QuickYield: An Efficient Global-Search Based Parametric Yield Estimation with Performance Constraints

Fang Gong[1], Hao Yu[2], Yiyu Shi[1], Daesoo Kim[1], Junyan Ren[3], and Lei He[1]

[1]University of California, Los Angeles, Los Angeles, US

[2]Nanyang Technological University, Singapore

[3]State Key Lab of ASIC, Fudan University, Shanghai, China

ABSTRACT

With technology scaling down to 90nm and below, many yield-driven design and optimization methodologies have been proposed to cope with the prominent process variation and to increase the yield. A critical issue that affects the efficiency of those methods is to estimate the yield when given design parameters under variations. Existing methods either use Monte Carlo method in performance domain where thousands of simulations are required, or use local search in parameter domain where a number of simulations are required to characterize the point on the yield boundary defined by performance constraints. To improve efficiency, in this paper we propose QuickYield, a yield surface boundary determination by surface-point finding and global-search. Experiments on a number of different circuits show that for the same accuracy, QuickYield is up to 519X faster compared with the Monte Carlo approach, and up to 4.7X faster compared with YENSS, the fastest approach reported in literature.

Categories and Subject Descriptors: B.7.[Hardware]: - Integrated Circuits-Design Aids
General Terms: Algorithms, Performance
Keywords: Parametric yield, Circuit simulation

1. INTRODUCTION

As integrated circuits enter into the nanometer era, process variation has become a major challenge for both design and fabrication [1, 2, 3, 4, 5, 6]. Circuit parameters such as effective channel length and threshold voltage of transistors can deviate significantly from their nominal values specified by designers. There are many uncertainties from the manufacturing process such as lithography, chemical mechanical polishing (CMP), etching and so on. The circuit performance such as delay and output swing may differ from the design specification under the nominal condition. Parametric yield, defined as the percentage of circuits that can work under a performance metric within the acceptable region, is a common measure to evaluate the design robustness in the presence of process variation.

DAC 2010, June 13-18, 2010, Anaheim, California, USA.

In general, the parametric yield can be estimated in either the performance domain or the parameter domain, as shown in Fig.(1). Monte Carlo simulation is commonly used for yield estimation in the performance domain. With the probability distribution of variable parameters, it first generates tens of thousands of random-sampling points in the parameter space, and then performs circuit simulations at each sampling point to evaluate the performance metric of interest. Within the obtained performance, Monte Carlo method screens out those missing the given performance constraints, and identify the acceptable or successful samplings. In this way, by dividing the number of successful samplings by that of the total sampling points, one can obtain the yield rate. The advantage of Monte Carlo simulation is its simplicity and generality. However, the Monte Carlo method is very time-consuming. The lengthy simulation at each sampling point limits its application for the yield estimation and hence a fast analysis is required.

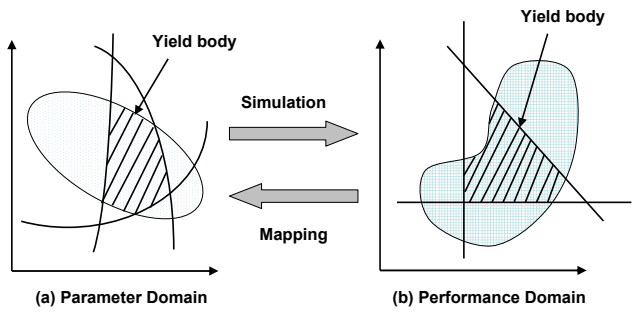


Figure 1: Yield estimation in (a) parameter domain and (b) performance domain.

Several works [1, 2, 3] have been proposed to utilize the *surface boundary*, which separates the success and failure regions to estimate the yield rate. These methods avoid the massive sampling simulations by only using points on the surface boundary. The approach in [1] uses linear constraints to approximate the yield surface. It has two limitations: first, the linear functions cannot approximate the yield body with a good accuracy, especially for yields defined by nonlinear performance constraints. Second, this method is based on the assumption that the yield body needs to be convex, which does not hold in general. To address these issues, approaches in [2] and [3] derive a nonlinear

procedure, called YENSS, which can handle nonlinear performance constraints as well as non-convex yield body. It starts from the nominal performance space, and searches along the tangent of the surface boundary to approach the surface boundary point, which is called *local search* in this paper. This method can provide the good accuracy without using the Monte Carlo method. However, expensive simulations are still required to locate each point on the surface boundary due to local searching.

In this paper, we propose an efficient parametric yield calculation method, called *QuickYield*. QuickYield applies a surface-point finding together with a global-search to locate points on the surface boundary. We first include performance constraints into the differential algebra equation (DAE) that describes the circuit and build an augmented equation system. The points on the surface boundary in the parameter domain can be determined by solving the augmented system equation. Then yield can be estimated from the obtained surface boundary. QuickYield is able to benefit problems where simulation is extremely expensive. Experimental results show that for the same accuracy, QuickYield is up to 519X faster compared with the Monte Carlo approach, and up to 4.7X faster compared with YENSS, the fastest approach reported in the literature.

The rest of the paper is organized in the following manner. We first review the background of the yield estimation in Section 2. In Section 3, we present the details about yield estimation in QuickYield. We present experimental results in Section 4 and conclude the paper in Section 5.

2. PRELIMINARIES

Generally, the variational parameters with performance constraint that defines the acceptable region can form a hyper-volume with respect to the nominal point in the parameter space, as shown in the Fig(2). The shaded region is the "safe" region where parameters can result in the acceptable performance. The parametric yield estimation, accordingly, is to calculate the ratio of the volume of the safe region to that of the entire space bounded by the min and max of all parameters. Here, all parameters are assumed independent after decomposing the correlation.

One direct approach to find the yield is to determine all the points in the hyper-volume. However, it is expensive and unnecessary, because the points on the hyper-surface boundary can provide enough information to calculate the volume of the yield body and thus the yield rate [3]. Therefore, it is desirable to directly identify the hyper-surface boundary (or called surface boundary).

The hyper-surface that encloses the acceptable or safe region can be mathematically described by the performance constraint as [1]

$$H(\gamma_p; f_m) = f_m(\gamma_p) - f_{worst} = 0, \quad (1)$$

where γ_p denotes the parameters subject to the process variations. Depending on the specific circuit problem under study, γ_p can be different parameters, such as the threshold voltage, width, length, load capacitance and etc. $f_m(\gamma_p)$ is the performance metric corresponding to γ_p, which can be swing, gate delay, oscillator frequency and so on. They can be obtained by SPICE simulations. f_{worst} is the worst-case performance that can be accepted. Unfortunately, in most cases it is either prohibitively expensive or technically impossible to obtain an explicit expression for $f_m(\gamma_p)$.

[1] For simplicity of our presentation and similar to [3], from now on we restrict our discussion to the simple case with only one performance constraint. It is understood, however, that our method can also be applied to multiple constraints scenario by handling one constraint at a time.

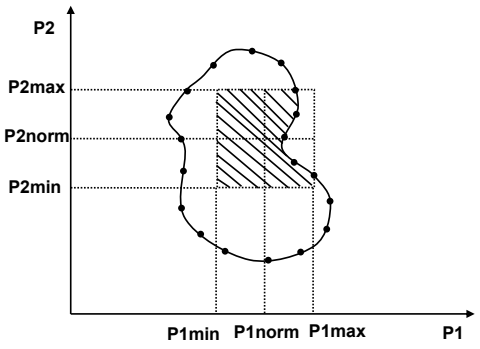
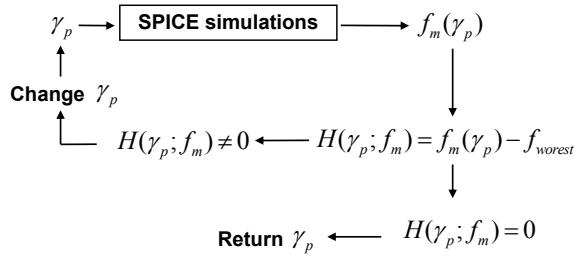
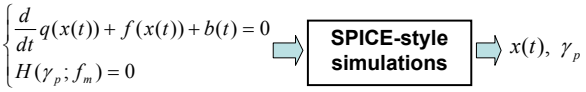


Figure 2: Performance Constraints defines hyper-volume in parameter space

There are some alternative ways proposed to obtain this surface. As shown in Fig.(3(a)), existing methods [1, 2, 3] start with an initial γ_p, and perform SPICE simulation with γ_p to get performance $f_m(\gamma_p)$. Then, the performance constraint $H(\gamma_p; f_m)$ is checked with the obtained f_m. If γ_p cannot satisfy the constraints, a new γ_p is selected and the above procedure is repeated until the constraint is satisfied. In this way, the final calculated parameter lies on the surface boundary. However, those methods all require quite a number of simulations to to identify points on the surface. The efficiency should be further improved.



(a) Existing methods



(b) QuickYield

Figure 3: Comparison of boundary surface finding methods by existing methods and QuickYield

3. YIELD ESTIMATION

In this section, we present details for the yield estimation in QuickYield. Without loss of generality, we assume that all the parameters are uniformly distributed. Non-uniform distribution and correlated parameters can be mapped into uniform parameter domain with pre-scale and PCA techniques[2, 3].

3.1 Algorithm Overview

In order to improve the efficiency when locating the yield surface boundary, the "surface-point finding" is developed to locate surface boundary points with the aid of a *global search*, which is in contrast to the approach in YENSS [3]. YENSS performs the search of surface point mainly based on a *local search*, where the tangent direction is calculated during each step of searching. The determination of the tangent direction requires to calculate the sensitivity $\partial f_m/\partial \vec{p}$, which is expensive for each step of searching.

Our approach switches the role of performance constraint $H(\gamma_p; f_m)$ and variable parameters γ_p, which is shown in Fig.(3(b)). We treat the γ_p as an unknown, while introducing $H(\gamma_p; f_m)$ as a extra equation into the simulation environment. 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 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. Moreover, during the searching, we develop a strategy by the global-search, which avoids the expensive local search used in [3] yet with a high accuracy. In this way, we can develop an efficient yield estimation algorithm, called *QuickYield*. The flow is presented below.

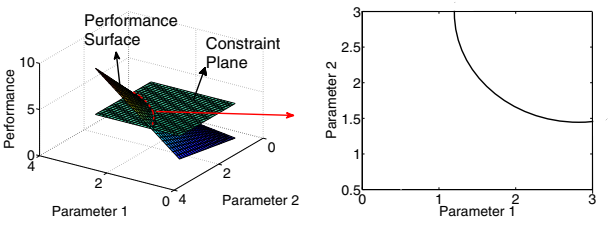
Algorithm 1 Flow of QuickYield

```
/* Setup */
Input circuit constructs, nominal parameters and performance constraints;
Build the augmented equation system with performance constraint equation;

/* Initial Yield Approximation */
Find the intersection of surface boundary with each axis using Surface-point Finding Strategy.
Initial yield approximation can be calculated with the yield volume defined by these intersections.

while Yield estimation can be refined do
    /* Surface-point Finding */
    Solve the augmented system to obtain the parameter values that fix a exact point on the surface boundary.
    /* Yield Refinement */
    Refine the yield estimate from the additional points on the surface boundary.
end while
```

3.2 Surface-point Finding Strategy

For the purpose of illustration, we first consider two uncorrelated and uniform-distributed parameters p_1 and p_2 as shown in ig.(2). Assuming that the performance constraint can be successfully added at one pair of nominal points (p_{1_norm},p_{2_norm}), there exists a "safe" region inside the region bounded by (p_{1_min},p_{1_max}) and (p_{2_min},p_{2_max}). The "safe" region thereby can be viewed as a nonlinear surface boundary separating the success and failure regions in the parameter domain. As such, the yield can be approximated by the area ratio of shaded region to the entire region bounded by min and max values of two parameters.

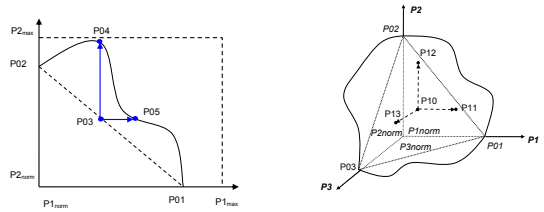
We elaborate the need to find surface points further from a geometrical perspective as shown in Fig.(4(a)). The equation system determines a performance surface in the parameter domain. The performance constraint $H(\gamma_p; f_m)$ further locates a constraint plane $f_m(\gamma_p) = f_{worst}$ [2] . It is clear that the yield surface boundary in the parameter domain, shown in Fig.(4(b)), is the intersection boundary of these two surfaces. Therefore, by solving the equation system together with the performance constraint, one can obtain their intersection boundary, which in fact separates the success and failure regions in the parameter domain.



(a) Rotated View (b) Topview

Figure 4: (a)rotated view and (b) topview of the intersection boundary of performance surface and constraint plane.

For this sake, we can first locate boundary points on the nonlinear yield surface boundary as many as possible, and then the exact surface boundary can be approximated by connecting those identified surface points with the further refinement beginning from the centroid point of those exsisiting surface points. Such a surface-point finding strategy is called global-search in this paper, which is different from the local-search in YENSS by computing tangent direction at each point.



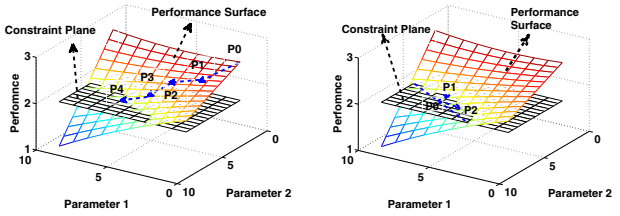
(a) 2D example (b) 3D example

Figure 5: Surface-point Finding Strategy

Let's illustrate the global-search by an example as follows. In order to find surface points, QuickYield first calculates the intersection points at each parameter axis: P_{01}, P_{02} shown in Fig.(5(a)). Then, it fixes one coordinate of P_{01} or P_{02} at its nominal values, while introducing the other coordinate as the extra unknown γ_p as shown in Fig.(3(b)). By connecting these points, an initial approximation to the yield body can be generated. To have an accurate approximation of the yield body, additional surface points such as P_{04} and P_{05} in between can be further found by fixing one parameter of of the middle point (P_{03}) of (P_{01}, P_{02}) and solving for the other parameter. This procedure is repeated to find more surface points to refine the yield estimation. Note that

[2]For the simplicity of illustration, a linear performance constraint is used in Fig.(4(a)). But QuickYield can handle fully nonlinear performance constraints in general.

this approach can be generalized to the case when there are multiple parameters. As shown in Fig.(5(b)), there exists a hyper-surface by connecting all intersections on axes, and hence the centroid point of the hyper-surface can be used to find additional surface points with the similar procedure in 2D problem. This will be addressed in the future work.



(a) YENSS[3]. (b) QuickYield

Figure 6: Comparison of Surface-point Finding Strategy

It is worthwhile to point out that our QuickYield uses the global-search, different from YENSS [3]. As shown in Fig.(6(b))), P_0 is the initial middle point of the line connecting two axis intersection points. Obviously, its determination needs no simulation. The identified surface points, such as P_1 and P_2, only require one simulation in QuickYield. Compared to YENSS as shown in Fig.(6(a))), YENSS involves more runs of simulations than QuickYield to locate a single surface point. This is because YENSS starts from the nominal design point and locally searches along the tangent direction of performance surface. The determination of the tangent direction is requires to calculate the sensitivity $\partial f_m/\partial \vec{p}$, which is expensive. In contrast, the surface-point finding strategy in QuickYield method is more efficient since it searches in global fashion by starting from the middle of the identified points. The efficiency of QuickYield is also demonstrated in experiments.

3.3 Surface-point Finding Method

To determine the surface point numerically, we show the application of the parameter finding used for device optimization [7, 8] in the content of yield estimation, i.e., the surface-point finding.

We start from differential algebra equation (DAE) in (2)

$$\frac{d}{dt}q(x(t)) + f(x(t)) + b = 0 \tag{2}$$

where $x(t)$ are unknown state variables.

First, QuickYield leverages the DAE with one performance constraint $H(\gamma_p; f_m)$ into one augmented system equation (3). As such, we have a new system with $x(t)$ and γ_p as the unknowns.

$$\begin{cases} \frac{d}{dt}q(x(t),\gamma_p) + f(x(t),\gamma_p) + b(t) = 0 \\ H(x(t),\gamma_p) = f_m(\gamma_p) - f_{worst} = 0 \end{cases} \tag{3}$$

We can use $F(x(t),\gamma_p)$ to denote the left hand side of equation (3), and $F : \mathbb{R}^m \times \mathbb{R}$ to rewrite the equations as $F(x(t),\gamma_p) = 0$. The system consists of a number of nonlinear equations, and can be solved with Newton-Raphson iterations.

The Newton-Raphson iteration[3] requires the Jacobian matrix $J(X) = \partial F/\partial X$, where $X = [x^T; \gamma_p]$ is the vector of the unknowns. For DC analysis, the Jacobian matrix becomes simple as:

$$J(X) = \begin{bmatrix} \partial f/\partial x & \partial f/\partial \gamma_p + b \\ \partial H(\gamma_p; x)/\partial x & \partial H(\gamma_p; x)/\partial \gamma_p \end{bmatrix} \tag{4}$$

Note that for the periodical circuits such as ring oscillators, the periodical steady state analysis (PSS) is required to find their periods T [9, 7, 8]. Similarly, one augmented system for PSS analysis can be built in discrete time domain with the finite difference method as [8, 7]:

$$\begin{bmatrix} \frac{1}{h}(q_1 - q_n) + f_1 + b \\ \frac{1}{h}(q_2 - q_1) + f_2 + b \\ \vdots \\ \frac{1}{h}(q_n - q_{n-1}) + f_n + b \\ H(x^T; \gamma_p) \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix} \tag{5}$$

Accordingly, the Jacobian matrix J_{fd} can be expressed as equation (6) at the top of the next page, where $C = \partial q/\partial x$, and $G = \partial f/\partial x$. When Newton-Raphson iteration converges, it returns the state-variable $x(t)$ and the variational parameter γ_p which resides on the boundary of the yield body in the parameter domain.

3.4 Surface-point Based Yield Estimation

To estimate the yield rate, we first employ an analytical formula [3] to compute the volume of the parallelotope and the simplex defined by the intersection points of the boundary surface with each parameter axis. We then use it as the initial approximation to the yield. When additional points on the yield boundary identified, QuickYield will refine the yield volume approximation by adding or subtracting the hyper-volume increments. Accordingly, the yield can be simply obtained if one divides the calculated volume by that of the parameter space.

4. EXPERIMENTAL RESULTS

We have implemented QuickYield in a Matlab-based circuit simulator, and all experiments are carried out on a Linux server with a 2.4GHz Xeon processor and 4GB memory. We use a Schmitt Trigger and a three-stage ring oscillator to compare the accuracy and efficiency of QuickYield with Monte Carlo and YENSS [3]. As an illustration, we use the widths of the MOSFETs as parameters γ_p subject to process variation, but QuickYield can handle other variations, such as threshold voltage, load capacitance and channel length as well.

4.1 Schmitt Trigger

We first use a Schimitt Trigger (shown in Fig.(7) to verify the accuracy and efficiency of QuickYield combined with DC analysis. In this case, we consider the lower swiching threshold V_{TL} as the performance metric. We consider the channel widths of NMOS W_{n1} and PMOS W_{p2} as variational parameters γ_p, which have 30% variations from their nominal values. As such, V_{TL} can differ from nominal value, which change the lower switching threshold V_{TL}. The performance constraint is given as: when the input V_{TL} is at 0.4V and the output is initially set to Vdd, the output V_{OUH} should be greater than 1.7V.

First, 30% random variations are introduced to W_{p2} and W_{n1} separately. The result from 6000 Monte Carlo DC

[3]For the simplicity of illustration, from now on we omit the iteration index k in our discussion about Newton-Raphson iteration.

$$J_{fd} = \begin{bmatrix} \frac{1}{h}C_1 + G_1 & & -\frac{1}{h}C_n & \frac{1}{h}(\frac{\partial q_1}{\partial \gamma_p} - \frac{\partial q_n}{\partial \gamma_p}) + \frac{\partial f_1}{\partial \gamma_p} + \frac{\partial b_1}{\partial \gamma_p} \\ \ddots & \ddots & & \vdots \\ & -\frac{1}{h}C_{n-1} & \frac{1}{h}C_n + G_n & \frac{1}{h}(\frac{\partial q_n}{\partial \gamma_p} - \frac{\partial q_{n-1}}{\partial \gamma_p}) + \frac{\partial f_n}{\partial \gamma_p} + \frac{\partial b_n}{\partial \gamma_p} \\ \frac{\partial H}{\partial x_1} & \cdots & \frac{\partial H}{\partial x_n} & \frac{\partial H}{\partial \gamma_p} \end{bmatrix} \quad (6)$$

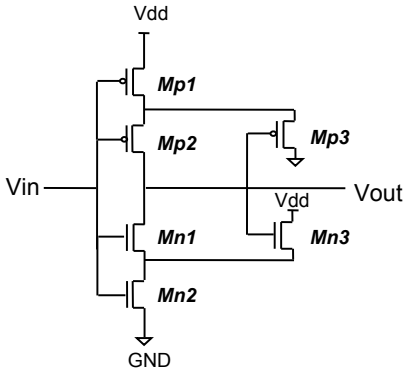
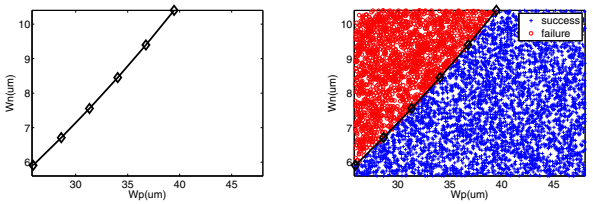


Figure 7: Schmitt Trigger design

simulations is shown as Fig.(8(b)) in the parameter space (W_p, W_n). The success and fail regions are separated by one boundary curve, and the hyper-volume can be estimated to calculate the yield.

Then we perform the QuickYield on the Schmitt Trigger example to find the boundaries, and result is shown in Fig.(8(a)). By comparing with the Monte Carlo results in Fig.(8(b)), we can observe that QuickYield can obtain the same boundary curve as that in Monte Carlo result.



(a) QuickYield (b) Monte Carlo

Figure 8: Comparison of simulation results from QuickYield and Monte Carlo

Moreover, we compare the accuracy as well as runtime for QuickYield and Monte Carlo in the Table(1). Since we do not have results from [3] for this example, QuickYield can not compare with YENSS here. From the table we can see that QuickYield can not only achieve 0.4% accuracy, but also gain $349X$ speedup over Monte Carlo method.

Table 1: Accuracy and Runtime Comparison

method	yield	time (second)	speedup
Monte Carlo (2000)	0.7014	66.2	2.97x
Monte Carlo (6000)	0.70185	197.1	1x
QuickYield (6 points)	0.70159	0.564	349x

4.2 3-Stage Ring Oscillator

We further consider a 3-stage ring oscillator as shown in Fig.(9). The oscillator period is chosen to be the performance metric of interest, which is determined by the delay of the inverters. The nominal period of the oscillator T_{norm} is $7.2028ns$ calculated via periodical steady state (PSS) simulation. The design specification requires that the variation in period δT should be within $\pm 2.5\%$ of T_{norm}. We consider the effect of random variations in the width of MOSFET in the first stage with 40% perturbation range from their nominal values. The nominal width of $Mp1$ is $3um$ and that of $Mn1$ is $2um$.

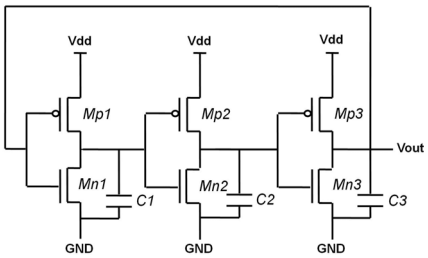
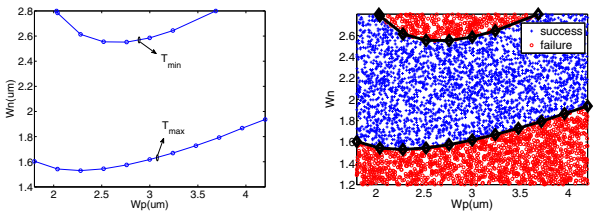


Figure 9: 3-Stage Ring Oscillator

Actually, there are two performance constraints for the oscillator as shown in equation (7), so there exist two boundary curves in the parameter space but they correspond to the T_{max} and T_{min}, respectively. We treat them individually in this case.

$$\begin{cases} H_1(\gamma_p; T) = T - T_{min} = 0 \\ H_2(\gamma_p; T) = T - T_{max} = 0 \end{cases} \quad (7)$$

First, the Monte-Carlo method is used to calculate the period T of the ring oscillator with width variations, as shown in Fig.(10(b)). The blue or success region is constitued of these samplings that lead to acceptable T, while the rest is the failure region. It is obvious that the success region is bounded by two nonlinear boundary curves. As such, the parametric yield can be estimated by the percentage of the samplings that locate in the success region.



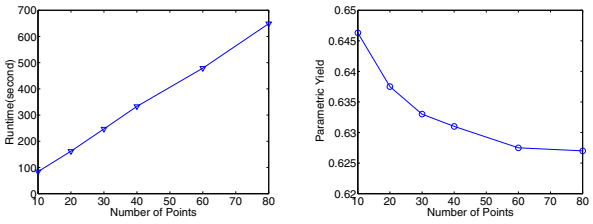
(a) QuickYield (b) Monte Carlo

Figure 10: Simulation results from QuickYield and Monte Carlo on Ring Oscillator

We further validate QuickYield against Monte Carlo simulation as shown in Fig.(10(a)). There are also two curves: upper bound for T_{min} and low bound for T_{max}, and they are identical to the result from Monte Carlo in Fig.(10(b)).

Moreover, we compare the accuracy and runtime of Monte-Carlo, YENSS obtained from [3] by normalizing with respect to its Monte Carlo runtime, and QuickYield in Table.(2). From this table, the Monte Carlo with 5000 simulations can generate a more accurate result, while it needs much more computation time. According to [3], YENSS can achieve 3.4% accuracy with $139X$ speedup over Monte Carlo. But QuickYield can obtain $519X$ speedup over Monte Carlo at a similar accuracy. Note that QuickYield and YENSS only use 10 points to locate two boundary curves. The accuracy can be improved with additional points.

We also study how the accuracy and runtime of Quick-Yield scales with the number of points, and the results is shown in Fig. 11. From the figure we can see that the runtime increases linearly, while the yield quickly converges when the point is over 60.



(a) Runtime (b) Yield

Figure 11: Runtime and accuracy scalability w.r.t. the number of points.

Table 2: Accuracy and Time Comparison for Ring Oscillator with two variational parameters

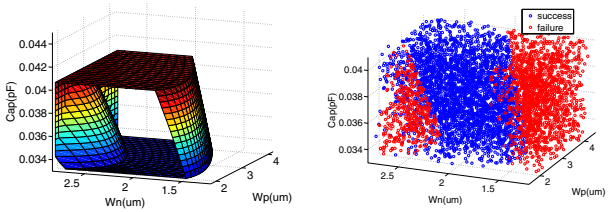
method	yield	time (second)	speedup
Monte Carlo (1000)	0.67465	8514.7	5.17x
Monte Carlo (5000)	0.62658	44073.8	1x
YENSS (10 points)	0.6482	-	139X
QuickYield (10 points)	0.6463	84.9	519x

To demonstrate the ability to handle many variational parameters, we increase the complexity of the problem by introducing random variation to the load capacitance $C1$ in the first stage. First, Monte Carlo samples the 3-D parameter space, and simulation results are plotted in Fig.(12(b)) (blue domain denotes the success region). Next, QuickYield is applied to find the boundary surfaces separating the success/fail regions shown in Fig.(12(a)), which is identical to the hyper-volume from Monte Carlo. Note that we remove the front and back surfaces to exhibit the surfaces for T_{min} and T_{max}.

Similarly, the accuracy and time are summarized in Table(3). With 20 points on each boundary surface, QuickYield can not only achieve up to 0.6% accuracy, but also obtain $267X$ speedup over Monte Carlo and $4.6X$ over YENSS.

5. CONCLUSIONS AND FUTURE WORK

In this paper, we have proposed a fast algorithm, Quick-Yield, to calculate the parametric yield with the performance constraint. QuickYield leverages the DAE equation with the performance constraint together to build an augmented system. By solving it, QuickYield locates the yield boundary points in the parameter space with the global-search, and further calculates the yield rate efficiently. Experimental results show that for the same accuracy, Quick-Yield is up to 519X faster compared with the Monte Carlo approach, and up to 4.7X faster compared with YENSS [2, 3]. Future work is to extend QuickYield in general to handle more variables under a problem formulation in the high-dimensional parameter space.



(a) QuickYield (b) Monte Carlo

Figure 12: Simulation results from QuickYield and Monte Carlo on Ring Oscillator

Table 3: Accuracy and Time Comparison for Ring Oscillator with three variational parameters

method	yield	time (second)	speedup
Monte Carlo (1000)	0.648	11840	5.33X
Monte Carlo (5000)	0.617	63128	1X
YENSS (20 points)	0.623	-	57X
QuickYield (20 points)	0.621	236.9	267X

6. REFERENCES

[1] P. Cox, P. Yang, and P. Chatterjee, "Statistical modeling for efficient parametric yield estimation of MOS VLSI circuits," in *IEDM'83*, 1983.

[2] S. Srivastava and J. Roychowdhury, "Rapid estimation of the probability of SRAM failure due to MOS threshold variations," in *CICC '07.*, 2007.

[3] 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," in *ASP-DAC '08*, 2008.

[4] S. C. G. Lucas and D. Chen, "Fastyield: Variation-aware, layout-driven simultaneous binding and module selection for performance yield optimization," in *ASP-DAC '09*, 2009.

[5] F. Gong, H. Yu, and L. He, "Picap: a parallel and incremental full-chip capacitance extraction considering random process variation," in *DAC '09*, 2009.

[6] H. Yu, X. Liu, H. Wang, and S. Tan, "A fast analog mismatch analysis by an incremental and stochastic trajectory piecewise linear macromodel," in *ASP-DAC '10*, 2010.

[7] I. Vytyaz, D. C. Lee, S. Lu, A. Mehrotra, U.-K. Moon, and K. Mayaram, "Parameter finding methods for oscillators with a specified oscillation frequency," in *DAC '07*, 2007.

[8] I. Vytyaz, P. K. Hanumolu, U.-K. Moon, and K. Mayaram, "Periodic steady-state analysis augmented with design equality constraints," in *DATE '08*, 2008.

[9] J. W. K. Kundert and A. Sangiovanni-Vincentelli, "Steady-state methods for simulating analog and microwave circuits," *Kluwer Academic Publishers*, 1990.