

# Fast Buffer Insertion Considering Process Variations \*

Jinjun Xiong  
EE Dept., Univ. of California  
Los Angeles, CA 90095  
jinjun@ee.ucla.edu

Lei He  
EE Dept., Univ. of California  
Los Angeles, CA 90095  
lhe@ee.ucla.edu

## ABSTRACT

Advanced process technologies call for a proactive consideration of process variations in design to ensure high parametric timing yield. Despite of its popular use in almost any high performance IC designs nowadays, however, buffer insertion has not gained enough attention in addressing this issue. In this paper, we propose a novel algorithm for buffer insertion to consider process variations. The major contribution of this work is two-fold: (1) an efficient technique to handle correlated process variations under nonlinear operations; (2) a provable transitive closure pruning rule that makes linear complexity variation-aware pruning possible. The proposed techniques enable an efficient implementation of variation-aware buffer insertion. Compared to an existing algorithm considering process variations, our algorithm achieves more than 25× speed-up. We also show that compared to the conventional deterministic approach, the proposed buffer insertion algorithm considering correlated process variations improves the parametric timing yield by more than 15%.

## Categories and Subject Descriptors

J.6 [COMPUTER-AIDED ENGINEERING]: Computer-aided design (CAD)

## General Terms

Algorithms, Design, Performance, Reliability.

## Keywords

Buffer Insertion, Process Variation, Transitive Closure, Pruning Rule, Dynamic Programming.

---

\*This work is supported in part by the NSF CAREER award CCR-0401682, a UC MICRO grant sponsored by Actel and Mindspeed, and a Faculty Partner Award by IBM.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

ISPD'06, April 9–12, 2006, San Jose, California, USA.  
Copyright 2006 ACM 1-59593-299-2/06/0004 ...\$5.00.

## 1. INTRODUCTION

Advanced process technologies impose significant challenges for modern IC designers as we move into the ultra-deep sub-micron era where manufactured circuits exhibit substantial process variations. Proactive consideration of process variations during the design stage is critical to ensure high parametric timing yield. Studies on process variations include variability modeling [1], statistical static timing analysis (SSTA) [2, 3], and recently statistical optimization, like gate sizing [4] and power reduction [5]. Critical for almost any high performance IC designs nowadays, however, buffer insertion has not gained enough attention in considering process variations, despite of the fact that the deterministic buffer insertion problem has been studied extensively with different objectives in literature [6, 7].

One of the difficulties in solving the buffer insertion problem considering process variations is to model buffer solutions as random variables that are correlated. Such correlations are not only due to the shared global variation or spatial variation as encountered in SSTA, but also due to the way how buffer solutions are computed. Because almost all existing buffering algorithms to some degree follow the same dynamic programming paradigm [6], where solutions are computed recursively from downstream nodes, solutions from the same subtree are inherently correlated. Moreover, operations involved in computing different buffer solutions are nonlinear, like the minimum and multiplication operations, which further complicates the problem of handling the correlated variations. Another difficulty is how to define the *pruning rule* between different solutions in the presence of process variations. Because without an efficient pruning rule, a straight-forward implementation of the dynamic programming based buffering algorithm would make its complexity to increase exponentially.

To the best of our knowledge, there are four recent publications that have attempted to solve the buffer insertion problem considering process variation [8, 9, 10, 11]. However, none of them has addressed the above difficulties with definite answers. For example, [8] considered only the effect of wire length variation albeit the fact that wire length variation is not a typical process variation; and it was assumed that there was no correlation between different solutions. Moreover, three heuristic pruning rules were proposed with none of them can bound the complexity of the algorithm. [9] proposed to capture the correlation between solutions by using the joint probability density function (JPDF), and compute the JPDF numerically to handle the nonlinear operations of the correlated random variables. A *two-*

*side threshold based* pruning rule was proposed to reduce the runtime complexity. However, the complexity of computing JPDP numerically is high, yet the effectiveness of the two-side threshold based pruning rule is not clear. [10] studied a similar buffer insertion problem but without modeling the interconnect variation as random variables, thus avoiding the problem of handling nonlinear operations of random variables. Under the assumption of an ideal setting (like infinity long two-pin nets) with simplification, [11] showed that buffer insertion is not sensitive to process variation. However, it is not clear whether or not that statement still holds for designs under the real setting (e.g., finite wire length and multiple pins).

The contribution of this work is as follows. We propose a novel algorithm for buffer insertion to consider correlated process variations. We develop an efficient technique to handle the correlated process variations under nonlinear operations. We present a provable transitive closure pruning rule that makes linear complexity variation-aware pruning possible. Equipped with the above techniques, we show an efficient implementation of the variation-aware buffer insertion algorithm. Compared to the existing algorithm considering process variations [9], our algorithm achieves more than  $25\times$  speed-up. We also show that compared to the conventional deterministic approach, our buffer insertion algorithm considering correlated process variations improves the parametric timing yield by more than 15%.

## 2. PRELIMINARY

### 2.1 Deterministic Buffer Insertion

For a given buffered routing tree, two figure-of-merits are associated with every legal buffer position  $t$  in the tree: i.e., the *input loading capacitance* (or *downstream loading capacitance*)  $C_t$  and the *required arrival time*  $T_t$ . The basic buffer insertion problem formulation is to find the locations of buffers in the given routing tree such that the required arrival time (RAT) at the root is maximized [6]. In the context of dynamic programming, we first traverse the routing tree in the reverse topological order and propagate solutions bottom up while book-keeping all intermediate solutions. At the end (root), we pick the optimal solution with the largest RAT. By backtracking the chosen optimal solution, we determine the optimal solutions for each sub-tree recursively.

We characterize a device (or buffer) in terms of its gate capacitance ( $C_b$ ), intrinsic delay ( $T_b$ ) and output resistance ( $R_b$ ). For a given interconnect segment in the layout, we characterize it by its lumped resistance  $R_w$  and capacitance  $C_w$ . When each interconnect segment in the routing tree is modeled by a  $\pi$  model, under the Elmore delay model,  $C_t$  and  $T_t$  can be computed recursively as follows. By adding a wire at node  $n$ , all solutions at  $n$  are propagated to the other end of the wire  $t$  as follows

$$C_t = C_n + C_w \quad (1)$$

$$T_t = T_n - R_w \cdot C_n - \frac{1}{2} \cdot R_w \cdot C_w. \quad (2)$$

By adding a buffer at node  $n$ , all solutions at  $n$  are propagated to the input of the buffer as follows

$$C_t = C_b \quad (3)$$

$$T_t = T_n - T_b - R_b \cdot C_n. \quad (4)$$

If the solution at node  $t$  is obtained by merging two solutions from its two sub-trees rooted at nodes  $m$  and  $n$ , respectively, then solutions at the merging point are computed by

$$C_t = C_n + C_m \quad (5)$$

$$T_t = \min(T_n, T_m). \quad (6)$$

Knowing the above three key operations, the deterministic dynamic programming based buffer insertion can be solved by recursively applying the above three operations to obtain new solutions as we traverse the routing tree bottom-up.

### 2.2 Process Variation Modeling

To model the impact of process variation on both device and interconnect characteristics, we represent these characteristics of interest as random variables, which are complicated functions of the underlying process parameters. We employ the first order approximation to capture the major component of the variation. The rationale is that if the underlying parametric variations are small, any nonlinear relationship can be reasonably captured by a first order approximation. Such an approximation has been well accepted for statistical timing analysis [2, 3]. Mathematically it can be described as:

$$A = a_0 + \sum a_i \cdot F_i, \quad (7)$$

where  $A$  is the characteristic of interest, and  $F_i$  are the underlying physical process parameters and they are not independent to each other in general. The nominal value of  $A$  is  $a_0$ , while sensitivities of  $A$  with respect to  $F_i$  are given by  $a_i$ . If the function that describes the characteristics of interest with respect to the process parameters is known, we can obtain (7) analytically via the first order Taylor expansion of the function. Otherwise, SPICE simulation can be used to extract the nominal value of  $a_0$  and the sensitivities of  $a_i$ .

We further identify the following three sources of variation to model the cause of variations in process parameter  $F_i$ : the inter-chip global variation, the intra-chip spatial correlation, and the purely random variation. The three sources of variations are independent by definition, as the mechanism of causing those variations are different.

$$F_i = f_{i,0} + X_{i,g} + Y_{i,s} + X_{i,r}, \quad (8)$$

where  $f_{i,0}$  is the nominal value for  $F_i$ ,  $X_{i,g}$  is the inter-chip global variation that is shared among all  $F_i$  within the same chip,  $Y_{i,s}$  is the intra-chip spatial variation, and  $X_{i,r}$  is the random variation. To capture the spatial correlation between different  $Y_{i,s}$ , we represent the spatial variation part as follows:

$$Y_{i,s} = \sum d_{i,k} X_{i,k}, \quad (9)$$

where  $X_{i,k}$  are independent random variables, and  $d_{i,k}$  are coefficients of  $X_{i,k}$  and determine the amount of correlation between different  $F_i$ . We can obtain (9) by using the methods in [12, 13]. The authors of [12], proposed to obtain (9) by using a quad-tree like spatial correlation model, while in [13] the authors proposed to obtain the coefficients of  $d_{i,k}$  by measuring the distance between  $F_i$ 's physical location and the modeled  $X_{i,k}$  location.

By plugging (9) and (8) into (7) we obtain

$$A = a_0 + \sum a_i f_{i,0} + \sum a_i X_{i,g} + \sum \sum a_i d_{i,k} X_{i,k} + \sum a_i X_{i,r},$$

which is further simplified as

$$A = A_0 + \alpha^T X. \quad (10)$$

It is easy to see that (10) is a *first order canonical form* for  $A$ , where  $X$  is a random vector that includes the inter-chip global source of variation  $X_g$ , the spatial correlation  $X_{i,k}$ , and the random variation  $X_{i,r}$ ; and  $\alpha$  is the corresponding coefficient vector of  $X$ . Because all random variables in  $X$  are mutually independent and follow a standard normal distribution, i.e.,  $X \sim N(0, I)$ , the mean value of  $A$  is given by  $A_0$  in (10), while the variance of  $A$  is given by  $\alpha^T \alpha$ .

### 3. VARIATION AWARE BUFFERING

To consider process variations, we model all characteristics of interests, like  $T_b$ ,  $C_b$  and  $R_b$  for devices, and  $R_w$  and  $C_w$  for interconnect, as a random variable in canonical form, i.e.,  $T_b = T_{b0} + \gamma_b^T X$ ,  $C_b = C_{b0} + \eta_b^T X$ ,  $R_b = R_{b0} + \zeta_b^T X$ ,  $C_w = C_{w0} + \eta_w^T X$ ,  $R_w = R_{w0} + \zeta_w^T X$ .

Applying these random variables to (1) to (6), we obtain solutions  $C_t$  and  $T_t$  that are functions of  $T_b$ ,  $C_b$ ,  $R_b$ ,  $R_w$  and  $C_w$ , hence they are also random variables. However, because of the non-linear operators (multiplication and minimum operations) involved in computing the new solution, the distributions are no longer in the first order canonical form. To compute the exact probability density function (PDF) of  $C_t$  or  $T_t$  after the above non-linear operations, [9] resorted to a numerical method where hyper-dimensional integration is performed to obtain the distribution of  $C_t$  and  $T_t$  exactly. However, such an approach is not desirable and it greatly complicates computation due to the following two reasons. First, the exact computation of PDF after nonlinear operations is both difficult and numerically expensive. Second, by computing the exact PDF numerically, it has to operate on totally different PDF representations at each iteration. Therefore, to make the computation efficient, we propose a novel approximation technique in the following that keeps  $C_t$  and  $T_t$  after nonlinear operations still in the first order canonical form without loss of much accuracy.

To proceed, we represent all downstream node solutions by the following first order canonical forms:

$$C_n = C_{n0} + \alpha_n^T \cdot X, \quad (11)$$

$$T_n = T_{n0} + \beta_n^T \cdot X. \quad (12)$$

At the sink,  $C_n$  is the loading capacitance and  $T_n$  is the required arrival time, which are known from design specification.

If the solution at node  $t$  is obtained by adding a wire at its direct downstream node  $n$ , we compute the new solution at node  $t$  as follows:

$$C_t = (C_{n0} + C_{w0}) + (\alpha_n^T + \eta_w^T) \cdot X, \quad (13)$$

$$\begin{aligned} T_t &= (T_{n0} - R_{w0} \cdot C_{n0} - \frac{1}{2} \cdot R_{w0} \cdot C_{w0}) \\ &\quad + (\beta_n^T - C_{n0} \cdot \zeta_n^T - R_{w0} \cdot \alpha_n^T) \cdot X \\ &\quad - \frac{1}{2} (R_{w0} \cdot \eta_w^T + C_{w0} \cdot \zeta_w^T) \cdot X \\ &\quad - X^T (\zeta_w \cdot \alpha_n^T + \frac{1}{2} \zeta_w \cdot \eta_w^T) X \\ &= T_{t0} + \delta_t^T \cdot X - X^T \Gamma X, \end{aligned} \quad (14)$$

where  $T_{t0} = T_{n0} - R_{w0} \cdot C_{n0} - \frac{1}{2} \cdot R_{w0} \cdot C_{w0}$ ,  $\delta_t = \beta_n - C_{n0} \cdot \zeta_n - R_{w0} \cdot \alpha_n - \frac{1}{2} \cdot (R_{w0} \cdot \eta_w + C_{w0} \cdot \zeta_w)$ , and  $\Gamma = \zeta_w \cdot \alpha_n^T + \frac{1}{2} \zeta_w \cdot \eta_w^T$ . It is

obvious that  $C_t$  in (13) is already a canonical form, but  $T_t$  in (14) is not due to the quadratic term  $X^T \Gamma X$ . To represent  $T_t$  as a canonical form, we have the following Theorems:

**THEOREM 1.** *Given random variables in vector form  $X$  that follow a standard multivariate Gaussian distribution as  $N(0, I)$ , i.e.,  $E(X) = 0$  and  $E(X^2) = I$ , for any vector  $\delta$  and matrix  $\Gamma$ , we have<sup>1</sup>:*

$$E(X^T \Gamma X) = \text{tr}(\Gamma), \quad (15)$$

$$E(X^T \Gamma X \delta^T X) = 0, \quad (16)$$

$$E((X^T \Gamma X)^2) = 2\text{tr}(\Gamma^2) + \text{tr}(\Gamma)^2. \quad (17)$$

where  $E(\cdot)$  is the expectation operation of a random variable, and  $\text{tr}(\cdot)$  is the trace operation of a matrix which takes the sum of diagonal elements of the matrix.

**THEOREM 2.** *Given  $\Gamma = \alpha \beta^T + \zeta \eta^T$ , we have*

$$\text{tr}(\Gamma) = \beta^T \alpha + \eta^T \zeta, \quad (18)$$

$$\text{tr}(\Gamma^2) = (\beta^T \alpha)^2 + (\eta^T \zeta)^2 + 2(\beta^T \alpha)(\eta^T \zeta), \quad (19)$$

where  $\alpha$ ,  $\beta$ ,  $\zeta$ , and  $\eta$  are all vectors.

We then compute the first two moments of  $T_t$  in (14) as follows:

$$E(T_t) = T_{t0} + \delta_t^T \cdot E(X) - E(X^T \Gamma X) = T_{t0} - \text{tr}(\Gamma),$$

$$E(T_t^2) = (T_{t0} - \text{tr}(\Gamma))^2 + \delta_t^T \delta + 2\text{tr}(\Gamma^2).$$

Knowing the first two moments, we compute the mean and variance of  $T_t$  in (14) as follows:

$$\mu(T_t) = E(T_t) = T_{t0} - \text{tr}(\Gamma), \quad (20)$$

$$\sigma^2(T_t) = E(T_t^2) - E(T_t)^2 = \delta_t^T \delta + 2\text{tr}(\Gamma^2). \quad (21)$$

We then approximate (14) by the following canonical form that matches its mean and variance with (20) and (21), i.e.,

$$T_t = (T_{t0} - \text{tr}(\Gamma)) + \sqrt{1 + \frac{2\text{tr}(\Gamma^2)}{\delta_t^T \delta}} \cdot \delta_t^T \cdot X. \quad (22)$$

The above approximation is justified because the amount of variation is relatively small compared to the nominal value. Therefore, by matching the mean and variance, we only lose accuracy for the higher order (third moment and above) terms. Experiment results confirm the effectiveness of this approximation.

If the solution at node  $t$  is obtained by adding a buffer at its direct downstream node  $n$ , we compute new solutions as follows:

$$C_t = C_{b0} + \eta_b^T \cdot X, \quad (23)$$

$$\begin{aligned} T_t &= (T_{n0} - T_{b0} - R_{b0} \cdot C_{n0}) \\ &\quad + (\beta_n^T - \beta_b^T - L_{n0} \cdot \zeta_b^T - R_{b0} \cdot \alpha_n^T) X - X^T \zeta_b \cdot \alpha_n^T X \\ &= T_{t0} + \delta_t^T \cdot X - X^T \Gamma X, \end{aligned} \quad (24)$$

where  $T_{t0} = T_{n0} - T_{b0} - R_{b0} \cdot C_{n0}$ ,  $\delta_t = \beta_n - \beta_b - C_{n0} \cdot \zeta_b - R_{b0} \cdot \alpha_n$ , and  $\Gamma = \zeta_b \cdot \alpha_n^T$ . It is obvious that  $C_t$  in (23) is already a canonical form, but  $T_t$  in (24) is not due to the quadratic term  $X^T \Gamma X$ . By using the similar technique as discussed above, we approximate (24) via the same canonical form as shown in (22) but with its own  $T_{t0}$ ,  $\delta_t$ ,  $\Gamma$  and  $X$ .

<sup>1</sup>A similar theorem has also been proved in [14].

If the solution at node  $t$  is obtained by merging two solutions from its two sub-trees rooted at nodes  $m$  and  $n$ , respectively, we compute the new solutions  $C_t$  by

$$C_t = (C_{n0} + C_{m0}) + (\alpha_n^T + \alpha_m^T) \cdot X, \quad (25)$$

which is already a first order canonical form. To express  $T_t$  after the minimum operation still to be a first order canonical form, we resort to the concept of *tightness probability* [3]. Denote  $\sigma_n^2$  and  $\sigma_m^2$  as the variance of  $T_n$  and  $T_m$ , respectively, and  $cov(T_n, T_m)$  as the covariance of  $T_n$  and  $T_m$ . Knowing the first order canonical form of  $T_n$  and  $T_m$ , we have  $\sigma_n^2 = \beta_n^T \beta_n$ ,  $\sigma_m^2 = \beta_m^T \beta_m$ , and  $cov(T_n, T_m) = \beta_n^T \beta_m$ . The tightness probability of  $t_{n,m}$  is the probability of  $T_n$  less than  $T_m$  and is computed by

$$t_{n,m} = P(T_n < T_m) = \Phi\left(\frac{T_{m0} - T_{n0}}{\sigma_{n,m}}\right), \quad (26)$$

where  $\Phi$  is the cumulative density function (CDF) of a standard normal distribution; and  $\sigma_{n,m}$  is given by

$$\sigma_{n,m} = (\sigma_n^2 - 2cov(T_n, T_m) + \sigma_m^2)^{1/2}. \quad (27)$$

According to [15], the mean and variance of the statistical version of  $T_t = \min(T_n, T_m)$  can be computed exactly as follows:

$$\begin{aligned} \mu(T_t) &= t_{n,m} \cdot T_{n0} + (1 - t_{n,m})T_{m0} - \sigma_{n,m} \cdot \phi \quad (28) \\ \sigma^2(T_t) &= (\sigma_n^2 + T_{n0}^2)t_{n,m} + (\sigma_m^2 + T_{m0}^2)(1 - t_{n,m}) \\ &\quad + (T_{n0} + T_{m0})\sigma_{n,m} \cdot \phi - \mu(T_t)^2, \quad (29) \end{aligned}$$

where  $\phi$  is the probability density function (PDF) of a standard normal distribution taking value at  $\frac{T_{m0} - T_{n0}}{\sigma_{n,m}}$ .

We then approximate the statistical  $T_t$  in (6) after merging two solutions as follows:

$$\begin{aligned} T_t &= t_{n,m} \cdot T_{n0} + (1 - t_{n,m})T_{m0} - \sigma_{n,m} \cdot \phi \\ &\quad + (t_{n,m} \cdot \beta_n^T + (1 - t_{n,m}) \cdot \beta_m^T) \cdot X \\ &= T_{t0} + \beta_t^T \cdot X, \quad (30) \end{aligned}$$

where  $\beta_t = t_{n,m} \cdot \beta_n + (1 - t_{n,m}) \cdot \beta_m$ , and  $T_{t0} = t_{n,m} \cdot T_{n0} + (1 - t_{n,m})T_{m0} - \sigma_{n,m} \cdot \phi$ , which is exactly the mean value of  $T_t$  as shown in (28). To match the variance of (30) to the exact variance as shown in (29), we scale the  $\beta_t$  in (30) to obtain the new canonical form as follows:

$$T_t = T_{t0} + \frac{\sigma(T_t)}{\sqrt{\beta_t^T \beta_t}} \beta_t^T \cdot X. \quad (31)$$

By replacing (13) and (22) with (1) and (2), (23) and (22) with (3) and (4), and (25) and (31) with (5) and (6), respectively, we have replaced all key operations needed in a deterministic buffer insertion algorithm with its respective variation counterpart. Therefore, we obtain a variation-aware buffer insertion algorithm. Moreover, because we always keep solutions in first order canonical form after each operation, we can apply the same technique recursively to compute all new solutions while traversing the routing tree bottom up.

Note that the same approximation method via moment matching techniques is not restricted to the first-order canonical model, and it can be extended to handle other nonlinear (like quadratic) forms as well.

## 4. VARIATION AWARE PRUNING

### 4.1 Review of Deterministic Pruning

The major complexity of dynamic programming based buffer insertion lies in the merging of two sets of solutions obtained from two different sub-trees. In general, the total number of possible combinations for merging is  $n \cdot m$ , where  $n$  and  $m$  are the number of solutions from two sub-trees, respectively. If all combinations are kept at each merging node, the number of solutions will grow exponentially towards the root. To avoid this problem, [6] proposed to define the *dominance relationship* (or *pruning rule*) between two solutions such that solution  $(C_1, T_1)$  dominates solution  $(C_2, T_2)$  if condition  $C_1 < C_2$  and  $T_1 > T_2$  are satisfied. In other words, solution  $(C_2, T_2)$  is redundant and can be removed. [6] proved that instead of  $n \cdot m$ , there would be no more than  $n + m$  number of solutions after pruning. Even though pruning helps reduce the total number of solutions, in general we still have to pay the price of  $O(n \cdot m)$  at each node in order to obtain all possible combinations for merging. In deterministic buffer insertion, such a procedure is further reduced to  $O(n + m)$  by using a merge sort like operation on the two sets of already sorted solutions. Based upon the above two linear operations on pruning and merging, [6, 7] proved that by keeping only dominating solutions at every node, the dynamic programming based algorithm can solve the buffer insertion problem in  $O(N^2)$  time without losing optimality, where  $N$  is the number of possible buffer locations. When there are  $B$  types of buffers in the library, [16] proved that the deterministic buffer insertion problem can be solved optimally in  $O(B \cdot N^2)$ .

### 4.2 Transitive Closure Based Pruning

We observe that for deterministic buffer insertion, the linear time operations for pruning and merging are made possible because of the following two properties: (1) for any two given solutions, there exists an *ordering property* between them so that comparing them is always possible, i.e.,  $T_1$  is either greater than  $T_2$  or less than  $T_2$ ; (2) there exists a *transitive ordering property* between solutions, i.e., if  $T_1 > T_2$  and  $T_2 > T_3$ , then  $T_1 > T_3$  (similarly, if  $C_1 < C_2$  and  $C_2 < C_3$ , then  $C_1 < C_3$ ). If we ensure that the above two properties hold for solutions considering process variations, we can achieve similar linear time complexity for both pruning and merging. In the following, we propose a new variation aware pruning rule that enables us to keep both merging and pruning operations in linear complexity even in the presence of process variations.

We first extend the deterministic dominance relation between  $(C_1, T_1)$  and  $(C_2, T_2)$  by enforcing:

$$P(C_1 \leq C_2) = 1, \quad (32)$$

$$P(T_1 \geq T_2) = 1. \quad (33)$$

The physical interpretation of this extension is that solution  $(C_1, T_1)$  has 100% probability (almost always) to result in a larger required arrival time but with a less loading capacitance when compared to solution  $(C_2, T_2)$ . We have the following Lemma<sup>2</sup>:

<sup>2</sup>For simplicity, we only use  $T$  to illustrate the idea. It is understood that same results can be applied to  $L$  as well.

LEMMA 1. Given  $T_1, T_2$  and  $T_3$  as three **dependent** random variables with arbitrary distributions, if  $P(T_1 > T_2) = 1$ ,  $P(T_2 > T_3) = 1$ , then  $P(T_1 > T_3) = 1$ .

**Proof:** Let  $X = T_1 - T_2$  and  $Y = T_2 - T_3$  and the JPDF of  $X$  and  $Y$  be  $f(x, y)$ . As  $P(T_1 > T_2) = P(X > 0) = \int_0^{+\infty} x \int_{-\infty}^{+\infty} f(x, y) dy = 1$ , we have  $\int_{-\infty}^0 dx \int_{-\infty}^{+\infty} f(x, y) dy = 1 - \int_0^{+\infty} dx \int_{-\infty}^{+\infty} f(x, y) dy = 1 - 1 = 0$ . Because  $f(x, y) \geq 0$  for all  $x$  and  $y$ , we have  $f(x, y) = 0$  for  $x < 0$ . Similarly, from  $P(T_2 > T_3) = P(Y > 0) = 1$ , we have  $f(x, y) = 0$  for  $y < 0$ . Therefore, we have  $\int_0^{+\infty} dy \int_0^{+\infty} f(x, y) dx = 1$ .

Then  $P(T_1 > T_3) = P(X + Y > 0) = \int_{-\infty}^{+\infty} dy \int_{-y}^{+\infty} f(x, y) dx = \int_0^{+\infty} dy \int_{-y}^{+\infty} f(x, y) dx \geq \int_0^{+\infty} dy \int_0^{+\infty} f(x, y) dx = 1$ . As we know  $P(T_1 > T_3) \leq 1$ , we must have  $P(T_1 > T_3) = 1$ .  $\square$

Lemma 1 shows that comparison between  $T_1$  and  $T_2$  based upon  $P(T_1 > T_2) = 1$  enforces the transitive ordering property between solutions. However, for any two given solutions, it is not always possible to compare them. Moreover, in practice, such a 100% probability requirement is too restrictive. Therefore, we relax such a requirement by adding two parameters such that solution  $(C_1, T_1)$  is said to dominate solution  $(C_2, T_2)$  if the following two conditions hold:

$$P(C_1 < C_2) > \overline{p}_L, \quad (34)$$

$$P(T_1 > T_2) > \overline{p}_T, \quad (35)$$

where  $\overline{p}_L$  and  $\overline{p}_T$  are two parameters between 0.5 and 1. Note in the context of pruning, it is not interesting to take a value less than 0.5 for either  $\overline{p}_T$  or  $\overline{p}_L$ . In other words, the probability of  $C_1$  less than  $C_2$  is greater than  $\overline{p}_L$ , while the probability of  $T_1$  greater than  $T_2$  is greater than  $\overline{p}_T$ . We call the pruning rule as defined by (34) and (35) as *transitive closure* based pruning rule. As we will see in the following, the values of the two parameters,  $\overline{p}_L$  and  $\overline{p}_T$ , determine how good the two desired properties (ordering property and the transitive ordering property between solutions) are approximately true. We have the following lemma:

LEMMA 2. Given  $T_1$  and  $T_2$  as two **dependent** random variables with arbitrary distributions, we have either  $P(T_1 > T_2) \geq 0.5$  or  $P(T_1 < T_2) \geq 0.5$ .

**Proof:** The proof follows directly from the fact that  $P(T_1 > T_2) + P(T_1 < T_2) = 1$ .  $\square$

Lemma 2 shows that comparison based upon  $P(T_1 > T_2) > 0.5$  results in a proper ordering between two random solutions  $T_1$  and  $T_2$ . Therefore, the remaining problem is whether or not such a pruning rule preserves the transitive ordering property between solutions. Unfortunately, for arbitrary distributions, we can show that it does not preserve the transitive ordering property in general. In other words, for arbitrary random distributions, when  $\overline{p}_L = 1$  and  $\overline{p}_T = 1$ , the transitive ordering property holds but not necessary the ordering property; when  $\overline{p}_L = 0.5$  and  $\overline{p}_T = 0.5$ , the ordering property holds but not necessary the transitive ordering property. Therefore, to have the two properties to hold simultaneously, we may have to impose some restrictions on the type of distributions for those random solutions. In the following, we prove that when the random solutions follow a joint normal distribution, both properties indeed hold simultaneously, which is stated in the following Lemma:

LEMMA 3. Given  $T_1, T_2$  and  $T_3$  as three **dependent** random variables with joint normal distributions, if  $P(T_1 > T_2) > 0.5$ ,  $P(T_2 > T_3) > 0.5$ , then  $P(T_1 > T_3) > 0.5$

**Proof:** To see this, we assume both  $T_1$  and  $T_2$  are normal, and we have the following closed form to evaluate the probability of  $T_1 > T_2$  according to [15],

$$P(T_1 > T_2) = \Phi\left(\frac{\mu_{T_1} - \mu_{T_2}}{\sigma_{T_1, T_2}}\right), \quad (36)$$

where  $\Phi$  is the cumulative density function (CDF) of a standard normal distribution;  $\mu_{T_1}$ , and  $\mu_{T_2}$  are the mean values of  $T_1$  and  $T_2$ , respectively; and  $\sigma_{T_1, T_2}$  can be computed by

$$\sigma_{T_1, T_2} = (\sigma_{T_1}^2 - 2 \cdot \rho_{T_1, T_2} \sigma_{T_1} \sigma_{T_2} + \sigma_{T_2}^2)^{1/2}, \quad (37)$$

where  $\sigma_{T_1}^2$  and  $\sigma_{T_2}^2$  are variance of  $T_1$  and  $T_2$ , respectively; and  $\rho_{T_1, T_2}$  is the correlation coefficient of  $T_1$  and  $T_2$ .

Because any CDF function is a non-decreasing function, and for the standard normal distribution  $\Phi(0) = 0.5$ , then we have  $\Phi(x) > 0.5$  for any  $x > 0$ . Therefore, to have  $P(T_1 > T_2) > 0.5$  is equivalent to have  $\frac{\mu_{T_1} - \mu_{T_2}}{\sigma_{T_1, T_2}} > 0$ . Because  $\sigma_{T_1, T_2}$  is positive according to (37), hence we have  $\mu_{T_1} > \mu_{T_2}$ . Knowing  $P(T_1 > T_2) > 0.5$  and  $P(T_2 > T_3) > 0.5$ , we have  $\mu_{T_1} > \mu_{T_2}$  and  $\mu_{T_2} > \mu_{T_3}$ . Therefore, we have  $\mu_{T_1} > \mu_{T_3}$ , which is equivalent to  $P(T_1 > T_3) > 0.5$ .  $\square$

According to the proof of the above lemma, it is easy to see that the following equivalent statement also holds:

LEMMA 4. Given  $T_1$  and  $T_2$  as two **dependent** random variables with joint normal distribution, for  $P(T_1 > T_2) > 0.5$  to hold, it is necessary and sufficient to have  $\mu_{T_1} > \mu_{T_2}$  with  $\mu_{T_1}$  and  $\mu_{T_2}$  being the mean for  $T_1$  and  $T_2$ , respectively.

Based upon the above discussion, we have the following theorem regarding the complexity of the dynamic programming based variation aware buffer insertion algorithm:

THEOREM 3. By utilizing the transitive closure based pruning rule as defined in (34) and (35) with  $\overline{p}_L = 0.5$  and  $\overline{p}_T = 0.5$ , the variation aware buffer insertion problem can be solved in  $O(B \cdot N^2)$ , where  $B$  is the number of buffer types in the library, and  $N$  is the number of legal buffer positions.

**Proof:** According to Lemma (2) and (3), we can compare two random solutions and order them much the same way as in the deterministic approach. Therefore, following similar arguments as in [6, 7] and [16], we conclude that our variation aware buffer insertion algorithm under the transitive closure based pruning rule has the same complexity as the deterministic algorithm, which is  $O(B \cdot N^2)$ .  $\square$

Next we discuss the extension of the above transitive closure pruning rule for other choices of  $\overline{p}_L$  and  $\overline{p}_T$  and see how the two desired properties are affected. In fact, we have the following theorem which proves that for  $\overline{p}_T$  (or  $\overline{p}_L$ ) between 0.5 and 1, the transitive ordering property always hold.

THEOREM 4. Given  $T_1, T_2$  and  $T_3$  as three **dependent** random variables with joint normal distributions, if  $P(T_1 > T_2) > \overline{P}_T$ ,  $P(T_2 > T_3) > \overline{P}_T$ , then  $P(T_1 > T_3) > \overline{P}_T$  for any constant  $\overline{P}_T$  between 0.5 and 1, i.e.,  $0.5 \leq \overline{P}_T \leq 1$ .

**Proof:** Define  $X = T_1 - T_2$  and  $Y = T_2 - T_3$ , then we have  $X + Y = T_1 - T_3$ . Therefore,  $P(T_1 > T_2) = P(X > 0)$ ,  $P(T_2 > T_3) = P(Y > 0)$ , and  $P(T_1 > T_3) = P(X + Y > 0)$ .

Because  $T_1, T_2$  and  $T_3$  are joint normal, then  $X$  and  $Y$  are also normal. Denote the PDF of  $X$  as  $N(\mu_x, \sigma_x)$  and the PDF of  $Y$  as  $N(\mu_y, \sigma_y)$ , where  $\mu_x$  and  $\sigma_x$  (similarly  $\mu_y$  and  $\sigma_y$ ) are the mean and standard deviation for  $X$  (similarly  $Y$ ), respectively. Hence we can obtain the

PDF of  $X + Y$ , which is also a normal distribution, as  $N(\mu_x + \mu_y, \sqrt{\sigma_x^2 + \sigma_y^2 + 2\rho\sigma_x\sigma_y})$  with  $\rho$  being the correlation coefficient between  $X$  and  $Y$ . We have

$$P(X > 0) = P\left(\frac{X - \mu_x}{\sigma_x} > -\frac{\mu_x}{\sigma_x}\right) = 1 - \Phi\left(-\frac{\mu_x}{\sigma_x}\right), \quad (38)$$

where  $\Phi$  is the CDF of a standard normal distribution. According to the property of the standard normal distribution, we have  $\Phi(-t) = 1 - \Phi(t)$ , therefore we have

$$P(T_1 > T_2) = P(X > 0) = \Phi\left(\frac{\mu_x}{\sigma_x}\right). \quad (39)$$

As we already know  $P(T_1 > T_2) > \overline{P_T}$ , we hence have  $\Phi\left(\frac{\mu_x}{\sigma_x}\right) > \overline{P_T}$ . Since any CDF function is also a non-decreasing function, we have

$$\frac{\mu_x}{\sigma_x} > \bar{t}, \quad (40)$$

where  $\Phi(\bar{t}) = \overline{P_T}$ . Moreover, for  $0.5 \leq \overline{P_T} \leq 1$ , we have  $\bar{t} > 0$ . Similarly, we have

$$\frac{\mu_y}{\sigma_y} > \bar{t}. \quad (41)$$

From (40) and (41), we have

$$\mu_x + \mu_y > (\sigma_x + \sigma_y)\bar{t}, \quad (42)$$

$$\frac{\mu_x + \mu_y}{\sqrt{\sigma_x^2 + \sigma_y^2 + 2\rho\sigma_x\sigma_y}} > \frac{(\sigma_x + \sigma_y)\bar{t}}{\sqrt{\sigma_x^2 + \sigma_y^2 + 2\rho\sigma_x\sigma_y}}. \quad (43)$$

Because  $-1 \leq \rho \leq 1$ , it is easy to show that

$$\frac{(\sigma_x + \sigma_y)}{\sqrt{\sigma_x^2 + \sigma_y^2 + 2\rho\sigma_x\sigma_y}} \geq 1. \quad (44)$$

As  $\bar{t} > 0$ , by multiplying both sides of (44) by  $\bar{t} > 0$  and then combining it with (43), we have

$$\frac{\mu_x + \mu_y}{\sqrt{\sigma_x^2 + \sigma_y^2 + 2\rho\sigma_x\sigma_y}} > \frac{(\sigma_x + \sigma_y)\bar{t}}{\sqrt{\sigma_x^2 + \sigma_y^2 + 2\rho\sigma_x\sigma_y}} \geq \bar{t}. \quad (45)$$

Therefore, by the fact that  $\Phi$  is a non-decreasing function and (45), we finally have

$$\begin{aligned} P(T_1 > T_2) &= P(X + Y > 0) \\ &= \Phi\left(\frac{\mu_x + \mu_y}{\sqrt{\sigma_x^2 + \sigma_y^2 + 2\rho\sigma_x\sigma_y}}\right) \\ &> \Phi(\bar{t}) = \overline{P_T}. \end{aligned} \quad (46)$$

□

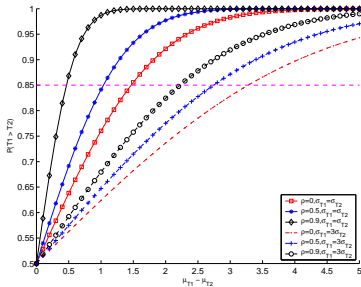


Figure 1: Probability of  $T_1$  greater than  $T_2$ .

Having proved the transitive ordering property, it is attempting to see whether the ordering property as in Lemma

(2) also holds for different choice of  $\overline{p_L}$  and  $\overline{p_T}$ . Unfortunately, such an extension is not true in general. However, in practice, we believe that the ordering property still holds (or approximately holds) for different choice of  $\overline{p_L}$  and  $\overline{p_T}$ . To see this, we plot the probability of  $T_1$  greater than  $T_2$  under three correlation coefficients (0, 0.5 and 0.9) as shown in Figure 1, where the x-axis is the mean difference of  $T_1$  and  $T_2$ , and the y-axis is  $P(T_1 > T_2)$ . The first three curves have  $\sigma_{T_1} = \sigma_{T_2}$ , and the rest three curves have  $\sigma_{T_1} = 3\sigma_{T_2}$ . According to Figure 1, we see that when the difference between  $\mu_{T_1}$  and  $\mu_{T_2}$  becomes larger, the probability of  $T_1$  greater than  $T_2$  is also increasingly larger. For a given required probability, say  $\overline{p_L} = 0.85$ , it only requires  $\mu_{T_1}$  being greater than  $\mu_{T_2}$  by less than 4 time unit. In practice, for a general routing tree, such a small delay difference will likely exist among different solutions, either due to the difference in routing or due to the difference in buffering. Moreover, such ordering becomes even better when two solutions have similar variances and have higher correlations, which is the case for our variation aware buffer insertion because solutions from the same sub-tree or nearby sub-trees are highly correlated inherently.

Based upon Theorem 4 and the above discussion, we believe that in practice the transitive closure based pruning rule for different choices of  $\overline{p_L}$  and  $\overline{p_T}$  other than 0.5 could also ensure the two desired properties, hence enabling us to carry out the variation aware buffer insertion algorithm much the same way as in the deterministic case. Therefore, similar runtime complexity conclusion can be drawn as shown in Theorem 3.

## 5. EXPERIMENT RESULTS

### 5.1 Experiment Setting

Two sets of benchmarks are obtained from the public domain for our experiments [17]. The characteristics of the benchmarks are shown in Table 1.

Bench	Sinks	Buffer Positions
p1	269	537
p2	603	1205
r1	267	533
r2	598	1195
r3	862	1723
r4	1903	3805
r5	3101	6201

Table 1: Characteristics of benchmarks.

Because of the lack of access to the real wafer data, we derive the process variation data based upon the literature that addresses similar process variation issues but in the context of statistical timing analysis [18]. In our experiment, the 65nm BSIM technology is assumed. We budget the random device variation, inter-die global variation, intra-die spatial variation, and interconnect variation all to be 5% of its nominal value, respectively. Moreover, to model the spatial variation, we divide the chip layout into different grids with the length of each grid as  $500\mu m$ . For devices located at a particular grid, their characteristics are affected by a set of nearby grids. We distribute the budgeted 5% spatial variation into different regions with the sensitivity of each region form an isotropic stationary Gaussian process with the value tapers off at a distance about  $2mm$ .

## 5.2 Comparison Base

Among the reported works [8, 9, 10] in literature, only [9] considered both device and interconnect process variations and provided an exact solution. Therefore, we use [9] as our comparison base in the following.

Because of the complication due to nonlinear operations in computing  $C_t$  and  $T_t$ , [9] employed a numerical method to compute the distribution of  $C_t$  and  $T_t$  explicitly. Moreover, to make the numerical method attractable, [9] further assumed that the variations in devices and interconnects are independent, hence ignoring their shared inter-chip global variation and intra-chip spatial correlations. The PDFs of  $C_t$  and  $T_t$  are computed numerically via hyper-dimensional integration, which is very expensive and slow to attain reasonable accuracy. In contrast, we employ the first order canonical form to represent both device and interconnect variations and the correlation between them is implicitly considered. Moreover, we compute the new canonical form for  $C_t$  and  $T_t$  via the novel approximation technique as discussed in section 3, thus avoiding the complexity of numerical integration yet maintain similar accuracy.

[9] also proposed a *two-side threshold* based pruning rule by relating the dominance relationship of solutions to designers' willingness of accepting uncertainty for a given design. That is, a threshold value  $\pi_\alpha$  gives a measure of designers preference for certainty in choosing the design parameter  $x$  in the presence of variations, such that the final design would have  $x$  less than  $\pi_\alpha$  with  $(100\alpha)\%$  certainty. Mathematically, this is given by

$$\alpha = \int_{-\infty}^{\pi_\alpha} f(x)dx, \quad (47)$$

where  $f(x)$  is the PDF of  $x$ . Given two different thresholds for either  $C_t$  and  $T_t$ , for example  $\pi_{\alpha_l}$  and  $\pi_{\alpha_u}$  for  $C_t$ , and  $\pi_{\beta_l}$  and  $\pi_{\beta_u}$  for  $T_t$ , such that  $0 \leq \alpha_l < \alpha_u \leq 1$  and  $0 \leq \beta_l < \beta_u \leq 1$ , solution  $(C_1, T_1)$  is said to dominate solution  $(C_2, T_2)$  if the following conditions are satisfied:

$$\pi_{\alpha_u}^{(1)} < \pi_{\alpha_l}^{(2)}, \quad (48)$$

$$\pi_{\beta_l}^{(1)} > \pi_{\beta_u}^{(2)}. \quad (49)$$

In other words,  $C_1$ 's upper threshold  $\pi_{\alpha_u}^{(1)}$  is smaller than  $C_2$ 's lower threshold  $\pi_{\alpha_l}^{(2)}$ , while  $T_1$ 's lower threshold  $\pi_{\beta_l}^{(1)}$  is larger than  $T_2$ 's upper threshold  $\pi_{\beta_u}^{(2)}$ . Despite of its intuitive definition, such a pruning rule is computationally expensive to use for large designs, because it is not guaranteed that the number of solutions after pruning is linearly.

## 5.3 Runtime Comparison

We first compare the runtime between our algorithm and [9] in terms of the largest benchmark that each algorithm can handle. As reported in [9], the largest routing tree has only nine (9) sinks. In contrast, our algorithm can easily handle routing trees with more than three thousand sinks ( $> 3000$ ), and it seems that nothing prevents our algorithm from handling even larger benchmarks<sup>3</sup>. In this sense, we improve the capacity of the algorithm by more than thousand times.

Furthermore, we note that one of the main reasons in preventing [9] from trying larger benchmarks is that their

<sup>3</sup>In fact, the largest benchmark we have tested in house is an eight-level H-tree clock network with more than 64,000 sinks.

two-side threshold based pruning rule is not very effective in both pruning and merging solutions. To verify this speculation, we obtain the source code from [9] and modify it based upon the same process variation models as used in this paper. We denote it as T2P because the two-side threshold based pruning rule is used. In other words, we avoid the high complexity in computing the JPDP numerically, but rather to use the same first order canonical form to represent the JPDP implicitly. Therefore, the only difference between our variation aware algorithm VAW and the newly implemented T2P algorithm is the pruning rule.

Bench	WORST	T2P [9]	VAW	Speedup
p1	0.00	25.4	1.0	25.4×
p2	0.01	-	4.3	-
r1	0.00	-	3.6	-
r2	0.00	-	15.0	-
r3	0.02	-	27.5	-
r4	0.04	-	88.9	-
r5	0.08	-	195.8	-

Table 3: Runtime comparison in second.

In Table 3, we report the runtime for both algorithms based upon the benchmarks we have tested. According to Table 3, we can see that the newly implemented T2P algorithm now can handle much larger benchmarks than what was originally reported in [9], and the largest tested benchmark is *p1* with 269 sinks. This improvement is mainly due to the avoidance of computing JPDP explicitly. However, we still fail to use the improved algorithm of [9] to run larger benchmarks. In fact, for the rest of tested benchmarks, it fails due to exceeding either memory capacity (2G) or tolerable time limit (4 hours in our setting). This observation is expected, because the two-side threshold based pruning rule only imposes partially ordering between solutions, rendering the complexity of merging and pruning very high. In contrast, by using the newly proposed transitive closure pruning rule, our VAW algorithm can easily run through all benchmarks and for the largest benchmark *r5*, the runtime is about 3 minutes. This significant runtime speedup is achieved because the transitive closure pruning rule as discussed in section 4 enforces a relatively strict ordering between solutions, thus enables an efficient implementation for both merging and pruning.

We also report the runtime for the conventional deterministic worst case buffering algorithm (WORST) in Table 3. We find the our VAW algorithm runs slower than WORST, but this is expected because of the additional computation needed to handle correlated process variations. If we plot the runtime of VAW versus the number of legal buffer positions, however, we can see that the runtime of our VAW algorithm scales almost linearly with respect to the benchmark size.

## 5.4 RAT Optimization

Enabled with the efficient implementation of buffer insertion considering process variations, we run our buffer insertion algorithm on the benchmarks for RAT optimization and study the effect of process variation on buffered interconnect design. We report the experiment results for both the conventional deterministic worst case design and our variation aware buffer insertion in Table 2. The *RAT* is defined as the 3-sigma *RAT* in the distribution for both algorithms. The 3-sigma yield of VAW is defined as 100%, which in term is used to find the timing yield for the worst

Bench	WORST			VAW		
	Buffer	RAT (%)	Yield	Buffer	RAT	Yield
p1	58 (-3.3%)	-2402.9 (-0.0%)	99.9%	60	-2402.9	100%
p2	149 (-4.5%)	-3203.5 (0.0%)	99.9%	156	-3204.0	100%
r1	59 (-9.2%)	-790.1 (0.1%)	99.9%	65	-790.7	100%
r2	112 (-17.0%)	-1127.8 (-1.5%)	65.2%	135	-1111.0	100%
r3	173 (-8.0%)	-1147.2 (-0.5%)	98.5%	188	-1141.9	100%
r4	320 (-14.4%)	-1722.6 (-1.4%)	45.6%	374	-1699.0	100%
r5	544 (-10.5%)	-1985.6 (-1.0%)	82.4%	608	-1966.6	100%
Avg	-9.6%	-0.6%	84.5%			

Table 2: Experiment results

case design. According to Table 2, we can see that compared to the conventional worst case design, our variation aware buffer insertion improves the 3-sigma  $RAT$  by 0.6% on average, and the parametric timing yield by more than 15%, respectively. This alarms the importance of developing efficient algorithms for IC designs to attack the process variation effects actively.

Interestingly, we observe that for some relatively small benchmarks, the improvement for  $RAT$  and yield is almost negligible, while for some large benchmarks, the improvement is quite significant. We also run the same set of experiments for different choices of  $\bar{p}_L$  and  $\bar{p}_T$  in (34) and (35). However, among all tested experiments, we observe almost no difference in the final optimal  $RAT$  at the root. These observations to some degree agree with what has been reported in [11] for infinity long two-pin nets. We are currently looking into the theoretical explanation to the above observations. We also report the number of buffers inserted for both algorithms in Table 2. We see that our variation aware buffering algorithm tends to put more buffers into the design to combat for the correlated process variations than the deterministic worst case design. This conclusion is in line with what has been reported in [9].

Finally, we verify the accuracy of our approach in predicting the  $RAT$  distribution via the Monte Carlo simulation. Given a buffered routing tree with process variations, we run our algorithm to compute the PDF of  $RAT$  at the root. After we obtain the  $RAT$  at the root, its PDF can be obtained by computing its mean and variance and it is approximated as a normal distribution. We find that our algorithm is reasonably accurate in predicating the PDF of  $RAT$  and tends to be conservative. On average, less than 10% difference is observed for all benchmarks we have tested.

## 6. CONCLUSION

A novel algorithm for buffer insertion considering process variation has been proposed. We have developed an efficient approximation technique to handle the correlated process variations under nonlinear operations. We have also proposed a provable transitive closure pruning rule that enables efficient implementation of the buffer insertion algorithm considering correlated process variations. We have applied the algorithm for timing optimization and concluded that process variation must be considered to achieve optimal designs for parametric timing yield, and buffer insertion considering correlated variation improves the timing yield by more than 15% on average.

## 7. REFERENCES

- [1] M. Orshansky, L. Milor, P. Chen, K. Keutzer, and C. Hu, "Impact of spatial intrachip gate length variability on the performance of high-speed digital circuits," in *IEEE Trans. on*

- Computer-Aided Design of Integrated Circuits and Systems*, pp. 544–553, May 2002.
- [2] H. Chang and S. Sapatnekar, "Statistical timing analysis considering spatial correlations using a single PERT-like traversal," in *Proc. Int. Conf. on Computer Aided Design*, pp. 621 – 625, Nov. 2003.
- [3] C. Visweswariah, K. Ravindran, K. Kalafala, S. G. Walker, and S. Narayan, "First-order incremental block-based statistical timing analysis," in *Proc. Design Automation Conf*, June 2004.
- [4] M. R. Guthaus, N. Venkateswaran, C. Visweswariah, and V. Zolotov, "Gate sizing using incremental parameterized statistical timing analysis," in *Proc. Int. Conf. on Computer Aided Design*, Nov 2005.
- [5] M. Mani, A. Devgan, and M. Orshansky, "An efficient algorithm for statistical minimization of total power under timing yield constraints," in *Proc. Design Automation Conf*, pp. 309–314, June 2005.
- [6] L. P. P. van Ginneken, "Buffer placement in distributed RC-tree networks for minimal Elmore delay," in *Proc. IEEE Int. Symp. on Circuits and Systems*, pp. 865–868, 1990.
- [7] J. Lillis, C.-K. Cheng, and T.-T. Lin, "Optimal wire sizing and buffer insertion for low power and a generalized delay model," in *Solid-State Circuits, IEEE Journal of*, vol. 31, pp. 437 – 447, March 1996.
- [8] V. Khandelwal, A. Davoodi, A. Nanavati, and A. Srivastava, "A probabilistic approach to buffer insertion," in *Proc. Int. Conf. on Computer Aided Design*, 2003.
- [9] J. Xiong, K. Tam, and L. He, "Buffer insertion considering process variation," in *Proc. Design Automation and Test in Europe*, 2005.
- [10] L. He, A. B. Kahng, K. Tam, and J. Xiong, "Simultaneous buffer insertion and wire sizing considering systematic CMP variation and random Leff variation," in *Proc. Int. Symp. on Physical Design*, April 2005.
- [11] L. Deng and M. D. Wong, "Buffer insertion under process variations for delay minimization," in *Proc. Int. Conf. on Computer Aided Design*, pp. 317–321, 2005.
- [12] A. Agarwal, D. Blaauw, V. Zolotov, S. Sundareswaran, M. Zhao, K. Gala, and R. Panda, "Statistical delay computation considering spatial correlations," in *Design Automation Conference, Proceedings of the Asia and South Pacific*, pp. 271 – 276, Jan 2003.
- [13] V. Khandelwal and A. Srivastava, "A general framework for accurate statistical timing analysis considering correlations," in *Proc. Design Automation Conf*, pp. 89 – 94, June 2005.
- [14] L. Zhang, W. Chen, Y. Hu, J. A. Gubner, and C. C.-P. Chen, "Correlation-preserved non-gaussian statistical timing analysis with quadratic timing model," in *Proc. Design Automation Conf*, pp. 83 – 88, June 2005.
- [15] M. Cain, "The moment-generating function of the minimum of bivariate normal random variables," in *The American Statistician*, vol. 48, May 1994.
- [16] Z. Li and W. Shi, "An  $o(bn^2)$  time algorithm for optimal buffer insertion with b buffer types," in *Design, Automation and Test in Europe*, pp. 1324–1329, March 2005.
- [17] W. Shi and Z. Li, "An  $o(n \log n)$  time algorithm for optimal buffer insertion," in *Proc. Design Automation Conf*, Jun 2003.
- [18] A. Agarwal, D. Blaauw, and V. Zolotov, "Statistical timing analysis for intra-die process variations with spatial correlations," in *Proc. Int. Conf. on Computer Aided Design*, pp. 900 – 907, Nov. 2003.