Propagating Last-Transition-Time Constraints in Gate-Level Timing Analysis

Maroun Kassab¹, Eduard Cerny¹, Sidi Aourid¹, Thomas Krodel²

¹Laboratoire LASSO, Dép. IRO, Université de Montréal
C.P. 6128, Succ. Centre-Ville, Montréal (Québec) H3C 3J7 Canada
e-mail: {kassab,cerny,aourid}@iro.umontreal.ca

²Nortel, PO Box 3511 Station C, Ottawa ON, K1Y 4H7 Canada
e-mail: krodel@nortel.ca

Abstract

Waveform narrowing is an attractive framework for circuit delay verification as it can handle different delay models and component delay correlation efficiently. The method can give false negative results because it relies on local consistency techniques. We present two methods to reduce this pessimism: 1) global timing implications and necessary assignments, and 2) a case analysis procedure that finds a test vector that violates the timing check or proves that no violation is possible. Under floating-mode, global implications eliminate timing check violation without case analysis in the c1908 benchmark, while for a tighter requirement case analysis finds a test vector after only 5 backtracks.

1. Introduction

Verifying delays in gate-level circuits is more difficult as the details of the design are fading away from the view of the designers who must rely on tools that synthesize blocks from descriptions in high level languages and then connect them together, often manually. Due to circuit complexity, traditional manual verification or simulation is not suitable, and considering just the topological delay of the circuit is too conservative and may be costly in useless re-design effort. Unfortunately, computing the delay of a combinational circuit is an NP-Complete problem. Consequently, exact methods are exponential in nature, and more research into heuristics and user interfaces is required to satisfy industrial expectations from static timing verification tools.

The complexity of the problem is caused by the fact that in general not all signal paths in a circuit can propagate transitions (false paths). When the longest paths are false, the actual delay of the circuit is less than this value, i.e., less than the topological delay of the circuit. Hrapcenko [12] presented early an extended discussion on the subject, and proved that minimal circuits may have delays that are less than the topological delays. The circuit shown in Figure 1 illustrates this point [12]. Many techniques have been developed to deal with the problem of false path [1]-[9]. Path oriented timing verifiers suffer from poor performance as they may have to enumerate a very large number of paths, however, it is possible to improve the performance by memorizing inconsistencies between sub-paths [9], and thus reduce the search space. In [5] the authors reduced the problem of determining whether the delay of the circuit is greater than δ to an ATPG problem. In [6] an exact method based on timed Boolean functions and an OBDD representation was formulated, however, it may experience exponential space explosion for certain circuits. Our group has developed a method based on abstract waveform narrowing [2] inspired by constraint logic programming (CLP) using relational interval arithmetic [17]. It can efficiently handle component delay correlation [1] and adapt to different circuit-delay modes (two-vector transition or floating mode) by a simple change in the abstract waveforms applied to the inputs of the circuit.

No exact delay calculation algorithm performs efficiently on all circuits, and the time complexity is exponential on some circuits. A trade-off between tightness of the upper bound on the max. circuit delay and the efficiency of the method is thus necessary. The method of [1, 2] is an excellent candidate for such a trade-off. Exact answers can still be obtained by performing case analysis over the waveforms on certain circuit nets, guided by heuristics. In this paper we show how the addition of global timing implications and a new heuristic for the selection of nets for the case analysis can considerably improve the performance of the algorithm. When combined with delay correlation [1], the resulting algorithm is closer to what is needed in industrial applications. Presently we are integrating our engine with a timing verifier developed at Nortel, with the objective to test the method on industrial circuits.

The paper is organized as follows: Section 2 gives basic definitions and terminology. Section 3 formalizes the problem of whether a circuit has a floating delay greater
3. Overview of Waveform-Narrowing

A timing-check \( \sigma = (\xi, s, \delta) \) is transformed into a constraint system that is consistent if and only if \( s \) has a delay greater than or equal to \( \delta \). A constraint system is composed of a finite set of variables \( \{X_1, X_2, \ldots, X_n\} \) which take values from their respective domains \( D_1, D_2, \ldots, D_n \) and a set of relational constraints \( \{C_1, C_2, \ldots, C_m\} \), each specifying which values of the variables are mutually compatible. The variables and the relational constraints represent the signal values on circuit nets and the gates, respectively. The specific circuit-delay mode and the output timing constraint \( \delta \) introduce further restrictions on the domains.

3.1 Domains

A real digital signal waveform is a mapping \( f: R \rightarrow R \). Abstracted to a binary waveform in discrete time, it becomes a mapping \( f: Z \rightarrow \{0, 1\} \). The space of all binary waveforms is \( BW = \{ f: Z \rightarrow \{0, 1\} \} \).

3.1.1 Abstract Waveforms

Definition 1: An abstract waveform (AW) is a subset of \( BW \) defined as \( w = v_{\min}^{\max} = \{ f \in BW \mid \forall t > \max. f(t) = v \land \exists' t' \in [l_{\min}, l_{\max}]. f(t') \neq v \} \). The abstract waveform space is \( AW = \{ v_{\min}^{\max} \mid v \in \{0, 1\}, l_{\min}, l_{\max} \in Z \} \).

A real waveform \( w \) contains binary waveforms that are stable at value \( v \) after time \( max \) and undergo the last transition at or after time \( l_{\min} \). Obviously, not any subset of \( BW \) can be represented by an abstract waveform. This leads to some approximations when the union operation is defined on abstract waveforms as it is not equivalent to the corresponding set union. References to \( v, l_{\min} \) and \( l_{\max} \) of an abstract waveform \( w \) are denoted \( w.v, w.l_{\min} \) and \( w.l_{\max} \), respectively. The \( w.v \) is the class and \( [w.l_{\min}, w.l_{\max}] \) the last-transition interval of \( w \). If \( w.l_{\min} > w.l_{\max} \) then \( [w.l_{\min}, w.l_{\max}] \) is empty and \( w \) itself is also empty, denoted by \( w = \emptyset \).

Relations and Operations on AW having the same class are defined as follows:

Equality: \( w_1 = w_2 \) iff \( w_2 \max = w_1 \max \land w_2.l_{\min} = w_1.l_{\min} \) or both are empty.

Narrowness: \( w_1 \) is said to be narrower than \( w_2 \), denoted \( w_1 < w_2 \) iff \( (w_1 \max \leq w_2 \max \land w_1.l_{\min} > w_2.l_{\min}) \lor (w_1 \max < w_2 \max \land w_1.l_{\min} \geq w_2.l_{\min}) \). \( w_1 \leq w_2 \) iff \( (w_1 < w_2) \lor (w_1 = w_2) \).

Narrowing an abstract waveform means changing its \( l_{\min} \) and/or \( l_{\max} \) to make it narrower than its previous value. An abstract waveform \( w_1 \) that is narrower than \( w_2 \) contains fewer binary waveforms than \( w_2 \).

Inclusion: \( w_1 \subset w_2 \) iff \( w_1 \leq w_2 \). Intersection: \( w_1 \cap \phi = \phi \land (w_1 \not\subset \phi) \Rightarrow w = w_1 \cap w_2 \) with \( w.v = w_1.v \lor w_2.v \) and \( w.l_{\min} = \max(w_1.l_{\min}, w_2.l_{\min}) \) and \( w.l_{\max} = \min(w_1.l_{\max}, w_2.l_{\max}) \). Union: \( w_1 \cup \phi = w_1 \land (w_1 \not\subset \phi) \Rightarrow w = w_1 \lor w_2 \) with \( w.v = w_1.v \land w_2.v \) and \( w.l_{\min} = \min(w_1.l_{\min}, w_2.l_{\min}) \) and \( w.l_{\max} = \max(w_1.l_{\max}, w_2.l_{\max}) \).

Lemma 1: If \( (w_1 \not\subset \phi) \land (w_2 \not\subset \phi) \) then \( (w_2 \max + 1 \geq w_1.l_{\min}) \land (w_1 \max + 1 \geq w_2.l_{\min}) \Rightarrow (w_1 \cup w_2 = \{ f \in BW \mid (f \in w_1 \lor f \in w_2) \}) \).

The result of performing AW union may in general include binary waveforms that were not originally included in the operands. \( w = w_1 \cup w_2 \) includes, beside \( w_1 \) and \( w_2 \), a minimal subset of BW that makes both \( w_1 \) and \( w_2 \) representable by a single AW, and no \( w' \) narrower than \( w \) contains both \( w_1 \) and \( w_2 \).
3.1.2 Abstract Signals

Definition 2: An abstract signal $S$ is a pair of abstract waveforms $(w, \bar{w})$ where $w, v = 0$ and $\bar{w}, v = 1$.

The components $w$ and $\bar{w}$ of $S$ are denoted $\bar{S}$ and $\hat{S}$, respectively. The space of all abstract signals is $AS = \{w \in AW | w,v = 0\} \times \{w \in AW | w,v = 1\}$; it is the domain of the variables in the constraints in our method.

Relations and Operations on $AS$.

Equality: $S_1 = S_2$ iff $(\bar{S}_1 = \bar{S}_2) \land (\hat{S}_1 = \hat{S}_2)$.

Narrowness: $S_1 \preceq S_2$ iff $(\bar{S}_1 \leq \bar{S}_2) \land (\hat{S}_1 \preceq \hat{S}_2)$.

Inclusion: $S_1 \subset S_2$ if $S_1 \preceq S_2$.

Intersection: $S_1 \cap S_2 = (\bar{S}_1 \land \bar{S}_2, \hat{S}_1 \land \hat{S}_2)$.

Union: $S_1 \cup S_2 = (\bar{S}_1 \lor \bar{S}_2, \hat{S}_1 \lor \hat{S}_2)$.

3.2 Gate Constraints

Gate constraints are derived from the Boolean gate functions. Let $g: BW \times BW \rightarrow BW$ be the timed Boolean function of a 2-input gate $G$. For example, in the case of a 2-input AND with fixed delay $d$, $g(I_1(t), I_2(t)) = I_1(t-d) \cdot I_2(t-d)$. Let $D_i$, $D_j$, and $D_s$ be the domains in $AS$ associated with the variables of the inputs $x_i$, $x_j$, and the output $x_s$, respectively. Let $x_i$, $x_j$, and $x_s$ be subsets of $BW$ such that:

$x_i = \{w \in BW | w \in D_i \lor w \in D_j\}$,
$x_j = \{w \in BW | w \in D_i \lor w \in D_j\}$,
$x_s = \{w \in BW | w \in D_i \lor w \in D_s\}$.

The projections of $D_i$, $D_j$, and $D_s$ to the terminals of $G$ are then as follows:

$x'_i = \{w_1 \in x_i | \exists w_2 \in x_j \cdot \exists w_3 \in x_s \cdot g(w_1, w_2) = w_3\}$,
$x'_j = \{w_1 \in x_i | \exists w_2 \in x_j \cdot \exists w_3 \in x_s \cdot g(w_1, w_2) = w_3\}$,
$x'_s = \{w_1 \in x_i | \exists w_2 \in x_j \cdot \exists w_3 \in x_s \cdot g(w_1, w_2) = w_3\}$.

The constraint relation $C_i(X_i, X_j, X_s)$ derived from $g$, is an operator that changes the values of $D_i$, $D_j$, $D_s$ as to become the narrowest possible to contain $x'_i$, $x'_j$, and $x'_s$, respectively. The projections and the operators on $AS$ are used to define a system of equations over $AS$ which is solved by computing the greatest fixpoint [1, 2].

Example 1: Let $D_i = (0, 33]_{10}, D_j = (0, 75]_{10}, D_s = (0, 125]_{10}$. Applying the constraints of the 2-input AND gate with delay 0 to $D_i$, $D_j$, $D_s$ (at output) yields the following new values:

$D'_i = (0, 100]_{10}, D'_j = (0, 100]_{10}, D'_s = (0, 100]_{10}$.

3.3 Constraint System

Given a timing-check $\sigma = (\xi, s, \delta)$, the construction of the constraint system is straightforward following the circuit description. Let $\xi(\{Gate_1, Gate_2, ..., Gate_m\}, \{Net_1, Net_2, ..., Net_n\})$ be the circuit of $m$ gates and $n$ nets where each gate is connected to a subset of the nets. We build a constraint system composed of $n$ variables $X_1, X_2, ..., X_n$, associated with the $n$ domains $D_1, D_2, ..., D_n$, respectively, and $m$ relational constraints $C_1, C_2, ..., C_m$, where $C_i$ operates on the domains corresponding to the variables of the nets connected to $Gate_i$. The initial values for all the domains of the constraint system of $\sigma$ are $(0, \infty]$ so as to contain any possible BW. For floating-mode delay calculation, we restrict the primary input domains to waveforms that are stable after time 0: $F = (0, 0, 1, 1, 1, 1)$. To verify if the output $s$ has a delay greater than or equal to $\delta$, we restrict the signal domain of $s$ to the waveforms having transitions at or after time $\delta$, i.e., $D_s = (0, 10, 10)$. The constraint system is tightened (solved) by repeatedly applying the local projections of domains as induced by the gate constraints (Section 3.2) until no narrowing of any domain is possible, i.e., the (unique) greatest fixpoint of the system of equations is reached. We implemented this iterative computation efficiently using an event-driven scheduler. It also includes selective state saving needed for backtracking in case analysis.

Definition 3: Given a timing-check $\sigma = (\xi, s, \delta)$ and its corresponding constraint system composed of the variables $X_1, X_2, ..., X_n$, their respective domains $D_1, D_2, ..., D_n$, and the constraints $C_1, C_2, ..., C_m$, a binary waveform $wD_k$ is said to be $\sigma$-compatible, iff it is part of a solution, i.e., iff there is a waveform in each $D_i$, $i \neq k$, such that with $w$ from $D_k$, the constraint system is satisfied. $w$ is said to be $\sigma$-incompatible if it is not $\sigma$-compatible.

Theorem 1: The fixpoint of the evaluation is reached in a finite number of steps.

Theorem 2: If $D_s = (\emptyset, \emptyset)$ then no transition is possible on output $s$ at or after time $\delta$.

Example 2: Consider the timing-check $\sigma = (\xi, s, 61)$ where $\xi$ is the circuit of Figure 1 [12]. Assuming the max. delay of 10 on the output of each gate, top = 70 and the floating-mode delay is 60, because the path $\{n_1, g_2, n_2, g_3, g_4, n_4, g_6, n_6, g_7, n_7, g_8\}$ is false. We now illustrate our method on this example. Let $D_{e_1}, D_{e_2}, D_{e_3}, D_{e_4}, D_{e_5}, D_{e_6}, D_{e_7}, D_{e_8}, D_{e_9}, D_{e_{10}}, D_{e_{11}}, D_{e_{12}}, D_{e_{13}}, D_{e_{14}}, D_{e_{15}}, D_{e_{16}}, D_{e_{17}}, D_{e_{18}}$, be the domains associated with the variables of the corresponding nets. The initial values are:

$D_{e_1} = (0, 10, 10), i.e., \{1, 2, 3, 4, 5, 6, 7\}$: the floating-mode inputs; $D_{e_1} = (0, 10, 10), i.e., \{1, 2, 3, 4, 5, 6, 7\}$: any domain is possible, i.e., the (unique) greatest fixpoint of the system of equations is reached. We implemented this iterative computation efficiently using an event-driven scheduler. It also includes selective state saving needed for backtracking in case analysis.

Waveforms propagation yields: $g_1 \Rightarrow D_{e_1} = (0, 10, 10)$; the maximal delay of $g_1$ is 10; therefore, no transition is possible on $n_1$ after time 10; $g_2 \Rightarrow D_{e_2} = (0, 20, 20)$;
4. Global Timing Implications

The method based on waveform narrowing uses local gate constraints, i.e., the global circuit function is not taken into account. By analyzing the circuit topology, however, we can deduce some of the functionality. Static learning [14] is used to identify some of the class-based implications. It is implemented in a pre-processing stage that determines tables of implications. When a class becomes empty in the domain of a net, learning tables are used to impose class restrictions on other domains. In addition we use the notion of static and dynamic timing dominators to identify global implications related to the existence of transitions at or after a certain time \( l_{min} \), as described next.

The propagation of the last-transition interval is the main mechanism in proving that no violation is possible. In Example 2, only one path was the potential carrier of the violation. There was no ambiguity in deciding which net is its cause when the gate constraints were applied: at gate \( g_8 \) it was able to decide that net \( n_5 \) cannot be the cause of the violation because \((\text{largest of } D_{n_{max}} \text{ and } D_{n_{max}} + 10) < (\text{smallest of } \sigma_{n_{max}} \text{ and } \sigma_{n_{max} + 10})\). This is why \( 1_{\infty} \) was narrowed to \( \sigma \) in \( n_5 \) and the last-transition interval was propagated to \( n_7 \). In more complex circuits, e.g., the carry-skip adder in Figure 2, we may not be able to make such an unambiguous decision. Consider the output \( C_7 \).

Suppose that the topological delay from \( C_2 \) to \( C_7 \) is 750 and that the timing constraint on \( C_7 \) is to require transitions at or after time 750, i.e., \( \sigma = (\xi_1, C_7, 750) \).

The gate constraints are able to propagate the last-transition interval from \( C_7 \) to \( X \) only because no other subpath can be the carrier of the transitions. Figures 2 and 3 illustrate the situation at \( X \). To simplify the presentation, assume that the NAND gate driving \( X \) is delayless. Class 0 of \( M \) (controlling) is removed because it blocks the way on \( N \) and \( P \).

The last-transition interval present at \( X \) propagates to class 0 on \( N \) and \( P \), but not to class 1, because both inputs do not need to have transitions to propagate to \( X \). We thus
cannot decide which net is responsible for transition propagation, i.e., \( l_{\text{min}} \) can be increased, (the domain narrowed) only in the controlling classes of the NAND gate. However, the circuit topology implies that all paths to \( C_7 \) longer than 749 contain \( C_6 \). Therefore, we can restrict waveforms on \( C_6 \) to those having transitions at or after time \((\min(D_X \cdot l_{\text{min}}, D_X \cdot l_{\text{min}}) - \text{max. topological delay from } C_6 \text{ to } X)\).

4.1 Static Timing Dominators

Definition 4: A net \( x \) of \( \xi \) is a \emph{static carrier} of \( \sigma = (\xi, s, d) \) iff \( \exists \) path in \( \xi \) containing \( x \) and \( s \) of length greater than or equal to \( d \).

Definition 5: The sub-circuit composed of the static carriers of \( \sigma = (\xi, s, d) \) and their driving gates of \( \xi \) is the \emph{static carrier circuit} of \( \sigma \). For example, the static carrier circuit of \( \sigma = (\xi_2, C_7, 750) \) where \( \xi_2 \) is in Figure 2, is the sub-circuit of \( \xi_1 \) composed of the shaded nets and their driving gates.

Definition 6: Let \( \Psi \) be the static-carrier circuit of \( \sigma = (\xi, s, d) \). Let \( \Psi' \) be a DAG derived from \( \Psi \) as follows: each net in \( \Psi \) corresponds to a vertex in \( \Psi' \); each gate in \( \Psi \) with \( k \) inputs \( x_1, x_2, \ldots, x_k \) and one output \( x_0 \) corresponds to \( k \) edges, from the vertex corresponding to \( x_0 \) to those corresponding to \( x_i \), \( i=1 \to k \). Add a terminal vertex \( T \) to \( \Psi' \), and an edge to \( T \) from each vertex of an input of \( \Psi \). \( \Psi' \) is a DAG with one source vertex \( S \) (corresponding to \( s \)) and one sink vertex \( T \). The nets of \( \Psi' \) corresponding to the dominators [15] of \( T \) (vertices lying on every path from \( S \) to \( T \)) are the \emph{static timing dominators} of \( \sigma \).

For example, for \( \sigma = (\xi_1, C_7, 750) \) where \( \xi_1 \) is in Figure 3, \( C_7, X, C_6, C_5 \) are static timing dominators of \( \sigma \).

Lemma 3: Let \( d \) be a static timing dominator of \( \sigma = (\xi, s, d) \). Waveforms on \( d \) that are stable at and after time \((d - \text{top}_{d,s})\) are \( s \)-incompatible.

Proof: Follows from Lemma 6.1 in [10].

4.2 Dynamic Timing Dominators

The propagation of the last-transition interval of the output to the static dominators of the circuit represents global necessary assignments. Additional global implications can be determined by analyzing the contents of the abstract signal domains.

Definition 7: Let \( \sigma = (\xi, s, d) \) be a timing-check, and \( C \) its constraint system. Let \( D_x \) be the domain associated with output \( s \). If \( D_x \neq (\phi, \phi) \) then \( s \) is said to be a \emph{0-dynamic carrier} of \( \sigma \). If net \( y \) is a \( k \)-dynamic-carrier and it is the output of gate \( g \) with max. delay \( d_{\text{max}} \), then an input net \( x \) of gate \( g \) is a \( k' \)-dynamic-carrier of \( \sigma \) where \( k' = (k + d_{\text{max}}) \), provided that the domain \( D_x \) satisfies \( D_x \cap (1,2^{\infty}) \neq (\phi, \phi) \). A net \( x \) is a \emph{dynamic carrier} of \( \sigma \) iff \( \exists k \geq 0 \) such that \( x \) is a \( k \)-dynamic carrier of \( \sigma \).

Definition 8: Let \( \Psi \) be the circuit composed of the dynamic carriers of \( \sigma = (\xi, s, d) \) and their driving gates. \( \Psi \) is the \emph{dynamic-carrier circuit} of \( \sigma \), and the topological delay \( \text{top}_{x,s} \) between \( x \) and \( s \) of \( \Psi \) is the \emph{dynamic distance} of \( x \). Intuitively, the dynamic distance of \( x \) is the maximum time a transition at \( x \) takes to reach \( s \), and is equal to the largest integer \( k \) such that \( x \) is a \( k \)-dynamic carrier of \( \sigma \). In fact, the concept of dynamic carriers is formulated by necessary conditions for a net to be the cause of a violation of the timing check, and the domain of a net that is not a dynamic carrier of \( \sigma = (\xi, s, d) \) does not contain transitions that propagate to the last-transition interval of \( s \).

Definition 9: The definition of dynamic timing dominators is obtained by replacing “static” with “dynamic” in Def. 6.

Theorem 3: For a timing-check \( \sigma = (\xi, s, d) \) and a dynamic timing dominator \( d \), let \( k \) be the largest integer such that \( d \) is a \( k \)-dynamic carrier of \( \sigma \). The waveforms on \( d \) that are stable at and after time \((d - k) \) are \( \sigma \)-incompatible.

Proof: Theorem 3 is a direct consequence of the fact that any net \( x \notin \Psi \) cannot be the cause of a timing violation, i.e., \( D_x \) does not contain transitions that propagate to within the last-transition interval on \( s \). This can be proven by contradiction: Suppose that there is a path \( p = (x, g_k, n_k, \ldots, n_1, g_1, s) \) such that the domain of \( x \) contains transitions that propagate along \( p \) to the last-transition interval on \( s \). This implies that the same property is true for all the nets of \( p \). Then \( n_k \) has transitions at or after time \((d - \text{max. delay of } g_k) \) and consequently \( n_k \) is \((d - \text{max. delay of } g_k)\)-dynamic carrier of \( \sigma \). Similarly \( x \) is \((d - \text{length of } p)\)-dynamic carrier of \( \sigma \). \( x \in \Psi \) contradicts the original assumption.

Corollary 1: Let \( d \) be a dynamic dominator of \( \sigma = (\xi, s, d) \) and \( k \) the dynamic distance of \( d \). Narrowing the domain of \( d \) by intersecting it with \((1, 2^{\infty}) - (\text{dynamic-distance of } d) \) maintains all the solutions of the original system.

The proof follows from Theorem 3.

Figure 4 exhibits the timing verification algorithm making use of Corollary 1.

5. Case Analysis

When the net domains remain non-empty after the fix-point calculation we cannot definitely conclude that a violation is possible. We adapted the FAN algorithm [13, 14]
function verify(ξ, s, δ) {
    (CS, EventQueue) = = NoViolation)
    return NoViolation;
    : if for an event do, then i = 0 to k-1
    : intersect domain of s with
    : determine dynamic dominators;
    : return NoViolation;
    : apply constraint on the domains;
    : apply constraint on the domains;
    : schedule all constraints operating on the
    : if the domain of s is empty return NoViolation;
    : else return PossibleViolation;
    : while EventQueue not empty {
    : take a constraint out of EventQueue;
    : if EventQueue is empty return PossibleViolation;
    : if (reach_fixpoint(CS, EventQueue) == NoViolation)
    : return evaluate(CS, EventQueue);
    : function evaluate(CS, EventQueue) {
    : if (reach_fixpoint(CS, EventQueue) == NoViolation)
    : function reach_fixpoint(CS, EventQueue) {
    : for each dominator d do {
    : intersect domain of d with
    : if the domain of d is changed then schedule all constraints
    : schedule all constraints operating on inputs and s on EventQueue;
    : return evaluate(CS, EventQueue);
    };
    : return evaluate(CS, EventQueue);
    : return evaluate(CS, EventQueue);
    : return NoViolation;
    : return NoViolation;
    : return NoViolation;
    : function verify(ξ, s, δ) returns NoViolation, the
    : When the function verify(ξ, s, δ) returns NoViolation, the
    : output s cannot have transitions at or after time δ .

Figure 4  Algorithm of the method

to perform case analysis by waveform splitting on nets, i.e.,
by restricting their domains to one class at a time with the
objective of finding a test vector or proving that no violation
is possible. We used SCOAP [16] controllability to
guide the algorithm.

The main idea is to compute the initial objectives so as to
set those nets which are inputs of gates in the dynamic-
carrier circuit Ψ of σ that are not dynamic carriers to a
non-controlling value regarding the gates they feed in Ψ .
This is justified by the following reasoning: the timing vi-
olation at output s is originating in Ψ , hence we need to
sensitize the paths in Ψ . To favor the longest paths, we es-
hablished the objectives to be a triplet (k, n0(k), n1(k)) as in
[13], but the semantics are different: a path to s of delay n0
(n1) is potentially enabled by setting the net k to 0 (1). The
backtrace procedures are identical to the ones in [13] and
[14], except that at fanouts, n0 (n1) receives the largest in-
coming n0 (n1) instead of their sum. In the context of ATPG,
backtrace is performed a minimal number of times. In our
case such a strategy resulted in poor performance, because
decisions on nets may have profound effect on Ψ , the

source of the violation. The backtrace is initiated each time
the size of the decision stack changes as a result of back-
tracks. Moreover, decisions are performed in 3 phases, fol-
lowing stem correlation pre-processing stage:

**Stem correlation:** We perform partial correlation on all re-
convergent fanout stems that are dynamic carriers. This is
done, for a stem Y, by computing the domain DX of each
variable X of the constraint system as follows: DX = DX0
∪ DX1, where DX0 and DX1 are the values of DX when DY
is intersected with (0]∞, ϕ) and (ϕ, 1]∞, respectively.
This has the effect of removing some of the incompatible
waveforms from the domains of the variables and no de-
cision is yet taken.

**Phase 1:** Let d0, d1, ..., dk be the consecutive dynamic-
dominators of (ξ, s, δ) computed before any decision is
taken. (d0 = s). Let ξdjd, djd+1 be the sub-circuit of ξ com-
posed of the fan-in cone of dj excluding dj+1. We fix the
class value of nets in ξdjd, djd+1, i = 0 to k-1, using the mod-
ified FAN algorithm. Then, we fix the class of nets in the
fan-in cone of dk .

**Phase 2:** We perform decisions on the whole circuit using
the modified FAN algorithm.

**Phase 3:** We perform decisions on s, and then on the pri-
mary inputs after complete backtrace from unjustified nets.
An output of a gate G is unjustified iff its domain is restrict-
et to one class and if we can intersect the domain on each
input with (0]∞, ϕ) or (ϕ, 1]∞,() to get non empty input
domains that are inconsistent with the gate constraint.

6. Experimental Results

Experiments were executed on a Sun SPARCstation
10. The basic constraint system evaluation without global
implications on timing dominators was able to eliminate
timing check violation in the c5315 and c7552 of the NOR-
gate implementations of the ISCAS’85 benchmarks [11]
with delays of 10 on the outputs of all gates. The use of timing
dominators eliminated timing violations from c1908 and
c3540. Stem correlation eliminated timing-check viola-
tion from c2670 and c6288. The case analysis found test
vectors for all circuits except c6288. Table 1 contains the
results. Note that the value of δ for which a test vector is
found represents the exact floating-mode delay of the cir-
cuit when the constraint system is inconsistent for (δ + 1)
on all outputs. The columns of Table 1 contain, from left to
right, the following information: 1) the circuit name, 2) the
circuit, 3) the timing dominators, 4) the result of the first evaluation of the con-
straint system before the use of timing dominators, 5) the result after the use of timing dominators, 6) the result after stem correlation, 7) the number of backtracks in the case analysis, 8) the result of case analysis, and 9) the total CPU
time.
Table 1 Results for ISCAS’85 circuits

<table>
<thead>
<tr>
<th>CIRCUIT</th>
<th>CIRCUIT MAX. TOP.</th>
<th>δ</th>
<th>BEFORE G.I.T.D.</th>
<th>AFTER G.I.T.D.</th>
<th>AFTER STEM C.</th>
<th>C.A. #BTRCK</th>
<th>C.A. RESULT</th>
<th>CPU (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>c17</td>
<td>50</td>
<td>340</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>0</td>
<td>V</td>
<td>0.05</td>
</tr>
<tr>
<td>c432</td>
<td>190</td>
<td>190F</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>1</td>
<td>V</td>
<td>18.82</td>
</tr>
<tr>
<td>c499</td>
<td>250</td>
<td>250F</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>5</td>
<td>V</td>
<td>7.10</td>
</tr>
<tr>
<td>c880</td>
<td>200</td>
<td>200F</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>0</td>
<td>V</td>
<td>3.06</td>
</tr>
<tr>
<td>c1355</td>
<td>270</td>
<td>270F</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>1</td>
<td>V</td>
<td>8.17</td>
</tr>
<tr>
<td>c1908</td>
<td>340</td>
<td>311</td>
<td>P</td>
<td>N</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>0.90</td>
</tr>
<tr>
<td>c1908</td>
<td>340</td>
<td>310F</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>5</td>
<td>V</td>
<td>11.58</td>
</tr>
<tr>
<td>c2670</td>
<td>250</td>
<td>241</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>0</td>
<td>N</td>
<td>3.67</td>
</tr>
<tr>
<td>c2670</td>
<td>250</td>
<td>240F</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>7</td>
<td>V</td>
<td>17.07</td>
</tr>
<tr>
<td>c3540</td>
<td>410</td>
<td>391</td>
<td>P</td>
<td>N</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>5.12</td>
</tr>
<tr>
<td>c3540</td>
<td>410</td>
<td>390F</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>3</td>
<td>V</td>
<td>56.00</td>
</tr>
<tr>
<td>c5315</td>
<td>460</td>
<td>451</td>
<td>N</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>1.56</td>
</tr>
<tr>
<td>c5315</td>
<td>460</td>
<td>450F</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>16</td>
<td>V</td>
<td>21.97</td>
</tr>
<tr>
<td>c6288</td>
<td>1230</td>
<td>1221</td>
<td>P</td>
<td>N</td>
<td>0</td>
<td>N</td>
<td>56.36</td>
<td></td>
</tr>
<tr>
<td>c6288</td>
<td>1230</td>
<td>1220F</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>A</td>
<td>A</td>
<td></td>
</tr>
<tr>
<td>c7552</td>
<td>380</td>
<td>371F</td>
<td>N</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>0.72</td>
<td></td>
</tr>
<tr>
<td>c7552</td>
<td>380</td>
<td>370F</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>1</td>
<td>V</td>
<td>8.34</td>
</tr>
</tbody>
</table>

Legend: (G.I.T.D. stands for Global Implications on Timing Dominators.)
P: Possible violation of the timing-check constraint. N: No violation of the timing-check constraint on any circuit output is possible. V: Test vector found. - (dash): Procedure not used (was not necessary). A: Abandoned due to excessive number of backtracks. F: Value represents exact floating-mode delay. (;): Value represents upper bound on the maximal floating-mode delay.

Not included in Table 1 is the timing check performed on a 16 bit carry-skip adder, partly shown in Figure 2. The adder has a topological delay of 2000 and a floating-mode delay of 1000. This was determined in 25 seconds of CPU time after a total of 1636 backtracks. For δ=1001 the case analysis proved that the constraint system is inconsistent on all outputs, and for δ=1000 a test vector was found.

The use of timing dominators was very effective on the traditionally difficult c1908 circuit. It proved that output 57_912 (topological delay of 340) cannot have a delay greater than 200 in 0.76 seconds. This particular case has 5 timing dominators, and no narrowing was performed on 3 of them by the original method.

7. Conclusions

We showed in this paper how global timing implications enhance the performance of the timing verification method based on waveform narrowing. Further refinements were achieved by enforcing correlation on convergent stems and new heuristics included in the case analysis. We are developing constraint models for complex gates (MUX, etc...), and we process SDR backannotation to test our method on industrial circuits.

References