Efficient SRAM Failure Rate Prediction via Gibbs Sampling

Changdao Dong and Xin Li
Electrical & Computer Engineering Department, Carnegie Mellon University
5000 Forbes Avenue, Pittsburgh, PA 15213
{changdao, xinli}@ece.cmu.edu

ABSTRACT

Statistical analysis of SRAM has emerged as a challenging issue because the failure rate of SRAM cells is extremely small. In this paper, we develop an efficient importance sampling algorithm to capture the rare failure event of SRAM cells. In particular, we adapt the Gibbs sampling technique from the statistics community to find the optimal probability distribution for importance sampling with minimum computational cost (i.e., a small number of transistor-level simulations). The proposed Gibbs sampling method applies an integrated optimization engine to adaptively explore the failure region by sampling a sequence of one-dimensional probability distributions. Several implementation issues such as one-dimensional random sampling and starting point selection are carefully studied to make the Gibbs sampling method efficient and accurate for SRAM failure rate prediction. Our experimental results of a commercial 65nm SRAM cell demonstrate that the proposed Gibbs sampling method achieves 3~10× runtime speed-up over other state-of-the-art techniques without surrendering any accuracy.

Categories and Subject Descriptors

B.7.2 [**Integrated Circuits**]: Design Aids – Verification

General Terms

Algorithms

Keywords

Integrated Circuit, Process Variation, Memory

1. INTRODUCTION

As deep sub-micron technology advances, process variations pose a new set of challenges on SRAM design. SRAM has been widely embedded in a large amount of semiconductor chips. For example, roughly half of the area of an advanced microprocessor chip is occupied by SRAM [13]. SRAM cells are generally designed with minimum-size devices [13] and can be significantly impacted by large-scale process variations (e.g., local mismatches caused by random doping fluctuations) at nanoscale technology [1]. For this reason, it becomes increasingly critical to evaluate the failure rate of SRAM cells both efficiently and accurately in order to achieve a robust design.

Towards this goal, a number of statistical analysis methods have been proposed for SRAM circuits [2]-[10]. For instance, analytical performance models have been derived to predict SRAM parametric yield [2]-[4]. While these models offer great design insights to understand SRAM circuits, they may not accurately capture the circuit behavior due to various approximations that are made. Another possible approach for SRAM failure rate prediction is based on transistor-level simulation, including both Monte Carlo analysis [5]-[9] and deterministic failure region prediction [10].

Since SRAM cells typically have extremely small failure probability (i.e., 10^{-6}~10^{-8}), a simple Monte Carlo method by directly sampling the variation space suffers from slow convergence rate, as only few random samples will fall into the failure region. To improve the sampling efficiency, a number of importance sampling methods have been proposed for fast SRAM failure rate prediction [5]-[9]. The key idea of importance sampling is to directly sample the failure region based on a distorted probability density function (PDF), instead of the original PDF of process variations.

Applying importance sampling to SRAM analysis, however, is not trivial. The efficiency of all existing importance sampling algorithms heavily relies on the choice of the distorted PDF that is used to generate random samples. Ideally, in order to maximize prediction accuracy, we should sample the failure region that is most likely to occur. Such a goal, however, is extremely difficult to achieve, since we never exactly know the failure region in practice. The challenging issue here is how to determine the *optimal* PDF for importance sampling so that the SRAM failure rate can be efficiently predicted.

In this paper, a novel Gibbs sampling method is proposed to improve the efficiency of SRAM failure rate prediction. Unlike the traditional Monte Carlo algorithm that samples a given PDF, the proposed Gibbs sampling approach does not need to know the sampling PDF explicitly. Instead, it adaptively searches the failure region and then generates random samples in it. When applied to SRAM failure rate analysis, Gibbs sampling can be conceptually viewed as a unique Monte Carlo method with an integrated optimization engine which allows us to efficiently explore the failure region. As a result, SRAM failure probability can be accurately predicted with a minimum number of sampling points (i.e., a small number of transistor-level simulations). Our experimental results of a commercial 65nm SRAM cell demonstrate that compared to other state-of-the-art techniques, the proposed Gibbs sampling algorithm achieves 3~10× runtime speed-up without surrendering any accuracy.

While the Gibbs sampling method was initially developed by the statistics community [11], [15], it is particularly tuned for our SRAM analysis application via three important new contributions. First, as previously mentioned, Gibbs sampling iteratively searches the failure region. At each iteration step, it needs to generate a random sample from an irregular one-dimensional PDF that is not simply uniform or Normal. Such a sampling task cannot be easily done by directly using an existing random number generator. For this reason, we propose to adopt the inverse-transform method [15] and incorporate it into our proposed Gibbs sampling engine. As such, randomly sampling an arbitrary one-dimensional PDF can be performed efficiently, which is a key technique to make the proposed Gibbs sampling of practical utility.

DAC'11, June 5-10, 2011, San Diego, California, USA

Second, an efficient implementation of Gibbs sampling requires a good starting point to speed-up the convergence. This is similar to most optimization algorithms where a good starting point facilitates fast convergence. In this paper, we propose a model-based optimization to determine a good starting point so that the Gibbs sampling algorithm can converge quickly (i.e., accurately predict the failure probability with a small number of sampling points).

Finally, to further improve prediction accuracy and reduce computational cost, a two-stage Monte Carlo flow is developed where Gibbs sampling is applied to create a set of random samples during the first stage and these samples are used to "learn" the optimal PDF for importance sampling. Next, during the second stage, a large number of random samples are efficiently generated from the optimal PDF to accurately estimate the failure probability.

The remainder of this paper is organized as follows. In Section 2, we briefly review the background on importance sampling and then propose the Gibbs sampling method in Section 3. A commercial 65nm SRAM cell is used to demonstrate the efficacy of the proposed Gibbs sampling method in Section 4. Finally, we conclude in Section 5.

2. BACKGROUND

Suppose that x is a d-dimensional random variable modeling process variations and its joint PDF is $p(x)$. Typically, x is modeled as a multivariate Normal distribution [2]-[10]. Without loss of generality, we further assume that the random variables $\{x_i; i = 1,2,\ldots,d\}$ in the vector x are mutually independent and standard Normal (i.e., with zero mean and unit variance):

$$p(x) = \prod_{i=1}^{d} \left[\frac{1}{\sqrt{2\pi}} \cdot \exp\left(-x_i^2\right) \right]. \tag{1}$$

Any correlated random variables that are jointly Normal can be transformed to the independent random variables in (1) by principal component analysis (PCA) [14].

The failure probability of an SRAM cell can be mathematically represented as [5]:

$$P_f = \int_{\Omega} p(x) \cdot dx \tag{2}$$

where Ω denotes the failure region, i.e., the subset of the variation space where the performance of interest (e.g., read margin, write margin, etc.) does not meet the specification. Alternatively, the failure probability in (2) can be defined as:

$$P_f = \int_{-\infty}^{+\infty} I(x) \cdot p(x) \cdot dx \tag{3}$$

where $I(x)$ represents the indicator function:

$$I(x) = \begin{cases} 1 & x \in \Omega \\ 0 & x \notin \Omega \end{cases}. \tag{4}$$

The failure probability P_f can be estimated by Monte Carlo analysis. The key idea is to draw N random samples from $p(x)$, and then compute the mean of these samples:

$$\widetilde{P}_f^{MC} = \frac{1}{N} \cdot \sum_{i=1}^{N} I\left[x^{(i)}\right] \tag{5}$$

where $x^{(i)}$ is the ith random sample generated by Monte Carlo analysis.

For our proposed SRAM application, the failure probability P_f in (3) is extremely small (e.g., 10^{-6}~10^{-8}) and most random samples created by Monte Carlo analysis do not fall into the failure region Ω. Hence, a large number of (e.g., over 10^9) samples are needed by the Monte Carlo method to accurately estimate the failure rate. Note that expensive transistor-level simulation is required to create each sampling point. In order words, 10^9 simulation runs must be performed in order to collect 10^9 random samples. It, in turn, implies that the aforementioned Monte Carlo method is extremely expensive, or even infeasible, when applied to most practical SRAM analysis problems.

To address this computational cost issue, importance sampling has been proposed to improve the efficiency of Monte Carlo analysis [5]-[9]. It aims to directly generate a large number of random samples in the failure region by using a distorted PDF $q(x)$. In this case, the failure probability can be expressed as [5]:

$$P_f = \int_{-\infty}^{+\infty} \frac{I(x) \cdot p(x)}{q(x)} \cdot q(x) \cdot dx . \tag{6}$$

In other words, Eq. (6) calculates the expected value of the function $I(x) \cdot p(x)/q(x)$ where the random variable x follows the PDF $q(x)$. If N sampling points $\{x^{(i)}; i = 1,2,\ldots,N\}$ are drawn from $q(x)$, the failure probability in (6) can be estimated by [5]:

$$\widetilde{P}_f^{IS} = \frac{1}{N} \cdot \sum_{i=1}^{N} \frac{I\left[x^{(i)}\right] \cdot p\left[x^{(i)}\right]}{q\left[x^{(i)}\right]} . \tag{7}$$

Note that the estimated failure probabilities in (5) and (7) are identical, if and only if the number of random samples (i.e., N) is infinite. In practice, when a finite number of sampling points are available, the results from (5) and (7) can be substantially different. If the distorted PDF $q(x)$ is properly chosen for importance sampling, Eq. (7) can be much more accurate than the simple Monte Carlo method in (5). In theory, the optimal PDF $q(x)$ with maximum estimation accuracy is [15]:

$$q^{OPT}(x) = \frac{I(x) \cdot p(x)}{P_f} . \tag{8}$$

Intuitively, if the PDF $q^{OPT}(x)$ in (8) is used, the function $I(x) \cdot p(x)/q^{OPT}(x)$ becomes a constant with zero variance. Hence, its expected value can be accurately estimated by (7) using few random samples.

Studying (8), we would notice two important properties of the optimal PDF $q^{OPT}(x)$. First, $q^{OPT}(x)$ is non-zero if and only if the variable x sits in the failure region. It, in turn, implies that we should directly sample the failure region to achieve maximum accuracy. Second, $q^{OPT}(x)$ is proportional to the original PDF $p(x)$ of process variations. In other words, the entire failure region should *not* be sampled uniformly. Instead, we should sample the variation space where performance failure is most likely to occur.

In practice, however, sampling the optimal PDF $q^{OPT}(x)$ in (8) is not trivial, as the indicator function $I(x)$ is not known in advance. Most exiting importance sampling algorithms apply various heuristics to approximate the optimal PDF $q^{OPT}(x)$ [5]-[9]. In this paper, we propose a new Gibbs sampling method that adaptively samples the optimal PDF $q^{OPT}(x)$ without explicitly knowing the indicator function $I(x)$. As such, the SRAM failure rate can be accurately predicted with minimum computational cost.

3. PROPOSED APPROACH

Our proposed SRAM failure rate analysis is facilitated by a combination of several novel techniques. In this section, we will discuss the details of these techniques and highlight their novelties.

3.1 Gibbs Sampling

As described in Section 2, directly sampling the optimal PDF

$q^{OPT}(x)$ in (8) is difficult due to two reasons. First, the indicator function $I(x)$ is not known in advance, as the failure region is unknown. Second, since $q^{OPT}(x)$ is not a simple multivariate statistical distribution such as uniform distribution or Normal distribution, it is extremely difficult, if not impossible, to directly draw random samples from $q^{OPT}(x)$.

In this paper, we adopt the Gibbs sampling method [11], [15] from statistics to predict the failure probability of SRAM cells. Compared to other traditional techniques [5]-[9], Gibbs sampling provides two promising features. First, it can efficiently search the failure region and determine the indicator function $I(x)$ on the fly. From this point of view, Gibbs sampling can be conceptually viewed as an integrated optimization engine that allows us to adaptively sample the optimal PDF $q^{OPT}(x)$ in (8). To the best of our knowledge, such optimal sampling is not possible by using other existing techniques [5]-[9].

Second, Gibbs sampling does not directly draw random samples from a multi-dimensional joint PDF. Instead, it iteratively samples a sequence of one-dimensional PDF's. These one-dimensional PDF's are not simply uniform or Normal, and, hence, cannot be directly sampled by a simple random number generator. However, as will be demonstrated in Section 3.2, the aforementioned one-dimensional sampling can be efficiently implemented with an inverse-transform method [15] with low computational cost.

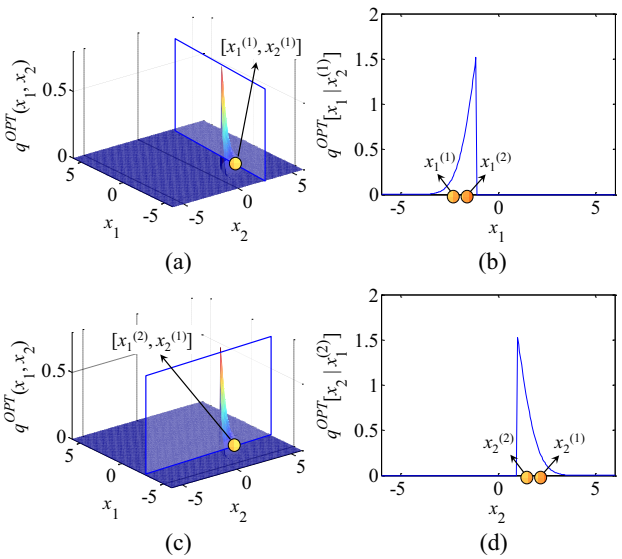


Figure 1. A simple two-dimensional Gibbs sampling example where $[x_1^{(1)},x_2^{(1)}]$ is the starting point. (a) The intersection between the joint PDF $q^{OPT}(x_1,x_2)$ and the plane $x_2 = x_2^{(1)}$ defines the conditional PDF $q^{OPT}[x_1|x_2^{(1)}]$. (b) A new sampling point $x_1^{(2)}$ is drawn from $q^{OPT}[x_1|x_2^{(1)}]$. (c) The intersection of the joint PDF $q^{OPT}(x_1,x_2)$ and the plane $x_1 = x_1^{(2)}$ defines the conditional PDF $q^{OPT}[x_2|x_1^{(2)}]$. (d) A new sampling point x_2^2 is drawn from $q^{OPT}[x_2|x_1^{(2)}]$.

To intuitively illustrate the Gibbs sampling algorithm, we first consider the simple two-dimensional example in Figure 1 where our goal is to sample the PDF $q^{OPT}(x_1,x_2)$. In this example, Gibbs sampling starts from an initial point $[x_1^{(1)},x_2^{(1)}]$. It first samples the conditional PDF $q^{OPT}[x_1|x_2^{(1)}]$ and replaces $x_1^{(1)}$ by a new value $x_1^{(2)}$, as shown in Figure 1(a)-(b). During this iteration step, we generate a new sampling point $[x_1^{(2)},x_2^{(1)}]$. Next, Gibbs sampling samples a different conditional PDF $q^{OPT}[x_2|x_1^{(2)}]$ and replaces $x_2^{(1)}$ by a new value $x_2^{(2)}$, as shown in Figure 1(c)-(d). A new sampling point $[x_1^{(2)},x_2^{(2)}]$ is created. Since the random variable x is two-dimensional in this example, Gibbs sampling varies x_1 again at the third iteration step. It draws a new random value $x_1^{(3)}$ by sampling the conditional PDF $q^{OPT}[x_1|x_2^{(2)}]$, resulting in a new sampling point $[x_1^{(3)},x_2^{(2)}]$. These iteration steps are repeatedly applied until a sufficient number of random samples are created. It can be proven that the aforementioned iteration yields a sequence of random samples that follows the given distribution $q^{OPT}(x_1,x_2)$ [11], [15].

The aforementioned two-dimensional Gibbs sampling can be extended to the general case where the PDF $q^{OPT}(x) = q^{OPT}(x_1,x_2,\ldots,x_d)$ is d-dimensional. Starting from an initial point $[x_1^{(1)},x_2^{(1)},\ldots,x_d^{(1)}]$, Gibbs sampling assigns a new value to one of the d random variables at each iteration step. This new value is determined by randomly sampling the conditional PDF. For instance, when sampling the ith random variable x_i, the following conditional PDF is used:

$$q^{OPT}\left(x_i \mid x_1,\cdots,x_{i-1},x_{i+1},\cdots,x_d\right) = q^{OPT}\left(x_i \mid x_{\setminus i}\right) \tag{9}$$

where $x_{\setminus i}$ denotes the vector x with x_i removed. Such random sampling is repeated with one random variable sampled at one time. Algorithm 1 summarizes the major steps of Gibbs sampling.

As discussed at the beginning of this section, directly sampling the multi-dimensional joint PDF $q^{OPT}(x)$ can be extremely difficult. By using Gibbs sampling, we only need to sample the one-dimensional conditional PDF $q^{OPT}(x_i|x_{\setminus i})$. As will be demonstrated in Section 3.2, such one-dimensional sampling can be efficiently implemented by using an inverse-transform method [15], even if the indicator function $I(x)$ in (4) and, hence, the PDF $q^{OPT}(x)$ in (8) are not explicitly known. In addition, a model-based optimization can be applied to determine a good starting point $[x_1^{(1)},x_2^{(1)},\ldots,x_d^{(1)}]$ so that the proposed Gibbs sampling algorithm converges quickly. In what follows, we will discuss these implementation issues in detail.

Algorithm 1: Gibbs Sampling

1. Start from a d-dimensional PDF $q^{OPT}(x_1,x_2,\ldots,x_d)$.
2. Select an initial starting point $[x_1^{(1)},x_2^{(1)},\ldots,x_d^{(1)}]$.
3. For $t = 1,2,\ldots$
4. For $i = 1,2,\ldots,d$
5. Draw $x_i^{(t+1)}$ from the conditional PDF $q^{OPT}(x_i|x_{\setminus i})$ to create a new sampling point $[x_1^{(t+1)},\ldots,x_i^{(t+1)},x_{i+1}^{(t)},\ldots,x_d^{(t)}]$.
6. End For
7. End For

3.2 One-dimensional Inverse-transform Sampling

During each iteration of Gibbs sampling, one of the d random variables (say, x_i) is sampled by the one-dimensional conditional PDF $q^{OPT}(x_i|x_{\setminus i})$. Such one-dimensional sampling can be efficiently performed by using an inverse-transform method [15]. To derive the one-dimensional sampling algorithm used in this paper, we first write $q^{OPT}(x_i|x_{\setminus i})$ as:

$$q^{OPT}\left(x_i \mid x_{\setminus i}\right) = \frac{q^{OPT}\left(x_i, x_{\setminus i}\right)}{q^{OPT}\left(x_{\setminus i}\right)} = \frac{q^{OPT}\left(x\right)}{q^{OPT}\left(x_{\setminus i}\right)} \tag{10}$$

where $q^{OPT}(x_{\setminus i})$ is the marginal PDF of $x_{\setminus i}$. Substituting (8) into (10) yields:

$$q^{OPT}\left(x_i \mid x_{\setminus i}\right) = \frac{I\left(x_i, x_{\setminus i}\right) \cdot p\left(x_i, x_{\setminus i}\right)}{P_f \cdot q^{OPT}\left(x_{\setminus i}\right)}. \tag{11}$$

Studying (11), we would have three important observations. First, $q^{OPT}(x_i|x_{\setminus i})$ is linearly proportional to the actual probability

distribution of process variations, i.e., $p(x_i,x_{\setminus i})$. Second, $q^{OPT}(x_i|x_{\setminus i})$ is non-zero if and only if the variable x_i, combined with $x_{\setminus i}$, sits in the failure region. Third, the denominator $P_f \cdot q^{OPT}(x_{\setminus i})$ in (11) is a constant, given any fixed value of $x_{\setminus i}$.

Since the integral of the PDF in (11) must equal 1, we have:

$$P_f \cdot q^{OPT}(x_{\setminus i}) = \int_{-\infty}^{+\infty} I(x_i, x_{\setminus i}) \cdot p(x_i, x_{\setminus i}) \cdot dx_i \,. \tag{12}$$

To calculate the value of $P_f \cdot q^{OPT}(x_{\setminus i})$ in (12) and, hence, determine the conditional PDF $q^{OPT}(x_i|x_{\setminus i})$ in (11), we consider two different scenarios shown in Figure 2. In Figure 2, the symbol Ω_i represents the failure region of the ith dimension x_i with a fixed value of $x_{\setminus i}$. Note that the failure region Ω_i is represented as either $\Omega_i = (-\infty, b_i]$ or $\Omega_i = [b_i, +\infty)$. In other words, we assume that there is only one single continuous failure region and $x_i = b_i$ is the boundary. Such an assumption is valid, if we consider only one failure mechanism (e.g., read margin when reading 0) at one time, similar to other previous works [5]-[9].

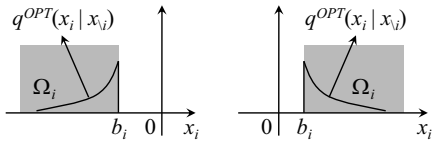


Figure 2. Two different scenarios of the SRAM failure region Ω_i: (a) $\Omega_i = (-\infty, b_i]$, and (b) $\Omega_i = [b_i, +\infty)$.

Based on the assumption that Ω_i is a single continuous region with the boundary $x_i = b_i$, we first find b_i by performing binary search along the ith dimension of process variations. Once b_i is known, Eq. (12) can be re-written as:

$$P_f \cdot q^{OPT}(x_{\setminus i}) = \begin{cases} \int_{-\infty}^{b_i} p(x_i, x_{\setminus i}) \cdot dx_i & if \quad \Omega_i = (-\infty, b_i] \\ \int_{b_i}^{+\infty} p(x_i, x_{\setminus i}) \cdot dx_i & if \quad \Omega_i = [b_i, +\infty) \end{cases} . \tag{13}$$

Since the random variable x follows a multivariate Normal distribution in (1), the integral in (13) can be calculated based on the Gauss error function [14]. Once $P_f \cdot q^{OPT}(x_{\setminus i})$ is solved, the analytical form of the conditional PDF $q^{OPT}(x_i|x_{\setminus i})$ in (11) is known.

It is important to note that $q^{OPT}(x_i|x_{\setminus i})$ in (11) is not a simple statistical distribution such as uniform distribution or Normal distribution. Hence, it cannot be directly sampled by using a random number generator. To apply the inverse-transform method to sample $q^{OPT}(x_i|x_{\setminus i})$, we need to calculate its cumulative distribution function (CDF):

$$Q(x_i) = \int_{-\infty}^{x_i} q^{OPT}(\tau \mid x_{\setminus i}) \cdot d\tau \,. \tag{14}$$

The key idea of the inverse-transform method is to sample a new random variable y_i that is uniformly distributed over the interval [0, 1]. Next, we map the sampling point y_i back to x_i based on the inverse CDF:

$$x_i = Q^{-1}(y_i) \tag{15}$$

where $Q^{-1}(\bullet)$ is the inverse function of $Q(\bullet)$. It can be proven that the sampling point x_i generated by the inverse-transform method follows the statistical distribution $q^{OPT}(x_i|x_{\setminus i})$ in (11) [15].

Algorithm 2 summarizes the major steps of the aforementioned inverse-transform method. It is important to emphasize that the computational cost of Algorithm 2 is dominated by Step 2 where multiple transistor-level simulations are required to find the boundary b_i of the failure region Ω_i. All other steps do not involve transistor-level simulation and, hence, can be completed with low computational cost.

Algorithm 2: One-dimensional Inverse-transform Sampling

1. Start from a d-dimensional PDF $q^{OPT}(x)$, a given random variable x_i for sampling, and the sampled value of all other random variables in the vector $x_{\setminus i}$.
2. Along the ith dimension x_i, apply binary search to find the boundary b_i of the failure region: $\Omega_i = (-\infty, b_i]$ or $\Omega_i = [b_i, +\infty)$.
3. Calculate the value of $P_f \cdot q^{OPT}(x_{\setminus i})$ by using (13).
4. Construct the conditional PDF $q^{OPT}(x_i|x_{\setminus i})$ in (11).
5. Construct the CDF $Q(x_i)$ in (14) and the inverse CDF $Q^{-1}(y_i)$ in (15).
6. Uniformly sample the random variable y_i over the interval [0, 1].
7. Map the sampling point y_i back to x_i based on (15).

3.3 Initial Starting Point Selection

To efficiently implement the proposed Gibbs sampling method, a good starting point $x^{(1)} = [x_1^{(1)}, x_2^{(1)}, ..., x_d^{(1)}]$ should be appropriately selected to speed-up the convergence of Algorithm 1. This is similar to most optimization algorithms where a good starting point facilitates fast convergence. Remember that the key idea of importance sampling is to directly draw random samples from the failure region that is most likely to occur. Hence, the initial starting point $x^{(1)}$ should meet the following two requirements: (1) $x^{(1)}$ is inside the failure region, and (2) the PDF of process variations, i.e., $p(x)$, takes a large value at $x^{(1)}$. These two requirements can be mathematically translated to the optimization problem:

$$\begin{aligned} \max_{x^{(1)}} \quad & p\left[x^{(1)}\right] \\ S.T. \quad & x^{(1)} \in \Omega \end{aligned} . \tag{16}$$

In (16), we want to find the failure point $x^{(1)}$ that is most likely to occur. Since the random variable x is modeled as a multivariate Normal distribution in (1), the optimization in (16) can be re-written as:

$$\begin{aligned} \min_{x^{(1)}} \quad & \left\| x^{(1)} \right\|_2 \\ S.T. \quad & x^{(1)} \in \Omega \end{aligned} \tag{17}$$

where $\|\bullet\|_2$ represents the L_2-norm of a vector. Eq. (17) aims to find the failure point x_1 that is closest to the origin $x = 0$. It is similar to the norm minimization problem defined in [7].

Note that solving the optimization problem in (17) is not trivial, since the failure region Ω is not explicitly known. To address this issue, we propose to approximate the performance of interest (e.g., read margin, write margin, etc.) as a linear or quadratic model of the random variable x. Once the model is available, the optimization in (17) can be solved by either quadratic programming (for linear model) or semi-definite programming (for quadratic model) [12]. The detailed algorithm for performance modeling and optimization can be found in [12].

Finally, it is important to mention that even though the linear or quadratic performance models may not be highly accurate to cover a large variation space, we can still obtain a good starting point required by the proposed Gibbs sampling. Note that our goal is *not* to exactly solve (17). Instead, we only want to find an *approximated* solution of (17) that can be used by Gibbs sampling to further explore the variation space with high failure probability.

3.4 Two-stage Monte Carlo Flow

To make the proposed Gibbs sampling algorithm efficient, there is one additional implementation issue that should be further addressed. As shown in Algorithm 2, multiple transistor-level simulations are required to perform binary search and then generate a single Gibbs sample. For this reason, once a set of (say, M) Gibbs samples are created, it is desired to "learn" the PDF $q^{OPT}(x)$ from these samples so that additional random sampling points can be directly drawn from $q^{OPT}(x)$ without running binary search. Such a strategy would help us to further reduce the computational cost and/or improve the prediction accuracy.

Towards this goal, we propose to adopt a two-stage Monte Carlo flow consisting of two sequential steps. First, Gibbs sampling (i.e., Algorithm 1 and Algorithm 2) is applied to generate M random samples inside the failure region. Next, in the second stage, we approximate the optimal PDF $q^{OPT}(x)$ as a multivariate Normal distribution $q(x)$ where the mean value and the covariance matrix of $q(x)$ are calculated from the M Gibbs samples. Once $q(x)$ is known, we directly sample it to generate N random samples and estimate the failure rate from these N samples by using (7). Algorithm 3 summarizes the major steps of the aforementioned two-stage Monte Carlo flow.

It should be noted that several traditional SRAM analysis techniques also draw importance samples from a multivariate Normal distribution [5], [7]. However, unlike the traditional techniques that determine the sampling PDF $q(x)$ by various heuristics, we apply Gibbs sampling to find the optimal PDF $q^{OPT}(x)$. Hence, the second-stage random sampling can converge quickly. Compared to other traditional techniques, Algorithm 3 provides 3~10× speed-up without surrendering any accuracy, as will be demonstrated by the experimental results in Section 4.

Algorithm 3: Two-stage Monte Carlo Flow

1. Start from a joint PDF $p(x)$, a given value of M (the number of Gibbs samples for the first stage), and a given value of N (the number of random samples for the second stage).
2. Apply Algorithm 1 and Algorithm 2 to generate M Gibbs sampling points.
3. Calculate the mean value and the covariance matrix of these M Gibbs samples. Determine a multivariate Normal distribution $q(x)$ to approximate the optimal PDF $q^{OPT}(x)$.
4. Generate N random samples from the multivariate Normal distribution $q(x)$.
5. Calculate the failure rate from these N samples by using (7).

4. EXPERIMENTAL RESULTS

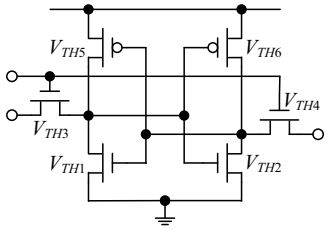
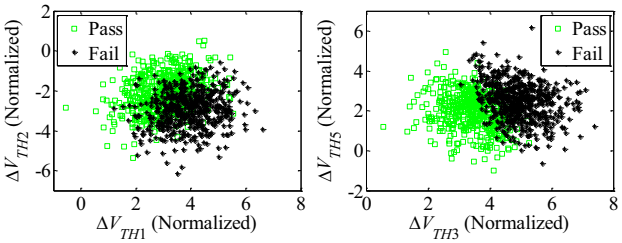


Figure 3. Circuit schematic of a 6-T SRAM cell.

In this section, a 6-T SRAM cell designed in a commercial 65nm process is used to demonstrate the efficacy of the proposed Gibbs sampling method. Figure 3 shows the circuit schematic of the SRAM cell where the local V_{TH} mismatch of each transistor is considered for Monte Carlo analysis. In this example, two performance matrices, read noise margin (RNM) and write noise margin (WNM), are used to assess the stability of the SRAM cell. For testing and comparison purpose, three different importance sampling methods are implemented: (1) mixture importance sampling (MIS) [5], (2) minimum-norm importance sampling (MNIS) [7], and (3) the proposed Gibbs sampling. All these three methods employ a two-stage analysis flow. Namely, a multivariate Normal distribution $q(x)$ is first constructed during the first stage and then importance samples are drawn from $q(x)$ to calculate the failure probability at the second stage.

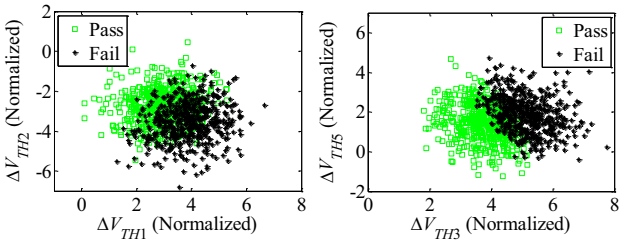
Table 1. Number of simulations at the first stage to determine the multivariate Normal distribution for importance sampling

	MIS [5]	MNIS [7]	Proposed
RNM	5600	2000	2026
WNM	3800	2000	1744



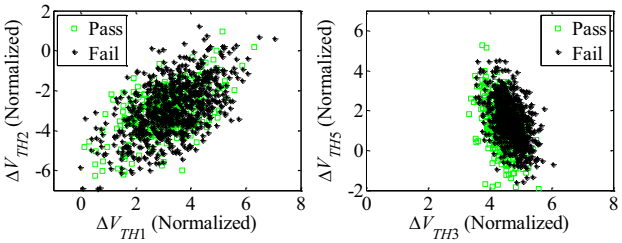
(a) (b)

Figure 4. Random samples generated by MIS [5] at the second stage: (a) RNM and (b) WNM.



(a) (b)

Figure 5. Random samples generated by MNIS [7] at the second stage: (a) RNM and (b) WNM.



(a) (b)

Figure 6. Random samples generated by the proposed Gibbs sampling at the second stage: (a) RNM and (b) WNM.

Table 1 shows the number of transistor-level simulations that are required by the first stage to determine the multivariate Normal distribution $q(x)$ for importance sampling. In this example, 400 Gibbs samples are generated for our proposed method at the first stage. Remember that each Gibbs sample involves multiple transistor-level simulations (see Algorithm 2). The total number of simulations is 2026 + 1744 = 3770 in this example. Note that MIS and MNIS require more transistor-level simulations in order to determine the PDF $q(x)$ at the first stage.

Next, to illustrate the difference between the three importance sampling algorithms, Figure 4~Figure 6 plot the first 1000 random samples that are generated at the second stage. For illustration purpose, these figures only show the V_{TH} mismatches of two transistors that are critical to the performance of interest. As shown in Figure 3, the mismatches between V_{TH1} and V_{TH2} are important for RNM and the mismatches between V_{TH3} and V_{TH5} are important for WNM. Studying Figure 4~Figure 6, we would have two key observations. First, both MIS and MNIS cannot accurately capture the optimal PDF $q^{OPT}(x)$ for importance sampling. These two traditional methods only attempt to identify the mean value of $q^{OPT}(x)$, while the mutual correlation between different random variables is completely ignored. Hence, a large number of simulation samples generated at the second stage do not fall into the failure region, as shown in Figure 4~Figure 5. On the other hand, the proposed Gibbs sampling is able to accurately capture both the mean value and the covariance matrix of $q^{OPT}(x)$ (see Algorithm 3). For this reason, most sampling points generated by Gibbs sampling are inside the failure region, thereby substantially improving the accuracy of failure rate prediction.

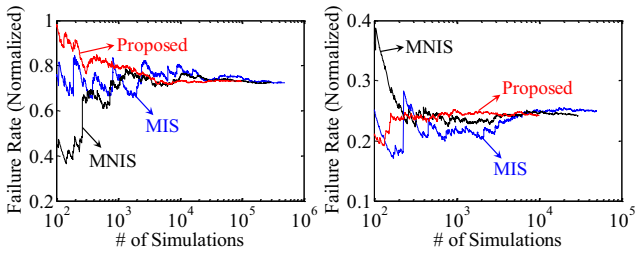


Figure 7. Estimated failure probability (normalized) as a function of the number of transistor-level simulations at the second stage: (a) RNM and (b) WNM.

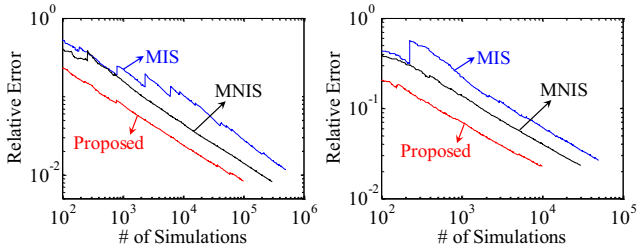


Figure 8. Relative error of failure rate prediction (defined by 95% confidence interval) as a function of the number of transistor-level simulations at the second stage: (a) RNM and (b) WNM.

Figure 7 shows the estimated failure probability (normalized) as a function of the number of transistor-level simulations at the second stage. Note that all three importance sampling methods yield the same failure probability, if the number of random samples is sufficiently large. In this example, the proposed Gibbs sampling is substantially more accurate than the other two traditional methods (i.e., MIS and MNIS) given the same number of random samples. Figure 8 further plots the relative prediction error as a function of the number of simulations. Here, the relative error is defined as the ratio of the 95% confidence interval over the estimated failure probability. As shown in Figure 8, the proposed Gibbs sampling achieves 3~10× runtime speed-up over MIS and MNIS. To achieve 5% error, the proposed Gibbs sampling only requires 3000~4500 simulations at the second stage, while both MIS and MNIS require more than 9000 simulations.

5. CONCLUSIONS

In this paper, a novel Gibbs sampling method is proposed for efficient failure rate prediction of SRAM circuits. The proposed Gibbs sampling adaptively explores the variation space so that a large number of random samples fall into the failure region. In particular, it iteratively samples a sequence of one-dimensional PDF's by an efficient inverse-transform method. As is demonstrated by our experimental results for a commercial 65nm SRAM cell, the proposed Gibbs sampling achieves 3~10× runtime speed-up over other state-of-the-art techniques without surrendering any accuracy. The Gibbs sampling technique can be further incorporated into a statistical analysis/optimization environment for accurate and efficient parametric yield prediction of SRAM circuits.

6. ACKNOWLEDGEMENTS

This work is supported in part by the Semiconductor Research Corporation under contract 1836.044 and the National Science Foundation under contract CCF–1016890.

7. REFERENCES

[1] B. Calhoun, Y. Cao, X. Li, K. Mai, L. Pileggi, R. Rutenbar and K. Shepard, "Digital circuit design challenges and opportunities in the era of nanoscale CMOS," *Proceedings of The IEEE*, vol. 96, no. 2, pp. 343-365, Feb. 2008.

[2] S. Mukhopadhyay, H. Mahmoodi and K. Roy, "Statistical design and optimization of SRAM cell for yield enhancement," *IEEE ICCAD*, pp. 10-13, 2004.

[3] K. Agarwal and S. Nassif, "Statistical analysis of SRAM cell stability," *IEEE DAC*, pp. 57-62, 2006.

[4] M. Abu-Rahma, K. Chowdhury, J. Wang, Z. Chen, S. Yoon and M. Anis, "A methodology for statistical estimation of read access yield in SRAMs," *IEEE DAC*, pp. 205-210, 2008.

[5] R. Kanj, R. Joshi and S. Nassif, "Mixture importance sampling and its application to the analysis of SRAM designs in the presence of rare failure events," *IEEE DAC*, pp. 69-72, 2006.

[6] A. Singhee and R. Rutenbar, "Statistical blockade: a novel method for very fast Monte Carlo simulation of rare circuit events, and its application," *IEEE DATE*, pp. 1-6, 2007

[7] L. Dolecek, M. Qazi, D. Shah and A. Chandrakasan, "Breaking the simulation barrier: SRAM evaluation through norm minimization," *IEEE ICCAD*, pp. 322-329, 2008.

[8] J. Wang, S. Yaldiz, X. Li and L. Pileggi, "SRAM parametric failure analysis," *IEEE DAC*, pp. 496-501, 2009.

[9] J. Jaffari and M. Anis, "Adaptive sampling for efficient failure probability analysis of SRAM cells," *IEEE ICCAD*, pp. 623-630, 2009.

[10] C. Gu and J. Roychowdhury, "An efficient, fully nonlinear, variability-aware non-Monte-Carlo yield estimation procedure with applications to SRAM cells and ring oscillators," *IEEE ASPDAC*, pp. 754-761, 2008.

[11] C. Andrieu, N. Freitas, A. Doucet and M. Jordan. "An introduction to MCMC for machine learning," *Machine Learning*, vol. 50, pp. 5-43, 2003.

[12] H. Zhang, T. Chen, M. Ting and X. Li, "Efficient design-specific worst-case corner extraction for integrated circuits," *IEEE DAC*, pp. 386-389, 2009.

[13] A. Chandrakasan, W. Bowhill and F. Fox, *Design of High-Performance Microprocessor Circuits*, Wiley-IEEE Press, 2000.

[14] A. Papoulis and S. Pillai, *Probability, Random Variables and Stochastic Processes*, McGraw-Hill, 2001.

[15] G. Fishman, *A First Course in Monte Carlo*, Duxbury Press, Oct. 2005