21.3

Computation and Refinement of Statistical Bounds on Circuit Delay

Aseem Agarwal, David Blaauw, *Vladimir Zolotov, **Sarma Vrudhula

University of Michigan, Ann Arbor, MI
*Motorola, Inc., Austin, TX
**University of Arizona, Tucson, AZ

Abstract

The growing impact of within-die process variation has created the need for statistical timing analysis, where gate delays are modeled as random variables. Statistical timing analysis has traditionally suffered from exponential run time complexity with circuit size, due to arrival time dependencies created by reconverging paths in the circuit. In this paper, we propose a new approach to statistical timing analysis that is based on statistical bounds of the circuit delay. Since these bounds have linear run time complexity with circuit size, they can be computed efficiently for large circuits. Since both a lower and upper bound on the true statistical delay is available, the quality of the bounds can be determined. If the computed bounds are not sufficiently close to each other, we propose a heuristic to iteratively improve the bounds using selective enumeration of the sample space with additional run time. We demonstrate that the proposed bounds have only a small error and that by carefully selecting an small set of nodes for enumeration, this error can be further improved.

Categories and Subject Descriptors

B.8.2 [Performance and Reliability]: Performance analysis

General Terms

Algorithms, performance, reliability

1 Introduction

Traditionally, the variation in the underlying process parameters have been modeled in static timing analysis (STA) using so-called case analysis. In this methodology, best-case, nominal and worst-case SPICE parameters sets are constructed and the timing analysis is performed several times. Each execution of static timing analysis is therefore deterministic, meaning that the analysis uses deterministic delays for the gates and any statistical variation in the underlying silicon is hidden. While this approach has been successfully used in the past to model die-to-die variations, it is not able to accurately model variations within a single die. Using a worst-case analysis for these so-called *within-die* variations leads to very pessimistic analysis results since it assumes that all devices on a die have worst-case characteristics, ignoring their inherent statistical variation. The emerging dominance of within-die variations therefore poses a major obstacle for deterministic STA.

Process variations are due to uncertainty in the device and interconnect characteristics, such as effective gate length, doping concentrations, oxide thickness and ILD thickness. In general, these variations can be divided into *between-die* variations (or inter-die variation) and *within-die* variations (or intra-die variations). Within-die variations can have a deterministic component due to topologically dependencies of device processing, such as CMP effects and lithographic distortions. In some cases, such topological dependencies can be directly accounted for in the analysis [1], whereas in other cases, such variations are treated as random.

DAC 2003, June 2-6, 2003, Anaheim, California, USA.

In this paper, we propose a statistical STA method for modeling random within-die process variations. Since between-die variations can be adequately captured using case analysis, we focus on within-die variations. We also treat all variations as random variations, meaning that topological dependencies are either removed prior to the analysis or are treated as random variations.

The extensive use of deterministic STA is in large part due to its linear run time with circuit size. In contrast, statistical STA has an underlying worst-case complexity that is exponential with circuit size, which poses a fundamental obstacle to its practical application. This high run time complexity is the result of reconverging paths in the circuit which causes correlations between path delays due to shared sections of such paths. A second source of correlation between arrival times results from spatial and topological correlation of the individual gate delays. Consequently, many statistical STA approaches [2-3] have high runtimes, which makes them applicable only to small circuits, or ignore the presence of correlations. Recently, a number of new methods have been proposed to address the increasing significance of process variations. In [4], a novel method using discretized probability distributions is proposed. However, the run time of the method is exponential. In [5], a method using statistical bounds is proposed with gate delays restricted to Gaussian distributions. In [6], a path based statistical delay computation is presented using an accurate delay model. However, the analysis is performed on one path at a time and the number of critical and near-critical paths in a circuit can be very large. In [7], a new circuit optimization method was proposed that reduces the number of near critical paths in a circuit, thereby improving the statistical delay of the circuit.

In this paper, we propose a new method for statistical STA. We first provide a formal model of statistical STA and formulate its exact solution. Since the computational complexity of this exact statistical STA is exponential with the circuit size, we present a new method for computing bounds on the exact probability distribution of the circuit. The computed bounds are themselves probability distribution functions that can be used to obtain a conservative estimate of the circuit delay at any desired confidence point. By restricting our analysis to bounds, we are able to preserve the linear run time complexity of deterministic STA. Since we provide both a lower and upper bound, we can determine the quality or error of the computed bounds. If the bounds are not sufficiently close to each other, we propose a heuristic method to iteratively improve the computed bounds using selective enumeration of the sample space. We show that by enumerating a small set of so-called *dependence* nodes, the computed bounds can be significantly improved with reasonable run time. The proposed methods were implemented and tested on benchmark circuits. The difference between the expected values of the upper and lower bound was shown to be small, ranging from 2 - 10%, and this difference could be reduced by 62% on average, using the proposed selective enumeration method.

2 Statistical STA Formulation and Exact Solution

The goal of statistical static timing analysis is to model the impact of gate delay variations due to within-die process variations on the circuit delay. Although at design time, the delay of each gate is unknown, after a chip has been manufactured, the gate delays are

fixed and have a deterministic value for each particular die. The randomness or variability of the circuit delay is therefore over the fabricated die, and it is the cumulative distribution of the circuit delay that statistical timing analysis aims to obtain. We now give the following definition of a timing graph:

Definition 1. A *timing graph* G is a directed graph having exactly one source and one sink node: $G=\{N,E,n_s,n_f\}$, where $N=\{n_1,n_2,...,n_k\}$ is a set of nodes, $E=\{e_1,e_2,...,e_l\}$ is a set of edges, $n_s \in N$ is the source node, and $n_f \in N$ is the sink node and each edge $e \in E$ is simply an ordered pair of nodes $e=(n_i,n_j)$.

In our formulation, a *deterministic* timing graph G_D represents a particular manufactured die, where each gate has a fixed delay value and each edge e in G_D is assigned a delay $D(e)$ accordingly. After fabrication, a deterministic timing graph G_D can be conceptually formulated for each die. However, during the design of a chip, the gate delays are unknown and must be modeled as random variables. Each gate delay is therefore specified either with a cumulative probability distribution function (*CDF*) or probability density function (*PDF*). A probabilistic timing graph G_P is a timing graph whose edges are assigned random variables of delay values. Since the PDF and CDF represent the variation of gate delays, they have the following obvious but important property:

Property 1. A delay CDF equals 0 for all delay values less than its minimum d_{min} and equals 1 for all values greater than its maximum d_{max}. A delay PDF is non-zero only on the interval $[d_{min}, d_{max}]$.

These properties follow from the fact that the delay of a real gate cannot be less that some finite minimum d_{min} or more than some finite maximum d_{max}. Similar to a number of previous statistical STA methods [2-4,6], we assume statistical independence of all edge delays. In practice, edge delays may be spatially or topologically correlated, which complicates the analysis by creating additional correlations between path delays. The contribution of this paper is therefore that it provides an efficient solution to the problem of path delay correlation due to path reconvergence. Note, however, that our method does not restrict the shape of the CDF of edge delays to some specific shape. To simplify the implementation of statistical STA, it is often more convenient to approximate continuous PDFs and CDFs with discrete functions although we will formulate the problem with continuous functions. A discrete PDF can be represented by a sequence of delay/probability pairs (d_i, p_i).

We now consider the sample space S of a probabilistic timing graph G_P, consisting of all deterministic timing graphs G_D with edge delays corresponding to the non-zero values of their probability distribution functions. Given a deterministic timing graph G_D in S, we can compute its delay $D(G_D)$, using any of the currently available means, such as traditional static timing analysis. The delay $D(G_P)$ is therefore defined on the sample space S and is a random variable. The objective of statistical timing analysis is to find the CDF of $D(G_P)$, which is defined as follows:

Definition 2. The cumulative probability distribution function of the delay of a probabilistic timing graph is expressed as:

$$P(D(G_P) \le t) = \int_{D(G_D) \le t} p_1(t_1)p_2(t_2)\ldots dt_1 dt_2 \ldots \quad \text{(EQ 1)}$$

where $p_i(t_i)$ is the probability density function of the delay of edge i and the integration is performed over the volume of sample space where delay $D(G_D)$ of timing graph G_D is less than t.

The cumulative probability distribution function of the graph delay can be used in a number of ways. First, given a particular performance constraint, the probability of obtaining a fabricated die that meets or exceeds this constraint can be determined, referred to as the performance yield. Conversely, given a required performance yield, the minimum expected performance can be obtained. For instance, a designer can determine the expected minimum performance of 95% of the dies.

Independent Statistical Timing Analysis

Deterministic timing analysis has traditionally used an approach where arrival times are propagated through the circuit in topological order. We therefore derive such a propagation based approach for computing the graph delay D_{Gp}. We first make the following useful definition.

Definition 3. A fanin subgraph $G_{S,n}$ of timing graph G_P at node n is a timing graph consisting of all edges and nodes of G_P that lie on a path from the source node n_s of G_P to node n.

Note also that the arrival time A_n at node n is equivalent to the graph delay of the fanin subgraph $G_{S,n}$. The objective of statistical timing analysis is to compute the arrival time CDF of node n based on the arrival time CDFs of its fanin nodes n_p. To compute the arrival time of n, we must consider whether the arrival times of its fanin nodes n_p are independent random variables. We therefore state the following theorem:

Theorem 1. Two arrival times $A_{n,i}$ and $A_{n,j}$ at nodes n_i and n_j are independent if the fanin subgraphs $G_{S,i}$ and $G_{S,j}$ at nodes n_i and n_j are disjoint (meaning they have no common edges) or if all common edges have a deterministic delay.

The validity of Theorem 1 is intuitively obvious and a proof is omitted for brevity. Given a node n with fanin nodes $n_{p,i}$, which have independent arrival time $A_{p,i}$, we can compute the arrival time A_n at node n as follows:

1. Sum the arrival time $A_{p,i}$ at each node $n_{p,i}$ with the edge delay of the edge connecting $n_{p,i}$ with n. Since the arrival time $A_{p,i}$ and the edge delay are independent, the PDF of A_n at node n is computed through convolution of the PDF of $A_{p,i}$ and the PDF of the edge delay [9].
2. Compute the arrival time CDF A_n at node n by taking the statistical maximum of the CDFs using the following equation:

$$P_{Max}(t) = P(t) \cdot Q(t) \quad \text{(EQ 2)}$$

As also noted in [3], the computation of the arrival time PDFs in statistical timing analysis is therefore very similar to arrival time propagation in deterministic timing analysis, where propagation of deterministic arrival times is replaced with convolution and selection of the latest arrival time is replaced with parallel reduction. The run time complexity is linear with the size of the circuit. Unfortunately, this procedure is only valid if the arrival times are independent for each node in the circuit.

Exact Dependent Statistical Timing Analysis

If the fanin subgraphs $G_{S,i}$ of fanin nodes $n_{p,i}$ of node n share one or more edges with random delays, the arrival times $A_{p,i}$ will be dependent random variables and the statistical maximum in EQ2 cannot be applied. An example of such a graph is shown in Figure 1(a). To determine for which portions the subgraphs $G_{S,i}$ share edges, we use the following definition of a *dependence* set.

Definition 4. Consider the set of k fanin nodes $n_{p,i}$ of node n, with fanin subgraphs $G_{S,i}$, and the intersection graph $G_I=\{N_I,E_I\}$ consisting of the union of edges and nodes shared by two or more subgraphs, excluding the source node n_s. The dependence set of n is the set of nodes $\{n_1, n_2,..., n_d,...\}$, such that n_d lies on the intersection graph, $n_d \in N_I$, and has one or more fanout edges e_i that do not lie on the intersection graph, $e_i \notin E_I$.

In Figure 1(a), the intersection graph G_I for node n_f is shaded and consists of nodes $\{q, r, a, b, c, d\}$. The set of dependence nodes for node n_f is $\{b, d\}$, since these nodes are part of the intersection graph G_I and have fanout edges that are not part of G_I. Note that nodes h and e have empty dependence sets since their intersection graphs are empty. Conceptually, dependence nodes mark the last points in the graph where the fanin subgraphs of two or more fanin nodes are shared and give rise to correlation between their arrival times. The concept of dependence nodes is similar to that used in probabilistic simulation [8]. We refer to a node in G_P as a convergence node n_c if it has a non-empty dependence set and define the *global* set of dependence nodes n_D as the union of the dependence sets $n_{d,i}$.

In order to compute the delay of a graph G_P with one or more dependence nodes, we sort the list of global dependence nodes in topological order. We then consider the first node $n_{D,1}$ in the ordered set n_D. In Figure 1(a), $n_D = \{a,b,d\}$, and $n_{D,1} = a$. By selecting the first node $n_{D,1}$ in the list, we ensure that the fanin subgraph $G_{S,1}$ at node $n_{D,1}$ does not contain any dependence nodes, and it follows that we can replace $G_{S,1}$ with a single edge e_1 connecting source node n_s and $n_{D,1}$, where the edge delay CDF D_1 of e_1 is equal to the arrival time CDF A_1 at $n_{D,1}$, as shown in Figure 1(b). Similarly, it is clear that the arrival time CDF A_1 at $n_{D,1}$ can be computed using independent arrival time propagation.

For simplicity, we assume that the edge delay PDF D_1 is discrete and is specified by a set of k delay, probability pairs (d_i, p_i). According to our construction, random variable D_1 does not depend on the edge delays of other edges in the transformed graph G_p. Then, using conditional probabilities [9], the arrival time PDF $p_x(t)$ at node x, can be computed as follows: $p_x(t) = \sum_{i=0}^{k} p_i \cdot p_{x,i}(t)$, where $p_{x,i}(t)$ is the arrival time PDF at node x when the delay D_1 of e_1 is equal to d_i and $p_i = P(D_1 = d_i)$. We therefore compute the arrival time PDF $p_x(t)$ by performing k arrival time computations, each weighted by the conditional probability p_i. Since during the computation of $p_{x,i}$, edge e_1 has a deterministic delay it is no longer a random variable and does not create dependence between arrival times. Node $n_{D,1}$ is therefore no longer a dependence node and we can propagate arrival times using independent arrival time propagation until we encounter the next global dependence node, $n_{D,2}$. Here, we repeat the same process, enumerating the arrival time PDF at $n_{D,2}$ using conditional probabilities and eliminating it as a dependence node.

Below is the procedure for dependent arrival time propagation:

1. Propagate arrival time PDFs in the circuit until the first dependence node n_d is encountered.

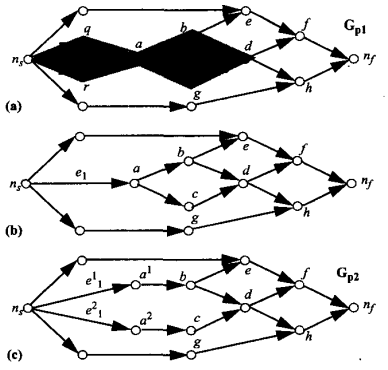


Figure 1. Dependent arrival time computation for node *n*. In (a), the intersection graph of n_f is shaded and dependence nodes of n_f are black.

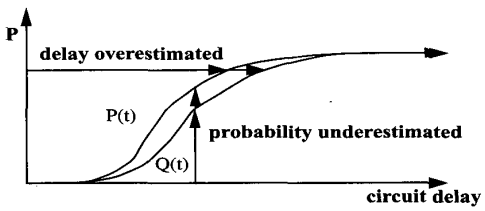


Figure 2. CDF $Q(t)$ is a conservative bound on CDF $P(t)$.

2. Enumerate the pairs (t_i, p_i) of the arrival time PDF at n_d and for each pair propagate t_i with conditional probability p_i.
3. Propagate t_i, using independent arrival time propagation until the next dependence node is encountered and repeat step 2.
4. Compute the final arrival time PDF at node x by summing the conditional arrival time PDFs weighted by the product of their conditional probabilities.

Since we recursively enumerate the arrival time PDFs of all dependence nodes, the complexity of this approach grows exponentially with the number of dependence nodes in a circuit. It can be shown that the set of nodes at which arrival times are enumerated is the sufficient and necessary set for exact computation of the arrival time CDF. It is therefore not possible to enumerate fewer nodes without creating arrival time dependencies in the circuit.

3 Statistical bounds

We now propose an efficient method for computing lower and upper bounds on the exact arrival time CDF of G_P. We define the upper and low bounds of a CDF as follows:

Definition 5. The arrival time CDF $Q(t)$ is a statistical upper bound of the arrival time CDF $P(t)$ if and only if for all t, $Q(t) \le P(t)$.

A similar definition can be formulated for lower bounds. Figure 2 shows two arrival time CDFs $P(t)$ and $Q(t)$, where $Q(t)$ is an upper bound on $P(t)$. Note that the upper bound $Q(t)$ is itself a valid CDF and that all confidence points are bounded by $Q(t)$ on $P(t)$. By using CDF $Q(t)$ instead of $P(t)$, we will overestimate the delay corresponding to a performance yield, resulting in a conservative analysis for late arrival times, as shown in Figure 2. Similarly, for a particular required delay, the probability that a die will meet this delay constrain will be underestimated.

To efficiently compute an upper bound on the exact graph delay CDF of G_p, we propose the following theorem for random variables:

Theorem 2. Let x, y and z be independent random variables that satisfy Property 1. Let x_1, x_2 be random variables with CDFs that are identical to the CDF of x. The CDF of random variable $max(x_1+y, x_2+z)$ is an upper bound on the CDF of the random variable $max(x+y, x+z)$.

The proof is omitted for brevity and is available in [10]. We can graphically illustrate Theorem 2 as follows. Consider the simple graph G_{p1} shown in Figure 3(a) with delay equal to $max((d_a+d_b+d_d), (d_a+d_c+d_e))$, where d_i is the delay of edge i. Fig-

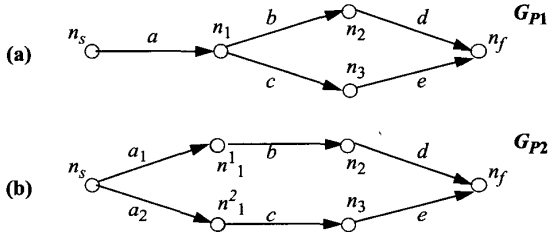


Figure 3. Bounded graph transformation through node splitting.

ure 3(b) shows the timing graph G_{P2} where edge a is split into edges a_1 and a_2, each with the same delay CDFs as a. The graph delay of G_{P2} is $max((d_{a1}+d_b+d_d), (d_{a2}+d_c+d_e))$. From Theorem 2, it follows that G_{P2} has a graph delay CDF that is an upper bound on graph delay CDF of the graph G_{P1}. In fact, it is clear that the CDF of arrival times at all nodes in G_{P2} are upper bounds on the CDF of arrival times of corresponding nodes in G_{P1} and hence we refer graph G_{P2} as an *upper bound* on graph G_{P1}. In general, G_{P1} can have a more complex structure with additional fanin and fanout edges at node $n2$, etc. It can be shown that for a general timing graph G_{P1}, splitting an edge into multiple edges, as illustrated in Figure 3, results in a graph G_{P2} that is an upper bound on G_{P1}. Based on Theorem 2, we now pose the following useful Corollary:

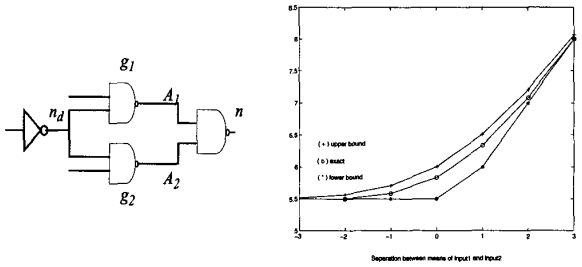
Corrolary 1. If for graph G_P, arrival times are computed for all nodes using the procedure of independent arrival time propagation, the computed arrival time CDFs will be an upper bound on the true arrival time CDFs at those nodes.

The validity of Corrolary 1 can be seen by considering the timing graph G_{P1} with dependence node a, as illustrated in Figure 1(a). Following the procedure for dependent arrival time propagation, we replace subgraph $G_{S,1}$ with a single edge e_1, as shown in Figure 1(b), where the edge delay CDF of e_1 is equal to the arrival time CDF at a. We now create a graph G_{P2}, as shown in Figure 1(c), which bounds G_{P1} by splitting edge e_1, such that a is no longer a dependence node in G_{P2}. By repeating this process for all dependence nodes, we obtain a timing graph $G_{P,k}$ that bounds the original timing graph G_{P1} and which has no dependence nodes. We can compute the exact arrival times of $G_{P,k}$ by performing independent arrival time propagation. Finally, it is easy to observe that we need not explicitly split edge e_1. Instead, we will compute identical arrival times to those of $G_{P,k}$ by simply performing independent arrival time propagation on graph G_{P1}, as stated in Corrolary 1. This leads to the useful observation that an upper bound on the arrival times of a timing graph G_p is obtained by ignoring the dependencies of arrival times and simply applying the procedure for independent arrival time computation, which has a linear run time complexity with circuit size. Similarly, a lower bound can be computed as discussed in [10].

4 Selective Enumeration

In order to improve the quality of the bounds described in Section 3, we combine the bound computation with enumeration of a small set of dependence nodes. Enumeration of dependence nodes improves the computed bounds in two ways. First, the enumeration partitions the sample space S and thereby reduces the dependencies of arrival time CDFs. Second, when all dependence nodes of a particular convergence node are enumerated, the arrival times at this convergence node become independent and the lower bound can be computed using their statistical maximum according to EQ2, instead of the minimum operation used for computing the lower bound of dependent arrival times. Our objective is to obtain the maximum improvement in the bounds through enumeration of a minimum number of dependence nodes. We therefore need to select those dependence nodes that most strongly impact the quality of the bounds at the sink node. For simplicity, we measure the difference between the upper and lower bounds as the difference of their expected values and refer to this measure as the *bound difference.*

To illustrate the factors that influence the effectiveness of enumerating a particular dependence node, we consider the circuit shown Figure 4(a) with two correlated arrival times A_1 and A_2 that converge at node n. We compare the upper and low bounds as well as the exact arrival time at node n, while shifting the alignment of the CDF of A_1 relative to the CDF of A_2, by varying the delay of gate g_1 as shown in Figure 4(b). As the shift between A_1 and A_2 increases, the two



(a) (b)

Figure 4. Expected values of bounds and exact arrival time at node n as a function of the shift between expected values of A_1 and A_2.

bounds rapidly converge to the true arrival time since the later arrival time dominates the result and the dependence between the two arrival times has little impact. Similarly, it is also possible that, while two dependent arrival times give rise to a large bound difference at a node, this bound difference does not propagate to the sink node. This occurs when along the propagation path, the arrival time bound combines with another arrival time bound that is aligned significantly later and that dominates. Therefore, enumeration of a dependence node is effective only if its arrival time PDFs align at one or more convergence nodes and if the sink node is not shielded from the arrival times at these convergence nodes.

Finding the minimum set of dependence nodes to obtain a required improvement in the bound difference is clearly an intractable problem and requires a heuristic solution. We first compute an *enumeration merit* for each dependence node n_d, which is a measure of the expected improvement in the bound difference by enumerating n_d, as discussed in the following section. We then use an iterative approach where dependence nodes are added to the set of enumerated nodes in decreasing order of their enumeration merit. If in a particular iteration the added dependence node does not significantly improve the bound, it is removed from the set of enumerated nodes before a new dependence node is added. The algorithm is shown in pseudo-code in Figure 5. The algorithm continues until either a required bound difference has been obtained, or the allowed run time is exceeded.

```
1. Order list of dependence nodes by decreasing merit
2. While (bound difference is not met and run time is not exceeded) {
3.   Add dependence node with maximum merit to enumeration list
4.   Recompute upper and lower bounds using the enumeration list
5.   If (bound difference at n_f did not improve sufficiently)
6.         remove last node from enumeration list.
7. }
```

Figure 5. Selective enumeration algorithm

In each iteration of the algorithm, we enumerate the selected dependence nodes and compute bounds on the CDF of the graph delay. Similar to exact statistical STA, we enumerate all possible arrival time values t_i of a selected dependence node n_d and compute bounds on the graph delay weighted by the conditional probability p_i that the arrival time at n_d has a value t_i. This approach can be generalized such that, instead of considering each individual value t_i in the discrete PDF of an arrival time, we split the PDF into several *partial* PDFs $p_j(t)$, each partial PDF consisting of delay values in an interval $[t_{j,s}, t_{j,e}]$, as shown in Figure 5. We then compute the upper bound $p_{j,upper}(t)$ and lower bound $p_{j,lower}(t)$ on the graph delay using the method discussed in Section 3, where the arrival time PDF at node n_d is replaced with one of the partial PDFs $p_j(t)$. Before propagating each partial PDF, we first scale it to have an area of 1, to ensure that it is a valid PDF. Each case j corresponds an arrival time at n_d in the interval $[t_{j,s}, t_{j,e}]$ and has a probability P_j of occurrence, equal to the

area of $p_f(t)$. We therefore again computed the final bound $p_{bound}(t)$ on the PDF of the graph delay using conditional probabilities, as follows:

$$p_{bound}(t) = \sum_j P_j \cdot p_{j,bound}(t) \qquad \text{(EQ 3)}$$

Partial enumeration of arrival times requires fewer bound computations than full enumeration, and therefore reduces the computational effort. The number of intervals used for enumeration of a dependence node therefore provides a trade-off between the number of nodes that can be enumerated and the granularity of their enumeration. In practice, it was found that two intervals resulted in the most efficient bound reduction.

Merit computation.

We employ a heuristic method to compute the enumeration merit for a dependence node. For simplicity, we limit our discussion to the enumeration merit for the upper bound, while similar considerations apply to the lower bound. For the upper bound, an improvement of the bound through selective enumeration can be thought of as a shift of the bound PDF to the left (corresponding to lower arrival time values) so that it matches more closely with the exact arrival time PDF. We therefore use the expected shift of the PDF as an indication of the effectiveness of enumeration of a dependence node.

For each dependence node n_d, arrival times with non-zero probability fall in a finite window $[t_d^{min}, t_d^{max}]$. We consider two intervals in this window: a left interval $w_{l,d} = [t_{l,d}^{min}, t_{l,d}^{max}]$ and right interval $w_{r,d} = [t_{r,d}^{min}, t_{r,d}^{max}]$, where both $t_{l,d}^{max}$ and $t_{r,d}^{min}$ are equal to the median of the arrival time PDF, as shown in Figure 6(a). The integral of the arrival time PDF over each interval is equal to 0.5 and each interval represents 50% of all possible arrival times at dependence node n_d, over all die.

We now propagate the start and end points of intervals $w_{l,d}$ and $w_{r,d}$, using conventional timing analysis, to obtain corresponding intervals $w_{l,f} = [t_{l,f}^{min}, t_{l,f}^{max}]$ and $w_{r,f} = [t_{r,f}^{min}, t_{r,f}^{max}]$ at the sink node n_f, as shown in Figure 6(b). During interval propagation, we compute a left interval $w_{i,l}$ and a right interval $w_{i,r}$ for a node n_i as follows:

- Given a left interval $w_{l,j} = [t_{l,j}^{min}, t_{l,j}^{max}]$ that is propagated through a gate g with minimum gate delay d_g^{min} and maximum gate delay d_g^{max}, the left interval at the output of the gate is $w_{l,i} = [t_{l,j}^{min} + d_g^{min}, t_{l,j}^{max} + d_g^{max}]$. The right interval is computed likewise.
- Given several left intervals $[t_{l,1}^{min}, t_{l,1}^{max}]$, $[t_{l,2}^{min}\ t_{l,2}^{max}]$,..., that converge at a node, the combined left interval is $[\max(t_{l,1}^{min}, t_{l,2}^{min},...), \max(t_{l,1}^{max}, t_{l,2}^{max},...)]$. The right interval is computed likewise.

For nodes that do not fall in the fanout cone of dependence node n_d, left and right intervals are taken to be equal to the total interval at that node.

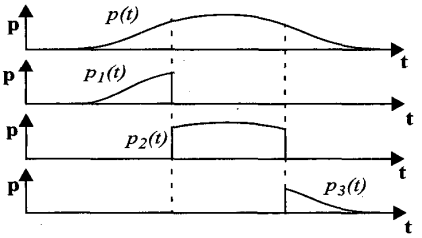


Figure 5. Arrival time PDF and partial PDFs for three intervals.

The computed interval $w_{l,f}$ ($w_{r,f}$) indicates the earliest and latest possible arrival times at the sink node n_f, resulting from an arrival time at the dependence node n_d that falls in the interval $w_{l,d}$ ($w_{r,d}$). Also, the two intervals $w_{l,f}$ and $w_{r,f}$ at the sink node will overlap, meaning that $t_{l,f}^{max} > t_{r,f}^{min}$ due to the uncertainty of gate delays and the merging of intervals at convergence nodes. Since the probability of an arrival time occurring in either interval $w_{l,d}$ or $w_{r,d}$ at node n_d is 0.5, the probability of an arrival time occurring in either interval $w_{l,f}$ and $w_{r,f}$ at the sink node will be greater than or equal to 0.5, due to the overlap of the intervals, as shown in Figure 6(b). Hence, *after* enumeration of dependence node n_d, the area of the arrival time PDF over either interval will be greater or equal to 0.5, as shown in Figure 6(c). However, due to the inherent error in bound computation, the arrival time PDF at the sink node *without* enumeration can be significantly skew to the right, as shown in Figure 6(d). In this case, the area under the left interval can be less than 0.5. The amount by which the area is less than 0.5 before enumeration is a good indicator of the expected shift of the arrival time PDF resulting from enumeration. The enumeration merit of dependence node n_d is hence computed as follows:

$$merit_d = \begin{cases} 0.5 - a_{l,f} & if\ (a_{l,f} < 0.5) \\ 0 & otherwise \end{cases}, \qquad \text{(EQ 4)}$$

where $a_{l,f}$ is the area of the arrival time PDF at the sink node over the left interval $w_{l,f} = [t_{l,f}^{min}, t_{l,f}^{max}]$. Note that the enumeration merit of a dependence node for *lower bound* computation can be computed similarly by observing the amount by which the arrival time PDF at the sink node over the *right* interval is less than 0.5.

A dependence node will have a high enumeration merit for upper bound computation if its left and right intervals at the sink node do not overlap significantly and if the PDF at the sink node is skewed towards the right interval, as shown in Figure 7(a). This occurs when the arrival times of the dependence node align at a convergence node and if the convergence node is not shielded from the sink node. On the other hand, if the arrival times do not align at a convergence node, the PDF at the convergence node has no significant skew relative to the intervals, as shown in Figure 7(b), and the computed enu-

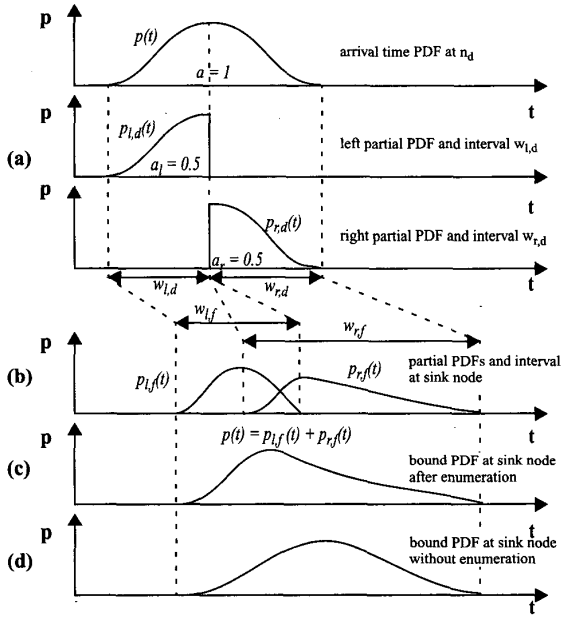


Figure 6. Computation of partial PDFs and intervals.

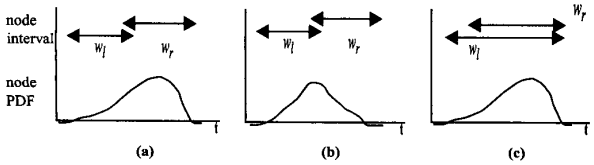


Figure 7. Left and right arrival time intervals and bound PDFs.

meration merit will be small or zero. Also, if the arrival time PDF at a convergence node is shielded from the sink node, the left and right intervals at the sink node will be largely overlapping, as shown in Figure 7(c), thereby also resulting in a low enumeration merit.

Each dependence node requires a separate propagation of left and right intervals. However, for implementation efficiency, we propagate intervals for all dependence nodes simultaneously. The list of propagated left and right intervals is pruned at each node in the graph and only the k smallest left and right intervals are propagated. This ensures linear computational complexity of the merit computation with circuit size. In our experiments k was set to 20. Note that different enumeration nodes are selected for the lower and upper bounds and each bound is computed separately. Finally, we verified the effectiveness of the proposed enumeration merit computation by comparing the ranking of all dependence nodes based on their enumeration merit with their ranking based on the actual improvement of the bound when a dependence node is enumerated. It was found that there is a good match between the two rankings and that 90% of the nodes matched between the top 10% nodes of both rankings.

5 Results

The upper and lower bound computation, as well as the proposed refinement method were implemented and tested on the ISCAS [11] benchmark circuits. The computed bounds were compared with either exact statistical timing analysis, discussed in Section 2, or with Monte Carlo simulation, for larger circuits. For each gate in the circuit, a delay distribution was specified with a standard deviation ranging from 10% - 15% of the mean of the distribution. A Gaussian distribution, truncated at the 3 sigma point, was used.

Table 1. Results of bound computation and refinement.

circuit	nodes		monte carlo	bound		selective enum bound		
	total	dep. n.		lower / upper	% dif	lower / upper	% impr	time (sec)
c17	13	3	1.4	1.37 / 1.42	4	1.40 / 1.40	100	8
c499	198	53	7.7	7.45 / 8.06	8	7.61 / 7.77	73	43
c432	245	59	5.2	4.73 / 5.28	10	4.98 / 5.22	58	40
c880	445	64	9.3	9.06 / 9.44	4	9.24 / 9.30	86	34
c1355	589	250	10.2	9.44 / 10.4	10	9.73 / 10.4	37	117
c1908	915	249	14.5	14.3 / 14.8	4	14.5 / 14.7	76	45
c2670	1428	268	12.8	12.5 / 13.1	5	12.7 / 13.0	65	49
c3540	1721	480	17.0	16.7 / 17.4	4	16.9 / 17.2	60	66
c5315	2487	264	17.3	17.3 / 17.7	2	17.3 / 17.5	43	76
c6288	2450	1197	46.9	45.2 / 48.6	7	45.7 / 48.3	22	187
c7552	3721	1050	15.9	15.6 / 16.1	3	15.8 / 16.0	67	150

Table 1 shows the results for the bound computation and selective enumeration. For each circuit, the total number of nodes (*column* 2) and the total number of dependence nodes (*column* 3) are shown. The statistical upper and lower bounds (*column* 5) have a relatively small difference in their expected value ranging from 2.25% to 10.45% (*column* 6). Although we only report the expected value in Table 1, the computed bounds are CDFs and allow the computation of other confidence points. For all circuits, the run time of the bound computation did not exceed 5 seconds. Also, the Monte Carlo results (*column* 4) fall between the computed bounds, as expected.

In Table 1 we also show the bounds after selective enumeration (*columns* 7) and the percentage improvement of their difference compared to the original bounds (*column* 8). Excluding the first circuit, the improvement of the bounds using selective enumeration ranged between 22 - 86% with an average of 62%. For these circuits, the number of dependence nodes selected for enumeration was small, ranging from 9 to 13 nodes, showing the somewhat surprising result that enumerating only a few carefully chosen nodes can significantly improve the bound. The run time of the selective enumeration method (*column* 9) is modest, not exceeding 187 seconds for all test cases. Figure 8 shows the CDFs for the proposed lower and

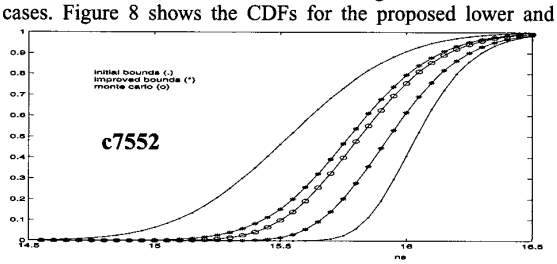


Figure 8. Comparison of CDF bounds and Monte-Carlo CDF

upper bound with and without selective refinement as well as the CDF obtained through Monte-Carlo simulation for the circuit c7552.

6 Conclusions

In this paper, we have proposed an efficient method for computing bounds on the statistical behavior of circuit delay due to within-chip process variations. We first presented an exact statistical timing analysis method. Since this method has exponential run time complexity with circuit size, we show how statistical bounds on the graph delay can be computed with linear run time complexity. In order to reduce the difference between the bounds, we proposed an iterative refinement technique which selectively enumerates dependence nodes in the circuit.

Acknowledgements

This research was supported by SRC contract 2001-HJ-959 and NSF grant CCR-0205227.

References

[1] V.Mehrotra, S.L.Sam, D.Boning, et al, "A methodology for modeling the effects of systematic within-die interconnect and device variation on circuit performance," Proc. DAC 2000.

[2] S. Devadas, H. F. Jyu, K. Keutzer, S. Malik, "Statistical timing analysis of combinational circuits ", Proc. ICCD 1992, pp. 38 - 43

[3] M. Berkelaar, "Statistical Delay Calculation, a Linear Time Method," Proc. TAU 1997

[4] J.J Liou, K.T. Cheng, S. Kundu, A. Krstic, "Fast Statistical Timing Analysis By Probabilistic Event Propagation", Proc. DAC 2001

[5] M. Orshansky, K. Keutzer, "A general probabilistic framework for worst-case timing analysis", Proc. DAC 2002.

[6] A. Gattiker, S.Nassif, R.Dinakar, C.Long "Timing Yield Estimation from Static Timing Analysis", Proc. ISQED 2001

[7] X. Bai, C. Visweswariah, P. Strenski, D. Hathaway, "Uncertainty-aware circuit optimization", Proc. DAC 2002.

[8] F. Najm, R. Burch, P. Yang, I. Hajj, "Probabilistic simulation for reliability analysis of CMOS VLSI circuits" IEEE Trans. on CAD, 1990

[9] Feller, W., P. "An Introduction to Probability Theory and its Applications", Vol. 1,2 John Wiley & Sons, New York, 1970.

[10] A.Agarwal, D.Blaauw, V.Zolotov, S.Vrudhula, "Statistical Timing Analysis using Bounds", Proc. DATE 2003.

[11] F. Brglez, H.Fujiwara, "A Neutral Netlist of 10 Combinatorial Benchmark Circuits", Proc. ISCAS, 1985, pp.695-698