Parametric Yield Estimation for SRAM Cells: Concepts, Algorithms and Challenges

Fang Gong[1], Yiyu Shi[2], Hao Yu[3] and Lei He[1]

[1] *EE Department, University of California, Los Angeles, CA, USA*
[2] *ECE Department, Missouri University of Science and Technology, Rolla, MO, USA*
[3] *EEE Department, Nanyang Technological University, Singapore*

ARTICLE: Yield Estimation

Parametric Yield Estimation for SRAM Cells: Concepts, Algorithms and Challenges

Fang Gong[1], Yiyu Shi[2], Hao Yu[3] and Lei He[1]
[1] *EE Department, University of California, Los Angeles, CA, USA*
[2] *ECE Department, Missouri University of Science and Technology, Rolla, MO, USA*
[3] *EEE Department, Nanyang Technological University, Singapore*

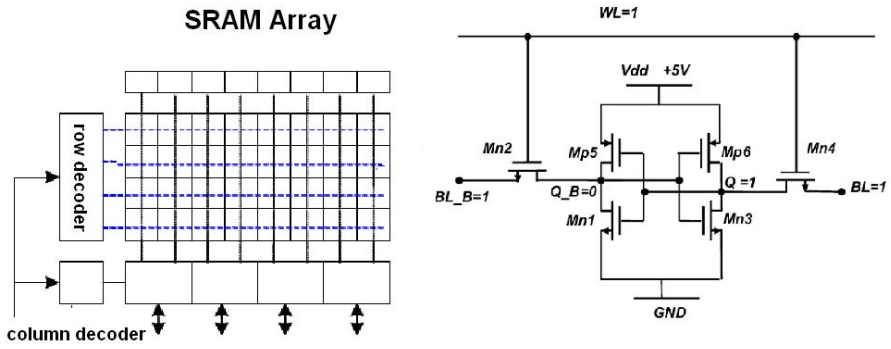
Abstract— With technology scaling down to 90nm and below, process variation has become a major challenge for both design and fabrication. Among all types of circuits, Static Random Access Memory (SRAM) is particularly vulnerable to process variation, as it contains a large number of nearly minimum-sized devices with ever-decreasing supply voltage and reduced noise margin. To determine the performance of the SRAM cell under process variation, we need to estimate its parametric yield efficiently and accurately. Existing parametric yield estimation methods can be classified into two categories: performance domain methods which require extensive Monte Carlo simulation, and parameter domain methods which require the characterization of a yield boundary defined by performance constraints without using Monte Carlo simulations. In this article, we review the pros and cons of these methods, and use a six-transistor (6T) cell as a basis for evaluation and quantitative comparison.

Index Terms— Parametric yield estimation, 6T SRAM bitcell, Monte Carlo method, parameter domain.

I. INTRODUCTION

As integrated circuits enter the nanometer era, process variation has become a major challenge for both design and fabrication. Many uncertainties can be introduced during manufacturing process steps such as lithography, chemical mechanical polishing (CMP), etching, etc. Consequently, circuit parameters such as effective channel length L_{eff} and threshold voltage V_{th} can deviate significantly from their nominal values. This in turn will cause circuit performance-determining parameters, such as maximum clock frequency and leakage power, to differ from design specifications. For example, Intel has observed 30% variation in chip frequency and 20X variation in chip leakage in 1,000 sample chips fabricated in 180nm technology [1].

Static Random Access Memory (SRAM) is among the circuits that are particularly vulnerable to process variation, as it contains a large number of nearly minimum-sized devices with ever-decreasing supply voltage and reduced noise margin [2]. These transistors constitute cells in an array structure as shown in Figure 1. Each cell is used to store one memory bit, and a typical implementation involves six transistors: the four transistors Mn1, Mn3, Mp5 and Mp6 have two stable states, i.e., either a logic '0' or '1', and the two additional access transistors Mn2 and Mn4 serve to control the access to the cell during read and write operations. The word line is used to determine whether the cell should be accessed (connected to bit line) or not, and the bit line is used to read/write the actual data from/to the cell. The quality (i.e., electrical robustness) of SRAM cells can be assessed using either the butterfly curve (Figure 2(a)) or the N-curve (Figure 2(b)). The butterfly curve can be used to measure the static noise margin (SNM) [2], while the N-curve provides information on both the read-stability and the write-ability based on static noise margin and write-trip voltage (WTV) [2]. If the width W, effective channel length L_{eff} and threshold voltage V_{th} of the transistors are altered by process variation, the noise margin, read-stability and write-ability can be affected, causing potential read/write failure. In light of this, we need to estimate the *parametric yield* of the SRAM cells, which is defined as the percentage of the cells that can function correctly and is a common measure to evaluate design robustness in the presence of process variation.



(a) (b)

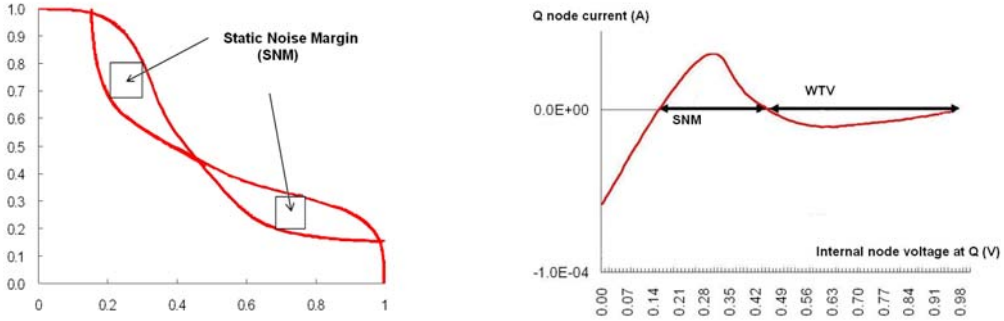
Figure 1: (a) SRAM array structure and (b) a 6T SRAM cell.

Compared with other circuits, yield estimation for SRAM cells is particularly challenging. Without redundancy implementation, the failure of any individual cell can result in the failure of the entire SRAM. If each cell fails independently with probability ρ and the total number of cells is *N*, then the failure probability of the SRAM is

$$P = 1 - (1 - \rho)^N \approx N\rho,$$

where the approximation holds for small ρ. In other words, *the failure probability for an SRAM is N times that of its individual cells*. For example, a 1M SRAM with cell failure probability 1e-9 has a

failure probability of 0.001. It is therefore critical to design cells that have extremely low failure probability. This brings significant challenges to accurate yield estimation of SRAM: typically, it is necessary to be able to predict yield higher than 0.9999, which is at the tail of the distribution (3σ-4σ for Gaussian distributions).



(a) (b)

Figure 2: (a) butterfly curve and (b) N-curve [3].

In the literature, yield estimation of SRAM cells is performed in either the *performance domain* or the *parameter domain*, as shown in Figure 3. The performance domain contains all possible performance metrics of interest (e.g., static noise margin, write-trip voltage) which can be obtained by circuit simulations over different parameter samples. By comparing against performance constraints, the parametric yield can be estimated as the percentage of successful samples among all samples. On the other hand, the parameter domain is defined as the space bounded by the min and max of all process parameters with consideration of their correlations; each combination of parameters corresponds to one performance point in the performance domain. As such, the boundary separating the success and failure regions can be located in the parameter space. If the parameters are each uniformly distributed, then the yield can be further estimated as the ratio of the hyper-volume of the success region to that of the entire parameter space.

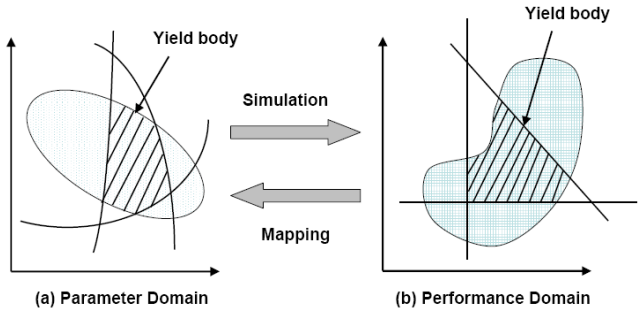


Figure 3: Yield estimation in (a) parameter domain and (b) performance domain.

To illustrate the difference between the two types of approaches, we use the example of a typical 6T SRAM cell design in 90nm technology. The parameters of the transistors are shown in Table 1. In this example, we estimate the yield based on the read failure, as read noise margin is typically a more stringent constraint than write noise margin. To read from the SRAM bitcell, both BL_B and BL are precharged to V_{dd}. We assume that *Q_B* stores '0' and *Q* stores '1'. While reading the SRAM cell (WL is charged to high), *BL* starts to discharge from V_{dd} and produces a voltage difference ΔV_{BL} between itself and *BL_B*. Because ΔV_{BL} is sensed by a sense amplifier connected to the end of the bit lines, ΔV_{BL} should be larger than a certain threshold ΔV_{th} at a specific time t_s, which is determined by the performance of the sense amplifier. In our experiment, we assume that t_s = 10ps and ΔV_{th} = 450mV.

We then introduce process variations to the threshold voltages V_{th} of Mn1 and Mn2, since these two MOSFETs are critical for read operation. We assume the variations to be uncorrelated Gaussian with a 0.1 standard-deviation-to-mean ratio[1].

	Width (um)	Length (um)	Source Region Area (um^2)	Drain Region Area (um^2)	Threshold Voltage (V)
Mn_1	0.375	0.1	0.10125	0.10125	0.2607
Mn_2	0.175	0.1	0.06125	0.06125	0.2607
Mn_3	0.375	0.1	0.10125	0.10125	0.2607
Mn_4	0.175	0.1	0.06125	0.06125	0.2607
Mp_5	0.225	0.1	0.08	0.08	-0.303
Mp_6	0.225	0.1	0.08	0.08	-0.303

Table 1: Parameters of the transistors in the 6T cell as shown in Figure 1(b).

To estimate the yield (probability of successful read operations) in *performance domain*, we first run 1,000 Monte Carlo simulations. The voltage difference ΔV_{BL} over time is depicted in Figure 3(a), with successful samples marked with blue and failure samples marked with red according to the constraint that at t_s = 10ps, and $\Delta V_{th} \geq 450$mV. As such, the yield can be approximated as the ratio of the number of blue curves over the total number of curves (908/1000). To estimate the yield in *parameter domain*, we further plot Figure 3(b), which captures the deviations in V_{th} of both Mn1 and Mn2 from their nominal values by a large range (σ=10% of nominal value). Again, the successful samples are marked with blue, and the failure samples are marked with red. For purposes of this illustration, we have converted the Gaussian distributed samples to uniformly distributed ones so that the reader can directly infer the yield from the ratio of the red area to the entire area. The yield can also be calculated as the ratio of the number of blue points to the total number of samples (908/1000). It should be anticipated that the yield calculated from the parameter domain should be identical with that from the performance domain, given the same design and parameter variation. While it seems that there is no difference between those two methods, as Monte Carlo simulations are used for both, we emphasize that for actual parameter domain methods, the boundary can be obtained without Monte Carlo simulations. This will be detailed in Section III.

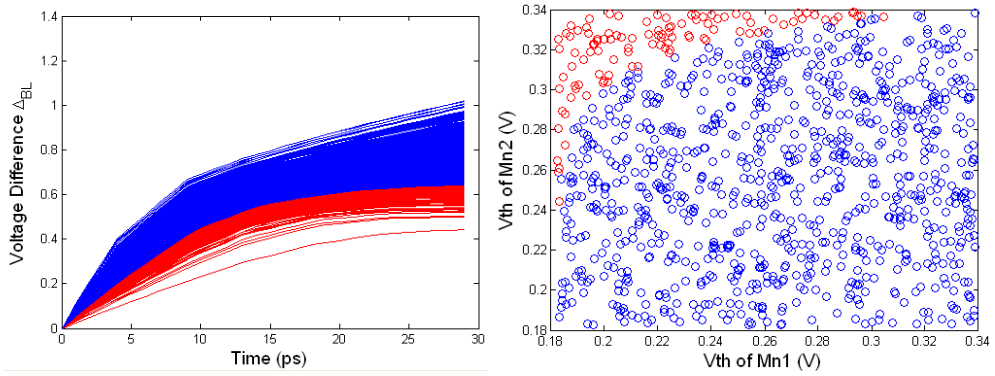


Figure 4: Yield estimation using Monte Carlo method in (a) performance domain and (b) parameter domain for a 6T SRAM cell (908 success points and 92 failure points).

[1] In actual applications, these parameters are correlated, and can be decorrelated with independent component analysis (ICA) (for non-Gaussian distributions) or principal component analysis (PCA) (for Gaussian distributions).

II. PERFORMANCE DOMAIN YIELD ESTIMATION

In this section, we discuss the yield estimation methods in performance domain, which mainly involve Monte Carlo methods and their derivations. For purposes of illustration, we set the yield of the SRAM cell to around 90% to generate all the figures in this section. A quantitative comparison between different performance domain methods is presented at the end of the section, where we set the yield to 99.9% to more closely approximate actual contexts.

A. Direct Monte Carlo

One straightforward way to estimate the yield is to perform *Monte Carlo* simulations in the performance domain. The direct Monte Carlo method [4, 5] usually involves hundreds of thousands samplings and simulations, especially when analytical solutions of stochastic problems are not available. In general, Monte Carlo approaches first perform sampling within the entire parameter domain according to the probability distributions $p(x)$. Then, circuit simulation is conducted with each set of sampled parameters $X_i \sim p(x)$ to obtain a *performance merit* value $g(X_i)$. Accordingly, Monte Carlo can estimate the expectation of performance merit

$$I(g) = \int g(x)\,p(x)dx$$

$$\text{with} \qquad I(g) \approx \frac{1}{n}\sum_{i=1}^{n} g(X_i).$$

Moreover, with given performance constraints, the yield can be estimated as the percentage of samplings with successful or acceptable performance.

The advantages of the direct Monte Carlo method are its simplicity and generality; it can be applied to arbitrary distributions of parameters and performance functions without any *a priori* information. On the other hand, direct Monte Carlo is very time-consuming to achieve high accuracy, since its convergence rate to the exact value is only $\mathrm{O}\left(1/\sqrt{N}\right)$, where *N* is the total number of samples or simulations. Therefore, direct Monte Carlo is not suitable for practical yield estimation.

B. Quasi-Monte Carlo

An alternative to the direct Monte Carlo approach is *quasi-Monte Carlo* (QMC) [6], which uses quasi-random sequences rather than random samplings. QMC starts with the generation of quasi-random numbers (or representative samples), such as Faure (1982), Neiderreiter (1987), Sobol (1967) or Halton (1960) sequences. It then converts the samples following those specific distributions to ones following the desired distribution. For example, the Sobol sequence follows a uniform distribution $u \sim U(0,1)$ and can be converted to Gaussian distribution using

$$x = F_X^{-1}(u)$$

where F_X^{-1} is the inverse cumulative distribution function (inverse CDF) of the Gaussian distribution. The resultant sequence x follows the Gaussian distribution. Circuit simulation can then be performed with each sampling of parameters to obtain performance merit values, and the yield is estimated as the percentage of successful samples.

Note that quasi-random numbers are deterministic samples rather than pure random numbers. Thus, QMC can cover the entire parameter space evenly with fewer samplings – and can therefore potentially improve both accuracy and efficiency – compared with direct Monte Carlo. For comparison, we apply QMC to the same example in Figure 4 to estimate the yield of the SRAM cell considering read failure. All settings remain the same, and the results in the performance and parameter domains

are shown in Figures 5(a) and 5(b), respectively. As in Figure 4, we convert the samples to follow uniform distributions so that the ratio of the blue area to the entire area represents the yield. It can be seen that QMC can achieve similar yield estimation (180/200) with only 200 samples; compared with 908/1000 in Figure 4, the relative error is about 0.9%. For the same number of samples, direct Monte Carlo has larger error in Figures 5(c) and 5(d) (177/200). The relative error is about 2.5%.

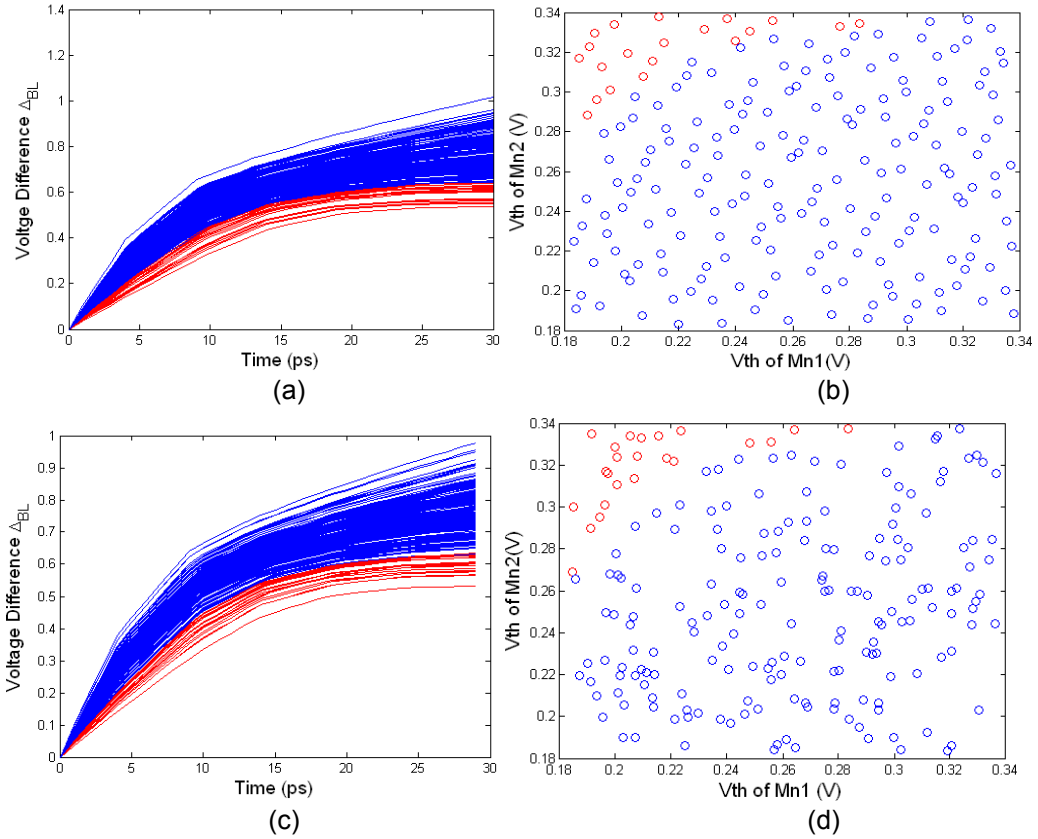


Figure 5: Yield estimation with quasi-Monte Carlo in (a) performance domain and (b) parameter domain for a 6T SRAM cell (180 success points and 20 failure points), and with direct Monte Carlo in (c) performance domain and (d) parameter domain (177 success points and 23 failure points).

The convergence rate of QMC can be $O(1/N)$ in optimal cases, much faster than that of direct Monte Carlo. However, the upper bound of estimation error (or the worst-case error) for multi-dimensional QMC is $O(\ln(N^d)/N)$ where *d* is the number of dimensions [7]. Thus the performance of QMC can decrease with the dimension.

C. *Importance Sampling*

Even though quasi-Monte Carlo can cover the entire parameter space evenly with fewer samplings, it is still possible to miss failure regions that are very small, and hence obtain misleading yield estimates. To avoid such cases, *importance sampling* (IS) has been proposed to estimate the SRAM yield [8] by shifting the sampling distribution to the failure region as shown in Figure 6 [8].

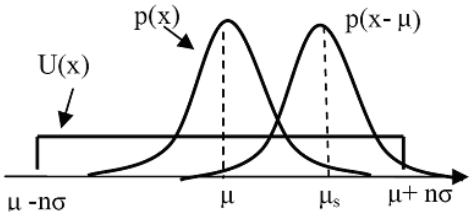


Figure 6: Shifting sampling distribution in Importance Sampling [8]

For the purpose of illustration, we take $p(x)$ as the probability density function (PDF) of variable parameter distribution, and assume that the failure region is located at the right tail of $p(x)$ around μ_s. The Monte Carlo method will generate more random samples around μ rather than μ_s, which does not improve the accuracy of yield estimation. To cure this, importance sampling tries to find a distorted sampling function $g(x) = p(x-\mu)$ which can increase the probability of sampling within the failure region. Readers are referred to [8] for more details.

When importance sampling is applied in the same context as in Figures 4 and 5, we can shift the sampling function around the failure region. As shown in Figure 7(a) and 7(b), more samples are scattered around the failure region and separation boundary so as to achieve higher accuracy. Since the right-down corner is a "safe" region wherein all parameters can lead to successful performance, fewer samples are needed to achieve the desired accuracy. In this case, 400 samples are used to achieve 0.4% relative error. For the same number of samples, quasi-Monte Carlo yields larger error in Figure 7(c) and Figure 7(d) (360/400). The relative error is about 0.8%.

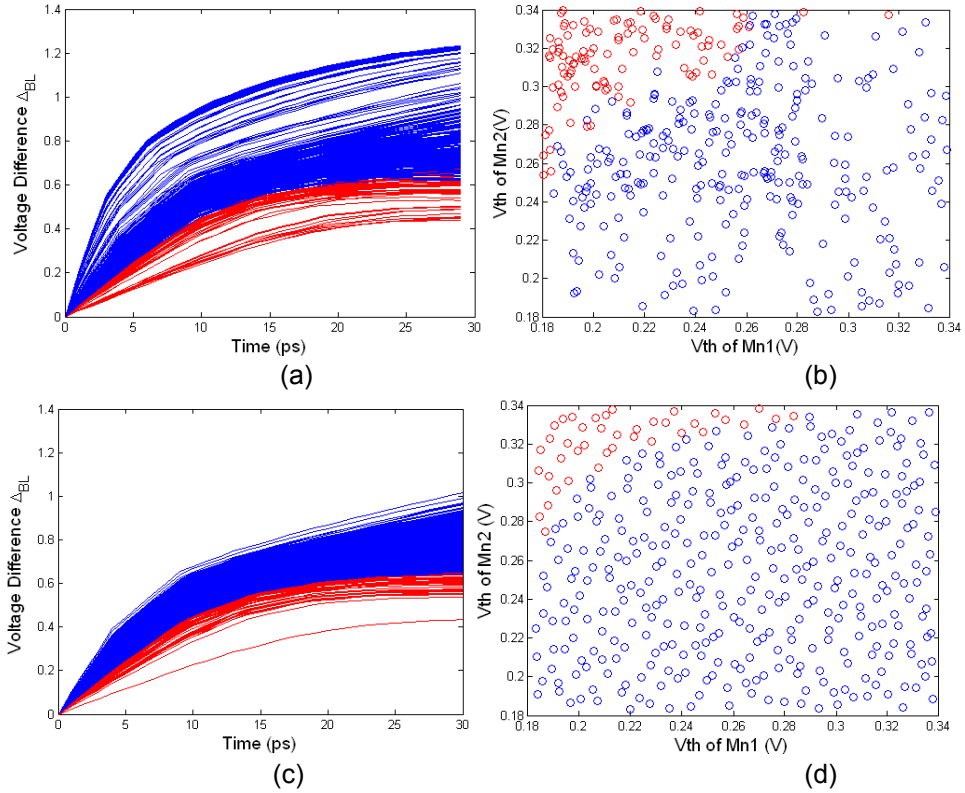


Figure 7: Yield estimation with importance sampling in (a) performance domain and (b) parameter domain for 6T SRAM cell (463 success points and 49 failure points after conversion), and with quasi-Monte Carlo in (c) performance domain and (d) parameter domain (360 success points and 40 failure points).

Importance sampling can reduce the number of samples required to achieve a desired accuracy, especially in the case where the failure region is small for rare failure events. However, it is always challenging to obtain an optimal sampling distribution $g(x)$ efficiently, since this depends on the actual distribution of the performance merit and is unknown beforehand.

In the above examples, we have set the yield to 90% for purposes of illustration. To indicate how the various methods might behave in a more realistic context, Table 2 shows the number of circuit simulations required by different methods when the yield is close to 99.9%. We can see that the previous conclusions can still hold: quasi-Monte Carlo can reduce the number of samplings by covering the entire space with deterministic sequences, and importance sampling can further reduce the number of samplings required.

# of parameters	Monte Carlo	Quasi-MC	Importance Sampling
1	1.0e+5 (1X)	2.5e+4 (3.8X)	1.5e+3 (67X)
2	1.8e+5 (1X)	4.6e+4 (3.9X)	2.5e+3 (72X)
3	3.5e+5 (1X)	9.6e+4 (3.6X)	5.8e+3 (61X)
4	3.5e+5 (1X)	9.6e+4 (3.6X)	5.8e+3 (61X)
5	3.5e+5 (1X)	9.6e+4 (3.6X)	5.8e+3 (61X)
6	3.5e+5 (1X)	9.6e+4 (3.6X)	5.8e+3 (61X)

Table 2: Number of circuit simulations required to achieve the same accuracy for SRAM cell yield estimation using performance-domain methods.

From the table, we see that the number of simulations required remains the same for more than three parameters. The reason is as follows: among all possible sources of process variations, the threshold voltage V_{th} is dominant due to random dopant effect [9], and the effects of other parameters are significantly masked. In addition, even though all six transistors are subject to V_{th} variations, not all of them are critical to the performance metric under study. For the read failure in this example, only V_{th} variations of Mn1, Mn2 and Mp6 will have significant impact on the yield.

III. PARAMETER DOMAIN YIELD ESTIMATION

To avoid the large number of simulations required for performance domain estimation, several approaches have been proposed to estimate the yield in parameter domain. Here we discuss two state-of-the-art techniques, *nonlinear surface sampling* and the *surface-point finding strategy* method.

A. Nonlinear Surface Sampling

Nonlinear surface sampling [10, 11] considers non-Monte-Carlo parametric yield estimation with fully nonlinear performance constraints. The key idea is to locate the yield boundary in the parameter domain, which can be approximated by connected boundary points as shown in Figure 8(b). For example, the performance surface can be defined with each point on the surface corresponding to a sampling point in the parameter domain, as shown in Figure 8(a). With the given performance constraint, the surface can be divided into success and failure regions, and the separation boundary projected into parameter domain as shown in Figure 8(b). As such, the yield can be estimated by the area (or, volume) ratio of the bounded region to that of the entire parameter domain[2].

[2] For techniques in parameter domain, the variable parameters are assumed to follow uniform distributions, because the yield estimation makes use of the area ratio of success region and entire parameter domain. Other distributions can be converted into uniform distribution based on the cumulative distribution function (CDF).

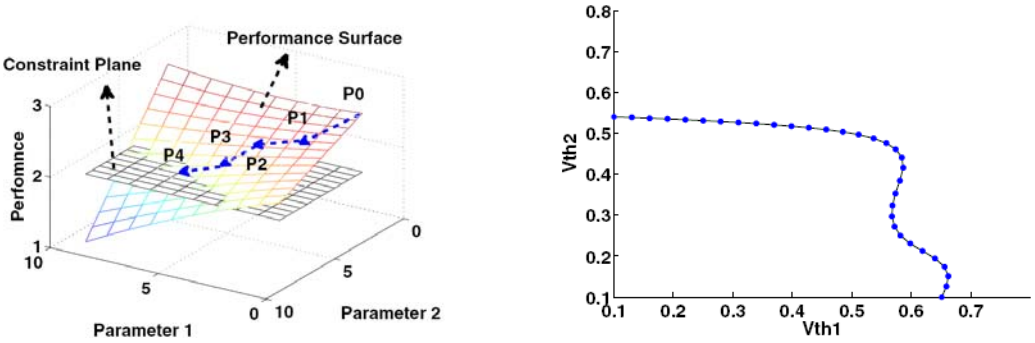


Figure 8: (a) Performance surface over parameter domain, and (b) the yield boundary in the parameter domain [11].

To locate a point on the yield boundary, nonlinear surface sampling starts with the nominal parameters P0, and searches along the tangent of the performance surface to approach the surface boundary points P4. As such, nonlinear surface sampling searches along the path as shown in Figure 7(a), and performs circuit simulations with the sampling parameters corresponding to P1, P2, P3 and P4. Sensitivity analysis is used to calculate the derivative of the performance merit function f_m with respect to each parameter p_i, so as to obtain the tangent direction at each sampling point.

This method can provide high accuracy without resorting to Monte Carlo. However, surface point samplings and corresponding expensive simulations are still required to locate each point, so further efficiency improvements are still needed.

B. Surface-Point Finding Strategy

Another approach, the so-called surface-point finding strategy [12], is a truly non-Monte-Carlo method because it can locate each boundary point using "one-shot" simulation without sampling. Per our discussion so far, previous methods mostly share a common yield estimation framework: generating samples, performing simulations at all samples, and then comparing with the given performance constraints to estimate the yield.

The work in [12] breaks the traditional framework by switching the role of performance constraints $H(\gamma_p; f_m)$ and variable parameters γ_p, where f_m is the performance merit function of interest. This method treats the parameter γ_p as an unknown, while introducing $H(\gamma_p; f_m)$ as an extra equation into the simulation system. It first combines the differential algebraic equation (DAE)

$$\frac{d}{dt}q(x(t)) + f(x(t)) + b = 0$$

with the performance constraint $H(\gamma_p; f_m)$ as

$$\begin{cases} \dfrac{d}{dt}q(x(t)) + f(x(t)) + b = 0 \\ H(\gamma_p; f_m) = f_m(\gamma_p) - f_{worst} = 0 \end{cases}$$

The objective of the simulation is to find the value of γ_p that can satisfy the performance constraint $H(\gamma_p; f_m)$ exactly. Therefore, the simulator needs to solve one augmented nonlinear system with Newton's method. With the obtained parameter values, the points on the surface

boundary can be located in the parameter domain, and the parametric yield can be evaluated accordingly. Therefore, the method provides both high accuracy and better efficiency, at the same time. Experiments show that this method can achieve up to 519X speedup over direct Monte Carlo and 4.7X over nonlinear surface sampling.

We must point out that parameter domain methods cannot handle – or at least are extremely challenged by – high-dimensional problems (more than 3 variables). This is because they approximate yield using the ratio of area or volume of the success region to that of the entire parameter space. It is challenging to calculate the hyper-volume of the success region in the high-dimensional space induced by multiple parameters of variation.

As for the yield optimization task, it usually estimates the yield of the initial design and then tries to improve yield by moving the design point (or tuning the design parameters in nominal case), with several iterations of yield estimation being required. For this purpose, Monte Carlo methods are not suitable, if not completely infeasible, since they usually need to conduct a huge number of (typically expensive) circuit simulations at each design point. Therefore, yield optimization methods should mainly deploy efficient parameter-domain techniques, such as those discussed in this section, to achieve efficiency.

IV.UPCOMING CHALLENGES AND CONCLUSIONS

In the foregoing, we have introduced the parametric yield estimation problem and discussed existing approaches in the context of SRAM yield. SRAM yield estimation still faces a number of challenges. First, none of the existing methods can be embedded into the yield optimization framework while retaining both efficiency and accuracy. Performance-domain methods cannot provide any sensitivity information to guide the optimization, and also require a large number of simulations. On the other hand, parameter-domain methods cannot efficiently handle multiple parameters and constraints.

Second, with the yield estimation of one SRAM cell, efficient optimization techniques are required to optimize the entire SRAM. Variations within each cell are spatially correlated, and accordingly the yields of each cell in an SRAM array are also correlated. It is still open how to answer such questions as: how many redundant cells will be enough to achieve a specific yield rate, how should the cells be tuned, and where they should be placed. This may involve joint architecture- and circuit-level optimization.

REFERENCES

[1] S. Borkar, T. Karnik, S. Narendra, J. Tschanz, A. Keshavarzi and V. De, “Parameter Variations and Impact on Circuits and Microarchitecture”, *Proc. DAC*, 2003, pp. 338-342.

[2] R. Heald and P. Wang, “Variability in Sub-100nm SRAM Designs” *Proc. ICCAD*, 2004, pp. 347-352.

[3] R. J. Greenway, K. Jeong, A. B. Kahng, C.-H. Park and J. S. Petersen, “32nm 1-D Regular Pitch SRAM Bitcell Design for Interference-Assisted Lithography”, *Proc. SPIE BACUS Symposium on Photomask Technology and Management,* 2008.

[4] N. Metropolis and S. Ulam, “The Monte Carlo Method”, *J. American Statistical Association* 44(247) (1949), pp. 335-341.

[5] J. F. Swidzinski and K. Chang. “Nonlinear Statistical Modeling and Yield Estimation Technique for Use in Monte Carlo Simulations”, *IEEE Trans. on Microwave Theory and Techniques* 48(12) (2000), pp. 2316–2324.

[6] H. Niederreiter, *Random Number Generation and Quasi-Monte Carlo Methods*, Society for Industrial and Applied Mathematics, 1992.

[7] P. Bratley, B. L. Fox and H. Niederreiter, "Implementation and Tests of Low-Discrepancy Sequences", *ACM Trans. on Modeling and Computer Simulation* 2(3) (1992), pp. 195-213.

[8] 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", *Proc. DAC*, 2006, pp. 69-72.

[9] P. Girard, A. Bosio, L. Dilillo, S. Pravossoudovitch and A. Virazel, *Advanced Test Methods For SRAMs: Effective Solutions For Dynamic Fault Detection In Nanoscaled Technologies*, Springer, 2009.

[10] S. Srivastava and J. Roychowdhury, "Rapid Estimation of the Probability of SRAM Failure Due to MOS Threshold Variations", *Proc. Custom Integrated Circuits Conf.*, 2007.

[11] 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", *Proc. ASP-DAC*, 2008.

[12] F. Gong, H. Yu, Y. Shi, D. Kim, J. Ren and L. He, "QuickYield: An Efficient Global-Search Based Parametric Yield Estimation With Performance Constraints", to appear in *Proc. DAC*, 2010.