Journal of Electronic Testing (2022) 38:165–180
https://doi.org/10.1007/s10836-022-05996-y

An Accurate Estimation Algorithm for Failure Probability of Logic Circuits Using Correlation Separation

Shuo Cai[1] · Binyong He[1] · Sicheng Wu[1] · Jin Wang[1] · Weizheng Wang[1] · Fei Yu[1]

Received: 26 August 2021 / Accepted: 16 March 2022 / Published online: 15 April 2022
© The Author(s), under exclusive licence to Springer Science+Business Media, LLC, part of Springer Nature 2022

Abstract
As the feature size of integrated circuits decreases to the nanometer scale, process fluctuations, aging effects, and particle radiation have an increasing influence on the Failure Probability of Circuits (FPC), which brings severe challenges to chip reliability. The accurate and efficient estimation of logic circuit failure probability is a prerequisite for high-reliability design. It is difficult to calculate FPC due to a large number of reconvergent fanout structures and the resulting signal correlation, particularly for Very Large-Scale Integrated (VLSI) circuits. Accordingly, this paper presents a Correlation Separation Approach (COSEA) that aims to efficiently and accurately estimate the FPC. The proposed COSEA divides the circuit into several different fanout-relevant and fanout-irrelevant circuits. Moreover, the error probability of the nodes is expressed as the result of interactions between different structures. As a result, the problem of signal correlation can be efficiently solved. Because the computational complexity of COSEA is linearly related to the scale of the circuit, it has good scalability. Compared with the Probabilistic Transfer Matrices (PTM) method, Monte Carlo simulation (MC), and other failure probability calculation methods in the literatures, the experimental results show that our approach not only achieves fast speed and good scalability, but also maintains high accuracy.

Index Terms Logic circuit · Failure probability · Correlation separation · Reliability design

1 Introduction

Over the past few decades, the advancement of the CMOS manufacturing process has been the main reason behind the improvement of semiconductor device performance [12]. As the feature size of CMOS further decreases, the integration of devices has improved and the threshold voltage has reduced [22, 31], these developments have improved the performance of CMOS devices, but also introduced reliability issues [4, 5, 27, 38]. Process fluctuations, aging effects, and the impact of external radiation lead to more serious instability among circuit elements [6, 11, 37], which brings great challenges to reliable circuit design [10, 16, 17, 32].

The soft error caused by transient fault is the main cause of circuit failure, so researchers pay attention to the modeling of transient fault and the calculation of FPC [13, 30, 34, 36, 41]. The most common method is to simulate the operation of the circuit by injecting a large number of faults and then calculating the FPC by determining whether the fault propagates to the output of the circuit. Monte Carlo simulation (MC) is a typical failure probability estimation method, the accuracy of which is related to the number of simulation times [26]. The theory of stochastic computation [1] is applied in Stochastic Computational Models (SCM) [16] to deal with the signal correlation and probability calculation of logic circuits [35], while the non-Bernoulli sequence is utilized to replace the Bernoulli sequence in these models in order to obtain better astringency. Cai et al. study the

Responsible Editor: M. B. Tahoori

✉ Shuo Cai
caishuo@csust.edu.cn

Binyong He
big_hebe@163.com

Sicheng Wu
wusichengcsuts@qq.com

Jin Wang
jinwang@csust.edu.cn

Weizheng Wang
greaquer_w@163.com

Fei Yu
yufeiyfyf@csust.edu.cn

[1] School of Computer and Communication Engineering, Changsha University of Science and Technology, Changsha 410114, HN, China

reliability bounds of the circuits during single event transient and multi-transient faults [8, 9]. Xiao et al. research the influence of different input vectors on the circuit failure probability and propose a method for calculating the reliability of different input vector circuits [39]. Cai and Chen propose a method that can handle signal correlation, which can be used to calculate the failure probability of large-scale circuits, but the calculation error is relatively large [7].

Signal probability theory is used to analyze the calculation of FPC caused by gate faults [2, 3, 14, 25]. These analysis methods study the calculation of signal propagation rules in the circuit and analyze the probability of the fault pulses propagating through the critical paths. Compared with simulation methods, these methods have lower computational cost and linear computational complexity. However, in precise calculations, the signal correlation caused by the reconvergent fanout in the circuit must be considered, as it will greatly increase the method's time consumption. Accordingly, it is not suitable for calculating the failure probability of large-scale and very large-scale circuits.

PTM methods [23, 24, 31, 33] employ probability transfer matrices to accurately calculate the reliability of the circuits by establishing the relationship between the inputs and outputs. The disadvantage of the PTM methods is that the storage space required for matrix operations is too large. Researchers subsequently achieved the optimization of the PTM methods by using Algebraic Decision Diagrams (ADD), but the extent of this improvement is limited [10, 28, 29]. The Probabilistic Gate Model (PGM) [15] takes the reconvergent fanout structure as the basic unit of calculation. In this approach, the conditional probability is used to calculate the signal probability of different types of reconvergent fanout structure in order to eliminate the influence of signal correlation. Nevertheless, its exponential time complexity is difficult to solve. The Critical Score Algorithm (CSA) is proposed in [18, 19, 21]. The author considers that the FPC depends primarily on the input signals and defines those input signals that lead directly to the output errors as critical. However, since CSA does not consider the correlation of signals, the accuracy of this algorithm is insufficient. Critical Gates Count algorithms (CGC) [20] verify whether the fault of the gate in the circuit affects the circuit. Here, the gate that affects the output of the circuit is defined as a critical gate. The FPC can be further calculated by locating the critical gates of the circuit.

Fast and accurate FPC estimation can help IC designers to create effective fault-tolerant designs in order to improve the reliability of circuits [40]. The purpose of most FPC calculation methods is to provide an effective trade-off between accuracy and runtime. This paper proposes a novel approach for calculating the FPC accurately, and with reasonable time and space complexity, called the correlation separation approach (COSEA). We focus specifically on the signal correlation caused by the reconvergent fanout structures of circuits and propose an accurate and efficient FPC estimation approach based on correlation separation. COSEA divides the circuit into Independent Circuit Structures (ICS) depending on the different types of fanout nodes and output nodes, then calculates the failure probability of all ICS, and finally obtains the set of ICS that affect the primary outputs. In this approach, the effect of high-order infinitesimal values on the result is ignored, meaning that the computational efficiency is greatly improved. Compared with the CSA, COSEA fully considers the reconvergent fanout structures in the circuit. Accordingly, it achieves higher accuracy and more stable performance. Furthermore, this approach uses the failure probability of the independent structures to represent the error probability of the primary outputs, allowing the failure probability of the multiple-outputs circuit to be accurately calculated according to the correlation of the independent structures. Since the calculation process of the proposed approach involves sequentially calculating the relevant information of each node, its time and space complexity are only linearly related to the number of gates in the circuit.

The remainder of this paper is organized as follows. Section 2 reviews the relevant knowledge of critical signals. Section 3 introduces the related concepts along with the specific analytic process of our approach. In Sect. 4, a large number of experimental results are presented and compared in detail with the existing classical methods. Section 5 theoretically analyzes the error of the proposed approach. Finally, Sect. 6 concludes this paper.

2 The Related Critical Score Schemes

According to the CSA, the FPC depends not only on the circuit structure, but also the applied input vector. The CSA can calculate FPC approximately based on specific input vectors and circuit structure. For each logic gate, the Error Probability of the Node (EPN) of the output depends on the EPNs of the inputs, the Fault Probability of the Gate (FPG), and the gate type. For example, the calculation results of EPN obtained from a two-input AND gate corresponding to four different input vectors are as follows (under the premise that the input signals are mutually independent). In the below, EPN_{in1} and EPN_{in2} represent the error probabilities of these two inputs respectively, while the FPG is the fault probability of this gate.

$$\begin{aligned}\mathrm{EPN}_{out(00)} &= (1-\mathrm{FPG})\mathrm{EPN}_{in1}\mathrm{EPN}_{in2} + \mathrm{FPG}\left(1-\mathrm{EPN}_{in1}\right)\\ &\quad \left(1-\mathrm{EPN}_{in2}\right) + \mathrm{FPG}\left(1-\mathrm{EPN}_{in1}\right)\mathrm{EPN}_{in2} + \mathrm{EPN}_{in1}\\ &\quad \left(1-\mathrm{EPN}_{in2}\right)\mathrm{FPG} = \mathrm{FPG} + \mathrm{EPN}_{in1}\mathrm{EPN}_{in2} - 2\mathrm{EPN}_{in1}\mathrm{EPN}_{in2}\mathrm{FPG}\end{aligned} \tag{1}$$

$$\begin{aligned}\mathrm{EPN}_{out(01)} &= \mathrm{EPN}_{in1}\left(1-\mathrm{EPN}_{in2}\right)(1-\mathrm{FPG}) + \mathrm{FPG}\\ &\quad \left(1-\mathrm{EPN}_{in1}\right) + \mathrm{EPN}_{in1}\mathrm{EPN}_{in2}\mathrm{FPG} = \mathrm{FPG} + \mathrm{EPN}_{in1}\\ &\quad - 2\mathrm{EPN}_{in1}\mathrm{FPG} - \mathrm{EPN}_{in1}\mathrm{EPN}_{in2} + 2\mathrm{EPN}_{in1}\mathrm{EPN}_{in2}\mathrm{FPG}\end{aligned} \tag{2}$$

$$\begin{aligned}\mathrm{EPN}_{out(10)} &= (1-\mathrm{FPG})\left(1-\mathrm{EPN}_{in1}\right)\mathrm{EPN}_{in2}+\mathrm{FPG}\left(1-\mathrm{EPN}_{in2}\right)\\ &\quad+\mathrm{EPN}_{in1}\mathrm{EPN}_{in2}\mathrm{FPG}=\mathrm{FPG}+\mathrm{EPN}_{in2}-2\mathrm{EPN}_{in2}\mathrm{FPG}-\mathrm{EPN}_{in1}\\ &\quad\mathrm{EPN}_{in2}+2\mathrm{EPN}_{in1}\mathrm{EPN}_{in2}\mathrm{FPG}\end{aligned} \tag{3}$$

$$\begin{aligned}\mathrm{EPN}_{out(11)} &= (1-\mathrm{FPG})\mathrm{EPN}_{in1}\left(1-\mathrm{EPN}_{in2}\right)+\mathrm{EPN}_{in2}\\ &\quad(1-\mathrm{FPG})\left(1-\mathrm{EPN}_{in1}\right)+(1-\mathrm{FPG})\mathrm{EPN}_{in1}\mathrm{EPN}_{in2}+\mathrm{FPG}\\ &\quad\left(1-\mathrm{EPN}_{in1}\right)\left(1-\mathrm{EPN}_{in2}\right)=\mathrm{FPG}+\mathrm{EPN}_{in1}+\mathrm{EPN}_{in2}\\ &\quad-\mathrm{EPN}_{in1}\mathrm{EPN}_{in2}-2\mathrm{EPN}_{in1}\mathrm{FPG}-2\mathrm{EPN}_{in2}\mathrm{FPG}\\ &\quad+2\mathrm{EPN}_{in1}\mathrm{EPN}_{in2}\mathrm{FPG}\end{aligned} \tag{4}$$

Here, $\mathrm{EPN}_{out(00)}$, $\mathrm{EPN}_{out(01)}$, $\mathrm{EPN}_{out(10)}$, and $\mathrm{EPN}_{out(11)}$ represent the error probability of the output node when the input vectors are "00", "01", "10", and "11" respectively.

Considering the process parameters and manufacturing level of CMOS IC at this stage, the product terms in the above equations can be regarded as high-order infinitesimal values, for the failure rate of nodes, it can be removed. Accordingly, the above equations can be simplified as follows:

$$\mathrm{EPN}_{out(00)}=\mathrm{FPG}. \tag{5}$$

$$\mathrm{EPN}_{out(01)}=\mathrm{FPG}+\mathrm{EPN}_{in1}. \tag{6}$$

$$\mathrm{EPN}_{out(10)}=\mathrm{FPG}+\mathrm{EPN}_{in2}. \tag{7}$$

$$\mathrm{EPN}_{out(11)}=\mathrm{FPG}+\mathrm{EPN}_{in1}+\mathrm{EPN}_{in2}. \tag{8}$$

In the simplified Eqs. (5) to (8), EPN_{out} represents the sum of input error probabilities that are correlated with the output. For example, $\mathrm{EPN}_{out(01)}$ can be regarded as the sum of the errors caused by the first input '0' (EPN_{in1}) and the gate (FPG) itself. Here, the input '0' is defined as a critical signal, which indicates that this signal plays a critical role in the occurrence of output error.

The CSA only needs to calculate the sum of the error probabilities of the critical inputs and the fault probability of the gate, an approach that simplifies the original polynomial multiplication operation into an addition operation with fewer terms. This greatly reduces the calculation cost.

Table 1 lists the EPN_{out} of other two-input logic gates. The calculation results of NAND and AND are the same, as are those of NOR and OR. Moreover, both inputs of the two-input XOR are critical signals. The EPN_{out} of logic gates with more inputs can also be calculated in this way.

According to the CSA, the factors affecting the node's error probability include three aspects: the error probability introduced by the upstream inputs, the fault probability of the logic gate itself, and the input vector applied to the logic gate.

The logic gate has a masking effect on non-critical input signals. The probability of error occurring within the critical input signals is propagated forward through the logic gate, continuing to affect the gates of the next level or propagating to the primary outputs.

Assuming that the primary inputs of the circuit carry no error probability, it follows that the FPC is entirely caused by the faults of the logic gates. We refer to the gates that directly affect the primary outputs of the circuit as Critical Gates (CGs). That is to say, the faults of CGs will not be masked by other logic gates in the circuit.

The CSA analyzes all CGs capable of propagating errors to the primary outputs and calculates the FPC according to their influence on the outputs. The steps in this calculation can be expressed as follows:

a) Search iteratively for the critical input signals and the corresponding CGs from the primary outputs;
b) Stop the iteration when there are no critical input signals or iterate to the primary inputs.

When CSA is used to calculate the FPC, the reconvergent fanout in the circuit is not taken into account, meaning that the relevant signals cannot be processed accurately.

For example, in the independent fanout structure shown in Fig. 1, the fanout node S fans out two identical signals (F1 and F2) to the AND gate G. Their failure probabilities are equal to EPN, while the fault probability of the AND gate G is FPG. Assuming that the logical value of S is 1, according to the CSA, F1 and F2 are both critical signals to G. Thus, the output failure probability of the AND gate is 2EPN + FPG. In fact, the fanout signals F1 and F2 are entirely related, while the error situations of F1 and F2 are also the same. Thus, if F1 fails, F2 must also fail, and vice versa. Therefore, the actual output failure probability of G is the sum of the error probability of S and the fault probability of G, i.e., EPN + FPG. This is the limitation of the

Table 1 $\mathrm{EPN}_{\mathrm{OUT}}$ of two-input gates using CSA

input vector	NAND	OR	NOR	XOR
00	FPG	$\mathrm{EPN}_{in1}+\mathrm{EPN}_{in2}+\mathrm{FPG}$	$\mathrm{EPN}_{in1}+\mathrm{EPN}_{in2}+\mathrm{FPG}$	$\mathrm{EPN}_{in1}+\mathrm{EPN}_{in2}+\mathrm{FPG}$
01	$\mathrm{EPN}_{in1}+\mathrm{FPG}$	$\mathrm{EPN}_{in2}+\mathrm{FPG}$	$\mathrm{EPN}_{\mathrm{in2}}+\mathrm{FPG}$	$\mathrm{EPN}_{in1}+\mathrm{EPN}_{in2}+\mathrm{FPG}$
10	$\mathrm{EPN}_{in2}+\mathrm{FPG}$	$\mathrm{EPN}_{in1}+\mathrm{FPG}$	$\mathrm{EPN}_{in1}+\mathrm{FPG}$	$\mathrm{EPN}_{in1}+\mathrm{EPN}_{in2}+\mathrm{FPG}$
11	$\mathrm{EPN}_{in1}+\mathrm{EPN}_{in2}+\mathrm{FPG}$	FPG	FPG	$\mathrm{EPN}_{in1}+\mathrm{EPN}_{in2}+\mathrm{FPG}$

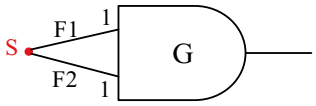


Fig. 1 Error of CSA method

CSA: as it assumes that the inputs of all gates are mutually independent, it is precisely the correlation in the actual circuits that results in the high complexity of calculating large-scale circuits.

3 Our Proposed Correlation Separation Approach

There are three steps involved in implementing COSEA. First, the circuit is divided into several independent structures according to the types of fanout nodes and output nodes and the failure probability of each structure is calculated. Second, the propagation of EPN is analyzed to obtain the set of ICS affecting the primary outputs. Finally, the FPC is calculated according to the ICS.

3.1 Separation of the Circuits

3.1.1 Classification of ICS

A whole circuit is divided into several ICS with fanout nodes as the interval points, and an ICS contains at least one logic gate. We classify the ICS as either Fanout-Relevant Circuits (C_{FR}) or Fanout-Irrelevant Circuits (C_{FI}) depending on the type of ICS output node. The output of C_{FR} is the fan out node. The failure probability of C_{FR} (PFR) is extended to other signals through its output. The output of C_{FI} is a non fan out node. The failure probability of C_{FI} (PFI) does not spread to other nodes, but only to its output node.

Taking Fig. 2 as an example, we divide the circuit into different C_{FR} and C_{FI}. As illustrated in Fig. 2, the circuit contains two fanout nodes and three primary outputs. For the independent structure between in1 and *S*1, its output *S*1 is the fanout node and its input *in*1 is the primary input. Thus, this structure is a C_{FR}, named $C_{FR(S1)}$. Similarly, the structure between *S*1 and *S*2 is also a C_{FR}, labeled $C_{FR(S2)}$. In addition, the structure between *out*1 and *S*2 is a C_{FI}, called $C_{FI(out1)}$. Similar structures are $C_{FI(out2)}$ and $C_{FI(out3)}$.

The properties of C_{FR} and C_{FI} are as follows:

a) The number of C_{FR} is equal to the number of reconvergent fanout structures in the circuit.
b) The number of C_{FI} is equal to the number of primary outputs.
c) The circuit is divided by fanout node, so there are no fanouts node in the ICS (C_{FR} and C_{FI});

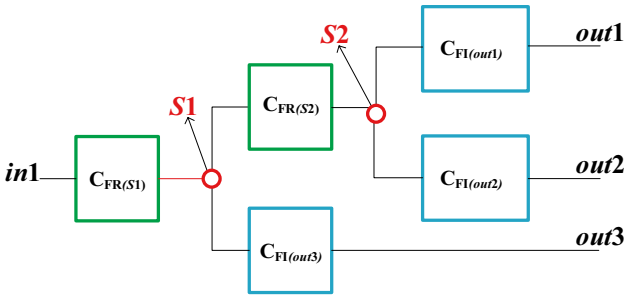


Fig. 2 Classification of C_{FR} and C_{FI}

d) The failure probabilities of different ICS are independent. That is to say, other ICS cannot affect the calculation of ICS (PFR or PFI) failure probability, it is only related to its internal topology and input vector.

3.1.2 Node Information

For a node *i* in the circuit, the EPN_i can be regarded as the accumulation of the error effects in its upstream circuit. Including n C_{FR} and one C_{FI}. C_{FI} is the adjacent ICS between node *i*, and they are all located upstream of node *i*.

In Fig. 3, $C_{FR(S)}$ and $C_{FI(i)}$ are indicated by green and blue dotted-line boxes, respectively. The target node *i* may be affected by multiple C_{FR}, but only one C_{FI}. We define $U_i = \{C_{FR1}, C_{FR2} \ldots, C_{FRn}\}$ as the set of C_{FR} that affects node *i*.

When CSA is used to calculate the failure probability of the circuit, there is an error because the correlation of the circuit signal cannot be separated. In our method, we divide the circuit into several independent ICS. Therefore, the EPN can be expressed as *function*(PFI, PFR_1, PFR_2, …, PFR_n). Actually, the calculation equation of EPN_i is as follows:

$$EPN_i = PFI_i + \sum_{j=1}^{n} PFR_j. \tag{9}$$

Here, *n* is the number of C_{FR} affecting node *i*.

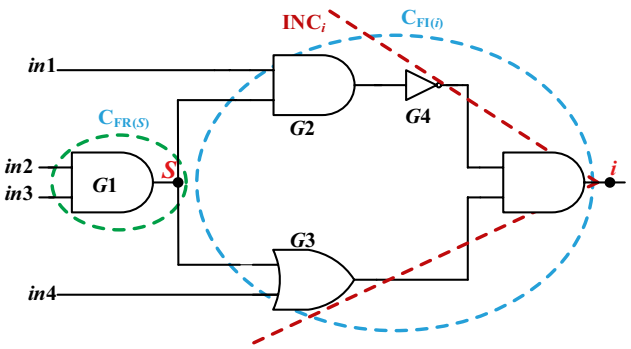


Fig. 3 Classification of C_{FR} and C_{FI}

We define $T_i = \{LV_i, PFI_i, U_i\}$ to represent the information of node i, including its logical value (LV_i), the failure probability of $C_{FI(i)}$, and the C_{FR} set that affects node i.

3.2 Calculation of PFI and PFR

3.2.1 Calculation of PFI

Since there is no fanout node in C_{FI}, we can use the CSA to calculate PFI. According to the calculation method of CSA, the EPN_i equals the sum of the error probability of the critical input signals and the FPG itself. Thus, the calculation formula of PFI_i is as follows:

$$PFI_i = + \sum_{j=1}^{n} PFI_j + FPG. \tag{10}$$

Here, n is the number of critical input signals of node i.

3.2.2 Calculation of PFR

In our approach, C_{FR} can be converted from C_{FI}. Figure 4 illustrates the conversion process from C_{FI} to C_{FR}.

In Fig. 4, a and b represent the logic gates in the circuit. When calculating the error probability of node a, $C_{FI(a)}$ is the C_{FI} of node a.

When b is calculated in turn, the range of $C_{FI(b)}$ includes $C_{FI(a)}$ and node b. Since b is a fan out node, the failure probability of $C_{FI(b)}$ will propagate along the fan out branch at point b. At this time, $C_{FI(b)}$ will be transformed into $C_{FR(b)}$, as shown in Fig. 4c. When C_{FI} is converted to C_{FR}, the corresponding node information also changes. For the target node i, its information is described as follows:

$$\begin{cases} PFR_i = PFI_i, \\ PFI_i = 0, \\ U_i = U_i \bigcup C_{FR(i)}. \end{cases} \tag{11}$$

Here, the sing "=" is the assignment operator.

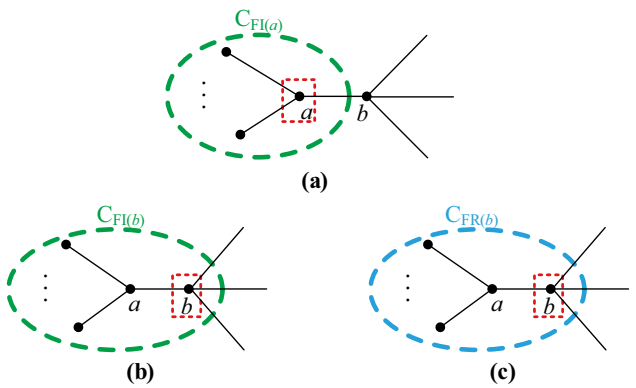


Fig. 4 Conversion process from C_{FI} to C_{FR}

3.2.3 Caculation of U

As discussed above, the node information T contains three elements: LV, PFI, and U. The LV is determined by the input logical values and the gate type. The method used to calculate PFI has been described above. Next, we will introduce the method use to calculate U in detail.

U is composed of multiple C_{FR}. Each C_{FR} can propagate from the input to the output of the gate, on the condition that the error of this C_{FR} will result in an error at the output of the gate. Among all input nodes, the C_{FR} that satisfies this condition is selected as the U of the output node.

Figure 5 presents an n-input AND gate, where $n0$ represents the number of input logical values of 0, $n1$ denotes the number of input values of 1, and $n = n0 + n1$. The node information with input values of 0 are $\{T_{Q1}, T_{Q2}, \ldots, T_{Qn0}\}$, while the node information with values of 1 are $\{T_{P1}, T_{P2}, \ldots, T_{Pn1}\}$, where $T_{Qi} = \{0, PFI_{Qi}, U_{Qi}\}$, $T_{Pi} = \{1, PFI_{Pi}, U_{Pi}\}$, $U_{Qi} = \{C_{FR(Qi(1))}, C_{FR(Qi(2))}, \ldots, C_{FR(Qi(kQi))}\}$, $U_{Pi} = \{C_{FR(Pi(1))}, C_{FR(Pi(2))}, \ldots, C_{FR(Pi(kPi))}\}$. Without loss of generality, we discuss the calculation methods of U_{out} and T_{out} ($T_{out} = \{LV_{out}, PFI_{out}, U_{out}\}$ in the case of three types of input vectors (outlined below).

a) As shown in Fig. 6a, $n1 = n$, $n0 = 0$, while the node information with values of 1 are $\{T_{P1}, T_{P2}, \ldots, T_{Pn1}\}$. 1) The logical value of output $LV_{out} = 1$; 2) Since all input signals are critical signals, $PFI_{out} = PFI_{P1} + \ldots + PFI_{Pn1} + FPG$; 3) All '1' signals are critical signals, meaning that the error of any C_{FR} will cause an error at the output of this gate. U_{out} is equal to the union of C_{FR} in the '1' signals, i.e., $U_{out} = U_{P1} \cup U_{P2} \cup \ldots \cup U_{Pn1}$. In this case, $T_{out} = \{1, PFI_{P1} + \ldots + PFI_{Pn1} + FPG, U_{P1} \cup U_{P2} \cup \ldots \cup U_{Pn1}\}$.
b) As shown in Fig. 6b, $n1 = n-1$, $n0 = 1$, the node information with values of 1 are $\{T_{P1}, T_{P2}, \ldots, T_{Pn1}\}$, while the node information with a value of 0 is T_{Q1}. 1) The logical value of output $LV_{out} = 0$; 2) Since the node '0' is a critical signal, $PFI_{out} = PFI_{Q1} + FPG$; 3) A C_{FR} error at the

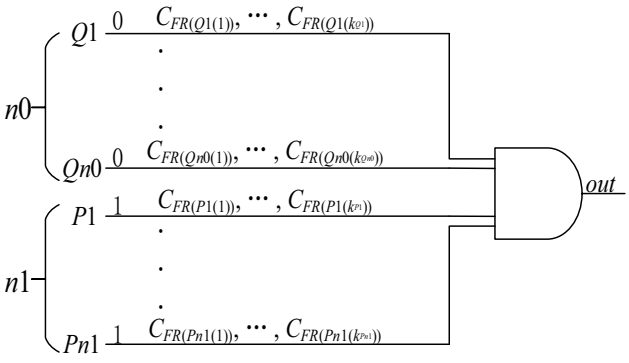
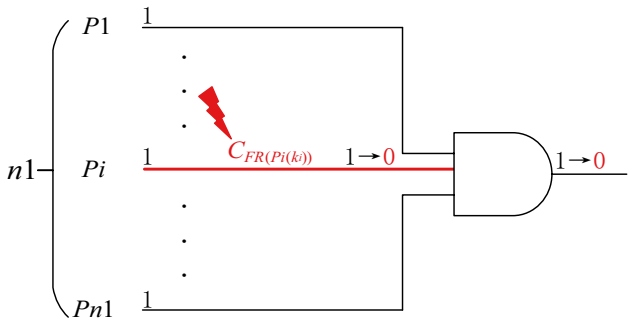
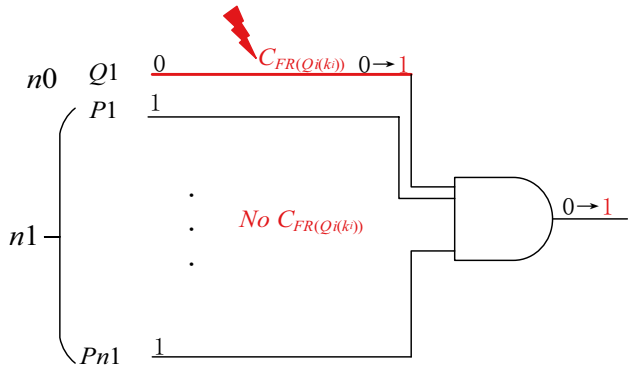


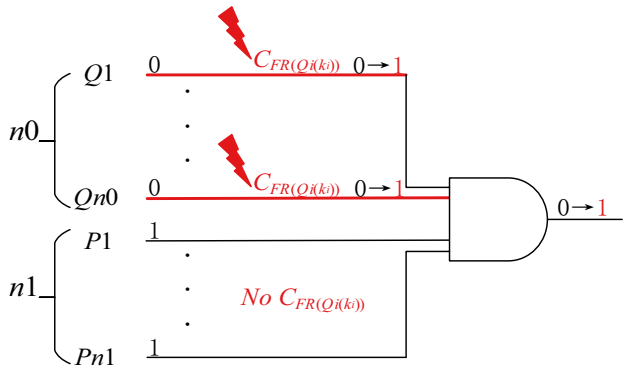
Fig. 5 Input information of n-input AND



(a) The first case: all inputs are 1



(b) The second case: one input is 0, the other inputs are 1



(c) The third case: two or more inputs are 0

Fig. 6 Calculation method of U for different inputs

'0' signal will cause an error at the output. However, if the C_{FR} also exists in other '1' signals, the C_{FR} error will change the '1' signal to '0', meaning that the output will still be '0'. Therefore, U_{out} is equal to the C_{FR} in the '0' signal minus the union of the C_{FR} in the '1' signals, i.e., $U_{out}=U_{Q1}–U_{P1}\cup\ldots\cup U_{Pn1}$. The output node information $T_{out}=\{0,PFI_{Q1}+FPG,U_{Q1}–U_{P1}\cup U_{P2}\cup\ldots\cup U_{Pn1}\}$.

c) As shown in Fig. 6c, n0 ≥ 2, $n1=n–n0$, the node information with values of 1 are $\{T_{P1}, T_{P2}, \ldots, T_{Pn1}\}$, while the node information with values of 0 are $\{T_{Q1}, T_{Q2}, \ldots, T_{Qn0}\}$. 1) The logical value of output $LV_{out}=0$; 2) As there is no critical signal at the input, $PFI_{out}=FPG$; 3) Similar to case (b), only C_{FR} errors that exist on all input 0 signals and do not exist on any input 1 signal can lead to output errors. Therefore, U_{out} equals the intersection of the C_{FR} in the '0' signals minus the union of the C_{FR} in the '1' signals, i.e., $U_{out}=U_{Q1}\cap U_{Q2}\cap\ldots\cap U_{Qn0}–U_{P1}\cup U_{P2}\cup\ldots\cup U_{Pn1}$. The output node information $T_{out}=\{0, FPG, U_{Q1}\cap U_{Q2}\cap\ldots\cap U_{Qn0}–U_{P1}\cup U_{P2}\cup\ldots\cup U_{Pn1}\}$.

The above method can be used to analyze not only the AND gate, but also other logic gates. Tables 2, 3 and 4 list the calculation formulas of T_{out} for the AND, OR and NOT gate respectively about von-Neumann error. Other types of faults can also be inferred using this method to get the calculation formula.

3.3 FPC Calculation for Multiple-output Circuits

The FPC of a single-output circuit is equal to the EPN of the primary output. However, for most logic circuits, there are multiple output nodes and they are also related. Thus, the FPC of the multiple-output circuit cannot be obtained by simply calculating the error probability of each primary output.

In our COSEA, the failure probability of a multiple-output circuit is equal to the comprehensive effects of PFR and PFI on all primary outputs. Since PFR and PFI are independent of each other, we can provide an accurate FPC calculation method for a multiple-output circuit.

The object of our study is a logic circuit with *m* outputs. We define U_{FR} as the set of C_{FR}, as follows:

$$U_{FR}=U_1\cup U_2\cup\cdots U_m. \tag{12}$$

The failure probability corresponding to U_{FR} is U_{PFR}, $U_{PFR}=\{PFR_1, PFR_2, \ldots, PFR_n\}$, while *n* is the number of C_{FR} in U_{FR}.

Define U_{PFI} as the set of PFIs at the outputs, i.e. $U_{PFI}=\{PFI_1, PFI_2, \ldots, PFI_m\}$. Since C_{FR} and C_{FI} are independent of each other, PFR and PFI are also independent.

Table 2 Calculation formulas of T_{out} for AND gate about Stuck-at-0

input	LV_{out}	PFI_{out}	U_{out}
$n1=n,n0=0$	1	$PFI_{P1}+PFI_{P2}+\ldots+PFI_{Pn1}+FPG$	$U_{P1}\cup U_{P2}\cup\ldots\cup U_{Pn1}$
$n1=n\text{-}1,n0=1$	0	$PFI_{P1}+FPG$	$U_{Q1}—U_{P1}\cup U_{P2}\cup\ldots\cup U_{Pn1}$
$n_0\geq 2$	0	FPG	$U_{Q1}\cap\ldots\cap U_{Qn0}—U_{P1}\cup\ldots\cup U_{Pn1}$

Table 3 Calculation formulas of T_{out} for OR gate Stuck-at-0

input	LV_{out}	PFI_{out}	U_{out}
$n1=0,n0=n$	0	$PFI_{Q1}+PFI_{Q2}+\ldots+PFI_{Qn0}+FPG$	$U_{Q1}\cup U_{Q2}\cup\ldots\cup U_{Qn0}$
$n1=1,n0=n\text{-}1$	1	$PFI_{P1}+FPG$	$U_{P1}—U_{Q1}\cup U_{Q2}\cup\ldots\cup U_{Qn0}$
$n_1\geq 2$	1	FPG	$U_{P1}\cap\ldots\cap U_{Pn1}—U_{Q1}\cup\ldots\cup U_{Qn0}$

The failure of any ICS will result in circuit failure, the FPC can be calculated as follows:

$$\mathrm{FPC}=1-\prod_{i=1}^{n}\left(1-\mathrm{PFR}_i\right)\times\prod_{j=1}^{m}\left(1-\mathrm{PFI}_j\right). \tag{13}$$

The algorithm about the calculation of T and COSEA are presented in Figs. 7 and 8, respectively.

3.4 An Example

Figure 9a, b and c show three types of convergence fanout circuits with independent structure, parallel structure and nested structure, respectively. In this section, we will describe the whole process of using COSEA in detail.

In the Fig. 9, *in* and *out* respectively represent the primary input and output, and S represents the fanout node. The parentheses in the figure contain node information T. The three elements of T are logical values, PFI and U. For example, the node information of *in*1 is $T_{in1}=(0, 0, \emptyset)$, and the output information of G1 is $T_{G1}=(1, \mathrm{FPG}, \emptyset)$.

COSEA is used to calculate the circuit including the independent convergence fanout structure (Fig. 9a). Assuming the input signal is fault- free, the three node information are $T_{in1}=T_{in3}=\{1, 0, \emptyset\}$, $T_{in2}=\{0, 0, \emptyset\}$ respectively. Starting from the input, calculate the propagation of the signal in the circuit. The *in*2 signal passes through the invert gate G1, the process of calculating the signal information of G1 according to the calculation formula of the Table 4 is as follows.

$$\begin{cases} T_{G1}(\mathrm{LV})=1, \\ T_{G1}(\mathrm{PFI})=T_{in2}(\mathrm{PFI})+\mathrm{FPG}=\mathrm{FPG}, \\ T_{G1}(\mathrm{U})=T_{in2}(\mathrm{U})=\emptyset. \end{cases} \tag{14}$$

The output signal of G1 passes through node S, S is the fanout node, which converts the G1 from C_{FI} to $C_{FR(S)}$, the failure probability of $C_{FR(S)}$: PFR(S) = FPG. Node S fanout two related signals, they convergence the *in*1 signal and *in*3 at G2 and G3 respectively, the output signals of G2 and G3 converge in G4. According to the first case of the Table 2, the calculation processes of G2, G3 and G4 are as follows.

Table 4 Calculation formulas of T_{out} for inverter Stuck-at-0

input	LV_{out}	PFI_{out}	U_{out}
0/1	1/0	$PFI_{in}+FPG$	U_{in}

$$\begin{cases} T_{G2}(\mathrm{LV})=1, \\ T_{G2}(\mathrm{PFI})=T_{in1}(\mathrm{PFI})+T_S(\mathrm{PFI})=0, \\ T_{G2}(\mathrm{U})=T_{in1}(\mathrm{U})\bigcup T_S(\mathrm{U})=C_{FR(S)}. \end{cases} \tag{15}$$

$$\begin{cases} T_{G3}(\mathrm{LV})=1, \\ T_{G3}(\mathrm{PFI})=T_{in3}(\mathrm{PFI})+T_S(\mathrm{PFI})=0, \\ T_{G3}(\mathrm{U})=T_{in3}(\mathrm{U})\bigcup T_S(\mathrm{U})=C_{FR(S)}. \end{cases} \tag{16}$$

$$\begin{cases} T_{G4}(\mathrm{LV})=1, \\ T_{G4}(\mathrm{PFI})=T_{G2}(\mathrm{PFI})+T_{G3}(\mathrm{PFI})+\mathrm{FPG}=3\mathrm{FPG}, \\ T_{G4}(\mathrm{U})=T_{G2}(\mathrm{U})\bigcup T_{G3}(\mathrm{U})=C_{FR(S)}. \end{cases} \tag{17}$$

So the FPC of Fig. 9a is as follows.

$$\mathrm{FPC(a)}=T_{G4}(\mathrm{PFI})+C_{FR(S)}=3\mathrm{FPG}+\mathrm{FPG}=4\mathrm{FPG}. \tag{18}$$

Analyze the convergence fanout structure of parallel type in Fig. 9b: the node information of the input signal $T_{in1}=Ti_{n2}=\{0, 0, \emptyset\}$. The input signal *in*1 propagates to the inverter G1, according to Table 4, the node information of G1: $T_{G1}=\{1, \mathrm{FPG}, \emptyset\}$. Similarly, tthe node information of G2: $T_{G2}=\{1, \mathrm{FPG}, \emptyset\}$. The output signal of G1 is propagated to the fanout node S1, and G1 is converted into $C_{FR(S1)}$, where the failure probability of $C_{FR(S1)}$: PFR(S1) = FPG, the converted node information $T_{S1}=\{1,$

1. ***Calculate_T***(*node*,*T*, *ICS*, *ICS.fp*)
2. Calculate *T*(node).*LV* according to the input logic, gate type
3. Calculate *T*(node).*PFI* according to critical score algorithm(CSA)
4. if node's type is the fanout
5. Add the node to the *ICS*
6. Add *T*(*node*).*PFI* to the *ICS.fp* as *ICS*(*node*)'s failure probability
7. *T*(*node*).*PFI* = 0
8. *T*(*node*).*U* =*T*(*node*).*U*∪*node*
9. else
10. Calculate *T*(*node*).*U* according to the input vector, gate type to select formula
11. end if
12. Return *T*(*node*)
13. **End *calculate_T***

Fig. 7 Algorithm for calculating T

// Calculate the FPC
1. set *NF* = node's input number
2. set *Number* = number of nodes
3. set *M* = number of primary inputs
4. set *N* = number of primary outputs
5. Create *ICS* as the *ICS* list
6. Create ICS_{fp} as the failure probability list of *ICS*
7. set *FPG* is the failure probability of gate
8. set $T = \{LV, PFI, U\}$ is the node information
9. ***COSEA***(*circuit netlist, input*)
10. read *circuit netlist* to get circuit information
11. $j = 1$
12. for *node* = 1 : *Number* // initialize *T* of each node
13. if node type is the primary input
14. *T*(*node*) = {*input*(*j*),0,0}//initial input node
15. $j = j + 1$
16. else
17. *T*(*node*) = {-1,0,0} // Initialize other nodes
18. end if
19. end for
20. *count* = 0
21. while (*count* < *Number*) // Calculate T for each node
22. for *node* = 1 : *Number*// Determine if the node's input has been calculated
23. for *node.in* = 1 : *NF*(*node*)
24. *flag* = 0
25. if *T*(*NF*(*node.in*)).*LV* == -1
26. *flag* = 1
27. end if
28. end for
29. if *flag* == 0 // The input of the node has been calculated
30. if node type is the fanout
31. Add the *node* number to the *ICS*
32. Add *T*(*node*).*PFI* to the *ICS.fp as ICS*(*node*)'s failure probability
33. *T*(*node*).*PFI* = 0
34. *T*(*node*).*U* = *T*(*node*).*U*∪*node*
35. else
36. Calculate *T*(*node*).*U* and *T*(*node*).*PFI* according to the input logic, gate type, *T*
37. end if
38. Calculate *T*(*node*).*LV* according to the input logic, gate type
39. *count* = *count* + 1 // After calculated a node
40. else
41. Continue
42. end for
43. end while
44. FPC = CALL *calculate_FPC*(*T*, *ICS*, *ICS.fp*)
45. return FPC
46. **END *COSEA***

Fig. 8 Algorithm of COSEA

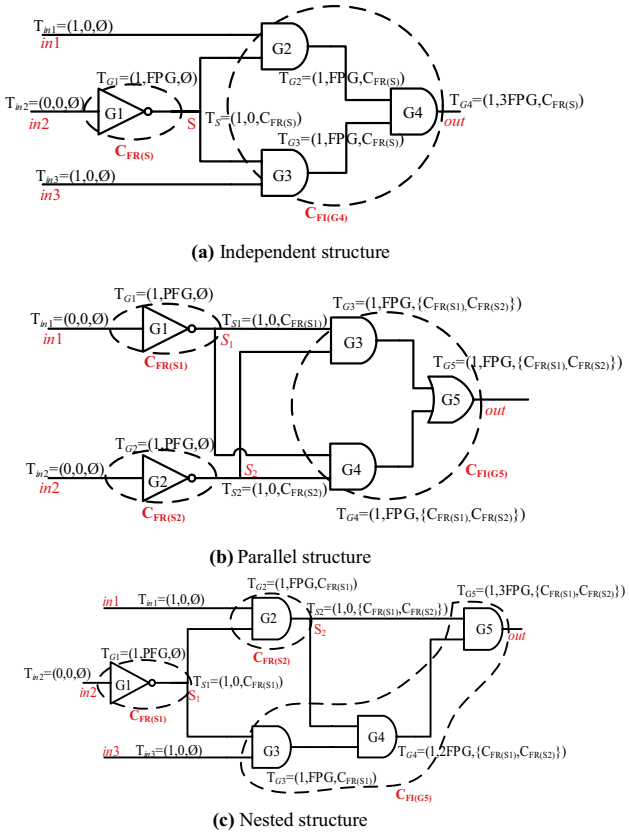


Fig. 9 Three simple circuits containing different types of convergence fanout structures (**a**) Independent structure (**b**) Parallel structure (**c**) Nested structure

0, $C_{FR(S1)}$}. Similarly, S2 also converts G2 into $C_{FR(S2)}$, PFR(S2) = FPG, $T_{S2} = (1, 0, C_{FR(S2)})$.

The fanout signals of S1 and S2 converge in AND gates G3 and G4 respectively, and their convergence is consistent with the first case of Table 2. Apply this formula to calculate the circuit information T_{G3} and T_{G4} of G3 and G4.

$$\begin{cases} T_{G3}(LV) = 1, \\ T_{G3}(PFI) = T_{S1}(PFI) + T_{S2}(PFI) + FPG = FPG, \\ T_{G3}(U) = T_{S1}(U) \bigcup T_{S2}(U) = \{C_{FR(S1)}, C_{FR(S2)}\}. \end{cases} \quad (19)$$

$$\begin{cases} T_{G4}(LV) = 1, \\ T_{G4}(PFI) = T_{S1}(PFI) + T_{S2}(PFI) + FPG = FPG, \\ T_{G4}(U) = T_{S1}(U) \bigcup T_{S2}(U) = \{C_{FR(S1)}, C_{FR(S2)}\}. \end{cases} \quad (20)$$

The output signals of G3 and G4 converge in G5, which conforms to the third rule of Table 3 or gate, so the calculation process of T_{G5} is as follows.

$$\begin{cases} T_{G5}(LV) = 1, \\ T_{G5}(PFI) = FPG, \\ T_{G5}(U) = T_{G3}(U) \bigcap T_{G4}(U) = \{C_{FR(S1)}, C_{FR(S2)}\}. \end{cases} \quad (21)$$

So the FPC of Fig. 9b is as follows.

$$\begin{aligned} FPC(b) &= T_{G5}(PFI) + PFR_{(S1)} + PFR_{(S2)} \\ &= FPG + FPG + FPG = 3FPG. \end{aligned} \quad (22)$$

Analyze the nested structure of the Fig. 9c: the node information $T_{in1} = T_{in3} = (1, 0, \emptyset)$, $T_{in2} = \{0, 0, \emptyset\}$. The input signal *in*2 propagates to G1, so the node information of G1 is $T_{G1} = \{1, FPG, \emptyset\}$. The output of G1 is the fanout node S1. S1 converts the C_{FI} represented by G1 into $C_{FR(S1)}$. After

the transformation, the node information of S1 is $T_{S1}=\{1, 0, C_{FR(S1)}\}$, and its failure probability is PFR(S1) = FPG. The fanout signal of S1 converges with *in*1 and *in*3 in G1 and G3, which conforms to the first case of Table 2, and their node information is shown in Fig. 9c. The C_{FI} circuit represented by G2 is converted to $C_{FR(S2)}$ by S2, and the node information of S2 after conversion is $T_{S2}=\{1, 0, C_{FR(S2)}\}$, where the failure probability of $C_{FR(S2)}$: PFR(S2) = PFG. The fanout of G3 and S2 converge on G4, the output of G4 and the other fanout signal of S2 converge on G5, and their input logic are both 1, which is in line with the first case of Table 2.

The FPC of Fig. 9c is as follows.

$$\begin{aligned} \mathrm{FPC(c)} &= T_{G5}(\mathrm{PFI}) + \mathrm{PFR}_{(S1)} + \mathrm{PFR}_{(S2)} \\ &= 3\mathrm{FPG} + \mathrm{FPG} + \mathrm{FPG} = 5\mathrm{FPG}. \end{aligned} \tag{23}$$

4 Performance Analysis

In order to evaluate the accuracy and effectiveness of the proposed COSEA, we compare our approach with PTM, MC, CSA, and CGC in this section. The experimental circuits include the benchmarks of ISCAS'85, ISCAS'89, along with some other small-scale circuits. The experiments were conducted on a computer equipped with a 3.0 GHz Pentium microprocessor and 8 GB of memory. The MC simulation is based on the C++ program running on the Visual Studio 2010 platform, while the COSEA, CSA, CGC, and PTM methods are all based on the MATLAB 2014a platform.

4.1 Experimental Setup

COSEA calculates FPC with the determined input vector. FPC_i represents the FPC corresponding to the *i*-th input vector. When the number of primary inputs for the circuit is small, the FPC corresponding to all input vectors can be calculated. However, as the number of inputs increases, the number of vectors also increases exponentially, making it difficult to estimate the corresponding FPC of all input vectors. We accordingly stipulate that for circuits with an input number less than or equal to 6, all input vectors are considered. By contrast, for circuits with more than 6 inputs, we consider 100 randomly selected vectors due to the time constraints associated with the MC simulation.

We compare COSEA with the PTM method, MC simulation, CSA, and CGC method. As the PTM method and MC simulation are accurate calculation methods when calculating circuits at different scales, we use them as references to compare the errors of COSEA in order to prove the accuracy of the proposed approach. The calculation formulas of Error(PTM) and Error(MC) are expressed in Eqs. (24) and (25). The CSA and CGC methods are estimation methods for calculating the FPC. The comparison between these two methods and COSEA includes a comparison of performance in terms of accuracy and speed; Error(CSA) and Error(CGC) are shown as Eqs. (26) and (27).

$$\mathrm{Error(PTM)} = \frac{1}{n} \times \sum_{i=1}^{n} \left| \frac{\mathrm{FPC}_{\mathrm{PTM}(i)} - \mathrm{FPC}_{\mathrm{COSEA}(i)}}{\mathrm{FPC}_{\mathrm{PTM}(i)}} \right|. \tag{24}$$

$$\mathrm{Error(MC)} = \frac{1}{n} \times \sum_{i=1}^{n} \left| \frac{\mathrm{FPC}_{\mathrm{MC}(i)} - \mathrm{FPC}_{\mathrm{COSEA}(i)}}{\mathrm{FPC}_{\mathrm{MC}(i)}} \right|. \tag{25}$$

$$\mathrm{Error(CSA)} = \frac{1}{n} \times \sum_{i=1}^{n} \left| \frac{\mathrm{FPC}_{\mathrm{COSEA}(i)} - \mathrm{FPC}_{\mathrm{CSA}(i)}}{\mathrm{FPC}_{\mathrm{COSEA}(i)}} \right|. \tag{26}$$

$$\mathrm{Error(CGC)} = \frac{1}{n} \times \sum_{i=1}^{n} \left| \frac{\mathrm{FPC}_{\mathrm{COSEA}(i)} - \mathrm{FPC}_{\mathrm{CGC}(i)}}{\mathrm{FPC}_{\mathrm{COSEA}(i)}} \right|. \tag{27}$$

Here, *n* is the number of vectors to be calculated for each circuit, $FPC_{PTM(i)}$ is the circuit failure probability of the *i*-th input vector as calculated by the PTM method, and $FPC_{MC(i)}$, $FPC_{CSA(i)}$, and $FPC_{CGC(i)}$ have similar representations to $FPC_{PTM(i)}$.

Since CSA cannot calculate the FPC of the multiple-output circuit, we decomposed the benchmarks of ISCAS'85 and ISCAS'89 according to the number of outputs and discarded the circuits with too few gates. The numbers of subcircuits of ISCAS'85 and ISCAS'89 used below are 438 and 543 respectively. In addition, for the convenience of illustration and comparison, FPG is set to 10^{-4} in the following experiments.

4.2 Experimental Results

4.2.1 COSEA VS. PTM

In this section, we compare the accuracy of COSEA and PTM. Due to the limitation of PTM, only a few small circuits are calculated here. Both COSEA and the PTM method can calculate the FPC of a single input vector.

Table 5 lists the errors of several different small-scale circuits as calculated by the PTM method and COSEA. It shows only the average error of all input vectors of each circuit. The calculation formula is shown in formula (24). The circuit with the smallest error is the Comparator, and the calculation error using the PTM method and COSEA is only 0.001%. The circuit with the largest error is Majority, with an average error of 0.033%, while the average error of the listed circuits is 0.012%. It can be seen that, for both the maximum error or the average error of all circuits, the errors of COSEA and PTM method are very small. The

Table 5 Comparison of COSEA and PTM methods

Circuits	*Characteristic*			Error(PTM) (%)
	gates	*input*	*outputs*	
C17	6	5	2	0.006
Full adder(XOR/NAND)	6	3	2	0.008
Comparator	6	5	2	0.001
Decoder2	6	2	4	0.002
MUX4	7	6	1	0.020
Majority	12	5	1	0.033
Average	7.2	4.3	2	0.012

experimental results demonstrate that COSEA has high precision in the FPC calculation of small circuits.

4.2.2 COSEA VS. MC

In order to evaluate the accuracy of COSEA in calculating large-scale and very large-scale circuits, we draw a comparison with the MC simulation. To obtain more accurate experimental results, the number of MC simulation times for each vector is 10^7. The experimental circuits include ISCAS'85 and ISCAS'89 benchmarks. The results for these are shown in Tables 6 and 7, respectively.

As can be seen from the above tables, the average Error(MC) of Table 6 and the average Error(MC) of Table 7 are only 0.678% and 0.480%, respectively. For some of the largest ISCAS'89 circuits, the MC simulation takes too long to calculate. Thus, only 50 or 30 input vectors are considered here.

The results demonstrate that COSEA has high accuracy when it comes to calculating the failure probability of large-scale and very large-scale circuits. In addition, the calculation accuracy of COSEA does not decrease as the circuit size increases, and the speed of this approach is 3–4 orders of magnitude faster than MC simulation.

4.2.3 COSEA VS. CSA

In this section, we compared the proposed COSEA with CSA, taking the results of the MC simulation as the reference standard. These three methods calculate the failure probability corresponding to 100 vectors that are randomly generated by each circuit.

We selected the 10 largest sub-circuits of C7552. Here, C7552_*i* represents the sub-circuit containing the *i*-th primary output node. More information about these ten experimental circuits is presented in Table 8.

Figure 10 plots the relative error of COSEA and CSA respectively compared with MC simulation. As can be seen from the figure, the calculation results of the sub-circuits of C7552 calculated by COSEA are essentially the same as the results of the MC simulation, while the relative error between these two methods is within 1%. By contrast, the accuracy of CSA is not as high. Because the influence of signal correlation is not considered, the FPC calculated by CSA and MC simulation differ greatly, with the maximum error reaching 130% and the average error exceeding 50%. The experimental results reveal that the accuracy of the proposed COSEA is much higher than that of CSA. Considering that the MC simulation is time-consuming, while COSEA is not only fast but also achieves similar accuracy, we will not list the calculation results of the MC simulation in the following experiments.

Figure 11 shows the comparison results of COSEA and CSA methods. The experimental circuits are single output sub-circuit of ISCAS'85 and ISCAS'89. Here, the abscissa represents the number of logic gates of each circuit. Figure 11a and d represent the FPC of two types of circuits. Figure 11b and e represent the relative errors of the two methods. As can be seen from Fig. 11b and e, the

Table 6 Comparison of COSEA and MC simulation (ISCAS'85)

Circuits	*Characteristic*			COSEA	MC	Number of input vectors	Error(MC) (%)	Speed-up ratio
	gates	inputs	outputs	Time (s)	Time (s)			
C432	268	36	7	0.08	348.13	100	0.839	4351.63
C499	826	41	32	0.26	1003.54	100	0.826	3859.77
C880	383	60	26	0.12	242.19	100	0.572	2018.25
C1355	546	41	32	0.18	329.47	100	0.600	1830.39
C1908	880	33	25	0.24	445.67	100	0.632	1856.96
C2670	1193	233	140	0.34	613.13	100	0.627	1803.32
C3540	1669	50	22	0.47	807.56	100	0.682	1718.21
C5315	2307	178	123	0.79	1235.38	100	0.632	1563.77
C6288	2416	32	32	0.83	1381.80	100	0.615	1664.82
C7552	3512	207	108	0.97	1740.47	100	0.750	1794.30
Average	1400	91	55	0.43	814.73	100	0.678	2246.14

Table 7 Comparison of COSEA and MC simulation (ISCAS'89)

Circuits	*Characteristic*			COSEA	MC	Number of input vectors	Error(MC) *(%)*	Speed-up ratio (%)
	gates	inputs	outputs	Time (s)	Time(s)			
S386	159	13	13	0.08	231.67	100	0.746	2895.88
S641	379	54	42	0.18	512.65	100	0.509	2848.06
S953	359	45	52	0.21	486.18	100	0.579	2315.14
S1196	529	32	32	0.26	768.49	100	0.553	2955.73
S1423	657	91	79	0.53	984.16	100	0.583	1856.91
S9234	5597	247	250	1.98	8417.65	100	0.482	4251.34
S13207	7951	700	790	2.35	14346.46	100	0.461	6104.88
S15850	9772	611	684	4.21	24410.95	50	0.350	5798.33
S35932	16065	1763	2048	7.35	37169.35	30	0.325	5057.05
S38417	22179	1664	1724	7.88	44399.88	30	0.207	5634.50
Average	4599	381	415	2.50	13172.74	81	0.480	3971.78

maximum error (CSA) exceed 300% and 160% respectively, while the average errors (Avr.Error) are 13.8% and 8.98% respectively. The proposed COSEA is more accurate than CSA because it fully considers the correlation of signals. Figure 11c and f represent the time consumption of the two methods. We can see that the time consumption is approximately linear with the circuit size. Therefore, both methods can be used to calculate the failure probability of very large-scale circuits, while the COSEA has higher accuracy than CSA.

4.2.4 COSEA VS. CGC

In this section, COSEA is compared with the variant methods of CGC to once again verify its accuracy and effectiveness. The CGC variant methods are divided into six types. Specifically, we selected the single-threaded methods (i.e., CGC-V1, CGC-V3, CGC-V4, and CGC-V6) to compare with COSEA to facilitate fair comparison. Among these, CGC-V1 is a relatively accurate method for locating critical gates. This method verifies the criticality of each node one by one. However, it takes too long to locate the critical gates of large-scale and very large scale circuits using this method. By contrast, CGC-V3 is able to rapidly locate critical gates and can be applied to VLSI circuits, but its accuracy is too low. For their part, CGC-V4 and CGC-V6 combine the features of the first two methods, balancing speed and accuracy.

Any CGC method is based on the input vector and calculates the FPC by locating the critical gates under the specific vectors. In this experiment, we selected 100 input vectors, used four different CGC variant methods to calculate the FPC and compared these results with those of the proposed COSEA.

Table 9 lists the experimental results of some ISCAS'85 circuits, including relative errors and the time consumption calculated by COSEA and the four CGC variant methods. We again use the FPC calculated by COSEA as references to compare these methods.

In terms of accuracy, COSEA and CGC-V1 are both identical and higher than the other three methods. In addition, the average

Table 8 Number of gates and inputs of sub-circuits of C7552

sub-circuits	gates	inputs
C7552_86	1096	194
C7552_106	676	94
C7552_73	606	124
C7552_59	600	124
C7552_60	600	124
C7552_105	598	80
C7552_93	500	94
C7552_94	500	94
C7552_77	499	94
C7552_87	498	94

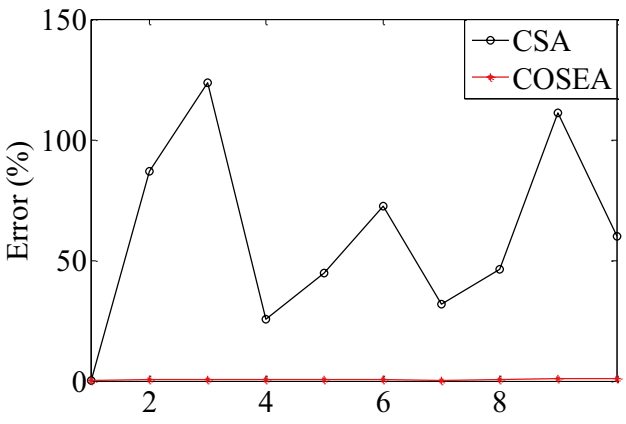
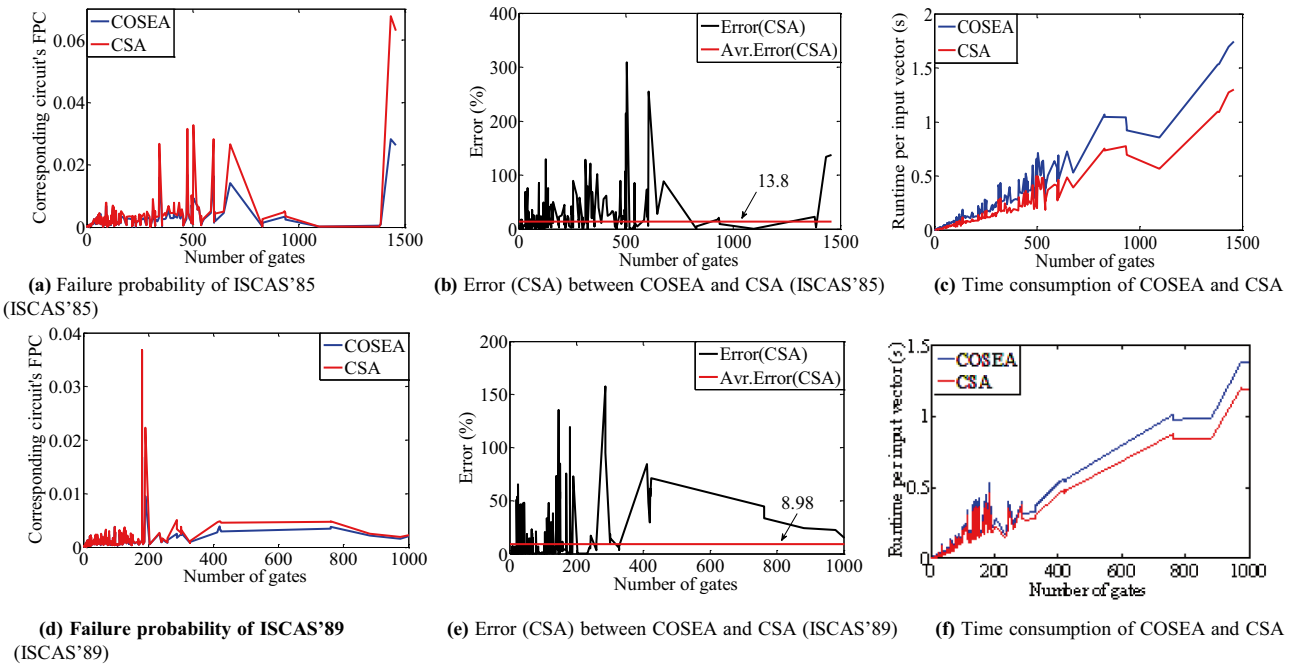


Fig. 10 Failure probability relative error of COSEA and CSA compared with MC simulation of 10 sub-circuits of C7552



(a) Failure probability of ISCAS'85 (ISCAS'85)

(b) Error (CSA) between COSEA and CSA (ISCAS'85)

(c) Time consumption of COSEA and CSA

(d) Failure probability of ISCAS'89 (ISCAS'89)

(e) Error (CSA) between COSEA and CSA (ISCAS'89)

(f) Time consumption of COSEA and CSA

Fig. 11 Comparison of COSEA and CSA (**a**) Failure probability of ISCAS'85 (**b**) Error (CSA) between COSEA and CSA (ISCAS'85) (**c**) Time consumption of COSEA and CSA (ISCAS'85) (**d**) Failure probability of ISCAS'89 (**e**) Error (CSA) between COSEA and CSA (ISCAS'89) (**f**) Time consumption of COSEA and CSA (ISCAS'89)

time required to calculate FPC corresponding to each input vector of the listed experimental circuits is only 0.16 s using COSEA, while the time consumption of CGC-V1, V4, and V6 is several orders of magnitude higher than that of COSEA. CGC-V3 is also slightly faster than COSEA when accuracy is poor. All things considered, the proposed COSEA not only achieves high accuracy, but also has great advantages in speed.

It should be noted that the time required to calculate the failure probability of C880 using the CGC methods is much lower than for other circuits. This is primarily because these CGC methods require few iterations to compute this circuit.

5 Error Analysis

This section will theoretically analyze the accuracy of the proposed COSEA, specifically the factors affecting the accuracy of this approach. There are two reasons for this error: the calculation error and the propagation error of the failure probability of ICS.

5.1 Error in the ICS Calculation Process

In the fault models corresponding to the PTM, PGM, CSA, and COSEA methods, the failure probability of the logic circuit stems from the internal elements of the circuit, while the failure of the circuit is caused primarily by the gate fault. As the high factorial product term is ignored, a calculation error will occur when using COSEA to calculate the FPC.

Next, we consider the method of calculating node error probability using CSA for a single logic gate and analyze the calculation error of failure probability caused by the logic gate. Here, we take a two-input AND gate as an example. The information of the gate inputs are $T_1 = \{1, PFI_{in1}, U_1\}$, and $T_2 = \{1, PFI_{in2}, U_2\}$, while the output PFI_{out} is:

Table 9 Comparison of COSEA and four CGC methods

circuit	COSEA	CGC-V1		CGC-V3		CGC-V4		CGC-V6	
	Time(s)	Error(%)	Time(s)	Error(%)	Time(s)	Error(%)	Time(s)	Error(%)	Time(s)
C432	0.08	0	118.19	3.65	0.04	3.06	20.30	5.81	10.33
C499	0.26	0	428.48	17.18	0.15	8.19	110.95	8.19	32.38
C880	0.12	0	5.07	4.02	0.06	1.18	2.45	2.38	0.96
C1355	0.18	0	1597.47	13.05	0.10	6.23	383.09	6.23	84.80
Average	0.16	0	537.30	9.48	0.09	4.67	129.20	5.65	32.12

$$\begin{aligned}\mathrm{PFI}_{out} = {} & \mathrm{PFI}_{in1} + \mathrm{PFI}_{in2} + \mathrm{FPG} - 2\mathrm{PFI}_{in1} \times \mathrm{FPG} \\ & -2\mathrm{PFI}_{in2} \times \mathrm{FPG} - \mathrm{PFI}_{in1} \times \mathrm{PFI}_{in2} + 2\mathrm{PFI}_{in1} \times \\ & \mathrm{PFI}_{in2} \times \mathrm{FPG}.\end{aligned} \tag{28}$$

The CSA ignores the joint probability of FPG, PFI_{in1}, and PFI_{in2}, the output of which is expressed as $\mathrm{PFI}_{out(\mathrm{cs})} = \mathrm{FPG} + \mathrm{PFI}_{in1} + \mathrm{PFI}_{in2}$. This process results in a loss of calculation accuracy.

We consider the three failure probability elements FPG, PFI_{in1}, and PFI_{in2} respectively and organize PFI_{out} as follows:

$$\begin{aligned}\mathrm{PFI}_{out} = {} & \left(1 + 2\mathrm{PFI}_{in2} \times \mathrm{FPG} - \mathrm{PFI}_{in2} - \mathrm{FPG}\right) \times \mathrm{PFI}_{in1} \\ & + \mathrm{PFI}_{in2} + \mathrm{FPG} - 2\mathrm{PFI}_{in2} \times \mathrm{FPG}.\end{aligned} \tag{29}$$

$$\begin{aligned}\mathrm{PFI}_{out} = {} & \left(1 + 2\mathrm{PFI}_{in1} \times \mathrm{FPG} - \mathrm{PFI}_{in1} - 2\mathrm{FPG}\right) \times \mathrm{PFI}_{in2} \\ & + \mathrm{PFI}_{in1} + \mathrm{FPG} - 2\mathrm{PFI}_{in1} \times \mathrm{FPG}.\end{aligned} \tag{30}$$

$$\begin{aligned}\mathrm{PFI}_{out} = {} & \left(1 + 2\mathrm{PFI}_{in1} \times \mathrm{PFI}_{in2} - 2\mathrm{PFI}_{in1} - 2\mathrm{PFI}_{in2}\right) \times \mathrm{FPG} \\ & + \mathrm{PFI}_{in1} + \mathrm{PFI}_{in2} - \mathrm{PFI}_{in1} \times \mathrm{PFI}_{in2}.\end{aligned} \tag{31}$$

Accordingly, the error of calculation (EC) about PFI_{in1} is:

$$\begin{aligned}\mathrm{EC}_{\mathrm{PFI}_{in1}} &= \frac{\mathrm{PFI}_{out}\left(\mathrm{PFI}_{in1}\right) - \mathrm{PFI}_{out(cs)}\left(\mathrm{PFI}_{in1}\right)}{\mathrm{PFI}_{out}\left(\mathrm{PFI}_{in1}\right)} \\ &= \frac{2\mathrm{PFI}_{in2} \times \mathrm{FPG} - \mathrm{PFI}_{in2} - 2\mathrm{FPG}}{1 + 2\mathrm{PFI}_{in2} \times \mathrm{FPG} - \mathrm{PFI}_{in2} - 2\mathrm{FPG}}.\end{aligned} \tag{32}$$

Similarly, the EC_{PFIin2} and EC_{FPG} are expressed as follows:

$$\mathrm{EC}_{PFIin2} = \frac{2\mathrm{PFI}_{in1} \times \mathrm{FPG} - \mathrm{PFI}_{in1} - 2\mathrm{FPG}}{1 + 2\mathrm{PFI}_{in1} \times \mathrm{FPG} - \mathrm{PFI}_{in1} - 2\mathrm{FPG}}. \tag{33}$$

$$\mathrm{EC}_{FPG} = \frac{2\mathrm{PFI}_{in1} \times \mathrm{PFI}_{in2} - 2\mathrm{PFI}_{in1} - 2\mathrm{PFI}_{in2}}{1 + 2\mathrm{PFI}_{in1} \times \mathrm{PFI}_{in2} - 2\mathrm{PFI}_{in1} - 2\mathrm{PFI}_{in2}}. \tag{34}$$

EC_{FPG} represents the error rate when the fault probability of the logic gate propagates to the next stage, while EC_{PFIin1} and EC_{PFIin2} represent the error rate when the failure probability of the upstream circuit propagates through the logic gate. The influence of the logic gate on the error of the FPC therefore includes both the error of the FPG and the upstream circuit failure probability error. The error of the logic gate is generated when calculating the logic gate. That is, EC_{FPG} denotes the error generated when calculating the failure probability of the logic gate. Moreover, EC_{PFin1} and EC_{PFin2} represent the propagation error when PF_{in1} and PF_{in2} respectively propagate to the logic gate.

EC depends on the absolute value of the molecule in the formula. The smaller the absolute value of the molecule, the smaller the error EC. The absolute value of the molecule is related to PFI_{in1}, PFI_{in2}, and FPG, while the size of PFI_{in1} and PFI_{in2} is also a function of the failure probability of the upstream circuit, i.e., $EC = function$ (FPG). Therefore, the size of EC depends on the FPG.

The signal has only one single propagation path in the ICS to reach the ICS output. On this singular path, it reaches the output of the ICS after multiple propagations, each of which is a change to the result of the previous propagation. Therefore, the error function of the logic gate for the failure probability of ICS is the cumulative multiplication of the propagation times, as shown below:

$$EC_G = \prod_{i=1}^{k} \left(1 + EC_i\right) - 1. \tag{35}$$

Here, EC_G is the error generated when the FPG_G propagates to the output of ICS, while k indicates the propagation times of G in the ICS and EC_i is the i-th error during propagation.

The failure probability of the ICS comprises multiple fault sources within it, so that the error of each ICS is the accumulation of its internal logic gates:

$$EC_{ICS} \approx \sum_{i=1}^{n_{ICS}} EC_{Gi}. \tag{36}$$

Here, EC_{ICS} is the error of the ICS, while n_{ICS} is the number of logic gates of the ICS and EC_{Gi} is the error of the i-th logic gate.

5.2 Error in the PFR Propagation Process

Similar to the way in which the ICS logic gates produce errors in the propagation path, the ICS will also produce errors when propagating in the circuit. ICS includes $\mathrm{C_{FR}}$ and $\mathrm{C_{FI}}$. As the failure probability of $\mathrm{C_{FI}}$ propagates directly to the primary output, the PFI has no transmission error. In this section, we focus solely on the propagation error of PFR. The PFR propagates through different paths, and the logic gates on the paths have different effects on it.

As shown in Fig. 12, the $\mathrm{C_{FR}}$ show their failure probability through the fanout nodes. The PFR propagates through two paths and produces corresponding errors due to the impact of G1 and G2, then converges at G3. According to the proposed COSEA, $\mathrm{U}_{G1} = \mathrm{U}_{G2} = \mathrm{U}_{G3} = \{\mathrm{C_{FR}}\}$, while the circuit failure probabilities of G1 and G2 are as follows:

$$\mathrm{EPN}_{(\mathrm{COSEA})G1} = \mathrm{PFR} + \mathrm{FPG}_{G1}. \tag{37}$$

$$\mathrm{EPN}_{(\mathrm{COSEA})G2} = \mathrm{PFR} + \mathrm{FPG}_{G2}. \tag{38}$$

In fact, the circuit failure probabilities of ICS after G1 and G2 are as follows:

$$\mathrm{EPN}_{G1} = \left(1 - \mathrm{FPG}_{G1}\right) \times \mathrm{PFR} + (1 - \mathrm{PFR}) \times \mathrm{FPG}_{G1}. \tag{39}$$

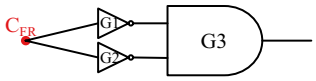


Fig. 12 PFR propagation error

$$\mathrm{EPN}_{G2} = \left(1 - \mathrm{FPG}_{G2}\right) \times \mathrm{PFR} + (1 - \mathrm{PFR}) \times \mathrm{FPG}_{G2}. \quad (40)$$

The error of PFR after G1 is as follows:

$$EP_{G1} = \frac{\mathrm{EPN}_{(\mathrm{COSEA})G1} - \mathrm{EPN}_{G1}}{\mathrm{EPN}_{G1}} = \frac{2\mathrm{FPG}_{G1} \times \mathrm{PFR}}{(1 - 2\mathrm{FPG}_{G1}) \times \mathrm{PFR} + \mathrm{FPG}_{G1}}. \quad (41)$$

Similarly, the error of PFR after G2 is:

$$EP_{G2} = \frac{2\mathrm{FPG}_{G2} \times \mathrm{PFR}}{\left(1 - 2\mathrm{FPG}_{G2}\right) \times \mathrm{PFR} + \mathrm{FPG}_{G2}}. \quad (42)$$

EP_{G1} and EP_{G2} are errors generated every time the PFR propagates through logic gates G1 and G2 in the circuit. Clearly, the magnitude of the error is related to the probability of logic gate failure on the propagation path.

Assuming that G1 and G2 are independent of each other, the error of PFR at G3 is:

$$EP_{\mathrm{PFR}} = EP_{G1} \times EP_{G2}. \quad (43)$$

Generally speaking, the PFR will experience multiple propagations on a path in the circuit. The error size of PFR (EP_{PFRs}) is as follows:

$$EP_{\mathrm{PFR}s} = \prod_{i=1}^{k1} EP_i. \quad (44)$$

Here, $k1$ refers to the number of propagation times of the PFR on the path, while EP_i represents the error of the PFR during the i-th propagation on the path. Unlike the propagation of logic gates inside the ICS, the PFR is able to propagate on multiple paths in the circuit. The PFR error in the propagation process is related to the number of propagation times on the different paths. The PFR reaches the primary outputs of the circuit through multiple paths, and each path has a different number of gates. The final error of PFR is the propagation times of all propagation paths. The calculation formula is as follows:

$$EP_{\mathrm{PFR}m} = \prod_{i=1}^{k2} EP_{PFR_{Si}}. \quad (45)$$

Here, EP_{PFRm} is the error caused by the propagation of PFR through multiple paths, $k2$ represents the number of paths to the output, and EP_{PFRsi} represents the error on each individual path.

The FPC calculated by COSEA is based on the ICS failure probability. There are some errors in the calculation and propagation process of PFR and some errors in the PFI calculation process. The total error of the proposed COSEA is as follows.

$$E = \sum_{i=1}^{n1} EC_{\mathrm{CFI}i} + \sum_{j=1}^{n2} EP_{\mathrm{PFR}mj} \times EP_{\mathrm{CFR}j}. \quad (46)$$

Here, $n1$ is the number of $\mathrm{C_{FI}}$, $EC_{\mathrm{CFI}i}$ is the calculation error of the i-th $\mathrm{C_{FI}}$, $n2$ denotes the number of PFR that reach the primary outputs, $EP_{\mathrm{PFR}mj}$ represents the propagation error of the j-th PFR, and $EC_{\mathrm{CFR}j}$ is the calculation error of the j-th $\mathrm{C_{FR}}$.

6 Conclusion and Future Work

As the circuit scale continues to expand, the accurate and efficient estimation of failure probability of large-scale and very large-scale circuits has become a difficult problem in fault-tolerant circuit design. This paper has accordingly introduced a correlation separation-based approach, called COSEA, to estimate the failure probability of logic circuits. The proposed approach can effectively deal with the influence of signal correlation caused by the reconvergent fanout in the circuit. The computational complexity of COSEA is linearly with the circuit scale, and can be used to accurately estimate the failure probability of large-scale and very large-scale circuits.

We also discuss the accuracy of the proposed COSEA. There are two main sources of error in COSEA: the first is the calculation error of each ICS's failure probability, while the second is the propagation error of $\mathrm{C_{FR}}$ caused by the influence of logic gates on different propagation paths. The experimental results demonstrate that the error resulting when FPC is calculated by COSEA is very small when FPG is 10^{-4}. For more reliable logical units, the error of our approach will be further reduced.

In the future, we could potentially embed COSEA into the reliable design process of integrated circuits. This approach could be used to help us accurately and efficiently locate the sensitive elements in the circuit, thereby improving the reliability of the circuit with a lower fault tolerance cost.

Funding This work was supported by the National Natural Science Foundation of China (NSFC) (Grant No.62172058, No.61702052), and the Hunan Provincial Natural Science Foundation of China (Grant No. 2020JJ4622).

Data Availability The authors declare that the data supporting the findings of this study are available within the article.

Declarations

Conflict of Interests The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

1. Alaghi A, Qian W, Hayes JP (2018) The Promise and Challenge of Stochastic Computing. IEEE Trans Comput Aided Des Integr Circuits Syst 37(8):1515–1531
2. Asadi H, Tahoori MB (2010) Soft error modeling and remediation techniques in ASIC designs. Microelectron J 41(8):506–522
3. Asadi H, Tahoori MB, Fazeli M et al (2012) Efficient algorithms to accurately compute derating factors of digital circuits. Microelectron Reliab 52(6):1215–1226
4. Borkar S (2005) Designing reliable systems from unreliable components: the challenges of transistor variability and degradation. IEEE Micro 25(6):10–16
5. Breuer MA, Gupta SK, Mak TM (2004) Defect and error tolerance in the presence of massive numbers of defects. IEEE Des Test Comput 21(3):216–227
6. Cai H, Petit H, Naviner JF (2011) Reliability aware design of low power continuous-time sigma–delta modulator. Microelectron Reliab 51(9):1449–1453
7. Cai J, Chen C (2017) Circuit Reliability Analysis Using Signal Reliability Correlations. In: Proceedings of 2017 IEEE International Conference on Software Quality, Reliability and Security Companion (QRS-C), Prague, pp 171–176
8. Cai S, He B, Wang W et al (2020) Soft Error Reliability Evaluation of Nanoscale Logic Circuits in the Presence of Multiple Transient Faults. J Electron Test 36(4):469–483
9. Cai S, Wang W, Yu F et al (2019) Single Event Transient Propagation Probabilities Analysis for Nanometer CMOS Circuits. J Electron Test 35(2):163–172
10. Cao X, Xiao L, Li J, Zhang R, Liu S, Wang J (2019) A Layout-Based Soft Error Vulnerability Estimation Approach for Combinational Circuits Considering Single Event Multiple Transients (SEMTs). IEEE Trans Comput Aided Des Integr Circuits Syst 38(6):1109–1122
11. Ebrahimi M, Evans A, Tahoori MB et al (2015) Comprehensive Analysis of Sequential and Combinational Soft Errors in an Embedded Processor. IEEE Trans Comput Aided Des Integr Circuits Syst 34(10):1586–1599
12. El-Maleh AH, Daud KAK (2015) Simulation-Based Method for Synthesizing Soft Error Tolerant Combinational Circuits. IEEE Trans Reliab 64(3):935–948
13. Fazeli M, Ahmadian SN, Miremadi SG, Asadi H, Tahoori MB (2011) Soft error rate estimation of digital circuits in the presence of Multiple Event Transients (METs). In: Proceedings of 2011 Design, Automation & Test in Europe, Grenoble, pp 1–6
14. Franco DT, Vasconcelos MC, Naviner L, Naviner J (2008) Reliability of logic circuits under multiple simultaneous faults. In: Proceedings of 2008 51st Midwest Symposium on Circuits and Systems, Knoxville, TN, pp 265–268
15. Han J, Chen H, Boykin E et al (2011) Reliability evaluation of logic circuits using probabilistic gate models. Microelectron Reliab 51(2):468–476
16. Han J, Chen H, Liang J et al (2014) A Stochastic Computational Approach for Accurate and Efficient Reliability Evaluation. IEEE Trans Comput 63(6):1336–1350
17. Hoefflinger B (2020) ITRS: The internationaltechnology roadmap for semiconductors. In: Hoefflinger B (ed) Chips. Springer, Berlin, pp 161–174
18. Ibrahim W (2016) Identifying the Worst Reliability Input Vectors and the Associated Critical Logic Gates. IEEE Trans Comput 65(6):1748–1760
19. Ibrahim W, Amer H (2016) Critical nodes count algorithm for accurate input vectors reliability ranking. In: Proceedings of the Summer Computer Simulation Conference, Montreal, Quebec, Canada, Article 19, pp 1–7
20. Ibrahim W, Ibrahim H (2019) Multithreaded and Reconvergent Aware Algorithms for Accurate Digital Circuits Reliability Estimation. IEEE Trans Reliab 68(2):514–525
21. Ibrahim W, Shousha M, Chinneck JW (2015) Accurate and Efficient Estimation of Logic Circuits Reliability Bounds. IEEE Trans Comput 64(5):1217–1229
22. Kish LB (2002) End of Moore's law: thermal (noise) death of integration in micro and nano electronics. Phys Lett A 305(3):144–149
23. Krishnaswamy S, Viamontes GF, Markov IL, Hayes JP (2005) Accurate reliability evaluation and enhancement via probabilistic transfer matrices. In: Proceedings of Design, Automation and Test in Europe, Munich, Germany, vol 1, pp 282–287
24. Krishnaswamy S, Viamontes GF, Markov IL et al (2008) Probabilistic transfer matrices in symbolic reliability analysis of logic circuits. ACM Trans Des Autom Electron Syst 13(1):8
25. Liu B, Cai L (2012) Reliability Evaluation for Single Event Transients on Digital Circuits. IEEE Trans Reliab 61(3):687–691
26. Liu B, Cai L (2017) Monte Carlo Reliability Model for Single-Event Transient on Combinational Circuits. IEEE Trans Nucl Sci 64(12):2933–2937
27. Meindl JD, Chen Q, Davis JA (2001) Limits on Silicon Nanoelectronics for Terascale Integration. Science 293(5537):2044–2049
28. Miskov-Zivanov N, Marculescu D (2006) MARS-C: modeling and reduction of soft errors in combinational circuits. In: Proceedings of the 43rd annual Design Automation Conference, San Francisco, CA, USA, pp 767–772
29. Miskov-Zivanov N, Marculescu D (2007) Soft Error Rate Analysis for Sequential Circuits 2007 Design. In: Proc Automation & Test in Europe Conference & Exhibition, Nice, pp 1–6
30. Miskov-Zivanov N, Marculescu D (2010) Multiple Transient Faults in Combinational and Sequential Circuits: A Systematic Approach. IEEE Trans Comput Aided Des Integr Circuits Syst 29(10):1614–1627
31. Naviner L, Liu K, Cai H, Naviner J (2014) Efficient computation of combinational circuits reliability based on probabilistic transfer matrix. In: Proceedings of 2014 IEEE International Conference on IC Design & Technology, Austin, TX, pp 1–4
32. Nicolaidis M (2005) Design for soft error mitigation. IEEE Trans Device Mater Reliab 5(3):405–418
33. Patel KN, Markov LL, Hayes JP (2003) Evaluating circuit reliability under probabilistic gate-level fault models. In: Proceedings of International Workshop on Logic Synthesis (IWLS), 2003, pp 59–64
34. Polian I, Hayes JP, Reddy SM, Becker B (2011) Modeling and Mitigating Transient Errors in Logic Circuits. IEEE Trans Dependable Secure Comput 8(4):537–547
35. Qian W, Li X, Riedel MD, Bazargan K, Lilja DJ (2011) An Architecture for Fault-Tolerant Computation with Stochastic Logic. IEEE Trans Comput 60(1):93–105
36. Rao RR, Chopra K, Blaauw DT, Sylvester DM (2007) Computing the Soft Error Rate of a Combinational Logic Circuit Using Parameterized Descriptors. IEEE Trans Comput Aided Des Integr Circuits Syst 26(3):468–479
37. Saha SK (2010) Modeling process variability in scaled CMOS technology. IEEE Des Test Comput 27(2):8–16
38. Shanbhag NR et al (2008) The Search for Alternative Computational Paradigms. IEEE Des Test Comput 25(4):334–343
39. Xiao J, Lou J, Jiang J (2019) A Fast and Effective Sensitivity Calculation Method for Circuit Input Vectors. IEEE Trans Reliab 68(3):938–953

40. Yan A, Lai C, Zhang Y et al (2018) Novel Low Cost, Double and Triple Node Upset Tolerant Latch Designs for Nano-scale CMOS. IEEE Trans Emerg Top Comput 9(1):520–533
41. Zhang M, Shanbhag NR (2006) Soft-Error-Rate-Analysis (SERA) Methodology. IEEE Trans Comput Aided Des Integr Circuits Syst 25(10):2140–2155

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Shuo Cai received the B.Sc. degree in information engineering from Zhejiang University, Hangzhou, China, and the M.Sc. degree in signal and information processing from Hunan University, Changsha, China, in 2004 and 2007, respectively. He received his Ph.D. degree in school of Computer Science and Electronic Engineering, Hunan University, Changsha, China in 2015. He is now an associate professor at school of Computer and Communication Engineering in Changsha University of Science and Technology, Changsha, China. His research interests include circuit reliability analysis, and fault-tolerant computing.

Binyong He is now a graduate student at school of Computer and Communication Engineering in Changsha University of Science and Technology, Changsha, China. His research interests include IC design and test, and reliability evaluation.

Sicheng Wu is now a graduate student at school of Computer and Communication Engineering in Changsha University of Science and Technology, Changsha, China. His research interests include fault-tolerant computing, and reliability evaluation.

Jin Wang received the B.S. and M.S. degree from Nanjing University of Posts and Telecommunications, China in 2002 and 2005, respectively. He received Ph.D. degree from Kyung Hee University Korea in 2010. Now, he is a professor at Changsha University of Science and Technology. He has published more than 300 international journal and conference papers. His research interests mainly include network performance analysis and optimization.

Weizheng Wang received his BS degree in applied mathematics from Hunan University in 2005 and the Ph.D. degree in technology of computer application from Hunan University in 2011, respectively. Presently, he is an associate professor at the Department of Computer & Communication Engineering, Changsha University of Science and Technology. His research interests include built-in self-test, design for testability, low-power testing, and Hardware security.

Fei Yu received the B.Sc. degree from Anhui Normal University in 2007, the M.Sc. and Ph.D. degrees from school of Computer Science and Electronic Engineering, Hunan University, Changsha, China, in 2010 and 2013, respectively. He is now an associate professor at school of Computer and Communication Engineering in Changsha University of Science and Technology, Changsha, China. He focuses on radio frequency integrated circuit design.