Clock Schedule Verification Under Process Variations[*]

Ruiming Chen and Hai Zhou
Electrical and Computer Engineering
Northwestern University
Evanston, IL 60208

Abstract

With aggressive scaling down of feature sizes in VLSI fabrication, process variations have become a critical issue in designs, especially for high-performance ICs. Usually having level-sensitive latches for their speed, high-performance IC designs need to verify the clock schedules. With process variations, the verification needs to compute the probability of correct clocking. Because of complex statistical correlations, traditional iterative approaches are difficult to get accurate results. Instead, a statistical checking of the structural conditions for correct clocking is proposed, where the central problem is to compute the probability of having a positive cycle in a graph with random edge weights. The proposed method only traverses the graph once to avoid the correlations among iterations, and it considers not only data delay variations but also clock skew variations. Experimental results showed that the proposed approach has an error of 0.14% on average in comparisons with the Monte Carlo simulations.

1 Introduction

With shrinking geometries in deep sub-micron technology, process variation becomes a prominent phenomenon in fabrication. These variations introduce random variables into the timing of a fabricated integrated circuit. These delay variations and clock skew variations present a new challenge on timing verification and yield prediction.

There are many recent researches that deal with the timing analysis under process variations [1, 2, 3, 4, 5]. These researches are mainly focused on timing analysis of combinational circuits. However, the validity of a circuit really depends on whether the correct signal values can be latched into the memory elements, and the results of timing analysis are to be used to check clocking conditions.

Level-triggered transparent latches are usually used in high-performance circuits because of their high performance and low power consumption [6]. The complexity introduced by latches is that a signal can pass transparently through a latch during its enabling period and this makes time borrowing across latch boundaries possible. Therefore, timing analysis can no longer be carried only on the separated combinational part since the output time of a latch is now dependent on its input time. Previous researches on clock schedule verification such as Szymanski and Shenoy [7, 8] used iterative approaches to find converged time as the stable behavior and checked it for set-up and hold conditions. Shenoy and Brayton [9] used iterative approaches to check positive cycles in constraint graphs based on structural properties of correct clocking.

In this paper, we formulate the clock schedule verification problem under process variations as computing the probability of correct clocking. Even though the delays are random variables under process variations, they assume fixed values after the fabrication. Therefore, when a signal pass through a gate or a wire multiple times, it must incur the same amount of delay each time. This means that in an iterative approach the delays in different iterations are tightly correlated, which makes the iterative approaches extremely difficult to be used under process variations. Based on this, we propose to use structural clock validity conditions with process variations. The relationship between valid clocking and the condition of no positive cycle in the latest constraint graph or negative cycle in the earliest constraint graph is first established. Then this structural condition with delays and clock skews being random variables is checked through non-iterative graph traversal techniques. One advantage of these techniques is that each element in the circuit is traversed only once and the problem of iterations is thus avoided. The proposed method is implemented and the experimental results show only an error of 0.14% on average compared with the Monte Carlo (MC) simulations.

The rest of the paper is organized as follows. In Section 2, the models of latches, clocking schemes, and the conditions on correct clock schedules are given without consideration of process variations. With the consideration of process variations, Section 3 formulates the statistical clock schedule verification problem and presents the difficulties involved in iterative approaches under process variations. Section 4 establishes the structural conditions for valid clock schedules by extending Szymanski and Shenoy's work [7]. In Section 5, the probability that these structural conditions are held under random element delays and clock skews is computed to give the probability of valid clocking. The experiments on the proposed approach and its comparison with the MC simulation are reported in Section 6. Finally, the conclusion is given in Section 7.

2 Deterministic clock schedule verification

2.1 Models of clock and transparent latches

Clock is the most important mechanism used in a synchronous circuit. A clock is a periodical signal used to regulate other signals in the circuit. With the help of a clock, the design of other signals is relaxed to tolerate glitches or false transitions if they pass the safety checking of the clock. Multiple phases of a clock may be used in a circuit. A *clock scheme* for a circuit is a set of periodical signals $\phi_1, \ldots, \phi_n$ with a common period c. A three-phase clocking scheme is shown in Figure 1. Selecting a period of length c as the *global time reference*, we can denote each phase ϕ_i by its starting and ending time (s_i, e_i) with respect to the reference. Note that it is possible to have $s_i > e_i$ based on the selection of global time reference. For simplicity, we assume that $s_i < e_i$ for all the phases, which can be easily done by shifting the global time reference. We generally order the phases such that $e_i < e_j$ if $i < j$. Also

[*]This work was supported by NSF under CCR-0238484.

note that w_i is used to represent the width of phase i.

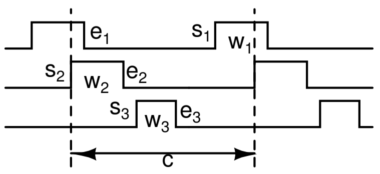

Figure 1: Three phase clocking with period c.

Memory elements are used in a circuit to store its state. In a synchronous circuit, clock signals are usually applied at memory elements to do the safety checking and to filter out unintended transitions. For this purpose, besides the data input and output, a synchronous memory element has a clock input. Only under a certain condition of the clock input does the memory element respond to the data input. Memory elements can be categorized into two groups according to how they respond to the clock input: *flip-flops* store the data when the clock switches; *latches* let the output have the input value when the clock level is high. Because of this, clock schedule verification is easy in a circuit with only flip-flops but is very hard when latches are used. We only focus on the latter problem in the sequel.

2.2 Clock validity conditions

Traditional approaches to clock schedule verification are based on a simple topological delay model of the combinational component. Given the pin-to-pin delay of each gate, the maximal and minimal delays from the output of one latch to the input of another latch can be computed by traversing the topology of the gate connections. Let Δ_{ij} and δ_{ij} represent the maximal and minimal combinational delays from latch i to latch j, respectively. Also let A_i and a_i represent the latest and the earliest signal arrival time on the input of latch i, and D_i and d_i the latest and the earliest signal departure time on the output of latch i, respectively. The famous SMO formulation [10] is

$$\begin{aligned} A_i &= \max_{j\to i}(D_j + \Delta_{ji} - E_{p_jp_i}) &(1)\\ D_i &= \max(A_i, c - w_{p_i}) &(2)\\ a_i &= \min_{j\to i}(d_j + \delta_{ji} - E_{p_jp_i}) &(3)\\ d_i &= \max(a_i, c - w_{p_i}) &(4)\end{aligned}$$

where p_i is the clock phase controlling latch i and E_{ij} is defined as

$$E_{ij} = \begin{cases} e_j - e_i & \text{if } j > i \\ c + e_j - e_i & \text{otherwise}\end{cases}$$

We must also note that used here are local time referring to local periods that end with the phase falling edges.

Ignoring initial hold condition violations, the SMO formulation is too aggressive on earliest time calculation. A *conservative formulation* of earliest time constraints is presented in [11], as

$$\begin{aligned} a_i &= \min_{j\to i}(d_j + \delta_{ji} - E_{p_jp_i}) &(5)\\ d_i &= c - w_{p_i} &(6)\end{aligned}$$

In practice, the aggressive formulation might yield a solution with a shorter period, but [12] showed that there are common situations, such as a latch driven by a qualified clock signal, in which the aggressive formulation is incorrect, and a similar problem arises in circuits which permit the clock to be stopped between adjacent latches to save power. Therefore, in this paper, we choose the conservative formulation.

The solution of the above equations should satisfy the setup and hold time conditions:

$$\begin{aligned} A_i &\leq c - S_i &(7)\\ a_i &\geq H_i &(8)\end{aligned}$$

where S_i and H_i are the setup time and hold time of latch i respectively.

3 Problem formulation

Process variations can be divided into *inter-die* variations and *intra-die* variations. Inter-die variations are variations that exist from one die to the next, while intra-die variations are variations within a single chip. Inter-die variations affect all the devices on the same chip similarly, while intra-die variations affect different devices differently in a same chip. Intra-die variations often exhibit spatial correlations, where devices that are close to each other are more alike than devices that are allocated far apart. So the process variations influence not only the data delay but also the clock network, and there exist correlations among the process variations of all the devices including clock network.

There are many recent researches that deal with the timing analysis under process variations [1, 2, 3, 4, 5]. In [1], a model of spatial correlations was shown. In [2], a block-based static timing analysis technique was presented. In [3], a method based on Bayesian Network is presented to do timing analysis. In [4], multiple arrival time propagation method is shown based on the model in [1]. In [5], a PERT-like circuit graph traversal method based on Principal Component Analysis (PCA) [13] is proposed to handle the complex situation considering path re-convergence and spatial correlations. All these researches only deal with timing analysis on combinational circuits. However, sequential circuits dominate the reality, and the validity of a circuit really depends on whether the correct signal values can be latched into the memory elements, so all these researches eventually should be used to check the clocking conditions, which is the focus of our work. Neves et al. [14] presented a graph-based algorithm to solve the clock skew optimization problem considering process variations, and it only considered the process variations of clock network while neglected the dominating data delay variations. Our work provided a framework for statistical clock schedule verification problems, which considered both data delay variations and clock network variations. The correlations among data delay variations and clock network variations are all considered here.

With consideration of process variations, the clock schedule verification problem can be formulated as

Problem 1 (statistical clock schedule verification) *Given a circuit and a clock schedule under process variations, compute the probability that the clock schedule is correct in the fabricated chips.*

Since there exist process variations in the clock network, the starting and ending time (s_p, e_p) are not the same for the latches controlled by the same clock phase p, so we cannot use a *single* pair of variables (s_p, e_p) to represent this phase. A pair of random variables (s_i, e_i) are used to represent the clock phase controlling latch i. However, since the influence of process variations on s_i and e_i is the same, $w_{p_i} = e_i - s_i$ is constant, and we only need one random variable e_i. We will have a constant w_p to represent the width of each phase p. Thus, we need n correlated random variables (e_i) and p constants (w_p) to represent all the clock phases instead of $2p$ constants (e_p, w_p) in the deterministic clock schedule verification problem, where n is the number of latches, while p is the number of clock phases. Since the delays are influenced by process variations, $\Delta_{ij}, \delta_{ij}, A_i, a_i, D_i$ are random variables. Process variations also influence the setup time and hold time of latches, so S_i and H_i are also random variables. These random variables may be correlated.

Now we use $A^{(s)}$ to denote that variable A is a random variable. We can translate the deterministic conservative formulation of time constraints to *statistical conservative formulation* of time constraints.

$$A_i^{(s)} = \max_{j \to i}(D_j^{(s)} + \Delta_{ji}^{(s)} - E_{ji}^{(s)}) \tag{9}$$

$$D_i^{(s)} = \max(A_i^{(s)}, c - w_{p_i}) \tag{10}$$

$$a_i^{(s)} = \min_{j \to i}(d_j + \delta_{ji}^{(s)} - E_{ji}^{(s)}) \tag{11}$$

$$d_i = c - w_{p_i} \tag{12}$$

where $E_{ji}^{(s)}$ is defined as

$$E_{ji}^{(s)} = \begin{cases} e_i^{(s)} - e_j^{(s)} & \text{if } p_i > p_j \\ c + e_i^{(s)} - e_j^{(s)} & \text{otherwise} \end{cases}$$

Similarly, the solution of equations (9)-(12) should satisfy the set-up and hold time conditions:

$$A_i^{(s)} \leq c - S_i^{(s)} \tag{13}$$

$$a_i^{(s)} \geq H_i^{(s)} \tag{14}$$

Now our object is to calculate the probability that a given clock schedule under process variation satisfies (9)-(14).

In clock schedule verification with process variations, when traversing the same element in different iterations, the element delay is the same. This means that in an iterative approach the delays in different iterations are tightly correlated, which makes the iterative approaches extremely difficult to be used under process variations.

The *max* and *min* operations on random variables are involved in the conservative formulation. Since no accurate analytical formula exists for any of them, the current practice always uses approximation techniques to handle them, and the correlation information cannot be maintained very well. So the inaccuracy of *max* and *min* operations on random variables introduces much inaccuracy into the final results.

4 Structural verification of clock schedule

4.1 Deterministic situation

Studying when the iterative approach to the clock schedule verification will converge and whether the converged solution is unique, Szymanski and Shenoy came up with some structural characterizations. A circuit C is modeled as a finite, edge bi-weighted, directed graph $G = (V, E, \Delta, \delta)$. For each memory element in C there is a vertex $i \in V$. G is called *latch graph.* The following two theorems were given in Shenoy's thesis [8].

Theorem 1 *The equation set composed by (1), (2) has a unique solution if and only if there is no zero Δ-weight cycle in the latch graph.*

Theorem 2 *If the latch graph has no zero δ-weight cycle, then the equation set composed by (3), (4) has a unique solution.*

Since it is very difficult to use iterative approach when the delays become random variables, a theory of structural conditions for a valid clocking will be established.

The *latest equation set* is composed by (1), (2) and (7). The *earliest equation set* is composed by (5),(6) and (8). We first translate the latest equation set into a system of inequalities.

$$\begin{aligned} A_i - D_j &\geq \Delta_{ji} - E_{p_j p_i} \\ D_i - A_i &\geq 0 \\ D_i &\geq c - w_{p_i} \\ -A_i &\geq S_i - c \end{aligned}$$

Based on the correspondence between a system of difference inequalities and the longest path problem on a graph [15], we can construct a *latest constraint graph* corresponding to this inequality set. A vertex will be introduced for each variable, and another vertex O will be introduced as the reference time 0.

Similarly, the earliest equation set can be translated into the following inequalities.

$$\begin{aligned} a_i - d_j &\leq \delta_{ji} - E_{p_j p_i} \\ d_i &\leq c - w_{p_i} \\ -a_i &\leq -H_i \end{aligned}$$

We can construct an *earliest constraint graph* for the earliest equation set.

As an example, consider a circuit given in Figure 2(a). Its latest and earliest constraint graphs are given in Figure 2(b) and (c), where the set-up time and hold time are assumed to be 0 for simplicity.

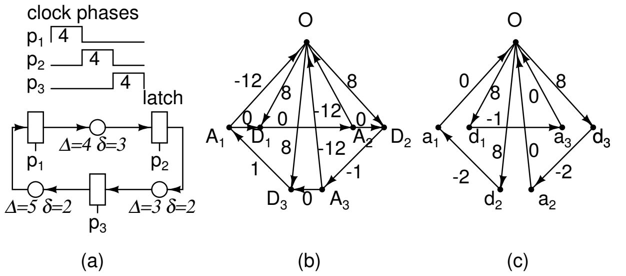

Figure 2: A clock schedule and its latest constraint graph (b) and earliest constraint graph (c).

Based on this construction, the following theorem can be established.

Theorem 3 *A given clock schedule is valid if and only if the latest constraint graph has no positive cycle and the earliest constraint graph has no negative cycle.*

4.2 Statistical situation

In statistical clock schedule verification problem, we can similarly construct the latest constraint graph and the earliest constraint graph based on constraints (9)-(14), but the weights of edges in these two graphs are random variables. The following theorem can be easily proved based on Theorem 3.

Theorem 4 *The probability that a given clock schedule is valid is equal to the probability that the latest constraint graph has no positive cycle and the earliest constraint graph has no negative cycle.*

When delays are constants, the structural condition for a valid clock schedule provides a different view on the clock schedule verification, and characterizes when an iterative approach converges to a unique solution. But a structural condition may not lead directly to a non-iterative approach–the current most efficient way of checking positive cycles is still iterative (e.g. Bellman-Ford [15]). As we know, there are no algorithms to deal with positive cycle detection problem when the edge weights are random variables.

5 Statistical checking of structural conditions

5.1 Statistical static timing analysis

The statistical static timing analysis algorithm introduced in [5] is used in our work to calculate the maximal delay between nodes. It used the PCA technique to transform a set of correlated parameters into an uncorrelated set. It assumed that the delay of gate or interconnect is normally distributed. After doing PCA, a delay can be represented by a linear function of principal components (independent random variables with standard normal distributions):

$$d^{(s)} = d_0 + k_1 \times p_1^{(s)} + ... + k_m \times p_m^{(s)},$$

where d_0 is the mean value, $p_i^{(s)}$ are independent principal components, and k_i are coefficients. The *sum* function and *max* function of normally distributed random variables are provided, which can maintain the correlation information. Especially for the *max* function, Clark's method [16] is used to approximate the result, which assumed that the maximal of two random variable with normal distribution is also normally distributed. Then it used a PERT-like traversal on the circuit graph to calculate the latest arrival time of primary outputs.

5.2 Latest time constraints

Statistical verification of the latest time constraints needs to calculate the probability that all cycles are non-positive in the latest constraint graph.

Let $C = \{c_1, c_2, \ldots, c_n\}$ be the set of cycles, and $|c_i|^{(s)}$ be the delay of cycle c_i. Let $F(|c_1|^{(s)}, |c_2|^{(s)}, \ldots, |c_n|^{(s)})$ be the joint cumulative distribution function (JCDF) of the delays of cycles, then the probability that all cycles are non-positive is $F(0, 0, \ldots, 0)$, which is equal to

$$P_r(|c_1|^{(s)} \leq 0, \ldots, |c_n|^{(s)} \leq 0).$$

However, we know that

$$P_r(|c_1|^{(s)} \leq 0, \ldots, |c_n|^{(s)} \leq 0) = P_r(\max_{j=1}^{n}(|c_j|^{(s)}) \leq 0).$$

Thus, if we can get the distribution of the maximum delay of all cycles, then we can get the probability of the circuit satisfying the latest time constraints. But the number of cycles is often exponential in the number of vertices of a graph, so enumeration method is prohibitive.

However, if we can classify the edges of a directed graph into two disjoint sets such that each cycle is just formed by two simple paths, one composed of edges from one set, then the enumeration of all cycles is not necessary. For example, there are eight cycles in the graph shown in Figure 3, we can classified its edges into two sets:

$$S_1 = \{(a,b)(b,d)(a,c)(c,d)(d,e)(d,f)(e,g)(f,g)\}$$

and

$$S_2 = \{(g,h)(g,i)(h,a)(i,a)\}$$

such that the subgraph by each of them is a directed acyclic graph(DAG). Based on this classification,

$$\max_{j=1}^{8}(|c_j|) = L_{max}(g,a) + L_{max}(a,g)$$

where $L_{max}(x,y)$ is the maximal distance between vertex x and y, which can be efficiently calculated by one pass of traversal.

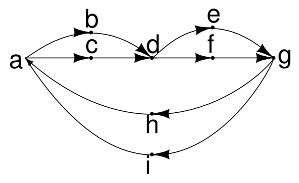

Figure 3: Partition a graph into DAGs

According to this, we designed an efficient algorithm called PCycleI to check the latest time constraints, shown in Figure 4. Here, we performed the depth first search on the latest constraint graph, and classified the edges into two sets: one set contains all the backward edges, and the other edges are in another set.

We assumed the random variables $\Delta_{ji}^{(s)}$, $\delta_{ji}^{(s)}$, $e_i^{(s)}$, $S_i^{(s)}$, $H_i^{(s)}$ in our statistical conservative formulation satisfy normal distribution. Any one of these random variables $r^{(s)}$ can be represented by the linear function of independent variables:

$$r^{(s)} = u + \sum_{i=1}^{m} a_i l_i^{(s)}$$

where u is the mean value of r, $l_i^{(s)}$ is an independent random variable, a_i is a constant, and m is the number of independent random variables. This representation can be always achieved using the models presented in [1] or [5]. Correlated variables can be represented by the linear function of independent random variables with standard normal distribution using PCA. So for simplicity, we assumed each $l_i^{(s)}$ is a standard normally distributed variable.

In step 2 of PCycleI algorithm, we used the statistical timing analysis method proposed in [5] to calculate path delay. Other statistical timing analysis method to calculate path delay can be also used here.

In step 2 of PCycleI algorithm, we deleted all the backward edges, and G becomes a DAG, if each cycle contains only one backward edge, this deletion does not lead to inaccuracy. But if any cycle contains more than one backward edges, the

```
Algorithm PCycleI

Input: constraint graph $G(V,E)$
Output: probability of no positive cycles
      existing in $G$
Notations:
$\mathcal{E}$:   backward edge set
$e$:   single edge
$L_e$:  the delay of edge $e$
$D_e$:  the maximal delay between the end vertex
     and start vertex of edge $e$
$C_e$:  the maximal delay of cycles containing
     the edge $e$

Procedures:
1.  Depth first search graph $G$ to find all
     backward edges, stored in set $\mathcal{E}$;
2.  For each edge $e$ in $\mathcal{E}$ {
      topological sort graph $G$ starting from
        the end vertex of $e$ ignoring backward
        edges in $\mathcal{E}$;
      calculate $D_e$ using statistical timing
        analysis method;
      $C_e = D_e + L_e$;
   }
3.  compute $\max_{e\in\mathcal{E}}(C_e)$;
4.  Probability of no positive cycles is equal
     to $P_r(max_{e\in\mathcal{E}}(C_e) \leq 0)$
```

Figure 4: The PCycleI Algorithm

algorithm may miss it. For example, Figure 5 shows that if the vertex order after topological sort is (a,c,b,d), then edge (b,c) and (d,a) are backward edges, and cycle (a,b,c,d,a) is not included using PCycleI. But if the vertex order is (a,b,c,d), edge (c,b) and (d,a) are backward edges, then cycle (a,b,c,d,a) is included. So different depth first search orders may lead to different cycle sets.

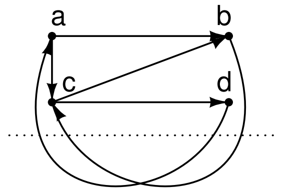

Figure 5: A cycle containing two backward edges

Unfortunately, our study shows that some cycles may still be missed no matter what order is used.

Theorem 5 *The edges in a directed graph cannot* always *be divided into two disjoint sets, such that any simple cycle is formed by two simple paths, one composed of edges from one set.*

Based on these discussions, we design a heuristic method by performing depth first search many times, and the order of choosing edges in each search is *randomly* selected. When the search is performed many times, the overall probability of missing cycles can be smaller. This new version algorithm PCycleII is shown in Figure 6. The complexity of this algorithm is $O(MN(V+E))$, where M is the number of depth first search iterations for each backward edge, N is the number of edges in backward edge set $\mathcal{E}$, V is the number of vertices in G, and E is the number of edges in G.

```
Algorithm PCycleII

New Notations:
$M$:  the number of depth first search iterations.
$C_e^t$:  the maximal length of cycles containing
      the edge $e$ in the $t$th iteration.

Procedure:
1.  Depth first search graph $G$ to find
     all backward edges, stored in set $\mathcal{E}$;
2.  For each edge $e$ in $\mathcal{E}$ {
     $t$ := 1;
     LOOP: While $t$<$M$+1 {
      depth first search graph $G$ starting
        from the end vertex of $e$ to find
        backward edges, stored in set $\mathcal{E}'$;
      topological sort graph $G$ starting
        from the end vertex of $e$ after
        ignoring all the edges in $\mathcal{E}'$;
      clear $\mathcal{E}'$;
      $t$ := $t$+1;
      if the resulting order is ever found
        goto LOOP;
      calculate $D_e$ using statistical timing
        analysis method;
      $C_e^t = D_e + L_e$;
     }
     $C_e := \max_{t=1}^{M}(C_e^t)$;
   }
3.  compute $\max_{e\in\mathcal{E}} C_e$;
4.  Probability of no positive cycles is equal to
     $P_r(max_{e\in\mathcal{E}}(C_e) \leq 0)$
```

Figure 6: The PCycleII Algorithm

5.3 Earliest time constraints

Statistical verification of the earliest time constraints needs to calculate the probability that all the cycles are non-negative in the earliest constraint graph.

Similar to the verification of latest time constraints, let $C = \{c_1, c_2, \ldots, c_n\}$ be the set of cycles, and $|c_i|^{(s)}$ be the length of cycle c_i. Let $F(|c_1|^{(s)}, \ldots, |c_n|^{(s)})$ be the JCDF of the lengths of cycles, then we can calculate the probability that all cycles are non-positive, that is,

$$P_r(|c_1|^{(s)} \geq 0, \ldots, |c_n|^{(s)} \geq 0).$$

However, we know that

$$P_r(|c_1|^{(s)} \geq 0, \ldots, |c_n|^{(s)} \geq 0) = P_r(\min_{j=1}^{n}(|c_j|^{(s)}) \geq 0).$$

This problem is similar to the latest time verification: we only need to calculate the distribution of the minimal length of all the cycles in the earliest constraint graph. However, since

```
Algorithm NCycle

Input:  constraint graph $G(V,E)$
Output:  probability of no negative cycles
      existing in $G$

Procedure:
1.  Split vertex $O$ into $O_1$ and $O_2$, all the
     outgoing edges of $O$ in original $G$ are
     outgoing from $O_1$, and all the incoming
     edges of $O$ in original $G$ are incoming
     edges of $O_2$;
2.  PERT-traversal the new $G$ to calculate the
     shortest distance $D^{(s)}$ from $O_1$ to $O_2$;
3.  Probability of no negative cycles is equal
     to $P_r(D^{(s)} \geq 0)$
```

Figure 7: The NCycle Algorithm

the earliest constraint graph is much simpler, where every cycle in the earliest constraint graph includes the vertex O, the verification can be done optimally. We split vertex O into two vertices O_1 and O_2: all the outgoing edges of O in original graph G are outgoing from O_1, and all the incoming edges of O in original G are incoming edges of O_2. Then G becomes a DAG, so we can traverse on the new G to calculate the shortest length from O_1 to O_2, which is also the shortest length of all cycles in the original G. NCycle, the earliest time constraints verification algorithm, is shown in Figure 7. Since all cycles are considered in NCycle, it can check the earliest time constraints accurately.

5.4 Combined verification

In the above two subsections, we have shown how to calculate the probability that the clock schedule does not violate the latest or earliest time constraints respectively. However, since the earliest and latest time is correlated, multiplying the probabilities of correct conditions in them does not give the right answer. Therefore, we need a combined verification.

In conservative formulation, the following relation is true:

$$\Delta_{ji} \geq \delta_{ji} \; \forall j \to i$$

It is obvious that cycles in the latest constraint graph share some elements with the cycles in the earliest constraint graph, so there exist correlations between the probabilities that the clock schedule does not violate the latest or earliest time constraints. In our algorithms, the vector representation of length maintains these correlations.

In PCycleII, let

$$A^{(s)} = \max_{e \in \mathcal{E}} C_e$$

and in NCycle, let

$$B^{(s)} = \min_{e \in \mathcal{E}} C_e$$

then the probability that latest constraint graph has no positive cycles and earliest constraint graph has no negative cycles is equal to

$$P_r(max(A^{(s)}, -B^{(s)}) \leq 0)$$

When the mean values of random variables ξ and η have large difference, Clark's method [16] selected the variable with larger mean value as the result of $max(\xi, \eta)$. But when the deviation of the variable with smaller mean value is much larger than the deviation of the other one, this approximation is not accurate. Here, $A^{(s)}$ and $-B^{(s)}$ often fall into this situation. Based on the representations of $A^{(s)}$ and $-B^{(s)}$, we discovered that it is better to use MC simulation method to accurately calculate the maximum of $A^{(s)}$ and $-B^{(s)}$.

6 Experimental results

PCycleII, NCycle and the combined verification algorithm NPCycle have been implemented and tested on the ISCAS89 benchmark circuits where the flip-flops are replaced by level-sensitive latches. For simplicity, we only considered the single phase clock, and since PCA can transform the correlated variables to uncorrelated variables, any random variable $r^{(s)}$ in statistical conservative formulation can be represented by a linear function of independent random variables with standard normal distribution:

$$r^{(s)} = r_0 + k_1 \times p_1^{(s)} + ... + k_m \times p_m^{(s)},$$

where r_0 is the mean value, $p_i^{(s)}$ are independent random variables with standard normal distribution, and k_i are coefficients. Then a random number generator is used to generate r_0 and k_i for $i = 1, ..., m$. All the random variables are normally distributed with 10-20% deviation. This test case generator introduces the spatial correlations of process variations of gates, interconnects and the clock network. For example, if for some $1 \leq i \leq m$, $k_i \neq 0$ for one gate, and $k_i \neq 0$ for another gate, there exists spatial correlation between the delay of these two gates. This test case generator can also successfully introduce temporal correlations between clock variations. All experiments were run on a Linux PC with a 2.4G Hz CPU and 2.0 GB memory.

To verify the results of the PCycleII, we used MC simulation as a comparison. In each iteration, MC simulation assigns values to $p_i^{(s)}$ for $i = 1, ..., m$, then calculates the gate delays and clock schedule, then performs Bellman-Ford shortest-path algorithm [15] to detect positive cycles. Here, the assigned values of $p_i^{(s)}$ for $1 \leq i \leq m$ satisfy standard normal distribution. We run 10,000 iterations for each case in the MC simulations. A comparison of results on the latest time constraints is shown in Table 1. The number of depth first search (M) in PCycleII is always less than 5. We can see that the results of PCycleII are very close to the MC results with an average error of 0.17%, which confirms that the probability of missing cycles in PCycleII is very small. PCycleII is also much faster than the MC simulations.

A comparison of the results by the NPCycle with those by the MC simulations is shown in Table 2. For each case, MC simulation performs the Bellman-Ford algorithm to detect positive cycles in the latest time constraint graph, if no positive cycles, then performs Bellman-Ford algorithm to detect negative cycles in the earliest time constraint graph. We can see that the results from NPCycle are very close to the MC simulation results with an average error of 0.14%. The circuit with the longest run time, s35932, was verified in 1342 seconds, while the MC simulation cannot finish within 3 days.

Table 1: Comparison results of PCycleII and Monte Carlo Simulation Method

circuit				PCycleII		Monte Carlo		error%
name	inputs#	latches#	gates#	yield%	runtime(sec.)	yield%	runtime(sec.)	
s27	4	3	10	94.41	0.01	94.39	96	0.02
s298	3	14	119	95.35	0.36	95.41	170	-0.06
s349	9	15	161	97.72	0.93	97.72	498	0.00
s526	3	21	193	95.35	0.22	95.25	287	0.10
s820	18	5	289	92.79	0.24	93.07	3381	-0.30
s1238	14	18	508	96.71	0.21	96.67	3312	0.04
s1488	19	6	653	87.49	4.43	87.86	3651	-0.42
s1494	8	6	647	88.69	6.47	88.32	3050	0.42
s35932	35	1728	16065	96.25	938.28	-	>3 days	-

Table 2: Comparison results of NPCycle and Monte Carlo Simulation Method

circuit	PCycleII	NCycle	NPCycle		Monte Carlo		error%
	yield%	yield%	yield%	runtime (sec.)	yield%	runtime (sec.)	
s27	94.41	97.78	92.15	0.05	92.13	180	0.02
s298	95.35	95.26	93.60	0.48	93.47	273	0.14
s349	97.72	95.35	92.91	2.00	92.71	970	0.22
s526	95.35	98.42	93.59	1.86	93.71	580	-0.13
s820	92.79	97.83	90.68	1.82	90.66	4310	0.02
s1238	96.71	92.92	89.30	1.81	89.47	4416	-0.19
s1488	87.49	96.41	83.88	6.82	83.89	4261	-0.01
s1494	88.69	95.99	84.91	8.23	84.59	4710	0.38
s35932	96.25	97.88	94.08	1342	-	>3 days	-

7 Conclusion

In statistical clock verification problem, because of the complex statistical correlations, traditional iterative approaches are difficult to get accurate results. Instead, a statistical checking of the structural conditions for correct clocking is proposed, where the central problem is to compute the probability of having a positive cycle in a graph with random edge weights. The proposed method only traverses the graph once to avoid the correlations among iterations. Experimental results showed that the proposed approach has an error of 0.14% on average in comparisons with the MC simulations. Our work provided a framework to solve statistical clock verification problem. The accuracy of our algorithms is mostly dependent on the accuracy of statistical timing analysis methods embedded in them, thus it will be improved even more with the improvements of statistical timing analysis.

References

[1] A. Agarwal, D. Blaauw, S. Sundareswaran, V. Zolotov, M. Zhou, K. Gala, and R. Panda. Path-based statistical timing analysis considering inter- and intra-die correlations. In *ACM Intl. Workshop on Timing Issues in the Specification and Synthesis of Digital Systems*, pages 16–21, 2002.

[2] A. Devgan and C. Kashyap. Block-based static timing analysis with uncertainty. In *Proc. Intl. Conf. on Computer-Aided Design*, pages 607–614, San Jose,CA, November 2003.

[3] A. Bhardwaj, S. B. Vrudhula, and D. Blaavw. Tau: Timing analysis under uncertainty. In *Proc. Intl. Conf. on Computer-Aided Design*, pages 615–620, San Jose,CA, November 2003.

[4] A. Agarwal, D. Blaauw, and V. Zolotov. Statistical timing analysis for intra-die process variations with spatial correlations. In *Proc. Intl. Conf. on Computer-Aided Design*, pages 900–907, San Jose,CA, November 2003.

[5] H. Chang and S. S. Sapatnekar. Statistical timing analysis considering spatial correlations using a single pert-like traversal. In *Proc. Intl. Conf. on Computer-Aided Design*, pages 621–625, San Jose,CA, November 2003.

[6] C. Ebeling and B. Lockyear. On the performance of level-clocked circuits. In *Advanced Research in VLSI*, pages 242–356, 1995.

[7] T. G. Szymanski and N. Shenoy. Verifying clock schedules. In *Proc. Intl. Conf. on Computer-Aided Design*, pages 124–131, 1992.

[8] N. V. Shenoy. *Timing Issues in Sequential Circuits.* PhD thesis, UC Berkeley, 1993.

[9] N. V. Shenoy and R. K. Brayton. Graph algorithms for clock schedule optimization. In *Proc. Intl. Conf. on Computer-Aided Design*, pages 132–136, Santa Clara, CA, 1992.

[10] K. A. Sakallah, T. N. Mudge, and O. A. Olukotun. *checkT*$_c$ and *mint*$_c$: Timing verification and optimal clocking of synchronous digital circuits. In *Proc. Intl. Conf. on Computer-Aided Design*, pages 552–555, November 1990.

[11] K. A. Sakallah, T. N. Mudge, and O. A. Olukotun. Analysis and design of latch-controlled synchronous digital circuits. In *Proc. of the Design Automation Conf.*, pages 111–117, 1990.

[12] T. G. Szymanski. Computing optimal clock schedules. In *Proc. of the Design Automation Conf.*, pages 399–404, Anaheim, CA, June 1992.

[13] W. J. Krzanowski. *Principles of Multivariate Analysis.* Oxford University Press, 2000.

[14] J. L. Neves and E. G. Friedman. Optimal Clock Skew Scheduling Tolerant to Process Variations. In *Proc. of the Design Automation Conf.*, pages 623–628, Las Vegas, NV, June 1996.

[15] T. H. Cormen, C. E. Leiserson, and R. H. Rivest. *Introduction to Algorithms.* MIT Press, 1989.

[16] C. E. Clark. The Greatest of a Finite Set of Random Variables. *Operations Research*, 1961.