Statistical Timing Analysis using Bounds

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 the dependencies created by reconverging paths in the circuit. In this paper, we propose a new approach to statistical timing analysis which uses statistical bounds. First, we provide a formal definition of the statistical delay of a circuit and derive a statistical timing analysis method from this definition. Since this method for finding the exact statistical delay has exponential run time complexity with circuit size, we also propose a new method for computing statistical bounds which has linear run time complexity. We prove the correctness of the proposed bounds. Since we provide both a lower and upper bound on the true statistical delay, we can determine the quality of the bounds. The proposed methods were implemented and tested on benchmark circuits. The results demonstrate that the proposed bounds have only a small error.

1 Introduction

Static timing analysis has become an indispensable part of performance verification. Traditionally, the variation in the underlying process parameters were 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 time using one case file. Each execution of the 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 in device and interconnect delay, it is not able to accurately model variations within a single die. With the continual scaling of feature sizes, the ability to control critical device parameters on a single die has become increasingly difficult. Using a worst-case analysis for these so-called *within-die* variations therefore leads to very pessimistic analysis results since it assumes that all devices on the 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, giving rise to the need for new statistical timing analysis approaches.

Variations in the delays of a circuit can be broadly classified into two categories: *environmental* variations and *process* variations. Environmental variation are caused by uncertainty in the environmental conditions during the operation of a chip, such as power supply and temperature variations. Process variations are due to uncertainty in the device and interconnect characteristics, such as effective gate length, doping concentrations, and oxide thickness. 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 topologically correlated lithographic distortions. In some cases, such topological dependencies can be directly accounted for in the analysis, thereby reducing the statistical variation [11], whereas in other cases, such variations are treated as random.

In this paper, we propose a formal model and efficient analysis method for statistical STA in the presence of 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. We initially also do not address environmental variations, although the proposed model and analysis methods can be extended to such variations.

The extensive use of deterministic STA in recent years is in large part due to its linear run time complexity 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 their path delays due to shared sections of such paths. Previous statistical STA approaches [4-9] have therefore suffered from very high runtimes, from the use of approximate methods with unclear accuracy impact, or from unrealistic assumptions, such as an assumed Gaussian distribution of gate delays.

In this paper, we propose a new method for statistical STA. Since the formulation of statistical STA has varied in subtle but important ways in the literature, we first provide a formal model of statistical STA. We then derive the proposed procedure for statistical STA in a strict manner from this problem formulation. Since the computational complexity of exact statistical STA is exponential with the circuit size, we present a new method for computing bounds on the exact statistical delay of the circuit and proof the correctness of these bounds. By computing only bounds of the true statistical behavior of the circuit, we are able to preserve the important characteristic of deterministic STA that it has a linear run time complexity with circuit size. Since we provide both a lower and upper bound on the true statistical delay, we can determine the quality or error of the computed bounds. The proposed method provides a statistical STA approach with linear run time, that is guaranteed conservative and has a bounded error. The proposed methods were implemented and tested on large benchmark circuits. The difference between the expected values of the upper and lower bound was shown to be small, ranging from 4 - 12%.

The remainder of this paper is organized as follows. In Section 2 we presents a formal model of statistical STA. In Section 3, we present a number of probabilistic timing graph transformations. In Section 4, we derive our method for exact statistical timing analysis. In Section 5, we present the computation of the lower and upper sta-

Proceedings of the Design,Automation and Test in Europe Conference and Exhibition (DATE'03)
1530-1591/03 $17.00 © 2003 IEEE

tistical bounds on the true statistical behavior. In Section 6 we present our results and in Section 7 our concluding remarks.

2 Statistical Timing Analysis Formulation

In this Section, we present a formal model of statistical static timing analysis. Our goal is to model the impact of gate delays variation 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.

At this point, we do not account for temporal variations of the gate delays due to environmental factors, such as power supply fluctuations, temperature dependence and noise, which must be modeled using case analysis. However, our general analysis approach can be extended to these types of variations as well. Also, for simplicity of formulation, we ignore the presence of false paths since these are orthogonal to the issues discussed in this paper. 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 $e=(n_i,n_j)$.

The nodes in the timing graph correspond to nets in the circuit, and the edges in the graph correspond to connections from gate inputs to gate outputs. Although circuits generally have multiple inputs and outputs, we can trivially transform them to graphs with a single source and sink by adding a virtual source and sink node.

In our formulation, a *deterministic* timing graph G_D represents a particular manufactured die, where each gate has a fixed delay value. Each edge e in G_D is assigned a delay $D(e)$, which represents the deterministic signal propagation delay from a gate's input to its output. Similar to other statistical STA methods, we ignore the dependence of gate delay on the transition time of the gate input signals.

A path P of a timing graph G is a sequence of nodes, such that each pair of adjacent nodes n_g and n_h has an edge $e_{gh}=(n_g,n_h)$. The path delay d_P of path P is the sum of all the delays $D(e_{ij})$ of edge e_{ij} on path P. Among all paths terminating at a node n, we define the path with the maximum delay as the critical path of n. The delay of the critical path of node n is equal to its *arrival time*, $t_a(n)$. Note that the arrival time for the source node n_s is a deterministic value equal to 0. The critical path of the sink node n_f of a timing graph is referred to as *the* critical path of the timing graph, and the arrival time of n_f, is referred to as its *graph delay.*

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 distribution function (*CDF*) or probability density function (*PDF*) and we define a *probabilistic* timing graph G_P as follows:

Definition 2. A probabilistic timing graph G_P is a timing graph whose edges are assigned random variables of delay values.

Figure 1 shows an example of a delay cumulative distribution function and its corresponding probability density function. Since these functions represent the variation of gate delays, they have the following obvious but important property:

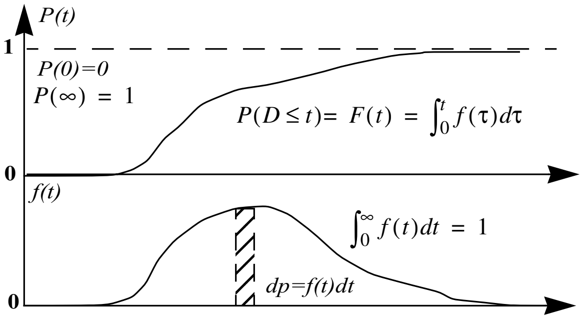


Figure 1. Delay CDF and PDF

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

This property follows from the fact that the delay of a real gate cannot be less that some finite minimum delay value d_{min} or more than some finite maximum delay value d_{max}. Similar to previous statistical STA methods [4 - 7], we assume statistical independence of all edge delays. In practice, edge delays may be spatially 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. The methods presented in this paper can also be extended to timing graphs with correlated edge delays. Note also that our method does not restrict the shape of the PDF of edge delays, which in general is not Gaussian.

To simplify the implementation of statistical STA it is often more convenient to approximate continuous probability density and distribution functions with discrete functions. A discrete PDF, corresponding to continuous PDF $f(t)$, can be represented by a sequence of pairs (d_i,p_i), where $d_i = i\Delta$ and $p_i = \int_{d_i - \Delta/2}^{d_i + \Delta/2} f(\tau)d\tau$. For computational efficiency, we use discrete PDFs and CDFs in the final implementation of our proposed statistical timing analysis approaches. However, for generality, we will formulate the statistical timing analysis task using continuous functions.

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 cumulative distribution functions. The probability that a timing graph G_D in S has an edge i with delay D_i between t_i and T_i is

$$P(t_1 \le \tau_1 \le T_1, t_2 \le \tau_2 \le T_2, \ldots) = \qquad \text{(EQ 1)}$$
$$\int_{t_1}^{T_1}\int_{t_2}^{T_2} \ldots p_1(\tau_1)p_2(\tau_2)\ldots d\tau_1 d\tau_2 \ldots$$

where $p_i(\tau_i)$ is the probability density function of edge i. 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 which completely defines its timing behavior. The CDF of $D(G_P)$ is defined as follows:

Definition 3. The cumulative distribution function of the delay of a probabilistic timing graph is expressed as:

$$P(D(G_P) \le t) \;=\; \int_{D(G_D) \le t} p_1(t_1)p_2(t_2)\dots dt_1 dt_2\dots \;, \qquad \text{(EQ 2)}$$

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 probability density function can be computed by simple differentiation. The CDF 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 this constraint, also called the performance yield, can be determined. Conversely, given a required performance yield, the maximum expected performance can be obtained.

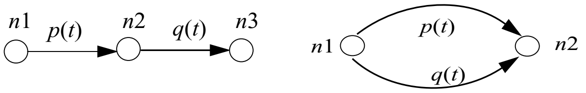
If we use discrete edge delay PDFs, we can compute the graph delay PDFs by enumerating the entire sample space consisting of all permutations of the non-zero delay probabilities of all edges. Of course, this method is exponential in its run time complexity with circuit size and is not useful as a practical solution. However, its formulation is useful as a formal definition and for understanding the underlying problem that needs to be solved.

3 Probabilistic Timing Graph Transforms

Before we discuss exact and bounded methods for computing the CDF of the graph delay, we briefly discuss three basic transformations for probabilistic timing graphs.

3.1 Series Reduction.

Figure 2(a) shows a probabilistic timing graph consisting of two series connected edges with delays described by pdf *p(t)* and *q(t)*. The total delay of the timing graph is the sum of its edge delays and by applying EQ2, the CDF of the graph delay $D_{Gp}(t)$ is:



(a) (b)

Figure 2. Series and parallel reduction

$$P(D_{Gp} \le t) \;=\; \int_{t_1 + t_2 \le t} p(t_1) \cdot q(t_2) dt_1 dt_2 \qquad \text{(EQ 3)}$$

The sum of the two independent edge delays is the convolution of its edge delays, as is well known from standard probability theory [3], and the two edges can be replaced with a single edge having the following probability density function:

$$p_{Gp}(t) \;=\; \int_0^{\infty} p(t - \tau) \cdot q(\tau) \cdot d\tau \qquad \text{(EQ 4)}$$

The cumulative distribution function of the graph delay is obtained through integration of EQ4.

3.2 Parallel Reduction.

Figure 2(b) shows a timing graph G_P consisting of two parallel edges with delays described by pdfs *p*(*t*) and *q*(*t*). Since the edge delays of both edges are statistically independent, the probability $P(D_{Gp} \le t)$ is the product of the probabilities that each edge delay is less than or equal to *t*:

$$P_{G_P}(t) = P(t) \cdot Q(t) \qquad \text{(EQ 5)}$$

Through differentiation, we obtain the probability density function of the graph delay $D(G_P)$ as follows:

$$p_{G_P}(t) = P(t) \cdot q(t) + p(t) \cdot Q(t), \qquad \text{(EQ 6)}$$

Therefore, the graph in Figure 2(b) can be replaced with a single edge have the pdf $p_{Gp}(t)$.

4 Statistical Timing Analysis

The initial formulation presented in the Section 2 relies on the enumeration of all possible edge delays with non-zero probability and is difficult to use for an efficient solution to the problem. 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 CDF D_{Gp}, in a manner that is consistent with the definition of D_{Gp} in Section 2. We first define the cumulative distribution of the latest arrival time, $A_n(t)$ at node *n* as follows:

Definition 4. The latest arrival time CDF, $A_n(t)$ at node *n* of G_P is the probability that a deterministic timing graph G_D in the sample space $S(G_P)$ has an arrival time $t_a(n) \le t$.

In the subsequent discussion, we will refer to the latest arrival time as simply *the* arrival time, noting that a similar derivation can be performed for the earliest arrival time. We also make the following useful definition.

Definition 5. 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*, and where node *n* is set as the sink node n_f of $G_{S,n}$.

From Definition 4 and 5, it follows that the arrival time A_n at node *n* is equivalent to the graph delay of subgraph $G_{S,n}$. Hence, computing arrival time distributions and graph delay distributions are equivalent problems. The objective of statistical timing analysis is to compute the arrival time CDF of node *n* based on the arrival time CDFs of its predecessor nodes n_p. We can then use such a method to propagate arrival times through the circuit in topological fashion. To compute the arrival time of *n*, we must consider if the arrival times of its predecessor nodes n_p are independent random variables. We now 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 any common edges have a deterministic delay.

The proof follows from the fact the arrival times $A_{n,i}$ and $A_{n,j}$ are composed of edges delays in their fanin subgraphs, and hence have no shared random variables. The proof is omitted for brevity.

Arrival time propagation.

If all arrival times are independent, the circuit has a tree like structure and arrival times can be computed using the max function of EQ5 and convolution of EQ4 in linear time. To determine if two arrival times are dependent, we consider a node *n* with predecessor nodes n_p which have arrival times A_p and fanin subgraphs $G_{S,p}$. If the fanin subgraphs $G_{S,p}$ share one or more edges with random delays, the arrival times A_p will be *dependent* random variables and the statistical maximum function cannot be applied. An example of such a graph is shown in Figure 3(a). To determine for which portions the subgraphs $G_{S,p}$ share edges, we use the following definition of a *dependence* set.

Definition 6. Consider the set of k predecessor 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 3(a), the set of dependence nodes for node n_f is {b, d}. Note that a is not in the dependence set of n_f since both its fanout edges belong to G_I. Also nodes h and e have empty dependence sets. Conceptually, dependence nodes mark the last points in the graph where the fanin subgraphs are shared and give rise to correlation between their arrival times.The concept of dependence nodes is similar to that used in probabilistic simulation [12]. We refer to a node in G_P as a *convergence* node n_c if it has a non-empty dependence set. We also define the global set of dependence nodes n_D as the union of the dependence sets $n_{d,i}$ and refer to a node as a dependence node, if it is an element in this list.

In order to compute the graph 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 3(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 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 3(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 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 [3] the arrival time pdf $p_x(t)$ at node x, can be computed as $p_x(t) = \sum_{i=0}^{k} p_i \cdot p_{x,i}(t)$, where $p_{x,i}(t)$ is the arrival time pdf at node x when the delay 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.

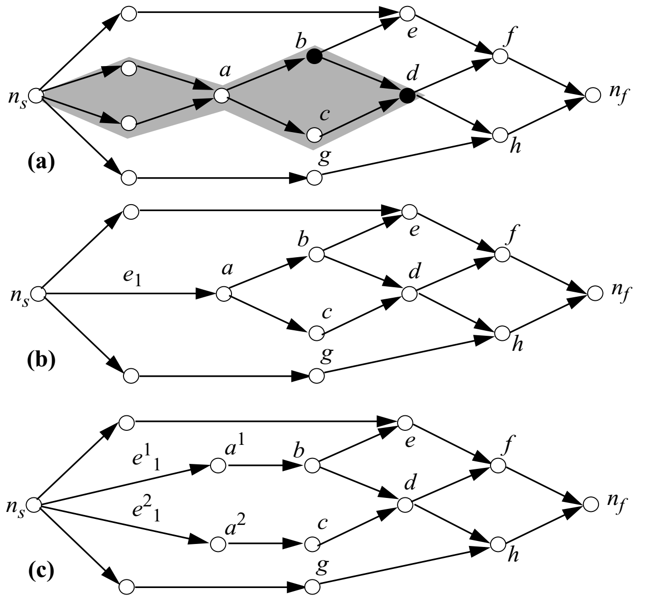


Figure 3. Dependent arrival time computation for node *f*. Intersection graph is shaded and dependence nodes are marked black.

Below is the procedure for dependent arrival time propagation:

1. Identify all dependence nodes in the circuit.
2. Propagate arrival time PDFs in the circuit until the first dependence node n_d is encountered.
3. Enumerate the pairs (t_i, p_i) of arrival time PDF A_d at n_d and for each pair propagate t_i with conditional probability p_i.
4. Propagate t_i, using independent arrival time propagation until the next dependence node is encountered and repeat step 3.
5. 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 total dependence nodes in a circuit. Note, however, that the number of dependence nodes is, in practice, significantly less then the number of edges in G_P. Dependent arrival time propagation therefore has a lower complexity than enumeration of the entire sample space. It can be shown that the set of nodes at which arrival times are enumerated is the sufficient and necessary set for exact arrival time computation. It is therefore not possible to enumerate fewer nodes without creating arrival time dependencies in the circuit. Nevertheless, dependent arrival time propagation is useful only for very small timing graphs or timing graphs with mostly tree-like structures.

5 Statistical bounds

We now propose an efficient method for computing lower and upper bounds of the exact arrival time CDF of G_P. We are interested in both upper and lower bounds since this allows us to determine the quality of the bounds by comparing their difference. Also, for digital circuits, analysis of both slow and fast paths are important for correct circuit operation and therefore a lower bound on the *earliest* arrival time CDF could be useful. We define the upper and low bounds of a CDF as follows:

Definition 7. The arrival time CDF $P(t)$ is an upper bound of the arrival time CDF $Q(t)$ if and only if for all t, $P(t) \le Q(t)$.

A similar definition and property can be formulated for lower bounds. Figure 4 shows two arrival time CDFs $P(t)$ and $Q(t)$, where $P(t)$ is an upper bound on $Q(t)$. Note that the upper bound $P(t)$ is itself a valid CDF and that not only the expected value μ_p of $Q(t)$, but also other characteristics, such as the 95% confidence point, are

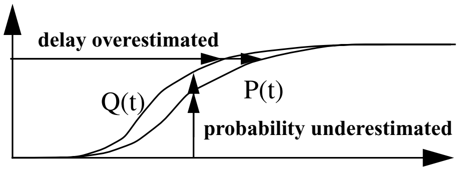


Figure 4. Lower and upper bounds of a CDF

bounded by $P(t)$ on $Q(t)$. By using CDF $P(t)$ instead of $Q(t)$, we will overestimate the delay corresponding to a particular probability or performance yield, resulting in a conservative analysis for late arrival times, as shown in Figure 4. Similarly, for a particular required delay, the probability that a die will meet this delay constrain will be underestimated by using $P(t)$ instead of $Q(t)$.

Upper bound computation.

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 random variables that satisfy Property 1. Let x_1, x_2 be random variables with cumulative distributions that are identical to x. Then, the CDF of random variable $max(x_1+y, x_2+z)$ is an upper bound on CDF of the random variable $max(x+y, x+z)$.

Proof. The cumulative distribution function of random variables $max(x+y, x+z)$ and $max(x_1+y, x_2+z)$ are:

$$Q(t) = \int_{x + max(y, z) \le t} p(x)q(y)r(z)dxdydz \qquad \text{(EQ 7)}$$

$$P(t) = \int_{max(x_1 + y, x_2 + z) \le t} p(x_1)p(x_2)q(y)r(z)dx_1dx_2dydz \qquad \text{(EQ 8)}$$

Multiplying equation EQ7 by the integral of probability density function $p(v)$ from minus infinity to infinity, rearranging some of the terms and renaming integration variables, gives us:

$$Q(t) = \int_{max(x + y, x + z) \le t, -\infty \le v \le \infty} p(x)p(v)q(y)r(z)dxdvdydz \qquad \text{(EQ 9)}$$

Integrals from formulae EQ8 and EQ9 for cumulative distributions $P(t)$ and $Q(t)$ have the same integration functions $f(x_1,x_2,y,z)=p(x_1)p(x_2)q(y)r(z)$ and $f(x,v,y,z)=p(x)p(v)q(y)r(z)$ and differ only in the names of the variables. We now split the $4D$ domain of both functions into two subdomains: $y \le z$ and $y > z$. cumulative distributions $P(t)$ and $Q(t)$ can be represented as the sum of two terms corresponding to the contribution of each subdomain.

$$\begin{aligned} Q(t) &= Q_{y \le z}(t) + Q_{y > z}(t) \\ P(t) &= P_{y \le z}(t) + P_{y > z}(t) \end{aligned} \qquad \text{(EQ 10)}$$

For subdomain $y \le z$ we define a one to one mapping (bijection) so that (x_1,x_2,y,z) corresponds to (v,x,y,z) i.e. $x_1=v$ and $x_2=x$. In this subdomain, inequality $max(x + y, x + z) \le t$ follows from inequality $max(x_1 + y, x_2 + z) \le t$. Therefore, the region of integration for computing $Q_{y \le z}(t)$ in this subdomain includes the integration region for computing $P_{y \le z}(t)$ and hence $P_{y \le z}(t) \le Q_{y \le z}(t)$ because in both cases we integrate the same function.

For subdomain $y > z$ we define a one to one mapping (bijection) so that (x_1,x_2,y,z) corresponds to (x,v,y,z) i.e. $x_1=x$ and $x_2=v$. Similar to the above consideration, $max(x + y, x + z) \le t$ follows from $max(x_1 + y, x_2 + z) \le t$ in this subdomain and the region of integration for computing $Q_{y > z}(t)$ includes the region of integration for computing $P_{y > z}(t)$. Therefore, $P_{y > z}(t) \le Q_{y > z}(t)$.

Combining inequalities for $P(t)$ and $Q(t)$ from each subdomain, we obtain the inequality $P(t) \le Q(t)$ for the whole sample space which proves the theorem. □

We now consider the simple graph G_{p1} shown in Figure 5(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. Figure 5(b) shows the timing graph G_{p2} where edge a is split into edges a_1 and a_2 with the same delay CDFs as a. From Theorem 2, it follows that G_{p2} has a delay CDF that is an upper bound on 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 of the CDF of arrival times of corresponding nodes in G_{p1} and hence we refer graph G_{p2} is an *upper bound* on graph G_{p1}. Based on this graph representation of Theorem 2, we therefore pose the following Corollary:

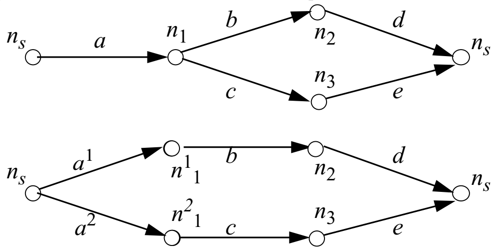


Figure 5. Bounded graph transformation through node splitting.

Corrolary 1. If for graph G_{P1} with one or more convergence nodes, 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 Corrolary 1 can be seen by considering the timing graph G_{p1} with dependence nodes n_D, as illustrated in Figure 3(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 3(b), where the edge delay CDF D_1 of e_1 is equal to the arrival time CDF A_1 at $n_{D,1}$. We now create a graph G_{p2}, which bounds G_{p1} by splitting edge e_1 as shown in Figure 3(c), such that $n_{D,1}$ 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 convergence nodes. We can compute the exact arrival time CDFs of $G_{p,k}$ by performing independent arrival time propagation. Finally, it is easy to observe that we need not explicitly replace subgraph $G_{S,1}$ with edge e_1 and split it, and that 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.

Lower bound computation.

We now discuss the computation of a low bound on the exact arrival time CDFs. Given the CDFs $X(t)$ and $Y(t)$ of two *dependent* random variables x and y and the random variable $z = max(x, y)$, it is clear that the CDF $min(X(t), Y(t))$, as shown in Figure 6, is a lower bound on the CDF of z. This can be seen by considering the graph in Figure 2(b), consisting of two parallel edges with delays x and y and edge delay CDFs $X(t)$ and $Y(t)$, respectively. The probability that the

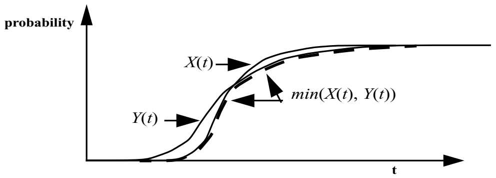


Figure 6. Lower bound for two dependent arrival times.

graph delay $z = max(x, y)$ exceeds a certain delay t is greater than or equal to the probability that either edge delay exceeds t, regardless of the correlation of the x and y. In other words $P(z > t) \geq P(x > t)$, and $P(z > t) \geq P(y > t)$. Since $P(x > t) = 1 - X(t)$, $P(y > t) = 1 - Y(t)$, and $P(z > t) = 1 - Z(t)$, it follows that $Z(t) \leq X(t)$ and $Z(t) \leq Y(t)$, from which it follows that $min(X(t), Y(t))$ is a lower bound on the CDF of z.

The lower bound computation is therefore identical to independent arrival time propagation, except that at convergence nodes, the CDF of the arrival time is computed by taking the minimum of the incoming arrival time CDFs for each time point. The lower bound computation has a linear run time complexity with circuit size.

6 Results

The proposed statistical timing analysis methods for computing upper and lower arrival time bounds were implemented. Also, the exact statistical timing analysis methods through edge enumeration described in Section 2 and dependent arrival time propagation described in Section 4 were implemented. In the cases where it was possible to compute the exact graph delay CDF with these methods, it was used to confirm the correctness of the computed bounds. For larger circuits, Monte Carlo simulation with 100,000 samples was used. To properly stress the proposed methods, fairly large standard deviations were used for the edge delay CDFs in the timing graphs, ranging between 15% and 35% of their mean.

In Table 1, we show the circuit characteristics and results for the ISCAS [10] benchmark circuits. The table shows the average and maximum number of dependence nodes per convergence node (*# dependence*). Some circuits have extensive reconvergence, indicated by their high number of dependence nodes, making them difficult test cases for statistical timing analysis.

Circuit			# dependence		det bound	monte carlo	bound	
name	# edges	# conv	avg	max	low/ upper		low/ upper	%diff
c17	19	3	0.8	3	1.6/2.4	2.24	2.18/2.28	5%
c499	481	38	6.3	56	6/9.6	8.48	7.64/8.7	12%
c432	379	39	7.9	32	8.8/15.0	12.5	11.9/13.1	9%
c880	815	85	6.4	48	11.6/18.4	14.9	14.5/15.3	6%
c1355	1137	240	2.5	26	11.6/19.0	16.7	15.2/17.1	11%
c1908	1556	103	5.9	85	17.4/28.8	23.4	22.7/23.9	5%
c2670	2449	259	5.5	51	15.2/24.6	20.7	19.9/21.4	6%
c3540	3011	480	23.5	275	20.2/33.6	27.4	26.5/28.1	6%
c5315	4687	152	4.2	15	21.4/34.0	27.9	27.6/28.6	4%
c6288	4864	1626	93.7	1142	54.4/89.8	75.7	72.5/79.0	8%
c7552	6459	391	6.1	98	19.6/30.8	25.5	24.7/26.0	5%

Table 1. Circuit statistics and exact reduction improvement

Table 1 also shows the results for the bound computation. The expected value of the lower and upper bound CDFs (*bound low/ upper*) and their relative difference (*bound diff*) is shown. Although we only report the expected value of the bounds in Table 1, the computed bounds are CDFs and allow the computation of other useful values, such as confidence points. We also show deterministic bounds (*det. bound*), obtained by selecting for each edge the minimum or maximum edge delay with non-zero probability and computing the graph delay with deterministic STA. The statistical upper and lower bounds have a relatively small difference in their mean value of 4% to 12%, showing their effectiveness. For all circuits, the run time of the bound computation did not exceed 5 seconds. Also, the Monte Carlo results fall between the computed bounds, as expected. Finally, Figure 7 shows the CDFs for the proposed lower and upper bound as well as the CDF obtained through Monte-Carlo simulation for circuit c7552.

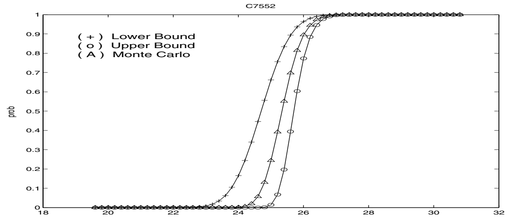


Figure 7. Comparison of CDF bounds and Monte-Carlo CDF

7 Conclusions

In this paper, we have proposed an efficient method for computing bounds on the statistical behavior of the circuit delay due to within-chip process variations. We presented a general method for statistical timing analysis. Since the exact statistical timing analysis 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. We prove the correctness of the upper and lower bounds and demonstrate that the obtained bounds are close.

Acknowledgements

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

References

[1] Hitchcock, R.B. “Timing verification and the Timing Analysis program”, Proc., Design Automation Conf., 1982, pp.594-604

[2] Jouppi, N.P. “Timing analysis for nMOS VLSI”, IEEE/ACM Design Automation Conf., 1983, pp. 411-418

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

[4] J.J Liou, K.T. Cheng, S. Kundu, A. Krstic, “Fast Statistical Timing Analysis By Probabilistic Even Propagation”, DAC 2001

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

[6] M. Berkelaar, “Statistical Delay Calculation, a Linear Time Method,” Proc. of TAU, 1997.

[7] R.B. Brawhear, N. Menezes, C. Oh, L. Pillage, R. Mercer, “Predicting circuit performance using circuit-level statistical timing analysis” European Design and Test Conference, 1994.

[8] R.-B. Lin; M.-C. Wu, “A new statistical approach to timing analysis of VLSI circuits”, Proc. Int. Conf. on VLSI Design, 1998

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

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

[11]M. Orshansky, L. Milor, P. Chen, K. Keutzer, C. Hu, “Impact of systematic spatial intra-chip gate length variability on performance of high-speed digital circuits“, ICCAD 2000, pp. 62 -67.

[12] F. Najm, R. Burch, P. Yang, I. Hajj, “Probabilistic simulation for reliability analysis of CMOS VLSI circuits” IEEE Trans. on CAD, Volume: 9 Issue: 4, April 1990