A Statistical Static Timing Analysis
Considering Correlations Between Delays

Shuji Tsukiyama*  
Masakazu Tanaka*  
Masahiro Fukui*

+ Dept. of EECE  
Chuo University  
Tokyo, JAPAN 112-8551  
tsuki@elect.chuo-u.ac.jp

* Advanced LSI Tech. Development Center  
Matsushita Electric Industrial Co., Ltd.  
Nagakakyo, JAPAN 617-8520  
{tanaka, fukui}@ngk.csdd.mei.co.jp

Abstract: In this paper, we present a new algorithm for the statistical static timing analysis of a CMOS combinatorial circuit, which can treat correlations of arrival times of input signals to a logic gate and correlations of switching delays in a logic gate. We model each switching delay by a normal distribution, and use a normal distribution of two stochastic variables with a coefficient of correlation for computing the distribution of output delay of a logic gate. Since the algorithm takes the correlation into account, the time complexity is \(O(n^m)\) in the worst-case, where \(n\) and \(m\) are the numbers of vertices and edges of the acyclic graph representing a given combinatorial circuit.

I. INTRODUCTION

The importance of statistical static timing analysis[1-5] is increasing in designing high density, high speed and low power VLSIs in deep sub-micron era. Because, designers often set excessive margins derived from the worst-case analysis in order to avoid the effect of the delay time uncertainty, and such excessive margins usually bring over-design of circuits. If designers can estimate the distribution of critical path delay caused by all local uncertainties, over-design of circuits may be eliminated so that high density and high performance VLSIs are produced with high yield[6-8].

Although several researches have been done on this topic[1-7], all of them except [5] assume that the distributions of all signal delays are independent each other. However, such an assumption is not realistic in combinatorial circuits with re-convergent paths. For example, if the delays of signals \(x\) and \(y\) of the circuit shown in Fig. 1 are heavily depend on the delay of signal \(v\), then we cannot ignore the correlation between delays of signals \(x\) and \(y\) in computing the maximum delay of signal \(z\).

In this paper, we present a new algorithm for the statistical static timing analysis of a CMOS combinatorial circuit, which can treat not only correlations of delays of signals denoted in Fig. 1 but also correlations between delays of transistors contained in a logic gate. This algorithm assumes that the delay of each logic gate is modeled by a normal distribution (Gaussian distribution), as usually done in the previous researches. In order to compute the distribution of the output delay of a logic gate, a maximum operation on the stochastic variables is necessary, and the proposed algorithm uses a normal distribution of two stochastic variables with a coefficient of correlation[9] for the maximum operation.

If delays of all signals are independent, the distribution of the maximum delay can be computed in \(O(n^m)\) time[3], where \(n\) and \(m\) are the numbers of vertices and edges of the graph representing a given combinatorial circuit. But, since the proposed algorithm takes the correlation into account, the time complexity of the algorithm becomes \(O(n^m)\) in the worst-case. However, for real combinatorial circuits, we can expect that the time complexity is much less than \(O(n^m)\).

II. PRELIMINARIES

In order to find the distribution of the maximum delay of a CMOS combinatorial circuit, we represent the circuit by an acyclic graph \(G=(V,E)\), as shown in Fig. 2. In the figure, each box denotes a logic gate, which is drawn to show the correspondence between the circuit and the graph. Each vertex contained in a box represents a terminal of the corresponding logic gate, and a vertex in the left or right side in a box corresponds to an input or an output terminal, respectively. Each sink, from which no edge goes out, corresponds to a primary output, and \(T\) denotes the set of

\[\text{Fig. 1 Arrival times of signals x and y have a correlation}\]
sinks. Vertex $v_0$ is the unique source into which no edge comes, and each edge going out from $v_0$ comes into a vertex corresponding to a primary input. These edges are used to introduce the differences and distributions of arrival times of primary inputs. Each edge in a box goes out from the vertex representing an input of the corresponding logic gate, and comes into the vertex representing the output of the gate. Edges exclusive of the ones going out from $v_0$ represent interconnects, if they are not contained in a box.

Each edge $e=(v,w)$ contained in a box has delays $t_d(e)$ and $t_l(e)$, such that $t_d(e)$ and $t_l(e)$ denote the switching times of pMOS and nMOS transistors, respectively. Henceforth, we use "b" to indicate 0 or 1, and $t_b(e)$ denotes $t_d(e)$ or $t_l(e)$. Delay $t_b(e)$ has a certain uncertainty, and is a stochastic variable. It is determined by saturated current $I_{sat}$, load capacitance $C_{load}$ of the transistor, and slew rate $t_{slew}$ of the gate voltage. The uncertainty of the delay comes from the distribution of $I_{sat}$, which mainly depends on the distribution of gate length $L_g$ of the transistor. Since the distribution of $L_g$ can be modeled by a normal distribution like the distribution of threshold voltage $V_t$ [10], we model the distribution of delay $t_b(e)$ of edge $e$ in a box by a normal distribution $N(t_b(e), \sigma^2_b(e))$, where $t_b(e)$ and $\sigma^2_b(e)$ are the mean and the variance of the distribution of $t_b(e)$.

The distribution of $L_g$ depends on the distributions of space $S_{poly}$ between adjacent polysilicon gates, and gate width $W_g$ and length $L_{diff}$ of diffusion area of the transistor. Therefore, by extracting these quantities from the mask pattern, the distribution of $L_g$ can be estimated, and hence the distribution of the switching delay $t_b(e)$ of a transistor. With the use of the distribution data and the proposed algorithm, we can execute post-layout timing analysis, gate size optimization[6,7], and layout optimization of a macrocell[8].

Since the distribution of $L_g$ depends on the distribution of space $S_{poly}$ between adjacent gates, delay $t_b(e)$ has a correlation with the delay of the edge corresponding to the adjacent transistor. Therefore, we introduce a correlation between delays $t_b(e')$ and $t_b(e'')$ of edges $e'$ and $e''$ which correspond to transistors with the same type in a logic gate, and denote the coefficient of correlation between these delays by $\rho_b(e',e'')$. Such correlations are introduced only for edges contained in a single logic gate.

For an edge $e$ representing an interconnect, delay $t_b(e)$ designates the interconnect delay, which is assumed to be constant, that is, $\sigma_b(e)=0$. This delay is evaluated by a method such as PRIMO[11]. If interconnect delay is also distributed by a normal distribution, then we may introduce a variance $\sigma_b(e)=\sigma$. But, in such a case, the proposed algorithm must be modified, because the distributions of $t_b(e)$ and $t_l(e)$ may have a correlation.

Now, let us show that the maximum delay to a vertex $w$ in $G$ can be estimated by using the edge-delays introduced above.

Let $d(v,0)$ and $d(v,1)$ be the maximum delays spent for transmitting signal 0 and 1 from source $v_0$ to a vertex $v \in V$, respectively. These are stochastic variables. In the following, the mean and variance of $d(v,b)$ are denoted by $m_b(v) = \text{Exp}[d(v,b)]$ and $s_b(v) = \text{Var}[d(v,b)]$, respectively.

Firstly, we consider the case where vertex $w$ corresponds to a sink or an input node of a gate. In this case, only one edge $e=(v,w)$ corresponding to an interconnect comes into $w$, and hence $d(v,0)$ and $d(v,1)$ can be calculated by the following equations:

$$d(v,0) = d(v,0) + t_d(e), \quad d(v,1) = d(v,1) + t_l(e).$$

Since $t_d(e)$ and $t_l(e)$ are constant, $m_b(w)$ and $s_b(w)$ are obtained as follows:

$$m_b(w) = m_b(v) + t_d(e), \quad m_l(w) = m_l(v) + t_l(e),$$
$$s_d(w) = s_d(v), \quad s_l(w) = s_l(v).$$

Moreover, correlation coefficient $\tau_{b}(w,v) = \text{Cov}[d(w,b), d(v,b)]$ between $d(w,b)$ and $d(v,b)$ of a vertex $u \neq w$ can be obtained by the following equations. Henceforth, we use $b'$ together with $b$ to indicate 0 or 1, and $b'$ is not necessarily equal to $b$.

$$r_{00}(w,u) = r_{00}(v,u), \quad r_{01}(w,u) = r_{01}(v,u),$$
$$r_{10}(w,u) = r_{10}(v,u), \quad r_{11}(w,u) = r_{11}(v,u).$$

Obviously, we have

$$r_{00}(w,w) = r_{11}(w,w) = 1, \quad r_{01}(w,w) = r_{10}(w,w) = r_{01}(v,v) = r_{10}(v,v).$$

Next, let us consider the case where $w$ corresponds to the output node of a logic gate with $k$ inputs. Let $v_i (i=1,2,...,k)$ be a vertex corresponding to an input node of the gate, and let $e_i = (v_i,w)$ be incoming edges of $w$. (See Fig. 3)

If the logic gate is a NAND gate, then $d(v,0)$ is calculated by the following equation:

$$d(v,0) = \max\{d(v_1) + t_d(e_1), \ldots, d(v_k) + t_d(e_k)\}$$

since $w$ becomes 0 when all inputs become 1. On the other hand, since $w$ becomes 1 when any input becomes 0, $d(v,1)$ seems to be determined by the equation

$$d(v,1) = \min\{d(v_1) + t_l(e_1), \ldots, d(v_k) + t_l(e_k)\}.$$
$$d(w,1) = \min\{d(v_i,0) + t_0(e_i) \mid 1 \leq i \leq k\}.$$ However, when all inputs other than $j$ are 1, $w$ becomes 1 after the delay $d(v_i,0) + t_0(e_i)$. Since $d(w,1)$ is the maximum among these delays $d(v_i,0) + t_0(e_i)$, we can calculate $d(w,1)$ by the following equation:

$$d(w,1) = \max\{d(v_i,0) + t_0(e_i) \mid 1 \leq i \leq k\}.$$ Similarly, if the logic gate is NOR gate, we can calculate $d(w,0)$ and $d(w,1)$ by the same maximum operations. If the logic gate is an inverter, we set $k = 1$ in the above equations and need not to take the maximum.

If the logic gate is a complex gate, for example, $v_i \Rightarrow v_j$, $w$ becomes 0 if $v_j = 1$ and $v_j = 1$ or $v_j = 1$. Therefore, $d(w,0)$ may be determined largest value of $d(v_i,0) + t_0(e_i) (1 \leq i \leq 3)$. On the other hand, $d(w,1)$ may also be determined by the largest values of $d(v_i,0) + t_0(e_i) (1 \leq i \leq 3)$. Thus, a complex gate consists of a series-parallel connection of MOS transistors, then we can determine $d(w,b)$ by the same manner as NAND or NOR gate. It is not hard to see that for an AND gate or an OR gate a similar discussion holds, and $d(w,b)$ can be computed by

$$d(w,b) = \min\{d(v_i,b) + t_0(e_i) \mid 1 \leq i \leq k\}$$

although the definition of $t_0(e_i)$ must be modified slightly.

For the logic gates discussed above, we only need the values of $d(v_i,b)$ and $t_0(e_i)$, and not $d(w,b)$ and $t_0(e_i)$ simultaneously, where $b$ denotes the complement of $b$. However, if the logic gate is an XOR gate, then output $w$ becomes 1 by input signal 1 or 0. Hence, $d(w,1)$ may be equal to $d(v_i,0) + t_0(v_j) + t_1(v_j) + t_0(v_j)$ (1 $\leq i \leq k$), where $t_0(v_j)$ (or $t_1(v_j)$) is the switching delay between the arrival time of signal 0 (signal 1) to input $v_j$ and the time when output $w$ turned to 1. Therefore, the definition of $d(w,1)$ is computed by the following equation.

$$d(w,1) = \max\{d(v_i,0) + t_0(v_j) + t_1(v_j) \mid 1 \leq i \leq k\}.$$ Through a similar discussion, we see that $d(w,0)$ is computed by

$$d(w,0) = \max\{d(v_i,0) + t_0(v_j) + t_0(v_j) \mid 1 \leq i \leq k\}.$$ Thus, in the case of XOR gate, we need new edge-delays different from $t_0(e_i)$ and a different maximum operation. In order to simplify the descriptions, we assume in the following that a given combinatorial circuit does not contain XOR gates. Therefore, the maximum operation is conducted only among $d(*,0)$ or $d(*,1)$. The following holds.

### III. Distributed of the Maximum

In this section, we show a method to compute mean $\text{Exp}[x(b)]$ and variance $\text{Var}[x(b)]$ of $x(b) = \max\{d(v_i,b), t_0(e_i) \mid 1 \leq i \leq k\}$. This is done by procedure $\text{DISTMAX}(\ln(w), U)$, where $\ln(w) = \{e_i = (v_i,w) \mid 1 \leq i \leq k\}$, and $U$ is the set of vertices to which there is no directed path from $w$. The procedure is computed recursively $\text{EXP}[x(b)]$ and $\text{VAR}[x(b)]$ for $b = 0$, and then for $b = 1$. Moreover, it computes correlation coefficient $R\{x(b), t(b), y\}$ between $x(b)$ and $t(b)$, and $x(b)$, respectively, mean $\text{EXP}[t]$ and variance $\text{VAR}[t]$ of $t = \max\{x, y\}$ are obtained by the following equations, unless $x_1 = x_2 = 0$:

$$\text{EXP}[t] = \mu_x \Phi(0) + \mu_y \Phi(0) + \sigma_x \varPhi(\sigma_x^2) \Phi(\beta)$$

$$\text{VAR}[t] = (\mu_x^2 + \sigma_x^2 - 2\mu_x \sigma_x) \Phi(2\beta)$$

where

$$\alpha = \frac{\sigma_x^2 + \sigma_y^2 - 2\sigma_x \sigma_y \rho}{\sigma_x^2 \sigma_y^2}, \quad \beta = (\mu_x - \mu_y) / \sigma_x$$

$$\varPhi(x) = \frac{1}{\sqrt{2\pi}} \exp \left( -\frac{x^2}{2} \right)$$

Moreover, correlation coefficient $R[t, z]$ between $t = \max\{x, y\}$ and stochastic variable $z$ with a normal distribution can be obtained by the following equation, if correlation coefficient between $x$ and $z$ is $R[x, z] = \rho_x$ and that between $y$ and $z$ is $R[y, z] = \rho_y$.

$$R[t, z] = R[t, \max\{x, y\}, z] = \frac{\alpha x \rho_y \Phi(\beta) + \alpha y \rho_y \Phi(\beta)}{\sqrt{\text{VAR}[t]}}$$

We use these equations in procedure $\text{DISTMAX}(\ln(w), U)$ as follows.

Let $x(b)^* = d(v_i,b) + t_0(e_i)$. Then, since $d(v_i,b)$ and $t_0(e_i)$ are independent, we have

$$\text{EXP}[x(b)^*] = \mu_x \Phi(0) + \mu_y \Phi(0)$$

$$\text{VAR}[x(b)^*] = \sigma_x^2 \sigma_y^2$$

355
Moreover, for \( x(b)_i \), and stochastic variable \( y \), the following equation holds

\[
\sqrt{\text{Var}[x(b)_i]} \cdot \text{R}(x(b)_i, y) = \frac{s(v,y) \cdot \text{Var}[d(v,b), y] + \alpha_b(c_e) \cdot \text{R}(t(b), y)}{\sqrt{\text{Var}[x(b)_i]}}.
\]

If \( y = d(u,b) \) of vertex \( u \in U \), then \( \text{R}(t(b), y) = 0 \). Therefore, for \( u \in U \), we have

\[
\text{R}(x(b)_i, d(u,b)) = \frac{s(v,y) \cdot \text{Var}[x(b)_i]}{\sqrt{\text{Var}[x(b)_i]}}.
\]

where \( r_b(v_i,u) = \text{R}[d(v,b), d(u,b)] \). If \( y = t_1(c) \) of edge \( e_i \in \text{In}(w) \), then \( \text{R}[x(0)_i, t_1(c)] = \text{R}[x(1)_i, t_1(c)] = 0 \), and we have

\[
\text{R}(x(b)_i, t_1(c)) = \alpha_c(c_e) \cdot \text{R}(x(b)_i, t_1(c)) / \sqrt{\text{Var}[x(b)_i]}.
\]

We consider correlation coefficients \( \text{R}(x(b)_i, t_j(c)) \) only for edges \( e_i \in \text{In}(w) \) such that \( i < j \), since we need these coefficients only.

Now, let \( x(b)_i = \max[d(v,b)_i + t_1(c)_j] \) for \( 1 \leq j \leq i \). Then, since \( x(b)_i = x(b)_i \), for \( i \geq 2 \), we can compute \( \exp(x(b)_i) \), \( \text{Var}[x(b)_i] \), \( \text{R}[x(b)_i, d(u,b)] \), and \( \text{R}[x(b)_i, t_1(c)] \) by using

\[
x(b)_i = \max[x(b)_1, x(b)_2, \ldots, x(b)_n]
\]

and the equations in Ref.[9]. Namely, let

\[
\alpha = \sqrt{\text{Var}[x(b)_i]} + \frac{\alpha}{\sqrt{\text{Var}[x(b)_i]}} - 2\gamma,
\]

\[
\gamma = \sqrt{\text{Var}[x(b)_i]} \cdot \frac{\text{R}[x(b)_i, d(v,b)] + \alpha \cdot \text{R}[x(b)_i, t_1(c) - \gamma]}{\sqrt{\text{Var}[x(b)_i]}}
\]

and

\[
\beta = \frac{\text{R}(x(b)_i, x(b)_i)}{\sqrt{\text{Var}[x(b)_i]}}.
\]

Then, we can compute \( \exp(x(b)_i) \), \( \text{Var}[x(b)_i] \), and \( \text{R}[x(b)_i, d(u,b)] \) for \( u \in U \) as follows.

\[
\exp(x(b)_i) = \exp(x(b)_i + \Phi(\beta)) + \exp(x(b)_i)^\Phi(\beta) + \alpha \cdot \exp(x(b)_i)^\Phi(\beta),
\]

\[
\text{Var}[x(b)_i] = ((\exp(x(b)_i))^2 + \text{Var}[x(b)_i]) \cdot \Phi(\beta) + ((\exp(x(b)_i))^2 + \text{Var}[x(b)_i]) \cdot \Phi(\beta) + (\exp(x(b)_i))^2 + \text{Var}[x(b)_i]) \cdot \Phi(\beta) - (\exp(x(b)_i))^2,
\]

\[
\text{R}(x(b)_i, d(u,b)) = \frac{X + Y}{\sqrt{\text{Var}[x(b)_i]}},
\]

\[
X = \frac{\text{Var}[x(b)_i]}{\sqrt{\text{Var}[x(b)_i]}} \cdot \text{R}(x(b)_i, d(u,b)) \cdot \Phi(\beta),
\]

\[
Y = \frac{\text{Var}[x(b)_i]}{\sqrt{\text{Var}[x(b)_i]}} \cdot \text{R}(x(b)_i, d(u,b)) \cdot \Phi(-\beta).
\]

Correlation coefficient \( \text{R}[x(b)_i, t_1(c)] \) for \( i > j \) can be computed as follows.

\[
\text{R}[x(b)_i, t_1(c)] = \frac{X + Y}{\sqrt{\text{Var}[x(b)_i]}}
\]

\[
X = \frac{\text{Var}[x(b)_i]}{\sqrt{\text{Var}[x(b)_i]}} \cdot \text{R}[x(b)_i, t_1(c)] \cdot \Phi(\beta),
\]

\[
Y = \frac{\text{Var}[x(b)_i]}{\sqrt{\text{Var}[x(b)_i]}} \cdot \text{R}[x(b)_i, t_1(c)] \cdot \Phi(-\beta).
\]

Repeating the computations stated above for \( i = 2, 3, \ldots, k \), we have the distribution of \( x(b)_k \), since \( x(b)_k = x(b)_k \). Hence, by conducting this repetition for \( b = 0 \), we have \( \exp(x(0)_i, \gamma \cdot \text{Var}[x(0)_i], \text{R}[x(0)_i, d(u,b)] \) for \( u \in U \). Then, conducting the repetition for \( b = 1 \), we obtain \( \exp(x(1)_i, \gamma \cdot \text{Var}[x(1)_i], \text{R}[x(1)_i, d(u,b)] \). In this case, however, we need not to compute \( \text{R}[x(1)_i, d(v,w)] \) for each such that \( e_i \in \text{In}(w) \). Because, we do not use these values. But, we compute \( \text{R}[x(1)_i, x(0)_i] \) as follows. Since \( t_1(c), x(0)_i = 0 \), we have

\[
\text{R}[x(1)_i, x(0)_i] = \frac{X + Y}{\sqrt{\text{Var}[x(1)_i]}}
\]

\[
X = \frac{\text{Var}[x(1)_i]}{\sqrt{\text{Var}[x(1)_i]}} \cdot \text{R}[x(1)_i, x(0)_i] \cdot \Phi(\beta),
\]

\[
Y = \frac{\text{Var}[x(1)_i]}{\sqrt{\text{Var}[x(1)_i]}} \cdot \text{R}[x(1)_i, x(0)_i] \cdot \Phi(-\beta).
\]

Hence, we have \( \text{R}[x(1)_i, x(0)_i] = \text{R}[x(1)_i, x(0)_i] = \text{R}[x(1)_i, x(0)_i] \).

IV. Algorithm

In this section, we describe the proposed algorithm by using procedure DISTMAX( \( \text{In}(w), U \) ) shown in the previous section.

Our problem is to find mean \( m(v) \) and variance \( s(v) \) of \( \{d(v,b) \} \) for each sink \( v \in T \) and correlation coefficient \( n_b(v,w) \) for each pair of sinks \( v, w \in T \). Once these values are obtained, we can compute the probability for the maximum delay \( M = \max \{d(v,b) : v \in T, b \in (0,1)\} \) to be not greater than \( D \), from the following equation

\[
\text{Prob} \{\text{Max} D \} = \int_{T} f_{n_b(v,w)} f_{d(v,x_1,\ldots,x_n)} \text{dx}_1 \cdots \text{dx}_n
\]

where \( n = 2|T| \), each \( x_i \) corresponds to \( d(v,b) \), and \( f_{d(v,x_1,\ldots,x_n)} \) is the probability density function of normal distribution with \( 2|T| \) variables. If we need its probability density function \( g(D) \) of distribution \( G(D) = \text{Prob} \{\text{Max} D \} \), it can be obtained by differentiating \( G(D) \). Although mean \( \text{Exp} \{g(D) \} \) and variance \( \text{Var} \{g(D) \} \) of the maximum delay \( M \) are computed numerically, they can be computed by a method similar to procedure DISTMAX by assuming \( g(D) \) as
a normal distribution. The outline of the proposed algorithm is as follows.

We first reduce a given circuit graph \( G=(V,E) \) by repeating the reduction of two series edges \( e^*=u(v) \) and \( e^*=v(w) \) into one edge \( e^*=(u,w) \). Since the delays of two series edges are independent, the mean and variance of \( t_b(e^*) \) are given by

\[
\mu_b(e^*) = \mu_b(e^*) + \mu_b(e^*),
\]

\[
\sigma_b^2(e^*) = \sigma_b^2(e^*) + \sigma_b^2(e^*),
\]

respectively, and correlation coefficient \( \rho_b(e^*,e) \) of \( t_b(e^*) \) with \( t_b(e) \) of edge \( e \) coming into \( w \) is given by

\[
\rho_b(e^*,e) = \frac{\sigma(e^*)\sigma(e) + \sigma(e)\sigma(e^*)}{\sqrt{\sigma_b^2(e^*)\sigma_b^2(e^*)\sigma_b^2(e)\sigma_b^2(e)}}.
\]

In these equations, if \( \sigma_b(*) = 0 \), then \( \rho_b(\cdot,\cdot) = 0 \).

Then, determine \( m_b(v) \) and \( s_b(v) \) of each vertex \( v \) of the reduced graph \( G^*=(V^*,E^*) \) in the topological order with the use of procedure DISTMAX as follows. Let \( Front \) be a set of vertices satisfying the following conditions.

(A) For any vertex \( v \in Front \), \( m_b(v) \) and \( s_b(v) \) of \( d(v,0) \) and \( m_1(v) \) and \( s_1(v) \) of \( d(v,1) \) are known.

(B) For any pair of vertices \( u, v \in Front \), \( t_{00}(u,v) \), \( t_{01}(u,v) \), \( r_{00}(u,v) \), and \( r_{11}(u,v) \) are known.

Initially, set \( Front \) as the set of vertices corresponding to primary inputs, and we see that conditions (A) and (B) can be satisfied for this \( Front \). Then, repeat the following procedure, until \( Front \) becomes \( T \).

1°: select a vertex \( w \notin Front \) such that all incoming edges of \( w \) come from vertices in \( Front \);

2°: let \( Del(w) \) be the set of vertices \( v \in Front \) such that all outgoing edges \( (v,u) \) from \( v \) come into vertices \( u \in Front \cup \{w\} \);

3°: set \( Front = Front - Del(w) \);

4°: conduct procedure DISTMAX (\( \{In(w) \in \text{Front}\), \( \text{and determine } m_b(w), s_b(w), \text{ and } r_{bb}(w,u) \) for \( u \in \text{Front} \), according to the type of the gate containing \( w \) as its output terminal. Namely, for example, if the gate is AND gate, then set \( m_b(w) = \text{Exp}[x(b)] \) and so on. Moreover, since we have \( R[x(0), x(1)] \), we can determine \( r_{bb}(w,u) \) as follows:

\[
t_{00}(w,w) = r_{11}(w,w) = 1,
\]

\[
t_{01}(w,w) = r_{10}(w,w) = R[x(1), x(0)].
\]

5°: add \( w \) to \( Front \);

If \( Front \) becomes \( T \), we compute \( \text{Exp}[M_b] = \text{Exp}\{d(v,b) \in T\}, \text{Var}[M_b], \text{ and } R[M_0, M_1] \), with the use of procedure DISTMAX (\( \{ (u,1) \in T \} \)), under the assumption that the delay of all edges \( (u,t) \) is constant and equal to 0. Then, we compute \( \text{Exp}[M] = \text{Exp}[M_0, M_1] \) and \( \text{Var}[M] \) by the equations in [9].

The time complexity of the proposed algorithm can be analyzed as follows.

Let \( k_w \) be the number of in-coming edges of a vertex \( w \in V \), and \( h_{\text{max}} \) be the maximum number of vertices contained in \( Front \) simultaneously. Then, the time required for adding a vertex \( w \) to \( Front \) can be denoted by \( O(k_w \cdot h_{\text{max}}) \), since for each edge \( e_i = (v_i,w) \), correlation coefficient \( R[x(b), d(u,0)] \text{ and } R[x(b), d(u,1)] \text{ are computed for each } u \in Front \text{ and } R[x(b), t_b(e_i)] \text{ for each edge } e_i = (v_i,w) \in \text{In}(w) \text{ such that } j \neq i \). Since the maximum value of \( k_w \) corresponds to the maximum number of inputs of a logic gate, we may assume \( h_{\text{max}} = k_w \) so that the time needed in addition of a vertex \( w \) to \( Front \) is \( O(k_w \cdot h_{\text{max}}) \). Therefore, the total time complexity is \( O(m \cdot h_{\text{max}}) \), since no vertex is added to \( Front \) more than once and \( \sum w \in V k_w \leq m \), where \( m = |E| \) is the number of edges of a given circuit graph \( G=(V,E) \).

In the worst-case, the maximum size \( h_{\text{max}} \) of \( Front \) is \( O(n) \), where \( n = |V| \) is the number of vertices. However, in real circuits, \( h_{\text{max}} \) is smaller than \( n \), so that the time complexity of the algorithm is less than \( O(n \cdot m) \).

V. EXPERIMENTAL RESULTS

In order to see the performance of the proposed algorithm, we first applied our algorithm to the graph in Fig. 4, which corresponds to the circuit in Fig. 1. The mean \( \mu \) and standard deviation \( \sigma \) of delays of edges \((a,x) \) and \((c,y) \) are 20 and 1.4, respectively, and those of edges \((v,x) \), \((v,y) \), \((x,z) \), and \((y,z) \) are 10 and 0.7, respectively. We changed the delay of \((a,v) \), and computed the maximum delay to vertex \( z \), as shown in Table 1. In the experiments, we assumed that all edge-delays are independent.

In the table, "Monte Carlo," "Ours," and "No Correlation" show the results obtained by Monte Carlo simulation (20,000 iterations), by our algorithm, and by ignoring correlation of delays \( d(x,b) \) and \( d(y,b) \), respectively, and "error" is the relative error to the result of Monte Carlo simulation. We showed in the table correlation coefficient \( r(x,y) \) and cpu-time spent in ULTRA-SPARC III 400MHz workstation, too. Fig. 5 shows the distributions of the maximum delay \( d(z) \) in the case of the top row in the table. Although the difference between "Ours" and "No Correlation" may not be large, the difference increases if \( \sigma \) becomes large.

![Graph representing the circuit of Fig. 1](image_url)
TABLE 1 DELAYS OF EDGE (u,v) AND THE MAXIMUM DELAY d(x)

<table>
<thead>
<tr>
<th>delay of edge (u,v)</th>
<th>maximum delay d(x)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Monte Carlo</td>
</tr>
<tr>
<td>μ</td>
<td>100.0</td>
</tr>
<tr>
<td>σ</td>
<td>5.6</td>
</tr>
<tr>
<td>cpu [sec]</td>
<td>4.48</td>
</tr>
<tr>
<td>μ</td>
<td>10.0</td>
</tr>
<tr>
<td>σ</td>
<td>2.8</td>
</tr>
<tr>
<td>cpu [sec]</td>
<td>4.10</td>
</tr>
<tr>
<td>μ</td>
<td>20.0</td>
</tr>
<tr>
<td>σ</td>
<td>1.4</td>
</tr>
<tr>
<td>cpu [sec]</td>
<td>3.91</td>
</tr>
<tr>
<td>μ</td>
<td>10.0</td>
</tr>
<tr>
<td>σ</td>
<td>0.7</td>
</tr>
<tr>
<td>cpu [sec]</td>
<td>3.92</td>
</tr>
</tbody>
</table>

![Fig. 5 The distributions of maximum delays](image)

VI. CONCLUSION

In this paper, we proposed a new algorithm for the statistical static timing analysis which can treat correlations of delays of re-convergent paths and correlations of delays of adjacent transistors. Since the algorithm takes the correlations into account, the worst-case time complexity is O(m^2), where n and m are the numbers of vertices and edges of a given acyclic graph representing a CMOS combinatorial circuit, respectively.

In order to see effects of correlation, we applied our algorithm to some benchmark circuits, and found that the difference between the results obtained by our algorithm and those obtained by ignoring correlations is not large enough, compared to the increase of the computational effort. However, since the difference increases, if the distribution becomes large, further research is necessary.

There still remain a few problems on the statistical static timing analysis, among which one of the most important issues is the deletion of false paths[12]. We are tackling this problem by using the proposed technique.

ACKNOWLEDGEMENT

The authors express their gratitude to Prof. T. Nakano, Dept. of Physics, Chuo University, for his valuable discussions about calculations of normal distributions, and Mr. Shoji Nishimoto, Dept. of EECE, Chuo University, for his programming of the proposed algorithm and collection of experimental results.

REFERENCES


Table 2 shows the results of a few circuits in ISCAS 85 benchmark, where n and m are the numbers of vertices and edges of the graph representing the circuits, respectively, and error is the percentage of the difference between "Ours" and "No Correlation." In the experiments, the circuits are laid out by a macrowell layout system[8], and the mean values of delays are extracted from the layout. The standard deviation σ is set as σ=0.15μ, and the edge delays are assumed to be independent again. The cpu-time is mainly depend on the number of numerical computations for normal distribution, and hence it can be reduced by using table look-up technique. Thus, further investigation about correlation and reduction of CPU time are future work.

<table>
<thead>
<tr>
<th>Table 2 RESULTS OF ISCAS85 BENCHMARK DATA</th>
</tr>
</thead>
<tbody>
<tr>
<td>c322</td>
</tr>
<tr>
<td>n = 541</td>
</tr>
<tr>
<td>m = 722</td>
</tr>
<tr>
<td>μ</td>
</tr>
<tr>
<td>6.64 nsec</td>
</tr>
<tr>
<td>6.70 nsec (0.92%)</td>
</tr>
<tr>
<td>σ</td>
</tr>
<tr>
<td>0.29 nsec</td>
</tr>
<tr>
<td>0.28 nsec (-6.23%)</td>
</tr>
<tr>
<td>cpu time</td>
</tr>
<tr>
<td>107.0 sec</td>
</tr>
<tr>
<td>3.7 sec</td>
</tr>
<tr>
<td>c499</td>
</tr>
<tr>
<td>n = 685</td>
</tr>
<tr>
<td>m = 921</td>
</tr>
<tr>
<td>μ</td>
</tr>
<tr>
<td>9.58 nsec</td>
</tr>
<tr>
<td>10.10 nsec (1.18%)</td>
</tr>
<tr>
<td>σ</td>
</tr>
<tr>
<td>0.28 nsec</td>
</tr>
<tr>
<td>0.200 nsec (-28.4%)</td>
</tr>
<tr>
<td>cpu time</td>
</tr>
<tr>
<td>208.4 sec</td>
</tr>
<tr>
<td>6.5 sec</td>
</tr>
<tr>
<td>c880</td>
</tr>
<tr>
<td>n = 1200</td>
</tr>
<tr>
<td>m = 1,570</td>
</tr>
<tr>
<td>μ</td>
</tr>
<tr>
<td>14.64 nsec</td>
</tr>
<tr>
<td>14.73 nsec (0.67%)</td>
</tr>
<tr>
<td>σ</td>
</tr>
<tr>
<td>0.58 nsec</td>
</tr>
<tr>
<td>0.55 nsec (-5.07%)</td>
</tr>
<tr>
<td>cpu time</td>
</tr>
<tr>
<td>371.7 sec</td>
</tr>
<tr>
<td>5.8 sec</td>
</tr>
</tbody>
</table>