Capturing Post-Silicon Variations using a Representative Critical Path

Qunzeng Liu and Sachin S. Sapatnekar

***Abstract*— In nanoscale technologies that experience large levels of process variation, post-silicon adaptation is an important step in circuit design. These adaptation techniques are often based on measurements on a replica of the nominal critical path, whose variations are intended to reflect those of the entire circuit after manufacturing. For realistic circuits, where the number of critical paths can be large, the notion of using a single critical path is too simplistic. This paper overcomes this problem by introducing the idea of synthesizing a representative critical path (RCP), which captures these complexities of the variations. We first prove that the requirement on the RCP is that it should be highly correlated with the circuit delay. Next, we present three novel algorithms to automatically build the RCP. Our experimental results demonstrate that over a number of samples of manufactured circuits, the delay of the RCP captures the worst case delay of the manufactured circuit. The average prediction error of all circuits is shown to be below 2.8% for all three approaches. For both our approach and the critical path replica method, it is essential to guard-band the prediction to ensure pessimism: on average our approach requires a guard band 31% smaller than for the critical path replica method.**

***Index Terms*— Algorithms, circuit analysis, design automation, timing.**

I. Introduction

In nanoscale technologies, performance variations in manufactured die are seen to span a large range, and post-silicon diagnosis is becoming especially important. For feature sizes in the tens of nanometers, it is widely accepted that design tools must take into account parameter variations during manufacturing. These considerations have gained a great deal of attention for design during the presilicon phase, with variation-tolerant algorithms and tools being developed for circuit analysis and optimization to ensure adequate manufacturing yield. However, even circuits optimized using these techniques can show a significant performance spread, and it is important to diagnose manufactured parts in an inexpensive way in the post-silicon phase.

Among the root causes of these performance drifts over a population of manufactured parts is the phenomenon of process parameter variation. The values of the process parameters in each part fluctuate from the values that are assumed in design, even under corner-based paradigms. These variations can be classified into two categories: die-to-die (D2D) variations and within-die (WID) variations. D2D variations correspond to parameter fluctuations from one chip to another, while WID variations are defined as the variations among different locations within a single die. WID variations of some parameters have been observed to be spatially correlated, i.e., the parameters of transistors or wires that are placed close to each other on a die are more likely to vary in a similar way than those of transistors or wires that are far away from each other. For example, among the process parameters for a transistor, the variations of channel length L and transistor width W are seen to have such a spatial correlation structure. However, not all parameter variations are spatially correlated [1], [2], parameter variations such as the dopant concentration N_A and the oxide thickness T_{ox} are generally considered not to be spatially correlated.

The presilicon design phase has seen substantial research in response to these challenges, and significant changes to the conventional deterministic corner-based paradigm have been proposed and implemented. For circuit timing, the notion of statistical static timing analysis (SSTA) has been proposed as an alternative [3]–[8]. The idea of SSTA is that instead of computing the delay of the circuit as a specific number, a probability density function (PDF) of the circuit delay is determined. Designers may use the full distribution, or the 3σ point of the PDF, to estimate and optimize timing. Efficient statistical timing analysis tools have been developed based on parameterized block-based statistical timing analysis [3], [4], taking into consideration spatial and structural correlations of the parameter variations in the circuit to be analyzed. The computational efficiency of these methods is made practical through a preprocessing step, proposed in [3], which has shown that Gaussian-distributed correlated variations can be orthogonalized using principal component analysis (PCA).

The abovementioned statistical timing analysis tools are useful for presilicon analysis over an entire population of die, and are intended to maximize the yield over the population. The post-silicon analysis and optimization problem is complementary to any such presilicon analysis. The diagnosis problem addresses the issue of estimating the performance of a manufactured die, or determining the critical path (or paths) on the manufactured die. While this information may be gathered using time-intensive delay testing schemes, there are many instances where faster diagnosis is necessary, e.g., in post-silicon tuning methods.

In the previous literature, the interaction between presilicon analysis and post-silicon measurements has been addressed in several ways. In [9], post-silicon measurements are used to learn a more accurate spatial correlation model to refine the SSTA framework. A path-based methodology is proposed in [10] to correlate post-silicon test data to presilicon timing analysis. In [11], a statistical gate sizing approach is presented to optimize the binning yield. The work is extended to

The authors are with the Department of Electrical and Computer Engineering, University of Minnesota, Minneapolis, MN 55455. This research was supported in part by the NSF under award under awards CCF-0205227 and CCF-0541367, and by the SRC under contract 2007-TJ-1572.

simultaneously consider the presence of post-silicon-tunable clock tree and statistical gate sizing in [12]. Post-silicon debug methods and their interaction with circuit design are discussed in [13]. A joint design-time and post-silicon tuning procedure is described in [14]. In [15], a critical path monitor is built to monitor the critical path of the circuit as well as measuring process variations. A path selection methodology is proposed in [16] to monitor unexpected post-silicon systematic timing effects.

In this paper, we focus on post-silicon tuning methods that require replicating the critical path of a circuit. Such techniques include adaptive body bias (ABB) or adaptive supply voltage (ASV) optimizations [17]–[19]. The approach that is used in [17]–[19] employs a replica of the critical path at nominal parameter values (we call this the *critical path replica*), whose delay can easily be measured to determine the optimal adaptation. However, this has obvious problems: first, it is likely that a large circuit will have more than a single critical path, and second, a nominal critical path may have different sensitivities to the parameters than other near-critical paths, and thus may not be representative. An alternative approach in [20] uses a number of on-chip ring oscillators to capture the parameter variations of the original circuit. However, this approach requires measurements for hundreds of ring oscillators for a circuit with reasonable size and does not address issues related to how these should be placed or how the data can be interpreted online.

Another post-silicon optimization technique uses dynamic voltage scaling [21], [22]. In [21], a delay synthesizer, composed of three delay elements, is used to synthesize a critical path as part of a dynamic voltage and frequency management system. However, the control signals of the synthesizer is chosen arbitrarily and therefore it is not able to adapt to the fact that the critical path may change as a result of process variations. In [22], the authors compensate this problem using a precharacterized look up table (LUT) to store logic speed and interconnect speed inside different process bins. A logic and interconnect speed monitor is then used as an input to select through the LUT control signals to program a critical path. However, the authors use simplified circuitry for the speed monitor, consisting of only one logic-dominated element and one interconnect-dominated element, and assume that the results are generally applicable to all parts of the circuit. In the presence of significant within-die (WID) variations, this assumption becomes invalid. Moreover, the approach requires substantial memory components even for process bins of a very coarse resolution, and is not scalable to fine grids. The same problem exists in [15], which relies on a 12-inverter delay line to be representative of the variations of the circuit in order to monitor the critical path in question.

We propose a new way of thinking about the problem. We automatically build an on-chip test structure that captures the effects of parameter variations on all critical paths, so that a measurement on this test structure provides us a reliable prediction of the actual delay of the circuit, with minimal error, for all manufactured die. The key idea is to synthesize a test structure whose delay can reliably predict the maximum delay of the circuit, under die-to-die (D2D) as well as within-die (WID) variations. In doing so, we take advantage of the property of spatial correlation between parameter variations to build this structure and determine the physical locations of its elements.

The test structure that we create, which we refer to as the *representative critical path* (RCP), is typically different from the critical path at nominal values of the process parameters. In particular, a measurement on the RCP provides the worst-case delay of the whole circuit, while the nominal critical path is only valid under no parameter variations, or very small variations. Since the RCP is an on-chip test structure, it can easily be used within existing post-silicon tuning schemes, e.g., by replacing the nominal critical path in the schemes in [17]–[19]. While our method accurately captures any correlated variations, it suffers from one limitation that is common to any on-chip test structure: it cannot capture the effects of spatially uncorrelated variations, because by definition, there is no relationship between those parameter variations of a test structure and those in the rest of the circuit. To the best of our knowledge, this work is the first effort that synthesizes a critical path in the statistical sense. The physical size of the RCP is small enough that it is safe to assume that it can be incorporated into the circuit (using reserved space that may be left for buffer insertion, decap insertion, etc.) without significantly perturbing the layout.

The remainder of the paper is organized as follows. Section II introduces the background of the problem and formulates the problem mathematically. Next, Section III illustrates the detailed algorithms of our approach. Experimental results are provided in Section IV, and Section V concludes the paper.

II. Problem Formulation

The impact of spatial correlations of parameter variations have been studied in the previous literature on SSTA. In this paper, such correlations are captured using the grid-based model from [3]. In this model, the chip is divided into a number of grids tailored for the size of the circuit. Variations of the same process parameter inside each grid are taken to be fully correlated, meaning the correlation coefficient is considered to be one, and the correlation follows a decreasing function with respect to distance: specifically, variations inside nearby grids show higher correlation than variations within grids that are far away. For different process parameters, it is assumed that there are no correlations.

In principle, to find the maximum delay in the circuit, it is necessary to measure the delay of each potentially critical path in every manufactured part: the cost of this can be prohibitive, particularly in a timing-optimized circuit, where numerous paths may have to be potentially critical. The basic idea of this work is to conceive a way to infer the value of an unknown variable by sampling the value of a variable related to it. In other words, given that the variations are correlated, one could build inferences on the circuit delay using a small set of measurements. As pointed out in [20], no single test structure can capture the variations if they are uncorrelated.

An obvious way to achieve this is to use the nominal critical path for this prediction: this is essentially the critical path replica method [17]–[19]. However, the delay sensitivities of this nominal path may not be very representative. For instance, under a specific variation in the value of a process

parameter, the nominal critical path delay may not be affected significantly, but the delay of a different path may be affected enough that it becomes critical. Therefore, we introduce the notion of building an RCP, and demonstrate that the use of this structure yields better results than the use of the nominal critical path.

The overall approach is summarized as follows. For the circuit under consideration, let the maximum delay be represented as a random variable, d_c. We build an RCP in such a way that its delay is closely related to that of the original circuit, and varies in a similar manner. The delay of this path can be symbolically represented by another random variable, d_p. Clearly, the ordered pair (d_c, d_p) takes on a distinct value in each manufactured part, and we refer to this value as (d_{cr}, d_{pr}). In other words, (d_{cr}, d_{pr}) corresponds to one sample of (d_c, d_p), which is a particular set of parameter values in the manufactured part. Since the RCP is a single path, measuring d_{pr} involves considerably less overhead than measuring the delay of each potentially critical path. From the measured value of d_{pr}, we will infer the value, d_{cr}, of d_c for this sample, i.e., corresponding to this particular set of parameter values.

To mathematically simplify the situation and explore the relationship between the variables d_c and d_p, we assume that all parameter variations are Gaussian-distributed, and the delay of both the circuit and the critical path can be approximated by an affine function of those parameter variations. These functions can be obtained by performing SSTA using existing techniques [3], and the end results of d_c and d_p can be represented by Gaussian-distributed PDFs.

Let $d_c \sim N(\mu_c, \sigma_c)$, $d_p \sim N(\mu_p, \sigma_p)$, and let the correlation coefficient of d_c and d_p be ρ. Then we know that the joint PDF of d_c and d_p is

$$f(d_c = d_{cr}, d_p = d_{pr}) = \frac{1}{2\pi\sigma_c\sigma_p\sqrt{1-\rho^2}} e^{C_1}, \qquad (1)$$

where
$C_1 = -\frac{1}{2(1-\rho^2)}\left(\frac{(d_{cr}-\mu_c)^2}{\sigma_c^2} + \frac{(d_{pr}-\mu_p)^2}{\sigma_p^2} - \frac{2\rho(d_{cr}-\mu_c)(d_{pr}-\mu_p)}{\sigma_c\sigma_p}\right)$.
Using basic statistical theory, the conditional PDF of $d_c = d_{cr}$, given the condition $d_p = d_{pr}$, can be derived to have the following expression.

$$f(d_c = d_{cr} | d_p = d_{pr}) = \frac{f(d_{cr}, d_{pr})}{f(d_{pr})} = \frac{1}{2\pi\sigma_c\sqrt{1-\rho^2}} e^{C_2}, \qquad (2)$$

where
$C_2 = -\frac{1}{2\sigma_c^2(1-\rho^2)}\left(d_{cr} - \left(\mu_c + \frac{\rho\sigma_c}{\sigma_p}(d_{pr}-\mu_p)\right)\right)^2$.
Therefore the conditional distribution of d_{cr} is a Gaussian with mean $\mu_c + \frac{\rho\sigma_c}{\sigma_p}(d_{pr}-\mu_p)$ and variance $\sigma_c^2(1-\rho^2)$.

The result of this conditional distribution can be used in various ways. For example, we can provide the entire conditional distribution as the output of this procedure, as in [20]. On the other hand, if the conditional variance can be made sufficiently small, we can be more specific and directly use the mean of the conditional distribution as the predicted value of the delay of the circuit, and the variance may be interpreted as the mean square error of infinite samples.

An alternative view, from a least squares perspective, is that it is desirable to minimize the variance, so that the mean is an estimate of the circuit delay with the smallest mean square error. For the term representing the variance of the conditional distribution, $\sigma_c^2(1-\rho^2)$, σ_c is fixed since the original circuit must remain undisturbed, implying that the variance of the conditional distribution is dependent only on ρ.

In other words, minimizing the variance is therefore equivalent to maximizing ρ. This formal result is also an intuitive one: the RCP should satisfy the property that the correlation of its delay with that of the original circuit is maximized. Hence, our focus is on developing an efficient algorithm to build such an RCP, with the objective of maximizing the correlation coefficient.

III. Generation of the Critical Path

A. Overview of the SSTA Framework

As mentioned in Section II, it is important to represent the variables d_c and d_p as affine functions with respect to the parameter variations. To achieve this, we will employ previously developed SSTA techniques. By way of background for our main algorithmic contribution, in this section, we will briefly introduce the SSTA technique used in this work.

Available techniques for SSTA can broadly be divided into path-based and block-based methods. Block-based parameterized statistical timing analysis procedures propagate the PDF of the arrival time, using a canonical form, at the output of each gate during a topological traversal of the circuit. This canonical form typically consists of a mean (i.e., the nominal value) and a set of normalized independent sources of variations. Spatial and structural correlations of the parameter variations can be taken care of by applying PCA to the covariance matrix of the correlated process parameters [3]. PCA is a linear transformation and converts correlated Gaussian variables into a set of independent ones. Spatially uncorrelated variations can be lumped into one variable [4] as a placeholder to provide information about the variance that these variables add to the delay of the circuit.

We use parameterized SSTA illustrated above to obtain d_c as an affine function in the canonical form. This canonical form shows the delay having a linear relationship with the m PCs, plus an independent parameter. As shown below, this form makes the calculation of the correlation coefficient ρ defined in Section II much easier.

The canonical expression for d_c is shown as:

$$d_c = \mu_c + \sum_{i=1}^{m} a_i p_i = \mu_c + \mathbf{a}^T\mathbf{p} + R_c, \qquad (3)$$

where μ_c, the mean of d_c obtained from SSTA, represents the nominal value of d_c. The random variable R_c is an independent term whose variance is recorded as SSTA is performed. The random variable p_i corresponds to the i^{th} PC, and is Gaussian-distributed as $N(0,1)$; note that p_i and p_j for $i \neq j$ are uncorrelated by definition, due to the property of principal components. The parameter a_i is the first order coefficient of d_c with respect to p_i. All a_i variables are stacked together to form the vector $\mathbf{a}$, and $\mathbf{p}$ is the vector containing all p_i.

Performing SSTA on the RCP yields another delay expression in canonical form:

$$d_p = \mu_p + \sum_{i=1}^{m} b_i p_i = \mu_p + \mathbf{b}^T\mathbf{p} + R_p \qquad (4)$$

where d_p, μ_p are defined in Section II, and $p_i, b_i, \mathbf{b}, \mathbf{p}, R_p$ are all inherited from Equation (3).

B. Finding the Correlation Coefficient: Computation and Intuition

Since the original circuit and RCP are on the same chip, the values of the principal components for a given manufactured part are identical for both, and therefore their delays are correlated. In the manufactured part, any alteration in the PCs affects both the original circuit and the RCP, and if the RCP can be constructed to be highly correlated with the original circuit, the circuit delay can be estimated to a good degree of accuracy. In the extreme case where the correlation coefficient $\rho = 1$, the circuit delay, d_c, can be exactly recovered from d_p; however, as we will show later, this is not a realistic expectation.

The correlation coefficient, ρ, can easily be computed as

$$\rho = \frac{\mathbf{a}^T \mathbf{b}}{\sigma_c \sigma_p} \tag{5}$$

where $\sigma_c = \sqrt{\mathbf{a}^T \mathbf{a} + \sigma_{R_c}^2}$ and $\sigma_p = \sqrt{\mathbf{b}^T \mathbf{b} + \sigma_{R_p}^2}$. An important point to note is that ρ depends only on the coefficients of the PCs for both the circuit and the critical path and their independent terms, and not on their means.

As discussed in Section II, the mean of the conditional distribution $f\left(d_c = d_{cr} | d_p = d_{pr}\right)$, which can be used as an estimate of the original circuit delay, is:

$$\bar{\mu} = \mu_c + \frac{\rho \sigma_c}{\sigma_p}\left(d_{pr} - \mu_p\right) = \mu_c + \frac{\mathbf{a}^T \mathbf{b}}{\sigma_p^2}\left(d_{pr} - \mu_p\right). \tag{6}$$

The variance, which is also the mean square error of the circuit delay estimated using the above expression, for infinite samples, is $\sigma_c^2\left(1 - \rho^2\right)$. Our goal is to build a critical path with the largest possible ρ.

Our theory assumes that the effects of systematic variations can be ignored, and we will show, at the end of Section IV, that this is a reasonable assumption. However, it is also possible to extend the theory to handle systematic variations in parameters that can be controlled through design: for a fully characterized type of systematic variation, we can compensate for it by choosing a shifted nominal value for the parameter.

It is also useful to provide an intuitive understanding of the ideas above. If we were to achieve our goal of setting $\rho = 1$, this would imply that

$$\rho = \frac{\mathbf{a}^T \mathbf{b}}{\sigma_c \sigma_p} = 1$$

This means that

$$\sum_{i=1}^{m} \bar{a}_i \bar{b}_i = 1$$

where $\bar{a}_i = a_i/\sigma_c$, and $\bar{b}_i = b_i/\sigma_p$. Note that $\bar{a}_i$ and $\bar{b}_i$ correspond to the entries of the normalized $\mathbf{a}$ and $\mathbf{b}$ vectors, respectively.

This may be achieved if $\bar{a}_i = \alpha \bar{b}_i$, i.e., $b_i = a_i/\alpha \ \forall\ 1 \leq i \leq m$, where $\alpha = \sigma_p/\sigma_c$. In other words, all of the PCA parameters for the original circuit and the RCP are identical, within a scaling factor of α. This could be achieved if the sensitivities of the delays of d_c and d_p to all process parameter variations are identical, within a fixed scaling factor.

The key issue here is that it is not essential for the delays d_c and d_p to be identical. In fact, the circuit delay and the RCP delay may have very different nominal values, since the nominal delays, μ_c or μ_p, never enter into Equation (5). All that matters is that the variations in the RCP should closely track those of the original circuit. This observation provides us with a significant amount of flexibility with respect to the critical path replica method, which attempts to exactly mimic all properties of the maximum delay of the original circuit, including the nominal delay.

C. Generating the Representative Critical Path

Next, we propose three methods for generating the RCP. The first is based on sizing gates on an arbitrarily chosen nominal critical path, while the second synthesizes the RCP from scratch using cells from the standard cell library, while the third is a combination of the two methods.

1) Method I: Critical Path Generation Based on Nominal Critical Path Sizing: As described in Section I, the nominal critical path is usually not a good candidate to capture the worst case delay of the circuit over all reasonable parameter variations. However, there is intuitive appeal to the argument that variations along the nominal critical path have some relationship to the variations in the circuit. Based on this idea, our first approach begins with setting the RCP to a replica of the nominal critical path, and then modifies transistor sizes on this path so that the sized replica reflects, as far as possible, the variation of the delay of the manufactured circuit. The objective of this modification is to meet the criteria described in Section II, in order to ensure that the RCP closely tracks the delay of the critical path in the manufactured circuit.

For an optimized circuit, it is very likely that there are multiple nominal critical (or near-critical) paths with similar worst-case delays at nominal parameter values. To make our approach as general as possible, we pick the one nominal critical path that has the maximum worst-case delay at nominal process parameter values, even if its delay is only larger than a few other paths by a small margin. If there are multiple such paths, we arbitrarily pick one of them. We show in Section IV that even with this relaxed initial choice, after the optimizations presented in this section, our method can produce very good results.

The problem can be formulated as a nonlinear programming problem as listed below:

$$\text{maximize} \quad \rho = \frac{\mathbf{a}^T \mathbf{b}(\mathbf{w})}{\sqrt{\mathbf{a}^T \mathbf{a} + \sigma_{R_c}^2}\sqrt{\mathbf{b}(\mathbf{w})^T \mathbf{b}(\mathbf{w}) + \sigma_{R_p}^2(\mathbf{w})}} \tag{7}$$

$$\text{s.t.} \quad \mathbf{w} \in \mathbf{Z}^n$$

$$\mathbf{w}_{min} \leq \mathbf{w} \leq \mathbf{w}_{max} \tag{8}$$

The objective function above is the correlation coefficient, ρ, between d_p and d_c, as defined by Equation (5), which depends on $\mathbf{a}$, $\mathbf{b}$, and $\sigma_{R_p}^2$. The values of the latter two quantities are both influenced by the transistor widths, which are allowed to take on any values between some user-specified minimum and maximum values.

Algorithm 1 illustrates our procedure for building the RCP under this approach. We begin with the nominal critical path

Algorithm 1 Variation-aware critical path generation based on sizing.

1: Perform deterministic STA on the original circuit and find the maximum delay path as the initial RCP. If there is more than one such path, arbitrarily pick any one.
2: Perform SSTA on the original circuit to find the PC coefficients corresponding to the vector $\mathbf{a}$ and the variance of the independent term.
3: Perform SSTA on the initial RCP to find its PC coefficients and the variance of its independent term. Calculate the correlation coefficient ρ^0 between the delay variables of the original circuit and the initial RCP.
4: $k = 1$
5: **while** (1) **do**
6: **for** each gate i on the critical path **do**
7: If not violating the maximum size constraint, bump up the size by multiplying it by a factor $F > 1$, keeping all other gate sizes unchanged from iteration $k-1$
8: Compute $\rho^k_{u,i}$, the correlation coefficient for this modified RCP with the original circuit. Change the size of the gate back to its size in iteration $k-1$.
9: If not violating the minimum size constraint, size down the gate by multiplying the size by the factor $1/F$, keeping all other gate sizes unchanged from iteration $k-1$
10: Compute $\rho^k_{d,i}$ as the correlation coefficient for the modified RCP by sizing gate i down. Change the size of the gate back to its size in iteration $k-1$.
11: **end for**
12: Choose j such that $\rho^k_{u,j}$ or $\rho^k_{d,j}$ is the largest among all correlation coefficients, and set ρ^k to be this correlation coefficient.
13: **if** $\rho^k > \rho^{(k-1)}$ **then**
14: Set the RCP to be the RCP from iteration $k-1$, except that the size of gate j is sized up or down by the factor F.
15: **else**
16: break
17: **end if**
18: **end while**

of the circuit, chosen as described above, and replicate it to achieve an initial version of the RCP. Note that this is similar to the critical path replica method described in [17], and guarantees our method is at least as good as that approach. This critical path is then refined by iteratively sizing the gates on the path, using a greedy algorithm, in such a way that its correlation with the original circuit delay is maximized.

The first step of this approach involves performing conventional static timing analysis (STA) on the circuit to identify a nominal critical path, which is picked as the initial version of the RCP. Next, we perform SSTA on the circuit to obtain the PDF of the circuit delay variable, d_c, in the canonical form. This analysis provides us with the coefficients of the PCs in the circuit delay expression, namely, the vector $\mathbf{a}$ of Equation (3), as well as the independent term. We repeat this procedure for the initial RCP, to obtain the coefficients of the PCs in the expression for the delay, d_p, of the RCP. Based on these two canonical forms, we can compute the correlation coefficient, ρ^0, between the two delay expressions.

The iterative procedure updates the sizes of gates on the current RCP, using a TILOS-like criterion [23], with one modification: while TILOS will only upsize the gates, we also allow for the gates to be downsized. The rationale is that TILOS, for transistor sizing, begins with the minimum-sized circuit; in contrast, our approach begins with the sized nominal critical path, with the intention that since this configuration lies within the solution space for the RCP, the final RCP is guaranteed to be no worse than the nominal critical path.

In the k^{th} iteration, we process each gate on the RCP one by one. As an example, for the gate i, we examine the case of multiplying its current size by a constant factor, F or $1/F$, to, respectively, up-size or down-size the gate, while leaving all other gate sizes identical to iteration $k-1$. We perform SSTA on this modified RCP to obtain the new coefficients for the PCs corresponding to this change, and calculate the new correlation coefficient, $\rho^k_{u,i}$ and $\rho^k_{d,i}$. We apply this procedure to all gates on the RCP during each iteration, and over all of these possibilities, we greedily choose to up-size or down-size the gate j whose size update provides the maximum improvement in the correlation coefficient. We then update the RCP by perturbing the size of the gate j, and set the value of ρ^k to the improved correlation coefficient. We repeat this procedure until no improvement in the correlation coefficient is possible, or until the sizes of gates in the RCP become too large.

We can save on the computation time by exploiting the fact that the RCP is a single path, and that SSTA on this path only involves sum operations and no max operations. When the size of a gate is changed, the delays of most gates on the critical path are left unchanged. Therefore, it is sufficient to only perform SSTA on the few gates and wires that are directly affected by the perturbation, instead of the entire path. However, we still must walk through the whole path to find the gate with the maximum improvement. If the number of gates of a nominal critical path is bounded by s, and the sizing procedure takes K iterations, then the run time of Algorithm 1 is $O(Ks)$.

The final RCP is built on-chip, and after manufacturing, its delay is measured. Using Equation (6) in Section III-A, we may then predict the delay of the original circuit.

As mentioned at the beginning of this section, a significant advantage of this approach is that by choosing the nominal critical path as the starting point for the RCP, and refining the RCP iteratively to improve its correlation with the circuit delay, this approach is guaranteed to do no worse than one that uses the unmodified nominal critical path, e.g., in [17]–[19]. For a circuit that is dominated by a single critical path, this method is guaranteed to find that dominating path, e.g., the optimal solution.

The primary drawback of this method is also related to the fact that the starting point for the RCP is the nominal critical path. This fixes the structure of the path and the types of gates that are located on it, and this limits the flexibility of the solution. Our current solution inherits its transformations in each iteration from the TILOS algorithm, and changes the

sizes of gates in the circuit. However, in principle, the idea could also be used to consider changes, in each iteration, not only to the sizes but also to the functionality of the gates on the RCP by choosing elements from a standard cell library, so that the delay of the modified RCP (with appropriately excited side-inputs) shows improved correlations with the circuit delay. Another possible enhancement could be to select the nominal critical path with the highest initial correlation coefficient with the circuit delay, instead of choosing this path arbitrarily. These extensions may be considered in future work, but Section IV shows that even without them, our approach still produces good results.

2) Method II: Critical Path Generation Using Standard Cells: The second approach that we explore in this work builds the RCP from scratch, using cells from the standard cell library that is used to build the circuit. In principle, the problem of forming a path that optimally connects these cells together to ensure high correlation with d_c can be formulated as an integer nonlinear programming problem, where the number of variables corresponds to the number of library cells, and the objective function is the correlation between the statistical delay distribution, d_p, of an RCP with n stages of logic composed of these cells, where a stage is defined as a gate together with the interconnect that it drives, and d_c.

The integer nonlinear programming formulation is listed below:

$$\begin{aligned} \text{maximize} \quad & \rho = \frac{\mathbf{a}^T\mathbf{b}(\mathbf{N}_s)}{\sqrt{\mathbf{a}^T\mathbf{a}+\sigma^2_{R_c}}\sqrt{\mathbf{b}(\mathbf{N}_s)^T\mathbf{b}(\mathbf{N}_s)+\sigma^2_{R_p}(\mathbf{N}_s)}} \\ \text{s.t.} \quad & \mathbf{N}_s \in \mathbf{Z}^n \\ & \mathbf{e}^T\mathbf{N}_s \leq s \\ & \mathbf{b} = \Sigma_{i=1}^n N_{si}\mathbf{b}_i \\ & \sigma^2_{R_p} = \Sigma_{i=1}^n N_{si}\sigma^2_{R_{pi}} \end{aligned} \tag{9}$$

The objective function above is the correlation coefficient, ρ, between d_p and d_c, as defined by Equation (5). The variable n represents the number of possibilities for each stage of the RCP, and the vector $\mathbf{N}_s = [N_{s1}, N_{s2}, \cdots, N_{sn}]^T$, where N_{si} is the number of occurrences of i in the RCP.

The first constraint states the obvious fact that each element of $\mathbf{N}_s$ must be one of the allowable possibilities. In the second constraint, $\mathbf{e} = [1, 1, \cdots, 1]^T$, so that the constraint performs the function of placing an upper bound on the total number of stages in the RCP. For the purposes of this computation, $\mathbf{a}$ and $\sigma^2_{R_c}$ come from the canonical form of the circuit delay, d_c, and are constant. The values of $\mathbf{b}$ and $\sigma^2_{R_p}$ are functions of $\mathbf{N}_s$, where the mapping corresponds to performing SSTA on the RCP to find the vector of PC coefficients $\mathbf{b}$ and the variance of the independent term R_p in the canonical form. The terms $\mathbf{b}_i, 1 \leq i \leq n$, are the PC coefficients corresponding to each stage of the RCP, and the R_{pi}s correspond to the independent terms, so that $\mathbf{b}$ and $\sigma^2_{R_p}$ are related to $\mathbf{N}_s$ through the last two constraints.

Since Equation (9) does not map on to any tractable problem that we are aware of, we propose an incremental greedy algorithm, described in Algorithm 2, which is simpler and more intuitive than any exact solution of the integer nonlinear program. While this algorithm is not provably optimal, it is practical in terms of its computational cost. We recall that the goal of our problem is to make the correlation coefficient between d_c and d_p as large as possible. The algorithm begins by performing SSTA on the original circuit to determine d_c.

Algorithm 2 Critical path generation using standard cells.

1: Initialize the RCP P to be the initial load INV.
2: Perform SSTA on the original circuit to find d_c in canonical form, and also compute the canonical form for the delay of each of the $p \times q$ choices for the current stage.
3: Calculate the load L^{k-1} presented by the $(k-1)$-stage RCP computed so far.
4: With L^{k-1} as the load, perform SSTA on the $p \times q$ choices for stage k.
5: Statistically add the canonical expressions for the delays of each of the $p \times q$ choices with the canonical form for the delay of the partial RCP computed so far, P. Calculate the correlation coefficient between the summed delays and the delay of the original circuit for each case.
6: Select the choice that produces the largest correlation coefficient as stage k in path P.
7: Go to Step 3.

During each iteration, the RCP is constructed stage by stage. If we have p types of standard gates, and q types of metal wires, then in each iteration we have $p \times q$ choices for the stage to be added. For an RCP with m stages, to find the optimal solution corresponds to a search space of $(p \times q)^m$. Instead, our method greedily chooses one of the $p \times q$ choices at each stage that maximizes the correlation of the partial RCP constructed so far with d_c, thereby substantially reducing the computation involved. In practice, we control the complexity even further by using the minimum driving strength gate of each functionality in the library, rather than considering all driving strengths for all gates.

The approach begins at the end of the critical path. We assume that the path drives a measurement device such as a flip-flop, and the part of the device that acts as a load for the critical path is an inverter INV . Therefore, for the first iteration, this inverter is taken as the load, and it corresponds to a known load for the previous stage, which will be added in the next iteration.

In iteration k, we consider appending each of the $p \times q$ choices to the partial RCP from iteration $k-1$, and perform SSTA for all of these choices to obtain the coefficients for the PCs, and the correlation with d_c, using Equation (5). The choice that produces the largest correlation coefficient is chosen to be added to the critical path. The load presented by this choice is then calculated for the next selection procedure, and the process is repeated. During the process of building the RCP, there may be cases where a wire on the RCP crosses the boundary between two correlation grids: if so, the current gate and the one it drives belong to two different grids, and the wire connecting them must be split into two parts to perform the SSTA. The maximum number of stages used in the RCP is a user-specified parameter. During our iterations, we keep a record of the correlation coefficient after adding each stage. Once all stages up to the maximum number are added, we find the maximum correlation coefficient saved and eliminate stages added beyond that point.

A complementary issue for this algorithm is related to

determining the physical layout of each stage. To simplify the search space, we assume that the RCP moves monotonically: for example, the signal direction on all horizontal wires between stages must be the same, and the same is true of signal directions on all vertical wires. Because of symmetry of the spatial correlation profile and hence the PCA results, we only choose the starting points to be from the bottom grids of the die. For a given starting point, the path would move towards the right and upper directions of the circuit.

It should be noted that systematic variations would affect the sensitivities of the parameter values, causing PC coefficients of identical cells at symmetric locations to become different. However, because systematic variations can be precharacterized before statistical analysis by a change of nominal values at different locations, we show in Section IV that a reasonable disturbance of the nominal values would not significantly affect the final results. The procedure continues until the number of stages in the RCP reaches a prespecified maximum, or when the monotonic path reaches the end of the layout.

If the number of stages of the RCP is bounded by s and the number of starting points that we try for the RCP is ω, the runtime of Method II is $O(\omega pqs)$, because at each of the s stages, we have $p \times q$ choices. In comparison to Method I, if the bound of maximum number of stages for each method is comparable, then the comparison between K and $\omega \times p \times q$ determines which method has the longer asymptotic run time.

This approach has the advantage of not being tied to a specific critical path, and is likely to be particularly useful when the number of critical paths is large. However, for a circuit with one dominant critical path, this method may not be as successful as the first method, since it is not guided by that path in the first place.

3) Method III: Combination of the Two Methods: As stated above, each of the above two methods has its strengths and weaknesses. If the circuit is likely to be dominated by the nominal critical path, it is likely that Method I will outperform Method II; moreover, by construction, one can guarantee that Method I will do no worse than the critical path replica method. However, the structure of the RCP from Method I is also closely tied to that of the nominal critical path, limiting its ability to search the design space, and Method II provides improved flexibility in this respect, although it loses the guarantee of doing no worse than the critical path replica method.

We can combine the two methods discussed above in different ways to obtain potentially better results than either individual method. Many combinations are possible. For example, we could first build the RCP by sizing the nominal critical path using Method I, and then add additional stages from the standard cell library to it using Method II. However, the number of stages may become too large in this case. The approach used in this paper combines the methods by first building the RCP from scratch using Method II, setting the size of each gate to be at its minimal value. Next, we update the sizes of the gates on this RCP using Method I to further improve the result. The procedure is listed as Algorithm 3 below for completeness.

Algorithm 3 Critical path generation using the combined method.

1: Build the initial RCP, P, using Method II. All gates are at their minimal size.
2: Perform the TILOS-like sizing of Method I on P

IV. Experimental Results

We demonstrate the effectiveness of the approaches presented in this paper, and compare it with the *critical path replica (CPR)* approach, which represents the conventional approach to solving the problem. Our experiments are shown on the ISCAS89 benchmark suite. We use 90nm technology and the related constants are extracted from PTM model. The netlists are first sized using our implementation of TILOS: this ensures that the circuits are realistic and have a reasonable number of critical paths. The circuits are placed using Capo [24], and global routing is then performed to route all of the nets in the circuits. The cell library contains the following functionalities: NOT, 2-input NAND, 3-input NAND, 4-input NAND, 2-input NOR, 3-input NOR, and 4-input NOR.

The variational model uses the hierarchical grid model in [25] to compute the covariance matrix for each spatially correlated parameter. Under this model, if the number of grids is G and the number of spatially correlated parameters being considered is P, then the total number of PCs is no more than ($P \times G$). The parameters that are considered as sources of variations include the effective channel length L, the transistor width W, the interconnect width W_{int}, the interconnect thickness T_{int} and the inter-layer dielectric H_{ILD}. The width W is the minimum width of every gate before the TILOS sizing. We use two layers of metal, and take the parameters for different layers of metal to be independent. The parameters are Gaussian-distributed, and their mean and 3σ values are shown in Table I. As in many previous works on variational analysis, we assume that for each parameter, half of the variational contribution is assumed to be from die-to-die (D2D) variations and half from within-die (WID) variations. We use *MinnSSTA* [3] to perform SSTA, in order to obtain the PCA coefficients for d_c. All programs are run on a Linux PC with a 2.0GHz CPU and 256MB memory.

TABLE I
Parameters used in the experiments.

	L (nm)	W (nm)	W_{int} (nm)	T_{int} (nm)	H_{ILD} (nm)
μ	60.0	150.0	150.0	500.0	300.0
3σ	12.0	22.5	30.0	75.0	45.0

We first show the results of the algorithm that corresponds to Method I, described in Section III-C.1, synthesizing the RCP by modifying a nominal critical path of the original circuit. The initial sizes of the gates are their sizes after timing optimization. We only show the results of the larger circuits, since these are more realistic, less likely to be dominated by a small number of critical paths, and are large enough to allow significant within-die (WID) variations. Of these, circuit s9234 is smaller than the others, and is divided into 16 spatial correlation grids, while all other circuits are divided into 256 grids.

In our implementation of Method I, we do not consider congestion issues. We assume both the CPR method and Method I replicate the nominal critical path, including the interconnects, to provide a fair comparison. In practice, Method I can route the replicated nominal critical path in the same way as any of the prior CPR methods reported in the prior literature.

We use a set of Monte Carlo simulations to evaluate the RCP. For each circuit being considered, we perform 10,000 Monte-Carlo simulations, where each sample corresponds to a manufactured die. For each Monte Carlo sample, we compute the delay of the RCP, the delay of the original circuit, and the delay of the nominal critical path that may be used in a CPR method, as in [17]–[19]. The delay of the RCP is then used to compute the circuit delay using Equation (6) in Section III-A, which corresponds to the mean of the conditional distribution in Equation (2).

Figure 1 illustrates the idea of the conditional PDF, described in Section II, based on a sample of the Monte-Carlo simulations for circuit s9234. The lower curve shown with a solid line is the result of SSTA, and the circled points represent the conditional PDF obtained using Equation (2). The mean of this conditional distribution is indicated using a dashed line, and this is used as the estimate of the true circuit delay. The figure uses a solid vertical line to display the true circuit delay, and it can be seen that the two lines are very close (note that the plot does not start at the origin, and distance between these two lines is exaggerated in the figure).

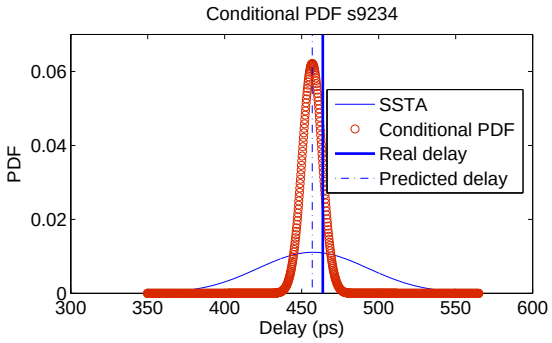


Fig. 1. Conditional PDF of s9234.

Since the probability of the true circuit delay being beyond the 3σ value of the conditional PDF is very low, the smaller the conditional variance, the smaller the errors. It is worth recalling that the conditional variance is given by $\sigma_c^2(1-\rho^2)$, each term of which is a constant for a specific RCP. Therefore, this variance is exactly the same for each die (corresponding to each sample of the Monte Carlo simulation), and if the RCP heuristic maximizes ρ as intended, we minimize this variance.

In our experiments, we compare the computed circuit delay, called the predicted delay, d_{predic}, with the true circuit delay of the circuit, referred to as the true delay, d_{true}. We define the prediction error as

$$\frac{|d_{true} - d_{predic}|}{d_{true}} \times 100\%. \tag{10}$$

In order to maximize yield, we must add a *guard band* for the predicted delay values to ensure that the predictions are pessimistic. Therefore in this set results we also compare the guard band needed to make 99% of the delay predictions pessimistic for both Method I and the CPR method, respectively.

TABLE II

A COMPARISON BETWEEN METHOD I AND THE CPR METHOD.

Circuit	Average error		Maximum error		Guard band (ps)	
	Method I	CPR	Method I	CPR	Method I	CPR
s9234	1.58%	2.84%	10.50%	14.93%	28.5	43.7
s13207	0.52%	1.07%	5.67%	6.61%	18.3	26.9
s15850	1.00%	2.15%	7.70%	10.88%	36.6	57.6
s35932	2.35%	5.77%	12.53%	21.46%	33.2	58.9
s38584	1.79%	3.23%	11.44%	17.89%	43.8	72.3
s38417	2.77%	5.24%	13.87%	21.22%	53.5	84.1

TABLE III

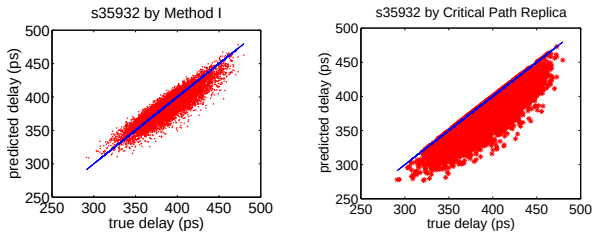
CONDITIONAL STANDARD DEVIATION, NUMBER OF STAGES FOR RCP, AND CPU TIME OF METHOD I.

Circuit	Avg $\frac{\sigma_{cond}}{\mu_{cond}}$	Max $\frac{\sigma_{cond}}{\mu_{cond}}$	Number of stages	CPU time
s9234	1.40%	1.84%	67	1.46m
s13207	1.06%	1.41%	71	10.98m
s15850	1.30%	1.74%	96	20.92m
s35932	2.51%	3.18%	36	8.23m
s38584	2.11%	2.68%	66	13.95m
s38417	3.12%	3.95%	41	3.88m

The results of the comparisons are presented in Table II, where the rows are listed in increasing order of the size of the benchmark circuit. For Method I as well as the CPR Method, we show the average error and maximum error over all samples of the Monte-Carlo simulation. All of the average errors of our approach are below 3% and both the average errors and maximum errors are significant improvements compared to the CPR method. The guard bands required by the two methods are listed in the last two columns. The guard band for Method I for each circuit is observed to be much smaller than the CPR method, and the advantage of Method I becomes particularly noticeable for the larger circuits.

The conditional variance derived in Section II defines the confidence of our estimate. Therefore we show the conditional standard deviation σ_{cond} as a percentage of the conditional mean μ_{cond} in Table III. Because μ_{cond} is different for each sample, we list the average and maximum $\frac{\sigma_{cond}}{\mu_{cond}}$ over all samples for each circuit. In order to provide more information about the RCP that we generate, we also show the number of stages for each RCP in the table. In this case, the number of stages for each RCP is the same as the nominal critical path for that circuit. The last column of the table shows the CPU time required by Method I for these benchmarks. The run time of Method I ranges from around one to twenty minutes. The conditional standard deviation is typically below 3% of the conditional mean on average. This method provides improved results in comparison with [26], which only upscales the transistors and does not consider downscaling as is done here.

For a visual interpretation of the performance of Method I, we draw scatter plots of the results for circuit s35932 in Figure 2(a) for Method I, and in Figure 2(b) for the CPR. The horizontal axis of both figures is the delay of the original circuit for a sample of the Monte-Carlo simulation. The vertical axis of Figure 2(a) is the delay predicted by our method, while the vertical axis of Figure 2(b) is the delay



(a) (b)

Fig. 2. The scatter plot: (a) true circuit delay vs. predicted circuit delay by Method I and (b) true circuit delay vs. predicted circuit delay using the CPR method.

of the nominal critical path, used by the CPR method. The ideal result is represented by the $(x = y)$ axis, shown using a solid line. It is easily seen that for the CPR method, the delay of the CPR is either equal to the true delay (when it is indeed the critical path of the Monte-Carlo sample) or smaller (when another path becomes more critical, under manufacturing variations). On the other hand, for Method I, all points cluster closer to the $(x = y)$ line, an indicator that the method produces accurate results. The delay predicted by our approach can be larger or smaller than the circuit delay, but the errors are small. Note that neither Method I nor the CPR Method is guaranteed to be pessimistic, but such a consideration can be enforced by the addition of a guard band that corresponds to the largest error. Clearly, Method I can be seen to have the advantage of the smaller guard band in these experiments.

Our second set of experiments implements the algorithm corresponding to Method II, presented in Section III-C.2. The maximum number of stages that we allow for the RCP for each circuit is 50, comparable to most nominal critical paths for the circuits in our benchmark suite. We use 7 standard cells at each stage, and 2 metal layers; therefore we have 14 choices for each stage. As in Method I, we do not consider congestion issues here and assume that the CPR method can perfectly replicate the nominal critical path. In practice, congestion considerations can be incorporated issues by assigning a penalty to congested areas when selecting wire directions. The setup of the Monte-Carlo simulations is similar to the first set of experiments, and the corresponding errors and guard bands are shown in Table IV. Since this Monte Carlo simulation is conducted separately from that in Table II, there are minor differences in the CPR error in these two tables, even though both tables use the same CPR as a basis for comparison. The average and maximum $\frac{\sigma_{cond}}{\mu_{cond}}$, the number of stages for each RCP, as well as the run times are shown in Table V. The advantage of Method II, again, increases with the size of the circuit.

It is observed that for almost all cases, the average and maximum errors for Method II are better than those for the CPR method. An exception to this is circuit s13207, which is dominated by a small number of critical paths, even after sizing using TILOS. We illustrate this using the path delay histogram in Figure 3(a), which aggregates the delays of paths in the sized circuit into bins, and shows the number of paths

TABLE IV

A COMPARISON BETWEEN METHOD II AND THE CPR METHOD.

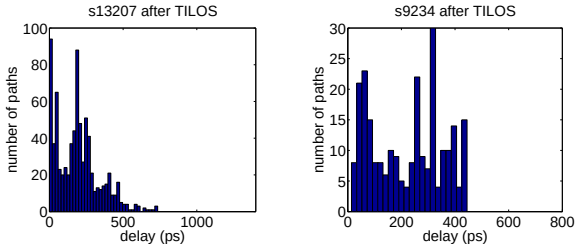
Circuit	Average error		Maximum error		Guard band (ps)	
	Method II	CPR	Method II	CPR	Method II	CPR
s9234	1.98%	2.84%	10.57%	15.15%	31.4	44.0
s13207	1.51%	1.06%	8.51%	7.22%	35.3	26.5
s15850	1.73%	2.14%	9.22%	10.97%	45.4	56.9
s35932	2.27%	5.80%	13.91%	21.34%	32.3	59.9
s38584	2.11%	3.29%	10.89%	17.12%	43.0	72.1
s38417	2.28%	5.27%	12.01%	22.88%	42.4	84.2

TABLE V

CONDITIONAL STANDARD DEVIATION, NUMBER OF STAGES FOR RCP, AND CPU TIME OF METHOD II.

Circuit	Avg $\frac{\sigma_{cond}}{\mu_{cond}}$	Max $\frac{\sigma_{cond}}{\mu_{cond}}$	Number of stages	CPU time
s9234	2.18%	2.79%	49	0.1s
s13207	1.75%	2.31%	30	15.7s
s15850	1.88%	2.45%	50	15.1s
s35932	2.19%	2.81%	50	16.7s
s38584	2.14%	2.73%	50	18.6s
s38417	2.13%	2.77%	50	15.5s

that fall into each bin. In this case, it is easily seen that the number of near-critical paths is small. In contrast, Figure 3(b) shows the same kind of histogram for circuit s9234, which is a more typical representative among the remaining benchmarks: in this case it is seen that a much larger number of paths is near-critical, and likely to become critical in the manufactured circuit, due to the presence of variations.

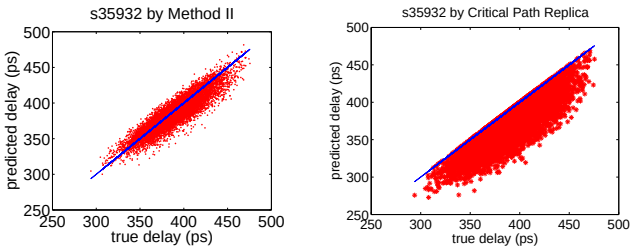


(a) (b)

Fig. 3. Histograms of path delays of (a) s13207 and (b) s9234 after TILOS optimization.

Under the scenario where the number of near-critical paths is small, it is not surprising that Method II does not perform as well as a CPR. First, as pointed out in Section III-C.2, Method II does not take advantage of any information about the structure of the original circuit, and is handicapped in such a case. Moreover, the unsized circuit s13207 was strongly dominated by a single critical path before TILOS sizing; after sizing, the optimized near-critical paths are relatively insensitive to parameter variations, meaning even if one of these becomes more critical than the nominal critical path on a manufactured die, it is likely to have more or less the same delay.

We also show scatter plots for both Method II and CPR in this case, in Figure 4(a) and Figure 4(b), respectively. The figures are very similar in nature to those for Method I, and similar conclusions can be drawn. In comparing Methods I and II by examining the numbers in Tables II and IV, it appears



(a) (b)

Fig. 4. The scatter plot: (a) true circuit delay vs. predicted delay by Method II and (b) true circuit delay vs. predicted delay using the CPR method.

that there is no clear winner, though Method II seems to show an advantage for the largest circuits, s35932 and s38417. With our limited number of choices for each stage of the RCP, referring to discussions about run time in Section III-C.2, it is not surprising that Method II is faster in terms of CPU time, as shown in Table V. The algorithm finishes within a few seconds for all of the benchmark circuits.

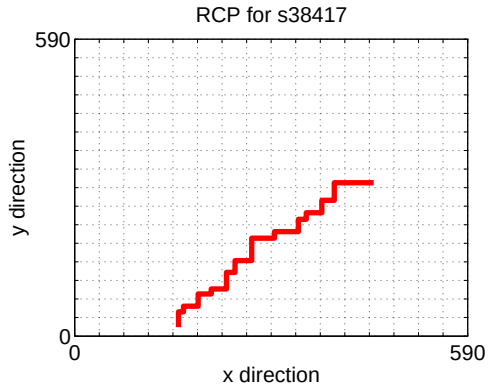


Fig. 5. The RCP created by Method II for circuit s38417.

Next we show the location of the RCP built for circuit s38417 using Method II on the chip in Figure 5. The figure shows the die for the circuit. The size of the die is determined by our placement and routing procedure, and the dashed lines indicate the spatial correlation grids. The solid bold lines are the wires of the critical path. The figure shows that the critical path grows in a monotonic direction and it starts from one of the grids at the bottom of the chip, both due to the layout heuristics discussed in Section III-C.2.

In order to gain more insight into the trend of improvement of the correlation coefficients, Figure 6 shows the correlation coefficient of Method II after each stage is added for one starting point: beyond a certain number of iterations, the marginal improvement flattens out. A similar trend is seen for Method I.

Our third set of experiments show the results of Method III, which is a combination of Method I and Method II. We first build an RCP from scratch using Method II, and then refine this RCP by the iterative sizing technique employed by Method I. Considering that Method II uses minimum-sized standard cells to restrict its search space, in the combined method, we

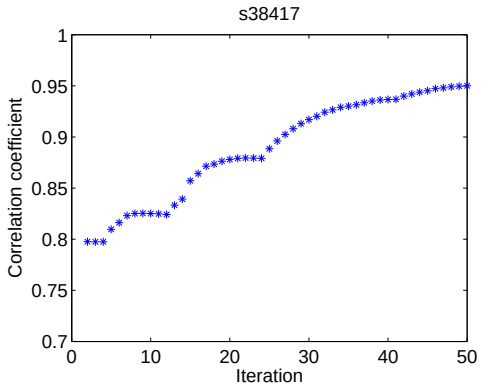
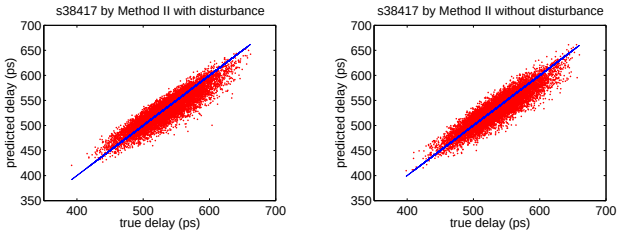


Fig. 6. Trend of correlation coefficient after each iteration.



(a) (b)

Fig. 7. Scatter plots of s38417 with and without nominal value disturbance for the RCP, to model systematic variations.

allow the cells to be sized up. In Table VI, we compare the correlation coefficients of the delay of the built RCP using Method II and using Method III. The correlation coefficients are calculated using Equation (5) and indicate of how closely the RCP can track the original circuit delay. The average and maximum errors, the guard band needed to ensure 99% of the predictions pessimistic, as well as the run time of the combined method are also listed in the table. It is observed that the sizing does indeed improve the results of Method II.

TABLE VI
RESULTS OF THE METHOD III AND ITS COMPARISON WITH METHOD II.

Circuit	ρ		Method III			
	Method II	Method III	Avg. error	Max. error	Guard band	CPU time
s9234	0.9732	0.9821	1.68%	7.64%	28.9ps	2.44s
s13207	0.9719	0.9750	1.38%	5.55%	31.2ps	16.97s
s15850	0.9613	0.9737	1.46%	7.2%	39.5ps	22.52s
s35932	0.9464	0.9492	2.26%	11.98%	34.1ps	20.38s
s38584	0.9493	0.9609	1.89%	9.13%	43.0ps	28.84s
s38417	0.9505	0.9526	2.17%	9.12%	37.4ps	15.06s

Finally, we experimentally demonstrate that our assumption of neglecting systematic variations is reasonable. We demonstrate this on Method II, and show that a reasonable change in the nominal parameter values of the RCP cells due to systematic variations would not affect the final results by much. This justifies our heuristic to only choose the starting point of the RCP at the bottom of the die.

The experiment proceeds as follows: after the RCP is built, we disturb the nominal values of all parameters associated with the RCP by 20%, while leaving those of the original circuit

unperturbed. This models the effect of systematic variations, where the RCP parameters differ from those of the original circuit. We show the final results of the scatter plots for circuit s38417, with and without disturbance, in Figures 7(a) and 7(b), respectively. It is shown that the plots are almost identical, and the average error is 2.26% with disturbance as compared to 2.28% for the normal case.

The intuition for this can be understood as follows. The correlation between the original circuit and the RCP depends on the coefficients of the PCs in the canonical expression. The coefficients depend on the sensitivities of the delay to variations, and not on their nominal values. Although the delay is perturbed by 20%, the corresponding change in the delay sensitivity is much lower, and this leads to the small change in the accuracy of the results.

V. Conclusion

In this paper, we have presented two novel techniques to automatically generate a critical path for the circuit to capture all of the parameter variations. A third approach is a simple hybrid of the two approaches that provides noticeable improvements over either individual approach. The key idea used in this paper is to take advantage of spatial correlations to build a test structure, the RCP, that captures variations in multiple critical paths in the circuit, exploiting the spatial correlation structure. Experimental results have shown that our methods produce good results.

Our current framework only handles process variations; environmental variations are addressed by adding adequate delay margins, since these are often worst-cased in practice. Addressing these through prediction remains an open area for future work. A straightforward extension of our method for very large circuits is to build not one, but a small number, of RCPs, one for each region of the circuit. The fundamental approach for building each path would be identical to the method proposed here.

VI. Acknowledgments

The authors would like to thank the anonymous reviewers, whose comments provided excellent feedback, resulting in an improvement in the quality of our paper.

References

[1] K. Agarwal, F. Liu, C. McDowell, S. Nassif, K. Nowka, M. Palmer, D. Acharyya, and J. Plusquellic, "A Test Structure for Characterizing Local Device Mismatches," in *IEEE Symposium on VLSI Circuits, Digest of Technical Papers*, pp. 67–68, 2009.

[2] L. Pang and B. Nikolic, "Impact of Layout on 90nm CMOS Process Parameter Fluctations," in *IEEE Symposium on VLSI Circuits, Digest of Technical Papers*, pp. 69–70, 2009.

[3] H. Chang and S. S. Sapatnekar, "Statistical Timing Analysis Considering Spatial Correlations using a Single PERT-Like Traversal," in *Proceedings of the IEEE/ACM International Conference on Computer-Aided Design*, pp. 621–625, Nov. 2003.

[4] C. Visweswariah, K. Ravindran, K. Kalafala, S. G. Walker, and S. Narayan, "First-Order Incremental Block-Based Statistical Timing Analysis," in *Proceedings of the ACM/IEEE Design Automation Conference*, pp. 331–336, June 2004.

[5] Y. Zhan, A. J. Strojwas, X. Li, L. Pileggi, D. Newmark, and M. Sharma, "Correlation-Aware Statistical Timing Analysis with Non-Gaussian Delay Distributions," in *Proceedings of the ACM/IEEE Design Automation Conference*, pp. 77–82, June 2005.

[6] J. Le, X. Li, and L. Pileggi, "STAC: Statistical Timing Analysis with Correlation," in *Proceedings of the ACM/IEEE Design Automation Conference*, pp. 343–348, June 2004.

[7] A. Agarwal, D. Blaauw, V. Zolotov, and S. Vrudhula, "Statistical Timing Analysis Using Bounds and Selective Enumeration," in *Proceedings of the ACM/IEEE International Workshop on Timing Issues in the Specification and Synthesis of Digital Systems*, pp. 29–36, Dec. 2002.

[8] A. Devgan and C. Kashyap, "Block-based Static Timing Analysis with Uncertainty," in *Proceedings of the IEEE/ACM International Conference on Computer-Aided Design*, pp. 607–614, Nov. 2003.

[9] B. Lee, L. Wang, and M. S. Abadir, "Refined Statistical Static Timing Analysis Through Learning Spatial Delay Correlations," in *Proceedings of the ACM/IEEE Design Automation Conference*, pp. 149–154, July 2006.

[10] L. Wang, P. Bastani, and M. S. Abadir, "Design-Silicon Timing Correlation–A Data Mining Perspective," in *Proceedings of the ACM/IEEE Design Automation Conference*, pp. 385–389, June 2007.

[11] A. Davoodi and A. Srivastava, "Variability Driven Gate Sizing for Binning Yield Optimization," in *Proceedings of the ACM/IEEE Design Automation Conference*, pp. 956–964, July 2006.

[12] V. Khandelwal and A. Srivastava, "Variability-Driven Formulation for Simultaneous Gate Sizing and Postsilicon Tunability Allocation," in *Proceedings of the International Symposium on Physical Design*, pp. 11–18, Mar. 2007.

[13] M. Abranmovici, P. Bradley, K. Dwarakanath, P. Levin, G. Memmi, and D. Miller, "A Reconfigurable Design-for-Debug Infrastructure for SoCs," in *Proceedings of the ACM/IEEE Design Automation Conference*, pp. 7–12, July 2006.

[14] M. Mani, A. Singh, and M. Orshansky, "Joint Design-Time and Post-Silicon Minimization of Parametric Yield Loss using Adjustable Robust Optimization," in *Proceedings of the IEEE/ACM International Conference on Computer-Aided Design*, pp. 19–26, Nov. 2006.

[15] A. Drake, R. Senger, H. Deogun, G. Carpenter, S. Ghiasi, T. Nguyen, N. James, M. Floyd, and V. Pokala, "A Distributed Critical-Path Timing Monitor for a 65nm High-Performance Microprocessor," in *Proceedings of the IEEE International Solid-State Circuits Conference*, pp. 398–399, Feb. 2007.

[16] N. Callegari, P. Bastani, L. Wang, S. Chakravarty, and A. Tetelbaum, "Path Selection for Monitoring Unexpected Systematic Timing Effects," in *Proceedings of the Asia-South Pacific Design Automation Conference*, pp. 781–786, Jan. 2009.

[17] J. W. Tschanz, J. T. Kao, S. G. Narendra, R. Nair, D. A. Antoniadis, A. P. Chandrakasan, and V. De, "Adaptive Body Bias for Reducing Impacts of Die-to-Die and Within-Die Parameter Variations on Microprocessor Frequency and Leakage," *IEEE Journal of Solid-State Circuits*, vol. 37, pp. 1396–1402, Nov. 2002.

[18] J. W. Tschanz, S. Narendra, R. Nair, and V. De, "Effectiveness of Adaptive Supply Voltage and Body Bias for Reducing the Impact of Parameter Variations in Low Power and High Performance Microprocessors," *IEEE Journal of Solid-State Circuits*, vol. 38, pp. 826–829, May 2003.

[19] J. W. Tschanz, S. Narendra, A. Keshavarzi, and V. De, "Adaptive Circuit Techniques to Minimize Variation Impacts on Microprocessor Performance and Power," in *Proceedings of the IEEE International Symposium on Circuits and Systems*, pp. 23–26, May 2005.

[20] Q. Liu and S. S. Sapatnekar, "Confidence Scalable Post-Silicon Statistical Delay Prediction under Process Variations," in *Proceedings of the ACM/IEEE Design Automation Conference*, pp. 497–502, June 2007.

[21] M. Nakai, S. Akui, K. Seno, T. Meguro, T. Seki, T. Kondo, A. Hashiguchi, H. Kawahara, K. Kumano, and M. Shimura, "Dynamic Voltage and Frequency Management for a Low-Power Embedded Microprocessor," *IEEE Journal of Solid-State Circuits*, vol. 40, pp. 28–35, Nov. 2005.

[22] M. Elgebaly and M. Sachdev, "Variation-Aware Adaptvie Voltage Scaling System," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 15, pp. 560–571, Nov. 2007.

[23] J. P. Fishburn and A. E. Dunlop, "TILOS: A Posynomial Programming Approach to Transistor Sizing," in *Proceedings of the IEEE/ACM International Conference on Computer-Aided Design*, pp. 326–328, Nov. 1985.

[24] A. Caldwell, A. B. Kahng, and I. Markov, "Capo: A Large-Scale Fixed-Die Placer," *available at http://vlsicad.ucsd.edu/GSRC/bookshelf/Slots/Placement*.

[25] A. Agarwal, D. Blaauw, V. Zolotov, S. Sundareswaran, M. Zhao, K. Gala, and R. Panda, "Path-Based Statistical Timing Analysis Considering Inter- and Intra-die Correlations," in *Proceedings of the ACM/IEEE International Workshop on Timing Issues in the Specification and Synthesis of Digital Systems*, pp. 16–21, Dec. 2002.

[26] Q. Liu and S. S. Sapatnekar, "Synthesizing a Representative Critical Path

for Post-Silicon Delay Prediction," in *Proceedings of the International Symposium on Physical Design*, pp. 183–190, 2009.

PLACE PHOTO HERE

Qunzeng Liu received the B.S. degree from Zhejiang University, Hangzhou, China, in Electrical Engineering in 2005. He is currently pursuing the Ph.D. degree in the Department of Electrical and Computer Engineering at the University of Minnesota. His research interests include statistical analysis and optimization of electronic circuits. He received the Best Paper award at ISPD2009 and the Best in Session award at SRC Techcon2009.

PLACE PHOTO HERE

Sachin S. Sapatnekar received the B.Tech. degree from the Indian Institute of Technology, Bombay in 1987, the M.S. degree from Syracuse University in 1989, and the Ph.D. degree from the University of Illinois at Urbana-Champaign in 1992. From 1992 to 1997, he was an assistant professor in the Department of Electrical and Computer Engineering at Iowa State University. He is currently the Robert and Marjorie Henle Chair in the Department of Electrical and Computer Engineering and a Distinguished McKnight University Professor at the University of Minnesota.

He is an author/editor of eight books, and has published mostly in the areas of timing and layout. He has held positions on the editorial board of the IEEE Transactions on VLSI Systems, and the IEEE Transactions on Circuits and Systems II, the IEEE Transactions on CAD, and has been a Guest Editor for the latter. He has served on the Technical Program Committee for various conferences, as Technical Program and General Chair for the Tau workshop and ISPD, and is currently the Vice Chair for DAC2009. He is a recipient of the NSF Career Award, five conference Best Paper awards, and the SRC Technical Excellence award. He is a Fellow of the IEEE.