Statistical Timing Analysis Using Bounds and Selective Enumeration

Aseem Agarwal, *Student Member, IEEE*, Vladimir Zolotov, *Member, IEEE*, and David T. Blaauw, *Member, IEEE*

***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 which uses statistical bounds and selective enumeration to refine these 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. 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. The proposed methods were implemented and tested on benchmark circuits. The results demonstrate that the proposed bounds have only a small error, which can be further reduced using selective enumeration with modest additional run time.**

***Index Terms*—Probability, process variation, ststistical analysis, yield prediction.**

I. Introduction

STATIC timing analysis has become an indispensable part of performance verification. 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 time using one case file. 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 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 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, 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 variations 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, oxide thickness and ILD thickness. A number of methods have been proposed to determine the impact of such variations on the delay of individual gates and interconnect [13]–[17]. 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 topological dependencies of device processing, such as CMP effects and topologically correlated lithographic distortions [3]–[5]. In some cases, such topological dependencies can be directly accounted for in the analysis, thereby reducing the statistical variation [18], [19], whereas in other cases, such variations are treated as random.

In this paper, we propose a formal model and an efficient analysis method for statistical STA in the presence of random within-die process variations. Since between-die variations are 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 also do not address environmental variations, although the proposed model and analysis methods can be extended to such variations.

Deterministic timing analysis has seen significant improvement since it was first introduced [1], [2], including methods to account for effects such as cross-coupling noise [6]–[11] and power supply noise [12]. The extensive use of deterministic STA 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. A second source of correlation between arrival times results from spatial and topological correlation of the individual gate delays. For instance, gates that are positioned within close proximity on a die, or that are similar in

Manuscript received September 24, 2002; revised December 23, 2002. This paper was recommended by Associate Editor F. N. Najm.

A. Agarwal and D. T. Blaauw are with the University of Michigan, Ann Arbor, MI 48105 USA (e-mail: abagarwa@umich.edu).

V. Zolotov is with the Motorola, Inc., Austin, TX 78759 USA.

Digital Object Identifier 10.1109/TCAD.2003.816217

layout topology, are more likely to have similar gate delays after fabrication. Due to the correlation between arrival times, many statistical STA approaches [20]–[27] have either high run times or they ignore the presence of these correlations. Recently, a number of new methods have been proposed to address the increasing significance of process variations. In [28], [29], a novel method using discretized probability distributions is proposed. However, the run time of the method is exponential and the proposed approaches to reduce the run time have an unclear impact on the accuracy. In [30], a method using statistical bounds is proposed with gate delays restricted to Gaussian distributions. Since the delay of CMOS gates is nonlinear with respect to its process parameters, the probability distribution of gate delays is typically not Gaussian. Also, Gaussian distributions are unbounded, meaning that they predict a finite probability of zero delay or very large delay for a gate, which is not physically feasible. In [31], a path based statistical delay computation is presented which has the advantage that it incorporates an accurate delay model that accounts for signal slope and output loading. 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, especially in highly optimized circuits. In [32], a new circuit optimization method was, therefore, proposed that reduces the number of near critical paths in a circuit, thereby improving the statistical delay of the circuit. The optimization, however, was performed using deterministic timing analysis.

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 a procedure for statistical STA in a strict manner from this problem formulation. Since the computational complexity of this exact statistical STA method is exponential with the circuit size, we also present a new method for computing bounds on the exact probability distribution of the circuit delay and proof the correctness of these bounds. 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 on the true statistical behavior of the circuit, we are able to preserve the important characteristic of deterministic STA which 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. Finally, we propose a heuristic method to iteratively improve the computed bounds using selective enumeration of the sample space with additional run time. Combined, the proposed methods provide a statistical STA approach with linear run time, that is guaranteed conservative, has a bounded error, and can be iteratively refined at the expense of additional computational effort. 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% to 10%, and this difference could be reduced by 62% on average, using the proposed selective enumeration method, with modest additional run time.

The remainder of this paper is organized as follows. In Section II, we present a formal model of statistical STA and our modeling assumptions. In Section III, we present a number of probabilistic timing graph transformations. In Section IV, we derive our method for exact statistical timing analysis. In Section V, we present the computation of the lower and upper statistical bounds on the true statistical behavior. In Section VI, we present methods for exact graph reduction and for refinement of the computed bounds using selective enumeration. In Section VII, we present our results and in Section VIII our concluding remarks.

II. 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 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 fabrication of a large set of die can be thought of as our probabilistic experiment, and each fabricated die as an experimental outcome. The randomness or variability of the circuit delay is, therefore, over the fabricated die, and it is the probability 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 are better modeled using case analysis. However, our general analysis approach could 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 $N = \{n_1, n_2, \ldots, n_k\}$ is a set of nodes, $E = \{e_1, e_2, \ldots, e_l\}$ 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 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 edges 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 terminating at a node n is equal to its *arrival time*, $t_a(n)$. The critical path of the

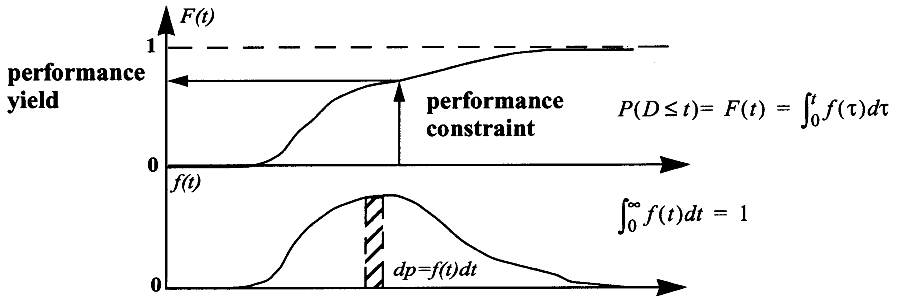

Fig. 1. Delay probability density and cumulative distribution functions.

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 probability 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.

Fig. 1 shows an example of a delay cumulative probability 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:

Property 1: A delay CDF equals 0 for all delay values less than a finite minimum value $d_{\min}$ and equals 1 for all values greater than a finite maximum value $d_{\max}$. A delay pdf is nonzero only on a finite 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 delay value $d_{\min}$ or more than some finite maximum delay value $d_{\max}$.

We assume statistical independence of all edge delays, similar to a number of previous statistical STA methods [20]–[25], [28], [29], [31]. In practice, edge delays may be spatially or topologically correlated, meaning that gates located within close proximity of each other or that have similar layout topologies will have an increased likelihood of having similar gate delays. The spatial and topological correlations between gate delays impacts the distribution of the circuit delay and complicates the analysis by creating additional correlations between path delays. We explicitly state the assumed independence of gate delay in a number of places and also note here that the entire formulation presented in this paper is based on this assumption. The contribution of this paper is, therefore, that it provides an efficient solution to the problem of path delay correlation due to path reconvergence. However, 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 CDF of edge delays to some specific shape, such as a Gaussian distribution. The shape of the edge delay CDFs are unrestricted as long as they satisfy Property 1 and they are valid probability distribution functions.

To simplify the implementation of statistical STA, it is often more convenient to approximate continuous probability density and cumulative 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 approach. 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 nonzero values of their probability distribution functions. The probability that a timing graph G_D in S has edges i with delay D_i between t_i and T_i is

$$P(t_1 \le \tau_1 \le T_1, t_2 \le \tau_2 \le T_2, \ldots) = \int_{t_1}^{T_1}\int_{t_2}^{T_2} \ldots p(\tau_1, \tau_2, \ldots)d\tau_1 d\tau_2 \ldots \quad (1)$$

where $p(\tau_1, \tau_2, \ldots, \tau_i, \ldots)$ is the joint probability density function of edge delays i. Since, as previously stated, we assume that all edge delays are independent random variables, this joint probability function is simply the product of the individual probability density functions of edge delays i and we can rewrite (1) as follows [35]:

$$P(t_1 \le \tau_1 \le T_1, t_2 \le \tau_2 \le T_2, \ldots) = \int_{t_1}^{T_1}\int_{t_2}^{T_2} \ldots p_1(\tau_1)p_2(\tau_2)\ldots d\tau_1 d\tau_2 \ldots \quad (2)$$

where $p_i(\tau_i)$ is the delay 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. The objective of statistical timing analysis is to find the CDF of $D(G_P)$, which is defined as follows.

Definition 3: The cumulative probability distribution function of the delay of a probabilistic timing graph with independent edge delays is expressed as

$$P(D(G_P) \le t) = \int_{D(G_D) \le t} p_1(t_1)p_2(t_2)\ldots dt_1 dt_2 \ldots \quad (3)$$

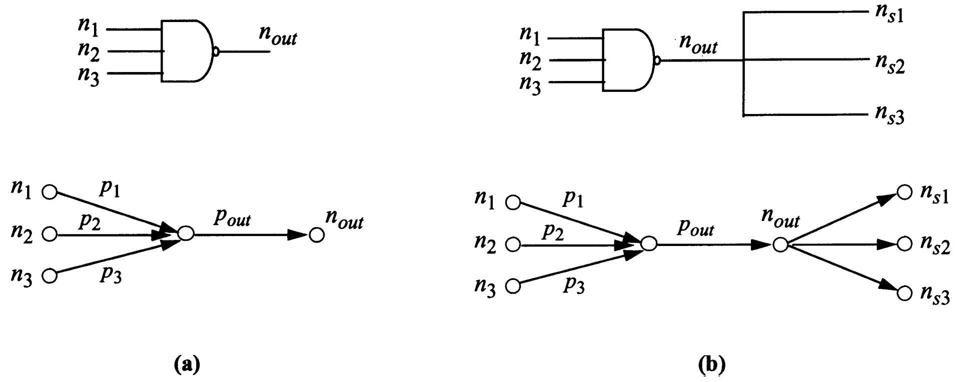

Fig. 2. Graph representation of gates with correlated pin-to-pin delays and interconnect delay variations.

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 the delay $D(G_D)$ of timing graph G_D is less than t.

The probability density function can be computed by simple differentiation of the CDF. 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, as illustrated in Fig. 1. The probability of obtaining a die that meets the specified performance is also referred to as the *performance yield*. A design, for instance, can be improved until its performance yield meets a sufficient level. The graph delay CDF is also useful to determine the number of expected dies in a certain performance range, allowing prediction of the performance "binning" of the design. Conversely, given a required performance yield, the expected performance can be obtained. This allows a designer to determine, for instance, the minimum expected performance of 95% of the fabricated dies.

If we use discrete edge delay pdfs, we can compute the graph delay pdf by enumerating the entire sample space consisting of all combinations of the nonzero delay probabilities of all edges. For each enumerated timing graph G_D in the sample space, we can then compute the graph delay and its probability of occurrence. By summing the probability of all samples with same delay, we then obtain the probability density function of the graph delay, shown in Fig. 1(b). 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.

Finally, we note that each edge in the timing graph is associated with a so-called *pin-to-pin* delay from an input node of a gate to the output node of that gate. An n-input gate is, therefore, represented with n edges in the timing graph. Since all edge delays are independent, this assumes that all pin-to-pin delays of a gate are independent. In practice, the pin-to-pin delays of a gate will be strongly correlated since they are determined by common transistors in the gate structure. To address this correlation, the graph representation of a gate can be extended to model this dependence using the representation as shown in Fig. 2(a). Each gate is represented with a set of fanin edges p_i in series

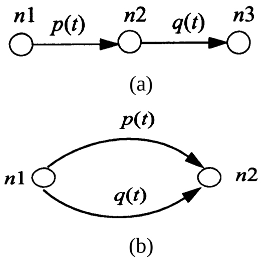

Fig. 3. Series and parallel reduction.

with a single fanout edge p_{out}. The total pin-to-pin delay of the gate is then distributed among edges p_i and p_{out}, where the delays of p_i represent the uncorrelated component of the pin-to-pin delays and p_{out} represents the correlated component of the pin-to-pin delay. Note that this model can be extended to include interconnect delay, as shown in Fig. 2(b), with additional edges representing the delay from the gate output node to the sink nodes of the interconnect.

III. 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 with independent edge delays.

A. Series Reduction

Fig. 3(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 (3), the CDF of the graph delay $D_{Gp}(t)$ is

$$P(D_{Gp} \le t) = \int_{t_1+t_2\le t} p(t_1)\cdot q(t_2)dt_1dt_2. \qquad (4)$$

The pdf of the sum of the two independent edge delays is the convolution of its edge delay pdfs, as is well known from standard probability theory [35], and the two edges can be replaced with a single edge having the following probability density function:

$$P_{Gp}(t) = \int_0^\infty p(t-\tau)\cdot q(\tau)\cdot d\tau. \qquad (5)$$

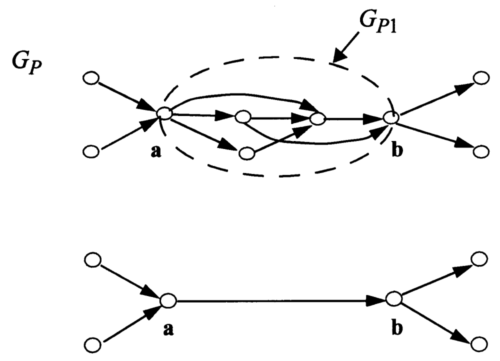

Fig. 4. Subgraph substitution with a single edge.

The probability distribution function of the graph delay is obtained through integration of (5).

B. Parallel Reduction

Fig. 3(b) shows a timing graph G_P consisting of two parallel edges with delays described by pdfs $p(t)$ and $q(t)$ and CDFs $P(t)$ and $Q(t)$. Since the delays of both edges are statistically independent, the probability $P(D_{Gp} \leq t)$ is the product of the probabilities that each edge delay is less than or equal to t:

$$P_{Gp}(t) = P(t) \cdot Q(t). \tag{6}$$

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

$$p_{Gp}(t) = P(t) \cdot q(t) + p(t) \cdot Q(t). \tag{7}$$

Therefore, the graph in Fig. 3(b) can be replaced with a single edge having the pdf $p_{Gp}(t)$.

C. Subgraph Substitution

Let graph G_P have subgraph G_{P1}, as shown in Fig. 4, such that:

1) G_{P1} and complementary graph $\overline{G}_{P1} = G_P - G_{P1}$ have only 2 common nodes a and b.
2) Node a (b) has only incoming (outgoing) edges belonging to subgraph $\overline{G}_{P1}$ and outgoing (incoming) edges belonging to subgraph G_{P1}.

Let the subgraph G_{P1} have a graph delay pdf $d(G_{P1})$. We can substitute subgraph G_{P1} in graph G_P with a single edge (a, b) having the same delay pdf $d(G_{P1})$. This results in a simpler timing graph with the same graph delay. This can be proven by rearranging the integration in (3) for the initial graph delay CDF and separating the integration over the random variables corresponding to the delays of subgraph G_{P1}.

IV. Statistical Timing Analysis

The initial formulation presented in the Section II 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 pdf, in a manner that is consistent with the definition of D_{Gp} in Section II. We first define the probability distribution of the latest arrival time, $A_n(t)$ at node n as follows.

Definition 4: The latest arrival time A_n at node n of G_P is a random variable where its CDF $A_n(t)$ is the probability that a deterministic timing graph G_D in the sample space $S(G_P)$ has an arrival time $t_a(n) \leq t$.

In the subsequent discussion, we will refer to the latest arrival time as simply *the* arrival time. We also note that a similar derivation can be performed for the *earliest* arrival time. Also, the arrival time for the source node n_s is a deterministic value equal to 0. We now 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 $G_{S,n}$.

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. We can then use such a method to propagate arrival times through the circuit in topological fashion. To compute the arrival time at node n, we must consider if the arrival times of its fanin nodes n_p are independent random variables. We, therefore, state the following theorem:

Theorem 1: For a timing graph G_P with independent edge delays, 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 validity of Theorem 1 is intuitively obvious from the fact that the sample space of $G_{S,i}$ and $G_{S,j}$ are disjoint and hence the arrival times $A_{n,i}$ and $A_{n,j}$ are independent. For completeness, a proof is given below.

Proof: The arrival time $A_{n,i}$ is equal to the delay of its fanin subgraph $D(G_{S,i})$ and arrival time $A_{n,j}$ is equal to the delay of its fanin subgraph $D(G_{S,j})$. The joint probability that $D(G_{S,i})$ is less than d_i and $D(G_{S,j})$ is less than d_j, assuming independence of the edge delays, can be written as (8), found at the bottom of the page, where $t_{i,k}$ is the delay of gate k in subgraph $G_{S,i}$ and $t_{j,k}$ is the delay of gate k in subgraph $G_{S,j}$. Using the fact that all $t_{i,k}$ belong to the domain of $D(G_{S,i})$ and

$$P(D(G_{S,i}) \leq d_i, D(G_{S,j}) \leq d_j) = \int\limits_{D(G_{S,i})\leq d_i, D(G_{S,j})\leq d_j} p_{i,1}(t_{i,1})p_{i,2}(t_{i,2})\ldots p_{j,1}(t_{j,1})p_{j,2}(t_{j,2})\ldots dt_{i,1}dt_{i,2}\ldots dt_{j,1}dt_{j,2} \tag{8}$$

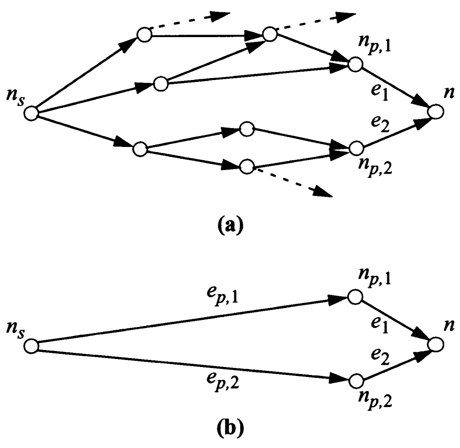

Fig. 5. Arrival time computation for node n. Dotted edges in (a) belong to graph G_P only and solid edges belong to $G_{S,n}$.

all $t_{j,k}$ belong to the domain of $D(G_{S,j})$ we can rewrite (8) as follows:

$$\begin{aligned} &P(D(G_{S,i}) \le d_i, D(G_{S,j}) \le d_j) \\ &= \int_{D(G_{S,i}) \le d_i} p_{i,1}(t_{i,1}) p_{i,2}(t_{i,2}) \ldots dt_{i,1} dt_{i,2} \\ &\cdot \int_{D(G_{S,j}) \le d_j} p_{j,1}(t_{j,1}) p_{j,2}(t_{j,2}) \ldots dt_{j,1} dt_{j,2}. \end{aligned} \tag{9}$$

From which follows:

$$\begin{aligned} &P(D(G_{S,i}) \le d_i, D(G_{S,j}) \le d_j) \\ &\qquad = P(D(G_{S,i}) \le d_i) \cdot P(D(G_{S,j}) \le d_j). \end{aligned} \tag{10}$$

From standard probability theory, it follows that the fanin subgraphs delays $D(G_{S,i})$ and $D(G_{S,j})$ are independent and, correspondingly, the arrival times $A_{n,i}$ and $A_{n,j}$ are independent. □

We first consider the case where the arrival times of fanin nodes n_p of node n are independent.

A. Independent Arrival Time Propagation

Without loss of generality, we consider a node n with two fanin nodes $n_{p,1}$ and $n_{p,2}$. We then consider fanin subgraph $G_{S,n}$ at node n, as shown in Fig. 5(a). Since the arrival times A_1 at node $n_{p,1}$ and A_2 at node $n_{p,2}$ are assumed to be independent, their fanin subgraphs $G_{S,1}$ and $G_{S,2}$ are disjoint, and using the subgraph substitution from Section III, each can be replaced with a single edge $e_{p,1}$ and $e_{p,2}$, as shown in Fig. 5(b). The edge delay CDF of $e_{p,1}$ and $e_{p,2}$ are set equal to the graph delay CDFs of $G_{S,1}$ and $G_{S,2}$, which is equal to the arrival time CDFs A_1 and A_2. The resulting graph, as shown in Fig. 5(b), can be reduced to a single edge by performing series and parallel reduction. The obtained edge delay CDF of the final edge between n_s and n is equal to the graph delay CDF of subgraph $G_{S,n}$, which, in turn, is equal to the arrival time CDF A_n at node n.

Given independent arrival time CDFs $A_{p,i}$ at fanin nodes $n_{p,i}$, we, therefore, compute the arrival time CDF A_n at node n as follows.

1) Convolve the arrival time CDF $A_{p,i}$ at each node $n_{p,i}$ with the edge delay CDF of the edge connecting $n_{p,i}$ with n.
2) Compute the arrival time CDF A_n at node n by applying (6) on the convolved CDFs.

As also noted in [24], 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 using (6). The run time complexity of this method is linear with the size of the circuit. Unfortunately, this procedure is only valid if the arrival times are independent. For this to be true for all nodes, the graph will have the form of a tree-like structure, with the root of the tree as the sink node of the graph and all leaf nodes of the tree connected to the source node n_s with an edge with deterministic delay. In practice, such timing graphs are rare and we, therefore, now discuss how to compute arrival times in the presence of dependent arrival times.

B. Dependent Arrival Time Propagation

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, even in case that edge delays are independent. Since application of (6) assumes statistical independence, it cannot be used to compute the maximum of arrival times $A_{p,i}$. An example of a timing graph with dependent arrival times is shown in Fig. 6(a). To determine for which portions the subgraphs $G_{S,i}$ share edges, we use the following definition of a *dependence* set.

Definition 6: Consider a pair of fanin nodes $n_{p,1}$ and $n_{p,2}$ of node n, with fanin subgraphs $G_{S,1}$ and $G_{S,2}$. The intersection graph G_I consists of edges and nodes shared by $G_{S,1}$ and $G_{S,2}$, excluding the source node n_s. The set of dependence nodes for the fanin node pair $n_{p,1}$ and $n_{p,2}$ is the set of nodes $\{n_1, n_2, \ldots, n_d, \ldots\}$, such that n_d lies on the intersection graph G_I, and such that n_d has one or more fanout edges that lie on either $G_{S,1}$ or $G_{S,2}$, but not on both. The set of dependence nodes for node n is the union of the dependence node sets over all possible pairs of its fanin nodes.

In Fig. 6(a), the intersection graph G_I for the fanin node pair f and h of 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 a fanout edge that lies on only one of the two fanin subgraphs of f and h. Note that a is not in the dependence set of n_f since all its fanout edges belong to the fanin subgraphs of both f and h. However, the dependence set of node d is $\{a\}$. Also 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 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 [33]. We refer to a node in G_P as a *convergence* node n_c if it has a nonempty dependence set. We also define the global set of dependence nodes S_D as the union of the dependence sets over all nodes and refer to a node as a dependence node if it is an element in this list.

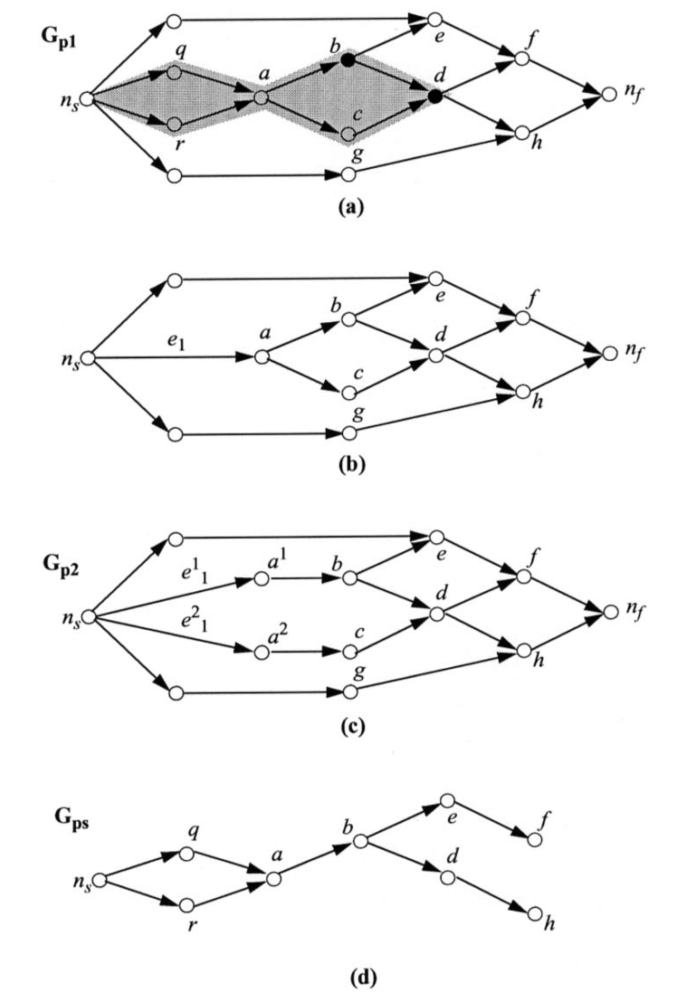

Fig. 6. Dependent arrival time computation for node n. In (a), the intersection graph of n_f is shaded and dependence nodes of n_f are black.

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 S_D. In Fig. 6(a), $S_D = \{a, b, d\}$, 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. 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 of A_1 at $n_{D,1}$, as shown in Fig. 6(b). Similarly, it is clear that the arrival time CDF of A_1 at $n_{D,1}$ can be computed using independent arrival time propagation, as explained in the previous Section.

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 [35], the arrival time pdf $p_x(t)$ at a convergence node x of $n_{D,1}$, 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 $p_i = \mathrm{P}(D_1 = d_i)$. 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 cir-
cuit until the first dependence node $n_d$ is
encountered.
2. Enumerate the pairs ($t_i, p_i$) of the ar-
rival 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
all graph nodes $x$ by summing their condi-
tional arrival time pdfs weighted by the
product of their conditional probabili-
ties.
```

Note that, although we refer to emuneration of *dependence* nodes, the enumeration is more precisely of *graph* instances according to the discretization of the arrival time at the dependence node. Also, above procedure computes the arrival time for all nodes in the timing graph, including the sink node, and requires enumeration of all dependence nodes. However, if the arrival time is needed for only a subset of nodes in the graph, it may not be necessary to enumerate all dependence nodes in the graph since only the dependence nodes associated with this subset of nodes needs to be enumerated. Of course, these dependence nodes may again have dependence nodes themselves that will need to be enumerated as well, leading to a recursive formulation.

We will now show that the set of nodes at which arrival times are enumerated is the sufficient and necessary set for exact computation of the graph delay CDF. First, from the construction of the exact statistical timing analysis algorithm in the above description, it is clear that enumeration of all dependence nodes is a sufficient condition. The enumeration of a dependence node eliminates that node as a dependence node from the timing graph. After enumeration of all dependence nodes, all arrival times converging at a node are independent in each enumerated instance of the timing graph. Therefore, the exact graph delay can be computed using independent arrival time propagation in each enumerated instance of the timing graph, showing the sufficiency of enumerating all dependence nodes.

Next, we show that enumeration of all dependence nodes is also a necessary condition, meaning that it is not possible to enumerate fewer nodes without arrival time dependencies remaining in the circuit. Without loss of generality, we consider convergence node n_f in Fig. 6(a) with fanin nodes f and h and with dependence nodes $\{b, d\}$. We now show how, based on the properties of dependence nodes given in Definition 6, enumeration of node b is necessary. Since we only use properties specified in Definition 6, the same arguments also translate to node d and all other dependence nodes in a circuit. We first observe that there exists a path from node b to both node f and h since, based on Definition 6, node b lies on the intersection of the fanin subgraphs of nodes f and h. Also, based on Definition 6, at least one fanout edge of b does not lie on both fanin subgraphs, but only on one of the fanin subgraphs. In Fig. 6(a) this is edge (b, e) which lies on the fanin subgraph of f but not the fanin subgraph of h. From this, it follows that there exists at least one pair of paths (p_1, p_2) from node b to nodes f and h, such that this pair of paths is disjoint, meaning they do not share any common edges. In Fig. 6(a), this pair of paths is $p_1 = (b, e, f)$ and $p_2 = (b, d, h)$.

We now consider subgraph G_{ps} of timing graph G_{p1}, consisting of all edges and nodes on paths p_1 and p_2 and on fanin subgraph of b, as shown in Fig. 6(d). We assume that nodes f and h have arrival times A_f and A_h, respectively. We now show that enumeration of node b is a necessary condition to ensure that arrival times A_f and A_h are independent. First, we note that enumeration of any nodes in the fanin subgraph of b (consisting of nodes r, q, and a) does not eliminate the dependence between A_f and A_h, since the fanin edge (a, b) of node b will still contribute a common, random delay to both A_f and A_h. Second, enumeration of any nodes that lie only on either path p_1 or p_2 (such as nodes d and e) also does not eliminate the dependence between A_f and A_h since these nodes affect only one of the two paths. Therefore, since enumeration of any node other that node b in subgraph G_{ps} does not eliminate the dependence of A_f and A_h, it follows that enumeration of node b is a necessary condition for eliminating this dependence.

In the original timing graph G_{p1}, the arrival times propagated along p_1 and p_2 combine with arrival times from other paths. However, it is clear that the dependence between arrival times propagated along p_1 and p_2 is sufficient to create dependence between the arrival times at nodes f and h in G_{p1}, even after the arrival times along p_1 and p_2 combine with other arrival times. From this it follows that enumeration of dependence node b is a necessary condition for eliminating the dependence between the arrival times at node f and h in G_{p1}. Since node b was chosen as a generic dependence node and we only relied on its properties as defined by Definition 6, the same arguments apply to all dependence nodes and it follows that enumeration of all dependence nodes is a necessary condition for exact statistical timing analysis using independent arrival time propagation.

The number of dependence nodes is typically 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. Nevertheless, the run time remains exponentially with the number of dependence nodes due to the recursive formulation of the method. Dependent arrival time propagation is, therefore, useful only for very small timing graphs or timing graphs with mostly tree-like structures. In the next section, we, therefore, show how to efficiently compute bounds on the arrival time CDFs and in Section VI, we discuss how these bounds are improved by enumeration of a small set of dependence nodes.

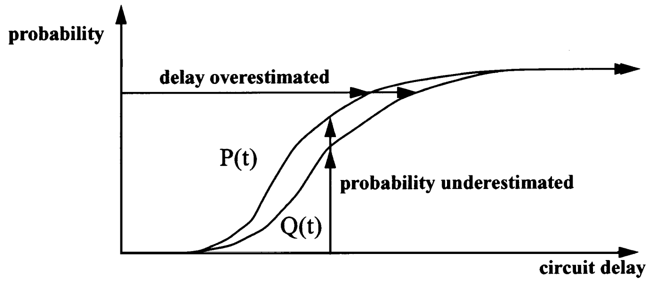

Fig. 7. CDF $Q(t)$ is a conservative bound on CDF $P(t)$. Probability corresponding to a particular delay is underestimated and delay corresponding to a particular probability is overestimated.

V. Statistical Bounds

We now propose an efficient method for computing lower and upper bounds of the exact arrival time CDF of G_P for timing graphs with independent edge delays. We are interested in both upper and lower bounds since this allows us to determine the quality of the bounds by comparing their difference. In general, several types of statistical bounds on a CDF have been defined. In this paper, we use the so-called Stochastic bound [35], which is defined as follows.

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

A similar definition can be formulated for lower bounds. The meaning of the defined bound is shown in Fig. 7 with 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 for all time points, the probability defined by $Q(t)$ is less than that of $P(t)$. Therefore, not only the expected value μ_p of $P(t)$, but also other characteristics, such as the 95% confidence point (performance yield), are bounded by $Q(t)$ on $P(t)$. By using CDF $Q(t)$ instead of $P(t)$, we will, therefore, overestimate the delay corresponding to a particular probability or performance yield, resulting in a conservative analysis for late arrival times, as shown in Fig. 7. Similarly, for a particular required delay, the probability that a die will meet this delay constraint will be underestimated by using $Q(t)$ instead of $P(t)$. Note, therefore, that $Q(t)$ is an upper bound on $P(t)$ from the perspective of arrival times but is a lower bound on $P(t)$ from the perspective of performance yield. In this paper, we consider bounds from the arrival time perspective and we, therefore, refer to $Q(t)$ as an *upper* bound on $P(t)$.

We can compute upper and lower bounds for both the *latest* arrival time, corresponding to slow paths in the circuit, as well as for the *earliest* arrival times, corresponding to fast paths in the circuit. Fast paths are important for detecting hold violations and race conditions in a circuit. To obtain a conservative analysis, an *upper* bound must be used for the *latest* arrival times and a *lower* bound must be used for the *earliest* arrival times. For clarity, we focus in this paper on late arrival times, although the analysis can be applied to early arrival times as well.

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

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

$$P(t) = \int_{x+\max(y,z)\le t} p(x)q(y)r(z)dxdydz \tag{11}$$

$$Q(t) = \int_{\max(x_1+y,x_2+z)\le t} p(x_1)p(x_2)q(y)r(z)dx_1dx_2dydz \tag{12}$$

where $p(x)$, $q(y)$, $r(z)$ are the probability density functions of x, y and z, respectively. Multiplying (11) by the integral of probability density function $\int_{-\infty}^{\infty} p(v)dv = 1$, rearranging some of the terms and renaming integration variables, gives us:

$$P(t) = \int_{\max(x+y,x+z)\le t, -\infty\le v\le\infty} p(x)p(v)q(y)r(z)dxdvdydz. \tag{13}$$

Integrals from formulae (12) and (13) for probability distributions $Q(t)$ and $P(t)$ 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. Note that random variable v is independent from random variables x, y and z. We now split the *4D* domain of both functions into two subdomains: $y \le z$ and $y > z$. Probability distributions $Q(t)$ and $P(t)$ can be represented as the sum of two terms corresponding to the contribution of each subdomain:

$$\begin{aligned} Q(t) =& Q_{y\le z}(t) + Q_{y>z}(t) \\ =& \int_{\max(x_1+y,x_2+z)\le t, y\le z} f(x_1, x_2, y, z)dx_1dx_2dydz \\ &+ \int_{\max(x_1+y,x_2+z)\le t, y>z} f(x_1, x_2, y, z)dx_1dx_2dydz \end{aligned} \tag{14}$$

$$\begin{aligned} P(t) =& P_{y\le z}(t) + P_{y>z}(t) \\ =& \int_{\max(x+y,x+z)\le t, -\infty\le v\le\infty, y\le z} f(x, v, y, z)dxdvdydz \\ &+ \int_{\max(x+y,x+z)\le t, -\infty\le v\le\infty, y>z} f(x, v, y, z)dxdvdydz. \end{aligned} \tag{15}$$

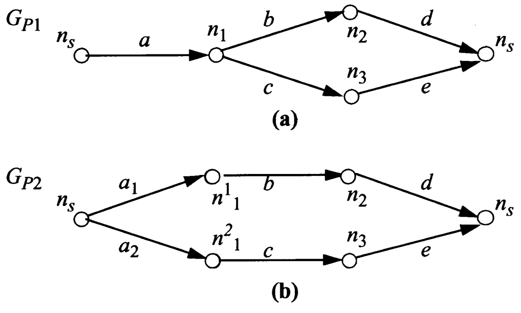

Fig. 8. Bounded graph transformation through node splitting.

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 $y \le z$ and therefore, inequality $\max(x+y, x+z) \le t$ follows from inequality $\max(x_1+y, x_2+z) = \max(v+y, x+z) \le t$. Therefore, the region of integration for computing $P_{y\le z}(t)$ includes the integration region for computing $Q_{y\le z}(t)$ and hence $Q_{y\le z}(t) \le P_{y\le z}(t)$ because in both cases we integrate the same function $f(x, v, y, z)$.

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 $\max(x_1 + y, x_2 + z) \le t = \max(x+y, v+z) \le t$ in this subdomain and the region of integration for computing $P_{y>z}(t)$ includes the region of integration for computing $Q_{y>z}(t)$. Therefore, $Q_{y>z}(t) \le P_{y>z}(t)$.

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

We can graphically illustrate Theorem 2 as follows. Consider the simple graph G_{p1} shown in Fig. 8(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. 8(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 Fig. 8, results in a graph G_{P2} that is an upper bound on G_{P1}, meaning that the arrival time CDFs of all nodes in G_{P2} are an upper bound on the arrival time CDFs of the corresponding nodes in G_{P1}.

Based on Theorem 2, we now pose the following useful corollary.

Corrolary 1: Given a graph G_P with independent edge delays and one or more convergence nodes. If 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.

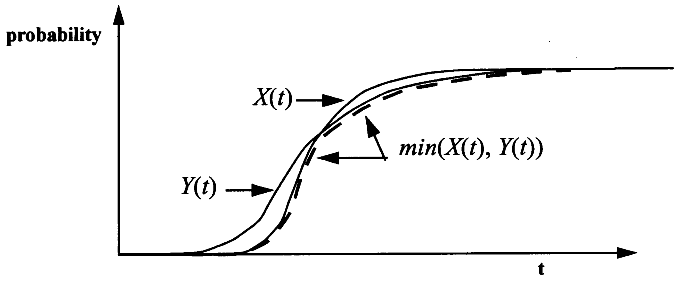

Fig. 9. Lower bound computation for two dependent arrival times.

The validity of Corrolary 1 can be seen by considering the timing graph G_{P1} with dependence node a, as illustrated in Fig. 6(a). Following the procedure for dependent arrival time propagation, we replace subgraph $G_{S,1}$ with a single edge e_1, as shown in Fig. 6(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 Fig. 6(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 time CDF 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 subsequently split it. 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.

B. Lower Bound Computation

We now discuss the computation of a lower 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 Fig. 9, is a lower bound on the CDF of z. This can be seen by considering the graph in Fig. 3(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 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 propagated arrival time is computed by taking the minimum of the incoming arrival time CDFs for each time point. The lower bound computation, therefore, has a linear run time complexity with circuit size. For nodes with empty dependence sets, such as nodes e and h in Fig. 6(a), the regular statistical maximum using (6) is taken, since their arrival times are independent.

It is important to note that the proposed bounds are not restricted to Gaussian gate delay and arrival time pdfs, but are valid for pdfs of any shape. Also, given gates with *bounded* gate delay pdfs, the proposed arrival time bounds have the property that their maximum and minimum values with nonzero probability match that of the exact graph delay pdf. The proposed bounds, therefore, have the useful property that the interval over which arrival times can occur matches that of the exact graph delay.

VI. Graph Reduction and Selective Enumeration

Our overall approach to statistical timing analysis consists of the following steps.

1) We perform exact graph reduction to decrease the problem size without altering the delay of the graph.
2) We compute upper and lower bounds using the methods explained in Section V.
3) We improve the computed bounds using enumeration of a select set of dependence nodes.

We now explain the exact graph reduction and the selective enumeration below in more detail.

A. Exact Graph Reduction

In order to reduce the size of the timing graph, we prune arrival times by considering their relative alignment at convergence nodes. We consider two arrival time CDFs, A_1 and A_2 that converge at node n, as shown in Fig. 10(a). If the minimum arrival time with a nonzero probability $t_{1,\min}$ of A_1, is greater than the maximum arrival time with a nonzero probability, $t_{2,\max}$ of A_2, as shown in Fig. 10(b), it is clear that in the entire sample space S, A_1 will be greater than A_2. We can, therefore, prune arrival time A_2 from the timing graph without changing its timing behavior. Note that, using this approach, entire subgraphs can at times be removed from the timing graph.

Also, when two arrival times A_1 and A_2 partially overlap, as shown in Fig. 10(c), the CDF of A_2 can be truncated at $t_{1,\min}$. Based on (6), values of A_2 are required only for $t \geq t_{1,\min}$ since for $t < t_{1,\min}$, $A_1(t) = 0$. Hence the CDF of A_2 needs to be computed only for the range $t_{1,\min}$ to $t_{2,\max}$. Based on this, we truncate the arrival time CDF at fanin nodes of n_2 by propagating the truncation time $t_{1,\min}$ across gates in reverse topological order, at each gate subtracting the maximum gate delay $d_{\max}$ with nonzero probability. The truncation time $t_{\text{trunc},i}$ at node n_i is, therefore, as follows: $t_{\text{trunc},i} = t_{1,\min} - \delta$, where δ is the sum of $d_{\max}$ over all gates between node n_i and n_2. An arrival time $t_{a,i} < t_{\text{trunc},i}$ at node n_i always results in an arrival $t_{a,2}$ at node n_2, such that $t_{a,2} < t_{1,\min}$ and can be truncated. Furthermore, we also truncate the CDFs of the gate delay for gates in the fanin cone of n_2. Given a fanin gate g with maximum delay $d_{g,\max}$, as shown in Fig. 10(d), the gate delay CDF can be truncated at $d_{g,\text{trunc}} = d_{g,\max} - t_p$, where $t_p = t_{2,\max} - t_{1,\min}$. Arrival time $t_{2,\max}$ at n_2 is the arrival time when all fanin gates of n_2 have their maximum delay $d_{\max}$. Hence, it follows that gate delays less than $d_{g,\text{trunc}} = d_{g,\max} - t_p$ will always result in arrival times at n_2 that are less than $t_{1,\min}$. The

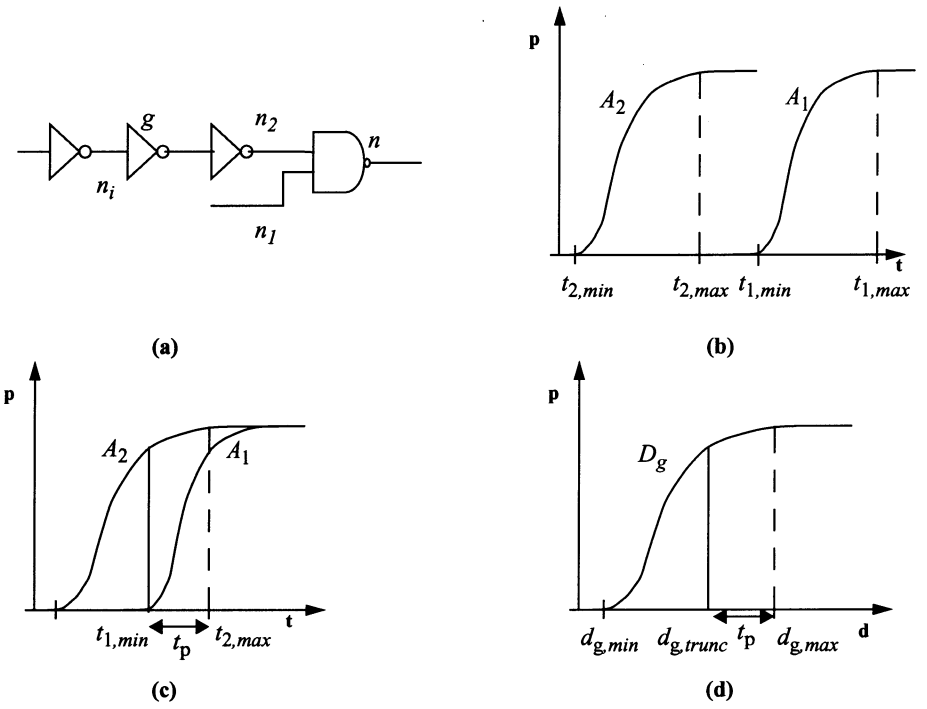

Fig. 10. Pruning of arrival times by considering their relative alignment.

delay CDF of gate g is, therefore, pruned at $d_{g,\mathrm{trunc}}$ thereby reducing the computational complexity of arrival time computation.

In addition to pruning, we also utilize the series and parallel graph transforms discussed in Section III to reduce the size of the timing graph. The pruning method and the series/parallel reduction are executed in a loop, until no further improvement is made. Note that while the series transform and pruning method reduce the graph size and the computational complexity, the parallel transform also resolves arrival time dependencies by removing local reconvergences and improves the quality of the computed bounds. It is, therefore, advantageous to reduce the graph size not only to improve the run time of the bound computation, but also to improve the quality of the bounds.

B. Selective Enumeration

In order to improve the quality of the bounds described in Section V, we combine the bound computation with enumeration of the sample space of the arrival time at 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 (6), instead of the minimum operation used for computing the lower bound of dependent arrival times. In this case, the upper and lower bounds at the dependence node become equal to each other. Selective enumeration, therefore, has the desirable property that as the number of enumerated dependence nodes increases, the quality of the bounds increases monotonically. Also, when all dependence nodes are enumerated, the lower and upper bounds will become equal to the exact arrival time CDF.

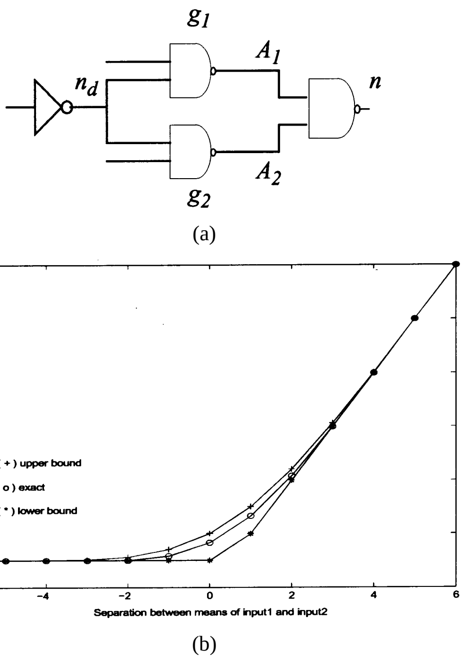

Fig. 11. Expected values of bounds and exact arrival time at node n as a function of the shift between expected values of A_1 and A_2.

Our objective is to obtain a 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 first consider the circuit shown Fig. 11(a) with two correlated arrival times A_1 and A_2 that converge at node n. We computed the upper and lower bounds as

```
1. Compute enumeration merit of each dependence node
2. Order list of dependence nodes by decreasing merit
2. While (bound difference is not met and run time is not exceeded) {
3.        Add an unselected 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) remove last node from enumeration list.
6.}
```

Fig. 12. Selective enumeration algorithm.

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. Fig. 11(b) shows the expected value of the upper and lower bounds and the exact arrival time at node n, plotted against the difference of the expected value of A_1 and A_2. As the shift between A_1 and A_2 increases, the two bounds rapidly converge to the true arrival time. This is caused by the fact that if one arrival time CDF lies significantly after the other, the later arrival time CDF dominates the result and the dependence between the two arrival times has little impact. In the extreme case, when the minimum time point of the later arrival time CDF falls after the maximum time point of the earlier arrival time CDF, as shown in Fig. 10(b), the later arrival time propagates unaltered and the computed bound matches exactly with the true arrival time distribution.

It is also possible that, two dependent arrival times give rise to a large bound difference at a node n_c, but this bound difference does not propagate to the sink node. This occurs when along the propagation path from n_c to the sink node n_f, the arrival time bounds combine with other arrival time bounds that are aligned significantly later and that dominate. In this case, the sink node n_f is shielded from node n_c and enumeration of its dependence nodes has little impact on the bound difference at n_f. 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. We, therefore, apply a heuristic which is iterative in nature and where the number of enumerated nodes is increased by one in each iteration. 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 Section VI-C. In each iteration of the algorithm, dependence nodes are added to the set of enumerated nodes in decreasing order of their enumeration merit. If in a particular iteration the bound does not improve significantly, the last added dependence node is removed from the set of enumerated nodes before a new dependence node is added. The algorithm is shown in pseudocode in Fig. 12. The algorithm continues until either a required bound difference has been obtained, or until the allowed run time is exceeded.

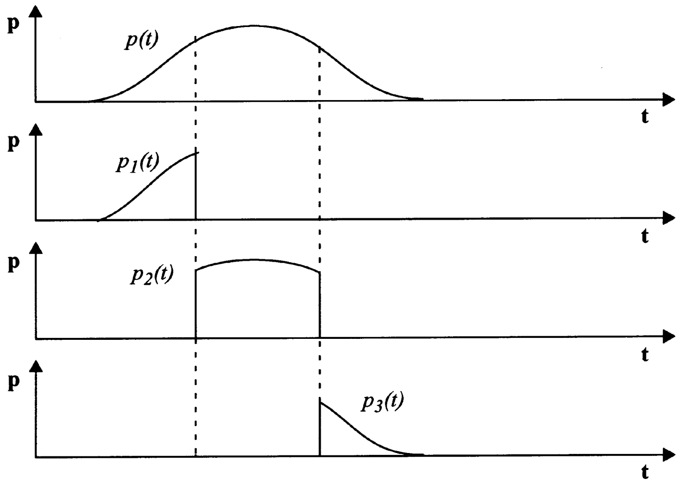

Fig. 13. Arrival time pdf and partial pdfs for three intervals.

In each iteration of the algorithm, we enumerate the selected dependence nodes and compute bounds on the CDF of the graph delay. For each enumerated dependence node n_d, we consider each possible arrival time value t_i and compute bounds on the graph delay weighted by the conditional probability p_i that the arrival time 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 Fig. 13. We then compute the upper bound $p_{j,\text{upper}}(t)$ and lower bound $p_{j,\text{lower}}(t)$ on the graph delay using the method discussed in Section V, 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 $p_j(t)$. We, therefore, again compute the final upper bound $p_{\text{upper}}(t)$ on the pdf of the graph delay using conditional probabilities, as follows:

$$p_{\text{upper}}(t) = \sum_j P_j \cdot p_{j,\text{upper}}(t). \tag{16}$$

Partial enumeration of arrival times requires fewer bound computations than full enumeration, and, therefore, reduces

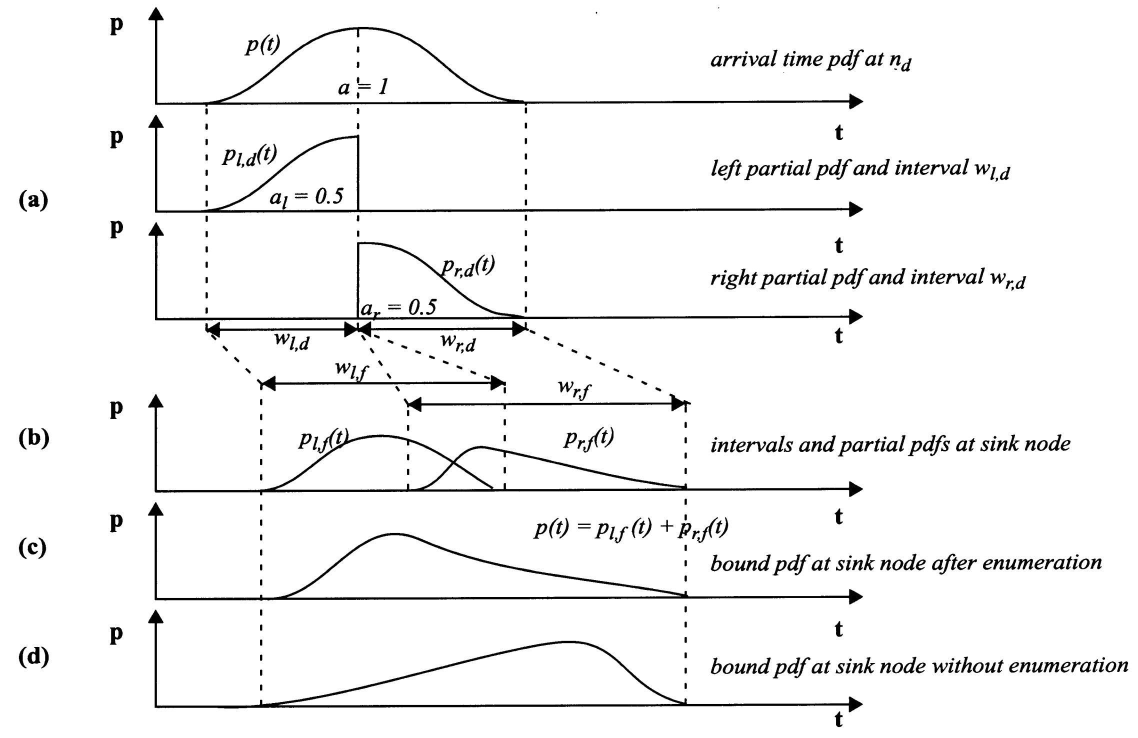
(a)
(b)
(c)
(d)
p
t
p(t)
a = 1
arrival time pdf at n_d
$p_{l,d}(t)$
$a_l = 0.5$
left partial pdf and interval $w_{l,d}$
$p_{r,d}(t)$
$a_r = 0.5$
right partial pdf and interval $w_{r,d}$
$w_{l,d}$
$w_{r,d}$
$w_{l,f}$
$w_{r,f}$
$p_{l,f}(t)$
$p_{r,f}(t)$
intervals and partial pdfs at sink node
$p(t) = p_{l,f}(t) + p_{r,f}(t)$
bound pdf at sink node after enumeration
bound pdf at sink node without enumeration

Fig. 14. Computation of partial pdfs and intervals.

the computational effort. The number of intervals used for enumeration, 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 the bound reduction was most efficient using partial enumeration with two intervals. This allows a much larger number of dependence nodes to be enumerated with reasonable run time and results in a greater reduction of the bound difference than when fewer dependence nodes are enumerated more finely.

C. Merit Computation

We employ a heuristic method to compute the enumeration merit for a dependence node, representing the expected improvement of the bound difference from enumeration of that node. For simplicity, we limit our discussion to the enumeration merit for the upper bound, although similar considerations apply to the lower bound. At the sink node, the pdf of the computed upper bound is skewed to the right (corresponding to higher arrival time values) relative to the pdf of the exact arrival time. Hence, improvement of the bound through selective enumeration results in 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 enumerating a dependence node.

For each dependence node n_d, arrival times with nonzero 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 Fig. 14(a). The integral of the arrival time pdf over each interval is equal to 0.5 and hence each interval represents 50% of all possible arrival times at dependence node n_d, over the fabricated die.

We can again consider partial enumeration, where partial pdfs $p_{l,d}(t)$ and $p_{r,d}(t)$, corresponding to the left and right intervals, are propagated to the sink node and the final arrival time pdf at the sink node is computed as their weighted sum. However, instead of propagating partial pdfs, we propagate only 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 Fig. 14(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 $[t_{l,1}^{\min}, t_{l,1}^{\max}], [t_{l,2}^{\min} t_{l,2}^{\max}], \ldots, [t_{l,n}^{\min}, t_{l,n}^{\max}]$, that converge at a node, the combined left interval at that node is $[\max(t_{l,1}^{\min}, t_{l,2}^{\min}, \ldots, t_{l,n}^{\min}), \max(t_{l,1}^{\max}, t_{l,2}^{\max}, \ldots, t_{l,n}^{\max})]$. The right interval is computed likewise.

For nodes n_i that do not fall in the fanout cone of dependence node n_d, initial left and right intervals are taken to be equal to the total interval at that node: $w_{l,i} = w_{r,i} = [t_i^{\min}, t_i^{\max}]$. Note that the computational effort of propagating the timing intervals from a dependence node to the sink node is equivalent to that of standard static timing analysis.

The computed interval $w_{l,f}(w_{r,f})$ at the sink node 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

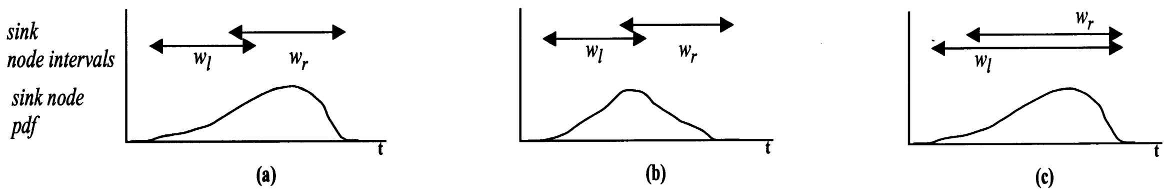

Fig. 15. Left and right arrival time intervals and bound pdfs.

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 Fig. 14(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 Fig. 14(c). However, due to the inherent error in bound computation, the arrival time pdf at the sink node without enumeration can be significantly skewed to the right, as shown in Fig. 14(d). In this case, the area under the left interval can be less than 0.5. Since after enumeration of n_d the area under the left interval will be equal to or greater 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, therefore, computed as follows:

$$merit_d = \begin{cases} 0.5 - a_{l,f}, & \text{if } (a_{l,f} < 0.5) \\ 0, & \text{otherwise} \end{cases} \tag{17}$$

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 toward the right interval, as shown in Fig. 15(a). This occurs when the arrival times of the dependence node align at a convergence node and when 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 Fig. 15(b), and the computed enumeration 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 Fig. 15(c), thereby also resulting in a low or zero 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. Furthermore, since the computational complexity of computing intervals for all dependence nodes grows quadratically with the size of the circuit, the list of propagated left and right intervals

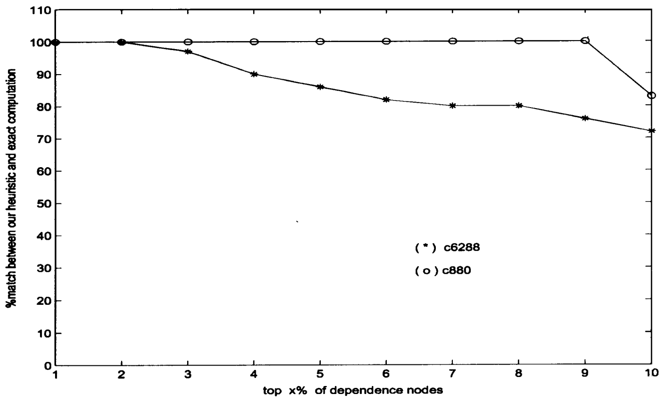

Fig. 16. Comparison of number of matching nodes between exact and heuristic rankings of dependence nodes.

is pruned at each node in the graph and only the smallest k intervals are propagated. This ensures linear computational complexity of the merit computation with circuit size. In our experiments k was set to 20 since the number of enumerated nodes was less than this in all cases. Note that different enumeration nodes are selected for the lower and upper bounds and that each bound is computed separately. Also, the accuracy of the enumeration merit can be improved by considering several intervals, although two intervals was found to provide good accuracy in our test circuits.

We verified the effectiveness of the proposed enumeration merit computation by comparing the ranking of dependence nodes based on their enumeration merit with their ranking based on the actual improvement of the bound when they were enumerated. We compared the top 10% of nodes using the exact ranking with the top $x\%$ nodes based on the heuristic ranking, where x was varied from 1 to 10%. The percentage of nodes in the heuristic ranking that matched with the top 10% in the exact ranking is shown in Fig. 16 for each value of x. The two selected circuits represent those with maximum and minimum improvement in bound difference using selective enumeration. The results demonstrate that in both cases, the heuristic ranking of nodes matches well with the exact ranking.

VII. Results

The proposed statistical timing analysis method, including the exact graph reduction, computation of upper and lower arrival time bounds and the method for selective enumeration, was implemented. Also, exact statistical timing analysis through edge enumeration, as described in Section II, and through dependent arrival time propagation, as described in Section IV, was implemented. This was used to confirm the correctness of the computed bounds for circuits where it was

TABLE I
CIRCUIT STATISTICS AND EXACT REDUCTION IMPROVEMENT

Circuit				# dependence nodes			exact graph reduction	
name	# node	# edges	# convergence node	total	per convergence node avg	per convergence node max	# edges	% improvement
c17	13	19	3	3	0.66	3	15	21%
c432	198	379	59	53	12.5	52	217	43%
c499	245	481	51	59	6.7	59	369	23%
c880	445	815	90	64	7.2	52	293	64%
c1355	589	1137	323	250	3.7	58	920	19%
c1908	915	1556	149	249	5.7	105	877	44%
c2670	1428	2449	277	268	5.2	56	1107	55%
c3540	1721	3011	522	480	22.6	312	1895	37%
c5315	2487	4687	348	264	4.1	68	1138	76%
c6288	2450	4864	1646	1197	95.7	1171	3653	25%
c7552	3721	6459	1479	1050	5.8	367	4228	35%

TABLE II
RESULTS OF BOUND COMPUTATION AND SELECTIVE ENUMERATION

circuit	monte carlo	expected value of computed bounds			expected value of bounds after selective enumeration				
		lower	upper	% difference	lower	upper	% improvement with enumeration	# enumerated dependence nodes	run time reduction / bound / enumeration (sec)
c17	1.399	1.369	1.428	4.167	1.399	1.399	100.000	3	0.5 / 0.2 / 7.0
c432	7.740	7.448	8.060	7.587	7.605	7.768	73.344	12	0.5 / 0.3 / 42
c499	5.168	4.730	5.282	10.451	4.984	5.215	58.152	13	0.5 / 0.3 / 39
c880	9.253	9.057	9.448	4.138	9.243	9.296	86.445	11	0.5 / 0.3 / 33
c1355	10.232	9.444	10.444	9.575	9.730	10.365	36.500	12	0.5 / 0.4 / 116
c1908	14.540	14.250	14.782	3.599	14.520	14.647	76.222	13	1.0 / 0.4 / 43
c2670	12.829	12.469	13.112	4.904	12.718	12.945	64.697	13	0.5 / 0.5 / 48
c3540	16.995	16.651	17.351	4.037	16.888	17.168	60.100	11	3.0 / 0.5 / 63
c5315	17.381	17.251	17.649	2.252	17.298	17.525	42.893	9	4.0 / 0.5 / 71
c6288	46.911	45.242	48.591	6.893	45.655	48.265	22.078	10	14 / 0.8 / 173
c7552	15.851	15.558	16.081	3.252	15.795	15.970	66.539	11	4.0 / 0.5 / 145

possible to compute the exact graph delay CDF. For larger circuits, Monte Carlo simulation with 100 000 samples was used. The statistical timing analysis was tested on the ISCAS benchmark circuits [34]. For each gate in the circuit, a delay distribution was specified with a standard deviation ranging from 10% to 15% of the mean of the distribution. A Gaussian distribution, truncated at the 3 sigma point, was used. This distribution of gate delay was based on Monte Carlo simulation for individual gate structures using SPICE simulation. In these simulations, intra-die gate length variations were applied, as measured for an industrial 0.13 μm process technology with test structures on a number of test chips. However, the proposed method for statistical timing analysis is not limited to any particular type of process variation and can also be used with delay variations resulting from other process factors.

In Table I, we show the circuit characteristics and results of the exact graph reduction for the benchmark circuits. The table shows the number of convergence nodes (*column 4*) and the total number of dependence nodes (*column 5*). The average (*column 6*) and maximum number of dependence nodes (*column 7*) per convergence node is also shown. Some circuits have extensive reconvergence, indicated by their high number of dependence nodes, making them difficult test cases for statistical timing analysis. The last two columns of Table I show the effectiveness of the exact reduction techniques. On average, the number of edges in the graph is reduced by 40% with a maximum reduction of 76%.

Table II shows the results for the bound computation and refinement through selective enumeration. The expected value of the lower bound (*column 3*) and upper bound (*column 4*) and their relative difference (*column 5*) is shown. Although we only report the expected value in Table II, the computed bounds are CDFs and allow the computation of other useful values, such as confidence points. The statistical upper and lower bounds have a relatively small difference in their expected value ranging from 2.25% to 10.45%, demonstrating their effectiveness. Also, the

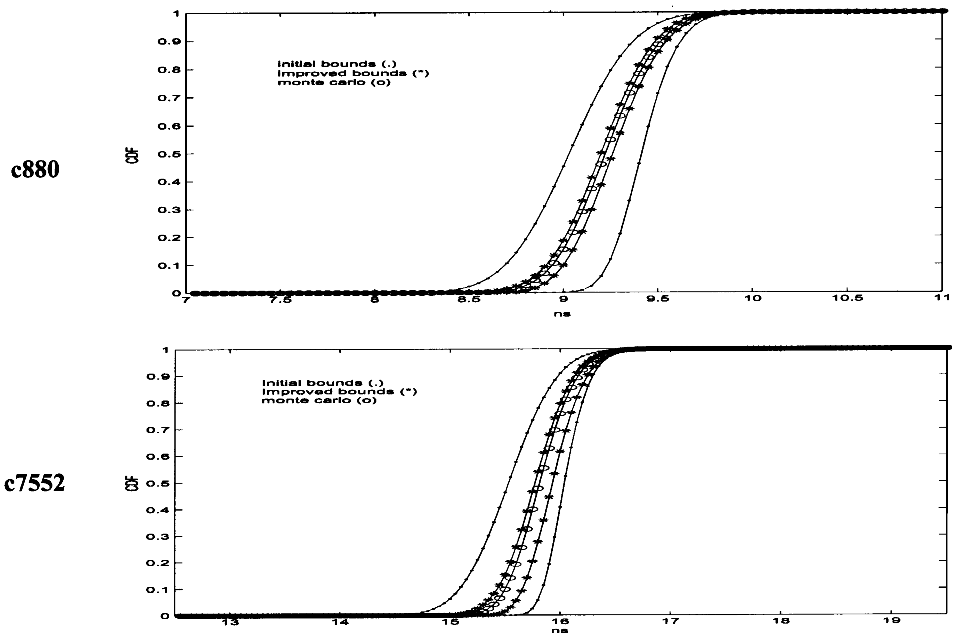

Fig. 17. Comparison of CDF bounds and Monte Carlo for c880 and c7552.

Monte Carlo results fall between the computed bounds, as expected.

Table II also shows the bounds after selective enumeration (*columns 6 and 7*) and the percentage improvement of their difference compared to the original bounds (*column 8*). The total number of dependence nodes enumerated during the bound computation is shown in column 9. Excluding the first circuit, where the selective enumeration obtained bounds that exactly matched the true graph delay, the improvement of the bounds using selective enumeration ranged between 22.07 – 86.44%, with an average of 62.45%. The number of dependence nodes selected for enumeration was small, ranging from 3 to 13 nodes, showing the somewhat surprising result that enumerating only a few carefully chosen nodes can significantly improve the bound computation.

Three run times are shown in column 10 of Table II: the run time required for circuit parsing, graph construction and graph reduction (*reduction*), the run time for computing the initial upper and lower bounds (*bounds*), and the run time for the iterative refinement of these bounds through selective enumeration (*enumeration*). The run time of the selective enumeration method is found to be modest, not exceeding 145 s for all test cases. Fig. 17 shows the CDFs for the proposed lower and upper bound with and without selective refinement as well as the CDF obtained through Monte Carlo simulation for the circuits c880 and c7552.

We compared the results obtained using the proposed statistical timing analysis method with traditional deterministic timing analysis. In Table III, the circuit delay, computed using the proposed statistical upper bound, is shown at three confidence points in columns 2–4. The 50% confidence point corresponds to the median circuit delay, while the 95% and 99% confidence points correspond to the expected delay at 95% and 99% performance yield. Columns 5–7, show the results from deterministic timing analysis and its percentage difference from the statistical timing analysis result. In each of the three deterministic timing analysis runs, each gate had a delay value fixed at, respectively, 50, 95, and 99% confidence points of its delay probability distribution. As can be seen from Table III, deterministic timing analysis, can be quite conservative for the higher confidence points, while it is optimistic for the median delay. The deterministic timing analysis overestimated the delay of the bounded statistical timing analysis by as much as 15% in the case of the 99% confidence point and underestimated the delay by as much as 16% for the 50% confidence point. Finally, the last two columns of Table III show the minimum and maximum circuit delay values with nonzero probability. As noted, these extreme values of the bound pdf correspond to the delay obtained with deterministic timing analysis when all gate delays are set at their minimum and maximum delay values with nonzero probability.

We next investigate the accuracy of approximating continuous gate delay distribution functions with discrete functions. We performed Monte Carlo simulation using both continuous and discrete gate delay pdfs, where the discrete pdfs were constructed using 8–10 delay/probability pairs. Fig. 18 shows the circuit delay distribution obtained with the discrete and continuous Monte Carlo simulations for circuit c7552 and shows that the two distributions are nearly indistinguishable. The error in the mean value of the circuit delay pdf obtained using discrete gate delay pdfs was 0.06%. It should also be noted that as arrival times are propagated in the circuit, the arrival time distributions increase in their width and number of discretizations. Since the computational complexity of convolution grows quadratically with the number of discretizations, the run time of bound computation can be improved significantly by pruning the long tails of such distributions. The analysis and run time results shown in this paper were obtained without such pruning. However, the literature reports a number of works exploring the trade-off between accuracy and run time using such pruning [28].

TABLE III
COMPARISON OF BOUNDED STATISTICAL STA WITH DETERMINISTIC STA

circuit	bounded statistical STA			deterministic STA			extreme values of bound	
	50%	95%	99%	50% / %diff	95% / %diff	99% / %diff	minimum	maximum
c17	1.38	1.52	1.58	1.25 / -9.29	1.57 / 2.97	1.72 / 7.93	0.85	1.85
c432	7.80	8.15	8.30	6.95 / -10.90	8.52 / 4.34	9.17 / 9.49	4.70	10.10
c499	5.20	5.39	5.46	4.33 / -16.83	5.42 / 0.55	5.89 / 7.30	3.00	6.30
c880	9.26	9.63	9.79	8.58 / -7.40	10.48 / 8.11	11.29 / 13.25	5.80	12.35
c1355	10.37	10.57	10.66	8.70 / -16.10	10.86 / 2.67	11.83 / 9.93	6.10	12.75
c1908	14.61	15.04	15.23	13.55 / -7.26	16.42 / 8.41	17.59 / 13.42	9.10	19.55
c2670	12.92	13.32	13.49	11.65 / -9.86	14.44 / 7.76	15.63 / 13.70	7.95	16.90
c3540	17.13	17.54	17.74	15.70 / -8.35	19.36 / 9.41	20.88 / 15.05	10.60	22.85
c5315	17.50	17.92	18.12	16.40 / -6.26	19.40 / 7.63	20.66 / 12.32	11.05	23.15
c6288	48.24	48.81	49.06	42.43 / -12.06	53.08 / 8.03	57.58 / 14.80	28.60	62.00
c7552	15.93	16.34	16.53	14.85 / -6.76	17.85 / 8.43	19.09 / 13.43	10.10	21.05

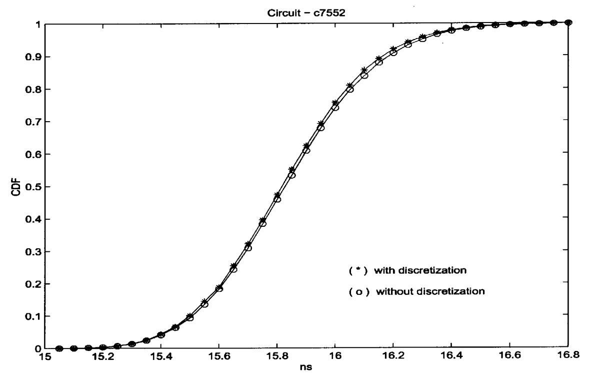

Fig. 18. Comparison of circuit delay distribution using continuous and discrete gate delay distributions.

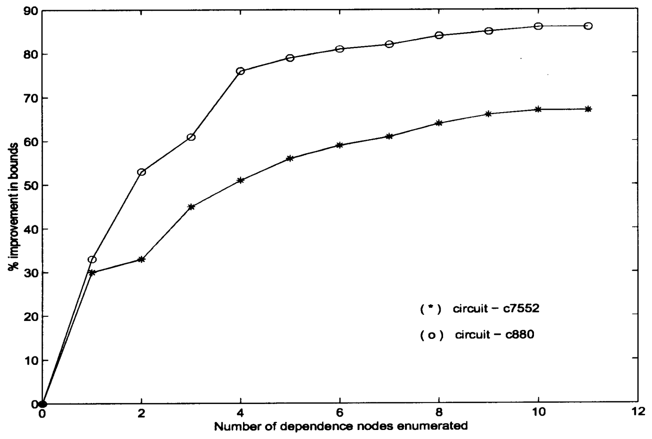

Fig. 19. Improvement in bound difference as a function of the number of enumerated dependence nodes.

Finally, we show the obtained improvement in the bound difference as a function of the number of enumerated dependence nodes for circuits C880 and C7552 in Fig. 19. The graph demonstrates that the bound difference decreases monotonically with the number of enumerated nodes. It also shows that the largest improvement in the bound difference is obtained from the first few selected dependence nodes and diminishes as the number of enumerated nodes increases. This demonstrates that enumeration of a few carefully selected dependence nodes can result in a significant improvement of the bound difference.

VIII. CONCLUSION

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 first presented a general method for statistical timing analysis as well as exact methods to reduce the problem size substantially. 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 on benchmark circuits that the obtained bounds are close in practice. In order to further reduce the difference between the bounds, we proposed an iterative refinement technique which selectively enumerates dependence nodes in the circuit. Using this technique, the difference in the expected value of the bounds could be reduced by 62.45% on average, with a modest run time increase.

REFERENCES

[1] R. B. Hitchcock, "Timing verification and the timing analysis program," in *Proc. IEEE/ACM Design Automation Conf.*, 1982, pp. 594–604.
[2] N. P. Jouppi, "Timing analysis for nMOS VLSI," in *Proc. IEEE/ACM Design Automation Conf.*, 1983, pp. 411–418.
[3] S. Nassif, "Delay variability: Sources, impacts and trends," in *Proc. ISSCC*, 2000, pp. 368–369.
[4] A. Kahng and Y. Pati, "Subwavelength optical lithography: Challenges and impacts on physical design," in *Proc. ISPD*, 1999, pp. 112–119.
[5] D. Boning *et al.*, "Manufacturing yield, reliability and failure analysis," in SPIE Symp. Microelectronic Manufacturing, Austin, TX, Oct. 1996.
[6] K. L. Shepard and V. Narayanan, "Noise in deep submicron digital design," in *Proc. ICCAD 1996*, San Jose, CA, pp. 524–53.
[7] D. Sylvester and K. Keutzer, "Getting to the bottom of deep submicron," in *Proc. ICCAD*, San Jose, CA, Nov. 1998, pp. 203–211.
[8] A. B. Kahng, S. Muddu, and E. Sarto, "On switching factor based analysis of coupled RC interconnects," in *Proc. IEEE/ACM Design Automation Conf.*, 2000, pp. 79–84.
[9] P. Chen, D. A. Kirkpatrick, and K. Keutzer, "Miller factor for gate-level coupling delay calculation," in *Proc. ICCAD*, San Jose, CA, 2000, pp. 68–74.
[10] S. Sapatnekar, "A timing model incorporating the effect of crosstalk on delay and its application to optimal channel routing," *IEEE Trans. Computer-Aided Design*, vol. 19, pp. 550–559, May 2000.

[11] A. Vittal, L. H. Chen, M. Marek-Sadowska, K.-P. Wang, and S. Yang, "Crosstalk in VLSI interconnections," *Trans. Computer-Aided Design*, vol. 18, pp. 1817–1824, Dec. 1999.
[12] G. Bai, S. Bobba, and I. N. Hajj, "Static timing analysis including power supply noise effect on propagation delay in VLSI circuits," *Proc. IEEE/ACM Design Automation Conf.*, pp. 295–300, 2001.
[13] K. Bowman, S. Duvall, and J. Meindl, "Impact of die-to-die and within-die parameter fluctuations on the maximum clock frequency distribution," in *Proc. ISSCC*, San Francisco, CA, 2001, pp. 278–279.
[14] V. Mehrotra, S. Nassif, D. Boning, and J. Chung, "Modeling the effects of manufacturing variation on high-speed microprocessor interconnect performance," *IEDM Tech. Dig.*, pp. 767–770, 1998.
[15] S. R. Nassif, "Modeling and analysis of manufacturing variations," in Proc. CICC, San Diego, CA, 2001, pp. 223–228.
[16] Y. Liu *et al.*, "Model-order reduction of RC(L) interconnect including variational analysis," *Proc. IEEE/ACM Design Automation Conf.*, pp. 201–206, 1999.
[17] E. Acar *et al.*, "Assessment of true worst-case circuit performance under interconnect parameter variations," *Proc. IEEE Int. Symp. Quality Electronic Design*, pp. 431–436, 2001.
[18] M. Orshansky, L. Milor, P. Chen, K. Keutzer, and C. Hu, "Impact of systematic spatial intra-chip gate length variability on performance of high-speed digital circuits," in *ICCAD 2000*, San Jose, CA, pp. 62–67.
[19] V. Mehrotra, S. L. Sam, D. Boning, A. Chandrakasan, R. Vallishayee, and S. Nassif, "A methodology for modeling the effects of systematic within-die interconnect and device variation on circuit performance," in *Proc. IEEE/ACM Design Automation Conf.*, Los Angeles, CA, 2000, pp. 172–175.
[20] Y. Deguchi, N. Ishiura, and S. Yajima, "Probabilistic CTSS: Analysis of timing error probability in asynchronous logic circuits," *Proc. IEEE/ACM Design Automation Conf.*, pp. 650–655, 1991.
[21] S. Devadas, H. F. Jyu, K. Keutzer, and S. Malik, "Statistical timing analysis of combinational circuits," in *Proc. ICCD 1992*, Cambridge, MA, pp. 38–43.
[22] H. F. Jyu and S. Malik, "Statistical timing optimization of combinational logic circuits," in *Proc. ICCD 1993*, Cambridge, MA, pp. 77–80.
[23] R. B. Brawhear, N. Menezes, C. Oh, L. Pillage, and R. Mercer, "Predicting circuit performance using circuit-level statistical timing analysis," in *Proc. Eur. Design and Test Conf.*, Paris, France, 1994.
[24] M. Berkelaar, "Statistical delay calculation, A linear time method," in *Proc. TAU 97*, Austin, TX, Dec. 1997, pp. 15–24.
[25] E. T. A. F. Jacobs and M. R. C. M. Berkelaar, "Gate sizing using a statistical delay model," *Proc. IEEE/ACM Design Automation and Test Eur. Conf.*, pp. 283–290, 2000.
[26] R.-B. Lin and M.-C. Wu, "A new statistical approach to timing analysis of VLSI circuits," in *Proc. Int. Conf. VLSI Design*, Chennai, India, 1998, pp. 507–513.
[27] S. Tongsima, C. Chantrapornchai, E. H.-M. Sha, and N. L. Passos, "Optimizing circuits with confidence probability using probabilistic retiming," *Proc. IEEE ISCAS*, pp. 270–273, 1998.
[28] J. J. Liou, K. T. Cheng, S. Kundu, and A. Krstic, "Fast statistical timing analysis by probabilistic even propagation," in *Proc. IEEE/ACM Design Automation Conf.*, Las Vegas, NV, 2001, pp. 661–666.
[29] J.-J. Liou, A. Krstic, L.-C. Wang, and K.-T. Cheng, "False path aware statistical timing analysis and efficient path selection for delay testing and timing validation," *Proc. ACM/IEEE Design Automation Conf.*, pp. 566–569, 2002.
[30] M. Orshansky and K. Keutzer, "A general probabilistic framework for worst-case timing analysis," in *Proc. ACM/IEEE Design Automation Conf.*, New Orleans, LA, 2002, pp. 556–569.
[31] A. Gattiker, S. Nassif, R. Dinakar, and C. Long, "Timing yield estimation from static timing analysis," in *Proc. ISQED 2001*, San Jose, CA, pp. 437–442.
[32] X. Bai, C. Visweswariah, P. Strenski, and D. Hathaway, "Uncertainty-aware circuit optimization," *Proc. ACM/IEEE Design Automation Conf.*, pp. 58–63, 2002.
[33] F. Najm, R. Burch, P. Yang, and I. Hajj, "Probabilistic simulation for reliability analysis of CMOS VLSI circuits," *IEEE Trans.Computer-Aided Design*, vol. 9, Apr. 1990.
[34] F. Brglez and H. Fujiwara, "A neutral netlist of 10 combinatorial benchmark circuits," *Proc. IEEE ISCAS*, pp. 695–698, 1985.
[35] W. P. Feller, *An Introduction to Probability Theory and its Applications*. New York: Wiley, 1970, vol. 1–2.

Aseem Agarwal (S'03) received the B.E. degree in electronics and communication from Gujarat University, India, in 2001. Since August 2001, he is a Doctoral Student in Department of Electrical Engineering and Computer Science at University of Michigan, Ann Arbor.

He is a Research Assistant in the Advanced Computer Architecture Lab working with Prof. D. Blaauw. His research focuses on timing analysis models and algorithms that consider process variability.

Vladimir Zolotov (M'97) received the Engineer degree from the Moscow Institute of Electronics, Moscow, Russia, and the Ph.D. degree from the Scientific Research Institute of Micro Devices, Moscow, Russia, both in electrical engineering.

He has been with the Advanced Tools Group, Motorola, Inc., Austin, TX, since 1998. He is involved in the development of EDA tools and methodology for high-performance VLSI designs. Previously, he was with the Moscow Research Laboratory, Motorola, Inc., Moscow, Russia. His research interests include signal integrity, reliability, on-chip inductance, timing analysis, and optimization of VLSI.

David T. Blaauw (M'01) received the B.S. degree in physics and computer science from Duke University, Durham, NC, in 1986, and the M.S. and Ph.D. degrees in computer science from the University of Illinois, Urbana, in 1988 and 1991, respectively.

He was with the Engineering Accelerator Technology Division, IBM Corporation, Endicott, NY, as a Development Staff Member, until August 1993. From 1993 to August 2001, he was with Motorola, Inc., Austin, TX, where he was the Manager of the High Performance Design Technology Group. Since August 2001, he has been a member of the faculty at the University of Michigan, Ann Arbor, as an Associate Professor. His work has focused on VLSI design and CAD with particular emphasis on circuit analysis and optimization problems for high-performance and low-power designs.

Dr. Blaauw was the Technical Program Chair and General Chair for the International Symposium on Low Power Electronics and Design, in 1999 and 2000, respectively, and was the Technical Program Co-Chair and member of the Executive Committee of the ACM/IEEE Design Automation Conference in 2000 and 2001.