	0.009
74L85	01	0.016	0.020	0.000	0.076	0.108	0.138	0.000	0.547	0.108	0.138	0.000	0.547
74L85	02	0.016	0.020	0.000	0.077	0.108	0.138	0.000	0.558	0.108	0.138	0.000	0.558
74181	00	0.008	0.014	0.000	0.044	0.044	0.094	0.000	0.259	0.044	0.094	0.000	0.259
74181	01	0.016	0.027	0.000	0.100	0.052	0.103	0.000	0.438	0.052	0.103	0.000	0.438
74181	02	0.020	0.029	0.000	0.123	0.059	0.094	0.000	0.380	0.059	0.094	0.000	0.380
74181	03	0.018	0.024	0.000	0.087	0.104	0.184	0.000	0.608	0.120	0.216	0.000	0.750
74181	04	0.012	0.019	0.000	0.070	0.090	0.172	0.000	0.590	0.107	0.208	0.000	0.746
74181	05	0.005	0.011	0.000	0.055	0.074	0.157	0.000	0.582	0.092	0.197	0.000	0.754
74181	06	0.001	0.005	0.000	0.040	0.055	0.146	0.000	0.588	0.070	0.186	0.000	0.760
74181	07	0.030	0.023	0.000	0.060	0.049	0.034	0.000	0.138	0.054	0.038	0.000	0.150
74283	00	0.016	0.023	0.000	0.071	0.065	0.119	0.000	0.408	0.065	0.119	0.000	0.408
74283	01	0.010	0.018	0.000	0.062	0.100	0.143	0.000	0.392	0.118	0.172	0.000	0.504
74283	02	0.005	0.013	0.000	0.055	0.086	0.134	0.000	0.337	0.105	0.168	0.000	0.498
74283	03	0.001	0.006	0.000	0.042	0.071	0.129	0.000	0.341	0.091	0.167	0.000	0.509
74283	04	0.000	0.001	0.000	0.006	0.056	0.125	0.000	0.337	0.075	0.170	0.000	0.505
c17	all	0.005	0.012	0.000	0.046	0.062	0.071	0.000	0.203	0.062	0.071	0.000	0.203
c499	all	0.002	0.003	0.000	0.011	0.007	0.030	0.000	0.340	0.006	0.038	0.000	0.506
c880	all	0.002	0.026	0.000	0.580	0.009	0.053	0.000	0.907	0.009	0.053	0.000	0.907

Table 2. EPP model accuracy (single faults)

Circuit	Output	E[ϵ_1]	S[ϵ_1]	L[ϵ_1]	U[ϵ_1]	E[ϵ_2]	S[ϵ_2]	L[ϵ_2]	U[ϵ_2]	E[ϵ_3]	S[ϵ_3]	L[ϵ_3]	U[ϵ_3]
74182	00	0.000	0.000	0.000	0.004	0.000	0.000	0.000	0.004	0.000	0.000	0.000	0.004
74182	01	0.018	0.021	0.000	0.051	0.187	0.223	0.000	0.601	0.187	0.223	0.000	0.601
74182	02	0.027	0.058	0.000	0.258	0.168	0.262	0.000	0.854	0.168	0.262	0.000	0.854
74182	03	0.030	0.063	0.000	0.286	0.182	0.226	0.000	0.688	0.182	0.226	0.000	0.668
74182	04	0.005	0.016	0.000	0.072	0.032	0.072	0.000	0.207	0.032	0.072	0.000	0.207
74L85	00	0.010	0.011	0.000	0.040	0.006	0.003	0.000	0.012	0.006	0.003	0.000	0.012
74L85	01	0.020	0.055	0.000	0.318	0.143	0.167	0.000	0.621	0.143	0.167	0.000	0.621
74L85	02	0.018	0.020	0.000	0.073	0.150	0.155	0.000	0.585	0.150	0.155	0.000	0.585
74181	00	0.017	0.023	0.000	0.113	0.100	0.130	0.000	0.420	0.100	0.130	0.000	0.420
74181	01	0.033	0.037	0.000	0.123	0.105	0.146	0.000	0.678	0.105	0.146	0.000	0.678
74181	02	0.030	0.030	0.000	0.117	0.114	0.124	0.000	0.560	0.114	0.124	0.000	0.560
74181	03	0.033	0.039	0.000	0.179	0.184	0.218	0.000	0.682	0.217	0.260	0.000	0.793
74181	04	0.024	0.037	0.000	0.191	0.179	0.220	0.000	0.609	0.218	0.274	0.000	0.750
74181	05	0.012	0.024	0.000	0.166	0.111	0.177	0.000	0.587	0.134	0.215	0.000	0.751
74181	06	0.001	0.005	0.000	0.040	0.108	0.195	0.000	0.582	0.143	0.253	0.000	0.754
74181	07	0.039	0.019	0.000	0.070	0.077	0.036	0.000	0.140	0.086	0.040	0.000	0.156
74283	00	0.036	0.029	0.000	0.100	0.178	0.183	0.000	0.534	0.178	0.183	0.000	0.534
74283	01	0.010	0.016	0.000	0.055	0.194	0.168	0.000	0.496	0.241	0.212	0.000	0.514
74283	02	0.007	0.015	0.000	0.056	0.126	0.144	0.000	0.362	0.157	0.185	0.000	0.500
74283	03	0.001	0.007	0.000	0.042	0.095	0.135	0.000	0.337	0.128	0.184	0.000	0.505
74283	04	0.000	0.001	0.000	0.004	0.093	0.149	0.000	0.338	0.128	0.207	0.000	0.506
c17	all	0.026	0.080	0.000	0.293	0.080	0.086	0.000	0.200	0.080	0.086	0.000	0.200
c499	all	0.005	0.005	0.000	0.024	0.015	0.043	0.000	0.343	0.013	0.056	0.000	0.511
c880	all	0.006	0.038	0.000	0.582	0.018	0.077	0.000	0.910	0.018	0.077	0.000	0.910

Table 3. EPP model accuracy (double faults)

Circuit	Output	E[ϵ_1]	V[ϵ_1]	L[ϵ_1]	U[ϵ_1]	E[ϵ_2]	V[ϵ_2]	L[ϵ_2]	U[ϵ_2]	E[ϵ_3]	V[ϵ_3]	L[ϵ_3]	U[ϵ_3]
74182	00	0.000	0.001	0.000	0.004	0.000	0.001	0.000	0.004	0.000	0.001	0.000	0.004
74182	01	0.024	0.051	0.000	0.225	0.162	0.253	0.000	0.825	0.162	0.253	0.000	0.825
74182	02	0.062	0.087	0.000	0.260	0.340	0.300	0.000	0.859	0.340	0.300	0.000	0.859
74182	03	0.014	0.022	0.000	0.081	0.128	0.192	0.000	0.480	0.128	0.192	0.000	0.480
74182	04	0.004	0.016	0.000	0.072	0.032	0.073	0.000	0.202	0.032	0.073	0.000	0.202
74L85	00	0.018	0.017	0.000	0.073	0.006	0.005	0.000	0.022	0.006	0.005	0.000	0.022
74L85	01	0.068	0.087	0.000	0.318	0.225	0.190	0.000	0.621	0.225	0.190	0.000	0.621
74L85	02	0.044	0.059	0.000	0.310	0.234	0.212	0.000	0.626	0.234	0.212	0.000	0.626
74181	00	0.021	0.024	0.000	0.114	0.130	0.132	0.000	0.414	0.130	0.132	0.000	0.414
74181	01	0.035	0.046	0.000	0.263	0.122	0.160	0.000	0.792	0.122	0.160	0.000	0.792
74181	02	0.044	0.040	0.000	0.229	0.152	0.146	0.000	0.652	0.152	0.146	0.000	0.652
74181	03	0.048	0.065	0.000	0.311	0.272	0.296	0.000	1.256	0.320	0.335	0.000	1.293
74181	04	0.046	0.068	0.000	0.385	0.250	0.246	0.000	1.154	0.290	0.279	0.000	1.231
74181	05	0.010	0.022	0.000	0.117	0.200	0.212	0.000	0.580	0.265	0.278	0.000	0.756
74181	06	0.003	0.016	0.000	0.129	0.132	0.198	0.000	0.612	0.183	0.262	0.000	0.756
74181	07	0.036	0.019	0.000	0.064	0.090	0.136	0.000	0.136	0.203	0.038	0.000	0.152
74283	00	0.047	0.072	0.000	0.354	0.170	0.188	0.000	0.642	0.170	0.188	0.000	0.642
74283	01	0.031	0.044	0.000	0.187	0.192	0.175	0.000	0.647	0.234	0.216	0.000	0.674
74283	02	0.013	0.027	0.000	0.147	0.157	0.153	0.000	0.412	0.205	0.207	0.000	0.517
74283	03	0.003	0.002	0.000	0.043	0.163	0.153	0.000	0.371	0.217	0.206	0.000	0.500
74283	04	0.001	0.006	0.000	0.006	0.148	0.165	0.000	0.338	0.200	0.225	0.000	0.505
c17	all	0.087	0.147	0.000	0.413	0.142	0.087	0.000	0.225	0.142	0.087	0.000	0.225
c499	all	0.006	0.006	0.000	0.024	0.020	0.048	0.000	0.342	0.016	0.060	0.000	0.510
c880	all	0.007	0.041	0.000	0.577	0.025	0.088	0.000	0.903	0.025	0.088	0.000	0.903

Table 4. EPP model accuracy (triple faults)

6 Conclusion

Motivated by the importance of FNR data in spectrum-based fault localization in this paper we have presented three EPP prediction models for logic circuits, viz. (1) a deterministic model that takes into account circuit topology, input values, and gate models, (2) a probabilistic model based on circuit topology and gate truth table pdf information, and, at the highest level of abstraction (3) a probabilistic model based on circuit topology only. Monte Carlo simulations show that the deterministic model performs best (average prediction error in the order of percents), while both probabilistic models perform somewhat worse (average error in the order of ten percent), while the difference between both models is small. As expected, the error increases with fault cardinality.

References

[1] R. Abreu, P. Zoeteweij, R. Golsteijn, and A.J.C. van Gemund, "A Practical Evaluation of Spectrum-based Fault Localization", *Journal of Systems and Software*, Vol. 82, No. 11, pp. 1780-1792, 2009.

[2] R. Abreu, P. Zoeteweij, A.J.C. van Gemund, "Spectrum-based Multiple Fault Localization". *Proc. 24th Int'l Conf. on Automated Software Engineering (ASE'09)*, Auckland, New Zealand, November 2009. pp. 88-99.

[3] R. Abreu, A.J.C. van Gemund, "Diagnosing Intermittent Faults Using Maximum Likelihood Estimation", *Artificial Intelligence*, Vol. 174, No. 18, pp. 1481-1497. 2010.

[4] H. Asadi, M.B. Tahoori, C. Tirumurti, "Estimating error Propagations Probabilities with Bounded Variance", *22nd IEEE Int'l Symp. on Defect and Fault Tolerance in VLSI Systems*, Sept. 2007, pp. 41-49.

[5] F. Brglez and H. Fujiwara, "A neutral netlist of 10 combinatorial benchmark circuits and a target translator in FORTRAN", *Int'l Symposium on Circuits and Systems, Special Session on ATPG and Fault Simulation*, pp. 695-698, 1985.

[6] A. Gonzalez-Sanchez, H-G. Gross, A.J.C. van Gemund, ”Empirical Study on the Usage of Testability Information to Fault Localization in Software”, *26th Symp. on Applied Computing (SAC’11)*, Taichung, March, 2011, to appear.

[7] M. Hansen, H. Yalcin, J. Hayes, ”Unveiling the ISCAS-85 benchmarks: A case study in reverse engineering”, *IEEE Design & Test*, 16(3):72-80, 1999.

[8] J.A. Jones, M.J. Harrold, J. Stasko, ”Visualization of test information to assist fault localization”, *Proc. of Int’l Conf. on Software Engineering (ICSE02)*, 2002.

[9] N. Mohyuddin, E. Pakbaznia, M. Pedram, ”Probabilistic error Propagation in Logic Circuits using the Boolean Difference Calculus”, *IEEE Int’l Conf. on Computer Design (ICCD’08)*, Oct. 2008, pp. 7-13.

[10] K.P. Parker, E.J. McCluskey, ”Probabilistic Treatment of General Combinational Networks”, *IEEE Trans. Computers*, June 1975, pp. 668-670.

[11] J.M. Voas, ”PIE: A Dynamic Failure-Based Technique”, *IEEE Trans. Softw. Eng’g*, Vol. 18, No. 8, Aug. 1992, pp. 717-727.