On Convergence of Switching Windows Computation in Presence of Crosstalk Noise

Pinhong Chen*+ Yuji Kukimoto* Chin-Chi Teng* Kurt Keutzer+

+Dept. of EECS, U.C. Berkeley
Berkeley, CA 94720
pinhong@eecs.berkeley.edu

*Silicon Perspective, A Cadence Company
3211 Scott Blvd, Suite 100
Santa Clara, CA 95054

ABSTRACT

Detecting overlapping of switching windows between coupling nets is a major static technique to accurately locate crosstalk noise. However, due to the mutual dependency between switching windows, the computation requires iterations to converge. In this paper, we discuss the issues and provide answers to the important questions involved in convergence and numerical properties, including the effect of coupling models, multiple convergence points, convergence rate, computational complexity, non-monotonicity, continuity and the effectiveness of bounds. Numerical fixed point computation results are used to explain these properties. Our contribution here builds a theoretical foundation for static crosstalk noise analysis.

Categories and Subject Descriptors

B.7.2 [**Design Aids**]: Verification—*Timing analysis, noise analysis*

General Terms

Theory,Algorithms,Verification

1. INTRODUCTION

In very deep submicron technologies, as the feature size decreases, the aspect ratio (height over width) of metal lines goes up. This results in larger lateral coupling capacitances, which dominate the total capacitance. Crosstalk noise is thus easily induced from the neighboring nets and causes the major timing variation. These effects can no longer be ignored and the analysis has been addressed extensively in state-of-the-art chip designs [1, 2].

Crosstalk noise can cause timing variation in two ways. If two nets switch at about the same time in the same direction, the delay is sped up, and for the opposite directions, the delay is slowed down. If the adjacent nets are quiet, there is no crosstalk noise. Therefore, it is important to identify the **switching window**, a possible timing duration in which a net or a timing node can possibly switch or make transitions. Identifying overlapping between switching windows can reduce pessimism involved in crosstalk noise analysis, because no crosstalk noise is induced between two nets when there is no overlap of the switching windows.

With crosstalk noise, switching windows are considered mutually dependent in static timing analysis(**STA**), and the computation cannot be completed in a single traversal of nets in general [3, 4, 5, 6, 7, 8, 9]. Iterations are therefore needed to resolve the dependency until they converge. The following questions thus arise:

- How many iterations are needed at most? What is the computational complexity? How fast does the process converge?
- Does it always converge? What coupling or overlapping models lead to divergence?
- Is there a unique convergence point independent of an initial condition?
- If a gate delay is reduced, is a circuit's longest path delay considering crosstalk noise reduced as well?
- If a gate delay is increased continuously, will the crosstalk noise also be increased continuously?

In [9], the authors suggest the use of a lattice theory to prove convergence of switching windows computation and show there exist multiple convergence points depending on the initial condition. However, the coupling model they used is very primitive and is not accurate due to its discrete nature. We will tackle this problem with a different point of view from a *numerical* fixed point computation perspective[10]. Besides, we will examine the impact of discrete and continuous coupling models on convergence and numerical properties of switching windows computation. Moreover, the questions listed above will be studied in this paper.

The rest of this paper is organized as follows. In Section 2. we introduce some definitions used in the paper, and give the upper and the lower bounds. Section 3 addresses fixed point computation applied to switching windows computation. The underlying models used in switching windows computation are examined in Section 4. Proof of convergence and efficiency issues are discussed in Section 5.

2. BACKGROUND

A **victim** is a net that suffers from noise effects, and an **aggressor** is a net that contributes noise. The roles may change depending on the context. A quantity is said to be **noisy** if the crosstalk noise effects have been included in it. On the contrary, a quantity is said to be **noiseless** or **nominal**, if the crosstalk noise effects are not included in it. For example, a **noisy delay** is the nominal delay plus the extra delay induced by crosstalk effects. Similarly, a **noisy switching window** is the nominal switching window including the extra noisy path delay induced by crosstalk effects.

ISPD'02, April 7-10, 2002, San Diego, California, USA.

A **coupling edge** exists from the victim to the aggressor in the STA timing graph if there is a coupling capacitance linked between them. A coupling edge is said to be **active** if the delta delay induced by this edge has been included in its victim's noisy switching window. In the process of switching windows computation, a coupling edge may change its **state** from inactive to active or vice versa due to overlapping of noisy switching windows.

Without loss of generality, we assume a single delay value on interconnect for all of the fanouts of each net in the following discussion. Let x_i and y_j be the latest arrival time of net i and the earliest arrival time of net j, respectively. Let $\theta_{ij} : R \to [0,1]$ be

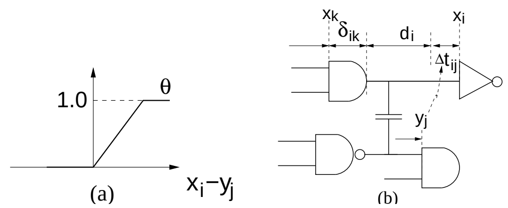

Figure 1: (a) θ function to represent the overlapping. (b) Variables to represent delays in a coupling subcircuit.

a switching window overlapping function of time difference. Figure 1(a) is an example of function $\theta_{ij}(x_i - y_j)$. If $x_i - y_j < 0$, the switching windows don't overlap, resulting in no coupling noise. As $x_i - y_j$ becomes greater than zero, the coupling noise starts to appear and eventually it saturates at a normalized value 1.0. The delta delay induced from aggressor net j to victim net i is thus written as $\theta_{ij}(x_i - y_j)\Delta t_{ij}^{max}$, where $\Delta t_{ij}^{max} > 0$ is the maximum delta delay induced from aggressor net j to victim net i.

An index set denotes a collection of net indices, representing a subset of all nets. Let d_i be the interconnect delay of of net i, A_i be an index set of aggressor nets of net i, D_i be an index set of fanin nets of net i's driving gate, and δ_{ik} be the gate delay from a fanin net k to net i. Figure 1(b) relates these variables to a circuit diagram. x_i and y_i are written as

$$x_i = \sum_{j \in A_i} \theta_{ij}(x_i - y_j)\Delta t_{ij}^{max} + d_i + \max_{k \in D_i}(x_k + \delta_{ik}), \text{ and} \tag{1}$$

$$y_i = -\sum_{j \in A_i} \theta_{ij}(x_j - y_i)\Delta t_{ij}^{max} + d_i + \min_{k \in D_i}(y_k + \delta_{ik}), \tag{2}$$

respectively.

2.1 Simple Upper and Lower Bounds for Switching Windows

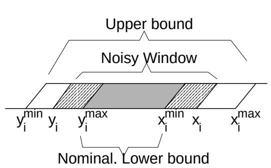

Figure 2: Bounds for switching window.

What are the possible latest and earliest arrival times considering crosstalk noise? Clearly, when all the crosstalk noises are active and induce the maximum extra delay to increase the latest arrival time and reduce the earliest arrival time, this is the largest switching window, i.e. **the upper bound**, that can be achieved with crosstalk noise. The upper bounds are written as

$$\begin{aligned} x_i^{\max} &= \sum_{j \in A_i} \Delta t_{ij}^{\max} + d_i + \max_{k \in D_i}(x_k^{\max} + \delta_{ik}) \\ &\geq \sum_{j \in A_i} \theta_{ij}(x_i - y_j)\Delta t_{ij}^{max} + d_i + \max_{k \in D_i}(x_k + \delta_{ik}), \text{ and} \quad (3) \\ y_i^{\min} &= -\sum_{j \in A_i} \Delta t_{ij}^{\max} + d_i + \min_{k \in D_i}(y_k^{\min} + \delta_{ik}) \\ &\leq -\sum_{j \in A_i} \theta_{ij}(x_j - y_i)\Delta t_{ij}^{max} + d_i + \min_{k \in D_i}(y_k + \delta_{ik}). \quad (4) \end{aligned}$$

On the contrary, when all crosstalk noises are excluded, the nominal switching window gives **the lower bound**, i.e. the smallest possible switching window. The lower bounds are written as

$$\begin{aligned} x_i^{\min} &= d_i + \max_{k \in D_i}(x_k^{\min} + \delta_{ik}) \\ &\leq \sum_{j \in A_i} \theta_{ij}(x_i - y_j)\Delta t_{ij}^{max} + d_i + \max_{k \in D_i}(x_k + \delta_{ik}), \text{ and} \\ y_i^{\max} &= d_i + \min_{k \in D_i}(y_k^{\max} + \delta_{ik}) \\ &\geq -\sum_{j \in A_i} \theta_{ij}(x_j - y_i)\Delta t_{ij}^{max} + d_i + \min_{k \in D_i}(y_k + \delta_{ik}). \end{aligned}$$

The relationship between the bounds and the final switching window are illustrated in Figure 2.

3. FIXED POINT COMPUTATION

Fixed point computation provides us a convenient vehicle to explore the underlying properties of how the computation precedes. In this section, we propose the formulation and point out some important properties.

3.1 Formulation

Let $\mathbf{x} \in \mathbf{R}^{2N}$ be a switching window configuration, where N is the number of nets or timing nodes in switching windows computation. For N nets, we need $2N$ variables to represent the latest arrival times, x_i's, and the earliest arrival times, y_i's, respectively (If rise and fall switching windows are considered separately, $4N$ variables are needed). Let $\mathbf{f} : \mathbf{R}^{2N} \to \mathbf{R}^{2N}$ be a mapping or transformation from $\mathbf{x}$ to a new switching window configuration $\mathbf{x}'$ considering the crosstalk noise based on the switching windows' overlapping calculated according to $\mathbf{x}$.

The objective of switching windows computation thus can be formulated as finding a **fixed point**, $\mathbf{x}^*$, such that $\mathbf{x}^* = \mathbf{f}(\mathbf{x}^*)$[9]. Specifically, iteration equations are written as

$$x_i^{(n+1)} = \sum_{j \in A_i} \theta_{ij}(x_i^{(n)} - y_j^{(n)})\Delta t_{ij}^{max} + d_i + \max_{k \in D_i}(x_k^{(n)} + \delta_{ik}), \text{ and}$$

$$y_i^{(n+1)} = -\sum_{j \in A_i} \theta_{ij}(y_i^{(n)} - x_j^{(n)})\Delta t_{ij}^{max} + d_i + \min_{k \in D_i}(y_k^{(n)} + \delta_{ik}).$$

With an initial guess, $\mathbf{x}^{(0)}$, we can perform **fixed point iterations** as

$$\mathbf{x}^{(1)} = \mathbf{f}(\mathbf{x}^{(0)}),\ \mathbf{x}^{(2)} = \mathbf{f}(\mathbf{x}^{(1)}),\ \dots\ \mathbf{x}^{(n)} = \mathbf{f}(\mathbf{x}^{(n-1)})\ \dots \tag{5}$$

until it converges. This process is usually referred to as **fixed point computation**[10].

3.2 Fixed Point Iteration for Switching Windows Computation

Let D be a closed and bounded domain in $\mathbf{R}^{2N}$, and let $\mathbf{f}(\mathbf{x}) \in D$ for all $\mathbf{x} \in D$. For any two points $\mathbf{x_1}, \mathbf{x_2} \in D$, if there exists a constant

L, such that

$$||\mathbf{f}(\mathbf{x_1}) - \mathbf{f}(\mathbf{x_2})|| < L||\mathbf{x_1} - \mathbf{x_2}||,$$

where $||\cdot||$ denotes a norm for the vector space D, $\mathbf{f}(\mathbf{x})$ is called a *Lipschitz function*, and L is called a *Lipschitz constant*. Using fixed point computation[10], if $0 \le L < 1$, the fixed point iteration converges and guarantees a unique convergence point(fixed point), given any initial $\mathbf{x}_0 \in D$. This is a sufficient condition for existence, convergence and uniqueness[10].

In practice, for one-dimensional case, L is roughly estimated as

$$L = \sup \frac{d f_i(x_i)}{dx_i} = \sup \sum_{j \in A_i} \theta'_{ij}(x_i - y_j)\Delta t_{ij}^{max} \quad \text{and} \quad \Delta t_{ij}^{max} = \Delta V_{max} \frac{\tau_i}{\tau_j},$$

where sup is an upper bound(maximum value) function, ΔV_{max} is the maximum noise induced by an aggressor and τ_i and τ_j are the slew time of a victim and an aggressor, respectively. In general, L is not bounded by 1.0. It depends on the underlying models, which are discussed in Section 4.

For example, consider a one-dimensional iteration function $f(x)$ in Figure 3(a) as

$$f(x) = \sum_{j \in A_i} \theta_{ij}(x - y_j)\Delta t_{ij}^{max} + d_i + \max_{k \in D_i}(x_k + \delta_{ik}), \qquad (6)$$

where $x^{\min}$ and $x^{\max}$ are the lower and the upper bounds described

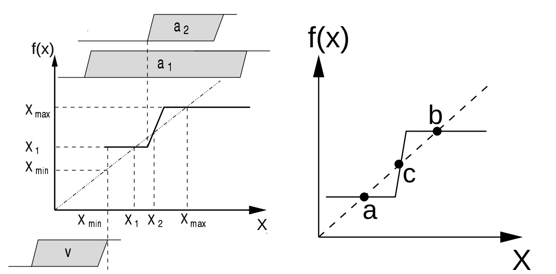

Figure 3: (a) Arrival time function. (b) Multiple convergence points.

in Section 2.1, and victim net v and two aggressors a_1 and a_2 together create function $f(x)$, where x_1, x_2 and $x_{\max}$ are the fixed points. Repeated substitution procedure which replaces the argument with its output value can be used to converge $x^{(n)}$ sequence. For one-dimensional case and a continuous function, a sufficient condition for convergence is given as $|f'(x)| < 1$. For $\mathbf{f}(\mathbf{x})$, the sufficient condition[10] is $||\mathbf{J}|| < 1$, where $\mathbf{J}$ is the Jacobian matrix of $\mathbf{f}(\mathbf{x})$.

3.3 Multiple Convergence Points and Unstable Fixed Point

Since L is not in general bounded by 1.0, it is easy to produce multiple convergence points in switching windows computation process. The actual convergence point depends on the initial switching windows [9].

For fixed point computation, a unique convergence point requires a *Lipschitz constant L* less than 1.0. In switching windows computation, L is not bounded for discrete models introduced in Section 4, and therefore there can be multiple fixed points. Even for a continuous model, it is possible for L to be greater than 1.0. For example, in Figure 3(b), points a, b and c are all fixed points, and the initial condition determines which fixed point the iteration converges to. Notice that point c is **unstable** since $f'(x)|_{x=c} > 1$. A small perturbation can drive convergence toward points a or b. Therefore, unstable fixed points cannot be obtained through fixed point computation.

3.4 Tightening Bounds

A conventional convergence scenario uses infinite switching windows for the initial condition to include all the noise effects in the beginning and shrinks the switching windows in the subsequent iterations[6, 8]. [9] shows starting from the infinite switching windows, the process can converge to a looser upper bound of switching windows.

Using Eq. (1) and (2), we can prove this monotonicity by induction as follows.

THEOREM 1. *If in the initial two steps, $x_i^{(0)} \ge x_i^{(1)}$ and $y_i^{(0)} \le y_i^{(1)}$ for all $i = 1..N$, $x_i^{(n)}$ forms a monotonically non-increasing sequence, and $y_i^{(n)}$ forms a monotonically non-decreasing sequence.*

Proof: The base case is clearly

$$x_i^{(0)} \ge x_i^{(1)} \quad \text{and} \quad y_i^{(0)} \le y_i^{(1)}, \quad \text{for} \quad i = 1..N.$$

By induction, we have

$$x_i^{(n-1)} \ge x_i^{(n)} \quad \text{and} \quad y_j^{(n-1)} \le y_j^{(n)} \quad \text{for} \;\; n \ge 1, \; i,j = 1..N,$$

so

$$x_i^{(n)} - y_j^{(n)} \le x_i^{(n-1)} - y_j^{(n-1)}.$$

θ_{ij} is a non-decreasing function, so we have

$$\theta_{ij}(x_i^{(n)} - y_j^{(n)}) \le \theta_{ij}(x_i^{(n-1)} - y_j^{(n-1)}), \;\; \text{and}$$

$$\Delta t_{ij}^{max} > 0 \quad \text{and} \quad \max_{k \in D_i}(x_k^{(n)} + \delta_{ik}) \le \max_{k \in D_i}(x_k^{(n-1)} + \delta_{ik}).$$

Therefore,

$$\begin{aligned} x_i^{(n+1)} &= \sum_{j \in A_i} \theta_{ij}(x_i^{(n)} - y_j^{(n)})\Delta t_{ij}^{max} + d_i + \max_{k \in D_i}(x_k^{(n)} + \delta_{ik}) \\ &\le \sum_{j \in A_i} \theta_{ij}(x_i^{(n-1)} - y_j^{(n-1)})\Delta t_{ij}^{max} + d_i + \max_{k \in D_i}(x_k^{(n-1)} + \delta_{ik}) \\ &= x_i^{(n)}. \end{aligned}$$

Besides, $y_i^{(n)}$ sequence can be proved symmetrically. □

LEMMA 1. *If the initial configuration starts from the maximum switching windows, $x_i^{(n)}$ forms a monotonically non-increasing sequence, and $y_i^{(n)}$ forms a monotonically non-decreasing sequence.*

This result actually shows the switching window shrinks starting from the maximum switching window. That is to say, the upper bound is reduced in each iteration. Thus, the accuracy of noisy switching windows' bound computed depends on how much run time is affordable. The results are still valid upper bounds of switching windows even before convergence. Similarly, using the minimum switching windows as the initial condition, the lower bound increases as the convergence process precedes. We have the following two lemmas.

LEMMA 2. *If in the initial two steps, $x_i^{(0)} \le x_i^{(1)}$ and $y_i^{(0)} \ge y_i^{(1)}$ for any $i = 1..N$, $x_i^{(n)}$ forms a monotonically non-decreasing sequence, and $y_i^{(n)}$ forms a monotonically non-increasing sequence.*

LEMMA 3. *If the initial configuration starts from the noiseless switching windows, $x_i^{(n)}$ forms a monotonically non-decreasing sequence, and $y_i^{(n)}$ forms a monotonically non-increasing sequence.*

This lemma actually provides a method to obtain tightest switching windows.

These results create a monotonic transformation during fixed point iteration suggested by [9].

4. COUPLING MODELS

In this section, we will consider the underlying models for calculating noise. Discrete models are easier and faster to calculate and in general give a bound of crosstalk noise. However, the error bound can be far off from the correct noise bound computed using a continuous model.

Crosstalk noise induces a voltage glitch on a victim and causes timing variation. The amount of the delta delay in timing calculation can be determined by aligning the noise peak with the victim waveform so that the superimposed waveform peak reaches the switching threshold (usually 50% of power rail voltage)[11]. If the victim's waveform is simplified as an ideal ramp with a slew time τ_v, the maximum delta delay $\Delta t^{\max}$ can be written as $\Delta t^{\max} = \tau_v \frac{\Delta V_{max}}{V_{DD}}$.

4.1 Noise Calculation Model

Traditional STA often ignores all the coupling effects and replaces any coupling capacitance with a grounded one. Conventionally, to calculate the coupling delay on each interconnect, a **discrete coupling factor model** uses 1X grounded capacitance when the neighboring net is quiet, 2X for the opposite direction switching and 0 for the same direction switching[12, 9]. Determining which factor to use depends on the overlapping of switching windows. However, it has been shown the coupling noise can result in as much as 3X capacitance effect when calculating the coupling delay[13, 14]. Besides, a discrete coupling factor model has discontinuity between the boundaries when a coupling factor is changed to another factor. On the contrary, a **continuous coupling factor model** can be used to avoid the discontinuity on the boundaries and increase accuracy[13, 14].

More accurate models have been proposed using superposition to calculate the total crosstalk noise without using any coupling factor nor decoupling the coupling capacitance, such as [6, 1, 5].

In [7], the authors propose a model where instead of direct substitution of x_i in Eq. (1) to evaluate the crosstalk noise, they decompose an arrival time into a component that is contributed from the driving gate as

$$\bar{x}_i = d_i + \max_{k\in D_i}(x_k + \delta_{ik}) \quad \text{and} \quad \bar{y}_i = d_i + \min_{k\in D_i}(y_k + \delta_{ik}),$$

and, based on these arrival times $\bar{x}_i$ and $\bar{y}_i$, evaluate the noisy arrival times as

$$x_i = \sum_{j\in A_i} \theta_{ij}(\bar{x}_i - \bar{y}_j)\Delta t_{ij}^{max} + \bar{x}_i \quad \text{and} \quad y_i = -\sum_{j\in A_i} \theta_{ij}(\bar{x}_j - \bar{y}_i)\Delta t_{ij}^{max} + \bar{y}_i.$$

It avoids pessimistically to take the noisy arrival time into account for switching windows' overlapping. This model still can be calculated using fixed point computation.

4.2 Switching Windows Overlapping Model

For multiple aggressors, the worst case noise should be calculated based on the switching window constraints. That is to say, due to the path delays, the aggressors' switching window may not be able to align arbitrarily to create the maximum noise.

In [15], the authors have proposed a mixed integer programming technique to find the worst possible noise. This model assumes "sharpness" of the noise peaks and it is considered a **discrete overlapping model**, which only allows either a complete noise contribution or zero noise. That is to say, $\theta(\cdot)$ is a step function in Eq. (1) and Eq. (2). Figure 4(a) shows a noise function in such kind of models. On the contrary, a **continuous overlapping model** allows the noise contribution to be fractional according to the noise glitch waveform on the victim net and the overlapping range. Figure 1 is an example of $\theta(\cdot)$ representing a continuous model. Many efficient methods have been proposed to find the maximum noise or the maximum delta delay using this kind of models. For example, [5, 7] have proposed envelope waveform methods, and [16, 17] formulate it as a weighted channel density problem and gives an algorithm with $O(MlogM)$ complexity, where M is the number of aggressors.

Note that if an infinite slope is assumed on the boundary of switching windows, a continuous overlapping model becomes a discrete overlapping model.

4.3 Discontinuity in Discrete Models

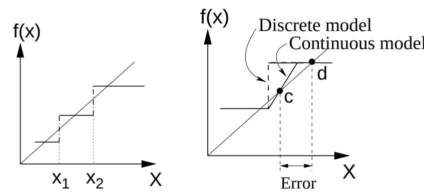

Figure 4: Discontinuity in $f(x)$ or noise when using a discrete model.

All of the discrete models mentioned above suffer from a drawback of discontinuity on the boundary when a discrete factor is changed to another or the overlapping condition is changed. The noise is discontinuous when increasing or decreasing gate delays. Figure 4(a) shows an example of discontinuity at x_1 and x_2 when using a discrete overlapping model.

If the discrete model is designed carefully, it can be an upper bound of crosstalk noise, which is considered very useful in STA. An example in Figure 4(b) shows that the discrete model converges at point d while the continuous model converges at point c, which is bounded by point d.

4.4 Error Bound between Discrete and Continuous Models

After convergence of switching windows, suppose no error was incurred in the previous stage delay for a single pair of coupling nets. The error incurred due to the use of a discrete model can be as large as $\Delta t^{\max}$, the maximum delta delay between a victim and an aggressor net pair, given the same initial configuration of switching windows. The error can be written as

$$\varepsilon_{ij} = (\theta_{ij}^{con}(x_i - y_j) - \theta_{ij}^{dis}(x_i - y_j))\Delta t_{ij}^{\max} \leq \Delta t_{ij}^{\max},$$

where $\theta_{ij}^{con}(\cdot)$ is the continuous overlapping function and $\theta_{ij}^{dis}(\cdot)$ is the discrete overlapping function. Figure 4(b) shows an example in which the discrete model converges to point d and the continuous model converges to point c. If every aggressor just aligns exactly at the same time, the error bound can be as large as the sum of the maximum delta delays:

$$\varepsilon_i^{net} \leq \sum_{j\in A_i} \theta_{ij}(x_i - y_j)\Delta t_{ij}^{\max} \leq \sum_{j\in A_i} \Delta t_{ij}^{\max}.$$

Moreover, the error may propagate forward along a timing path and accumulate to include all the maximum delta delays induced from all the aggressors. Therefore, the error bound for a noisy longest path delay is equal to difference between the upper bound and the lower bound of ending net's switching window of the path:

$$\varepsilon(x_i) \le x_i^{\max} - x_i^{\min} = \sum_{j \in A_i} \Delta t_{ij}^{\max} + \max_{k \in D_i}(x_k^{\max} + \delta_{ik}) - \max_{k \in D_i}(x_k^{\min} + \delta_{ik}),$$

and similarly, the error bound for a noisy shortest path delay is equal to:

$$\varepsilon(y_i) \le y_i^{\max} - y_i^{\min} = \sum_{j \in A_i} \Delta t_{ij}^{\max} + \min_{k \in D_i}(y_k^{\max} + \delta_{ik}) - \min_{k \in D_i}(y_k^{\min} + \delta_{ik}),$$

4.5 Non-Monotone Property

When decreasing a gate delay, the noisy longest delay of a circuit may increase. For example, consider a case in Figure 5(a), where if the arrival time of switching window *a* is reduced, net *a* may start to attack net *v* due to the overlapping of switching windows resulting in delay increase in net *v*. In terms of Eq. (1), as y_j decreases,

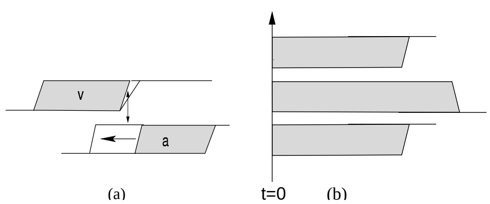

Figure 5: (a) Reducing a gate delay resulting in a longer path delay. (b) Floating mode delay model.

$\theta_{ij}(x_i - y_j)$ increases, resulting in increase of x_i on the left hand side. Using just one operational condition to determine x_i and y_j may result in an optimistic evaluation of crosstalk noise.

Using a floating delay model[18](e.g. see Figure 5(b)), which assumes zero for any earliest arrival time, i.e. $y_j = 0$ in Eq. (1), is still impossible to solve this problem. If we set $y_j = 0$, Eq. (1) becomes

$$\begin{aligned} x_i &= \sum_{j \in A_i} \theta_{ij}(x_i) \Delta t_{ij}^{max} + d_i + \max_{k \in D_i}(x_k + \delta_{ik}) \\ &= \sum_{j \in A_i} \Delta t_{ij}^{max} + d_i + \max_{k \in D_i}(x_k + \delta_{ik}) = x_i^{\max}, \end{aligned}$$

which does not provide any useful information about switching windows.

5. CONVERGENCE OF SWITCHING WINDOWS COMPUTATION

In this section, we will argue the convergence of switching windows computation.

5.1 Proof of Convergence

THEOREM 2. *The iteration in Eq. (5) converges to a fixed point* $\mathbf{x}^* = \mathbf{f}(\mathbf{x}^*)$, *given the initial switching window configuration as* $\mathbf{x}^{(0)} = \mathbf{x}^{\min}$ *or* $\mathbf{x}^{(0)} = \mathbf{x}^{\max}$

Proof: Convergence of the iteration can be proved by the following facts[6, 9]:

1. x_i's and y_i's have an upper and a lower bounds as shown in Section 2.1,
2. if starting from $\mathbf{x}^{(0)} = \mathbf{x}^{\min}$, by Lemma 3, $x_i^{(n)}$ forms a non-decreasing sequence and $y_i^{(n)}$ forms a non-increasing sequence. The iteration converges, since the sequences are bounded.
3. Using Lemma 1, we can prove similarly for the initial condition starting from $\mathbf{x}^{(0)} = \mathbf{x}^{\max}$. □

Some switching window overlapping model may not have monotonicity for $\mathbf{f}(\mathbf{x})$ with respect to x_i. For example, when an aggressor's latest arrival time is much less than a victim's, the switching window of the victim is not affected. This effect can be captured by adding an extra term $-\theta_{ij}(x_i - x_j)$ to Eq. (1) as:

$$x_i = \sum_{j \in A_i} (\theta_{ij}(x_i - y_j) - \theta_{ij}(x_i - x_j)) \Delta t_{ij}^{max} + d_i + \max_{k \in D_i}(x_k + \delta_{ik}) \quad (7)$$

However, it can be shown in Figure 6(a) Eq. (7) has a decreasing portion, leading to oscillation among points *a*, *b*, *c* and *d*, and the iteration cannot converge. To remedy the decreasing portion,

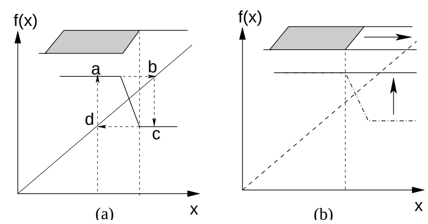

Figure 6: (a) A decreasing portion resulting in non-convergence. (b) Extending the aggressor's switching window to infinity.

the aggressor's switching window needs to be extended to infinity such as shown in Figure 6(b) when calculating crosstalk noise between coupling nets, and hence Eq. (1) is mostly used. Without the decreasing portion, Eq. (1) is monotonically increasing for x_i and x_k and monotonically decreasing for y_j. Theorem 2 equivalently shows there exists at least one fixed point in *D*. Now, let's see if there exists some looping structure in Eq. (1), i.e. oscillation among points in the iteration:

$$\mathbf{x}^{(n+k)} = \mathbf{f}^{(n)}(\mathbf{x}^{(k)}) = \mathbf{x}^{(k)} \text{ and } n > 1.$$

Without loss of generality, we assume x_i affects y_j and y_j affects x_k, and x_k affects x_i in Eq. (1). Any increase in x_i leads to decrease in y_j and subsequently increase in x_k, which in turns increases x_i. This excludes the possibility to have oscillation in the iteration. That is to say, the iteration converges given any initial conditions with $x_i^{\min} \le x_i^{(0)} \le x_i^{\max}$ and $y_i^{\min} \le y_i^{(0)} \le y_i^{\max}$.

5.2 Computational Complexity

For a discrete overlapping model, we have at most $O(NM)$ coupling edges, where N is the number of nets and M is the maximum number of aggressor nets for any victim net. In each pass of calculation of $\mathbf{f}(\mathbf{x})$, at least one coupling edge's state is finalized such that the edge either contributes the noise to its victim or not for all the subsequent passes. If no coupling edge changes its state, there is no update to the noisy switching windows, so the iteration converges. For each pass, we need to examine $O(N)$ nets against their $O(M)$ aggressors to identify switching windows' overlapping. Thus, the total complexity is $O(N^2M^2)$. In [12], an algorithm with complexity $O(N^2M)$ is suggested without counting the cost of checking against $O(M)$ aggressors for detecting overlaps on each net. In practice, this algorithm converges quite fast within 3 to 5 iterations.

If an event-driven style of computation is used[7], the total complexity is still the same since there are $O(N)$ nets in an STA timing graph to update, and for each update to a switching window we need to check against $O(M)$ aggressors to detect overlapping and trigger new update events. There are $O(NM)$ edges, and we thus need $O(N^2M^2)$ operations for an event-driven style of computation. They conclude with an efficient algorithm similar to the approach of using Gauss-Seidel method.

5.3 Convergence Rate

For fixed point computation under a continuous overlapping model, the convergence rate needs to be considered. For some local regions, there exists a local Lipschitz constant L such that $L < 1$, and the fixed point iteration converges. The convergence rate is determined by this local Lipschitz constant L. Provided $\mathbf{x}^{(n)}$ is close enough to $\mathbf{x}^*$, the error is bounded by[10]:

$$||\mathbf{x}^{(n)} - \mathbf{x}^*|| \leq \frac{L^n}{1-L}||\mathbf{x}^{(1)} - \mathbf{x}^{(0)}||, \text{ or}$$

$$||\mathbf{x}^{(n)} - \mathbf{x}^*|| \leq \frac{L}{1-L}||\mathbf{x}^{(n)} - \mathbf{x}^{(n-1)}||.$$

Since L may be greater than 1.0 in some local areas, it may have some local divergent sequences as shown in Figure 7 for one-dimensional

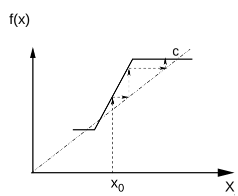

Figure 7: Local divergence.

case. However, the ending game of convergence is still dominated by the convergent L value which is closest to 1.0.

5.4 Speed-up of Convergence

Quite a few speed-up methods for convergence have been proposed[8, 7, 12, 9]. Most of the techniques are similar to Gauss-Jordan, Gauss-Seidel, or update(event)-driven calculations. A Gauss-Seidel style calculation for fixed point computation uses any updated information as soon as it is available. For example, the iteration function can be modified as

$$x_i^{(n+1)} = \sum_{j \in A_i} \theta_{ij}(x_i^{(n)} - y_j^{(n+1)})\Delta t_{ij}^{max} + d_i + \max_{k \in D_i}(x_k^{(n+1)} + \delta_{ik}).$$

Usually, $y_j^{(n)}$ is used if $y_j^{(n+1)}$ is not available at the moment when calculating $x^{(n+1)}$. Techniques are thus focused on how to maximize the use of $y_j^{(n+1)}$ instead of $y_j^{(n)}$, and the use of $x_k^{(n+1)}$ instead of $x_k^{(n)}$.

Another speed-up method is to replace fixed point computation by a Newton-Raphson iteration, which has a quadratic convergence rate. However, computing each Jacobian matrix element needs $O(N)$ operations, and the total complexity to build the Jacobian matrix is $O(N^3)$. Inverting the Jacobian matrix may need as many as $O(N^3)$ operations besides the original computation cost of $\mathbf{f}(\mathbf{x})$. Moreover, singularity problem should be handled with care. Also, this method does not guarantee to find the tightest noisy switching windows.

6. CONCLUSION

Switching windows computation can be well-controlled by careful selection of the underlying models. In this paper, we show, formulate and prove the various numerical properties by a numerical fixed point computation perspective and these could serve as a theoretical foundation for switching windows computation.

7. REFERENCES

[1] R. Levy, *et al.* “Clarinet: A noise analysis tool for deep submicron design”. In *Proc. of Design Automation Conference*, pp. 233–238, 2000.

[2] R. Kumar. “Interconnect and Noise Immunity Design for Pentium 4 Processor”. In *Intel Reports downloaded from http://developer.intel.com*.

[3] S. S. Sapatnekar. “On the Chicken-and-Egg Problem of Determining the Effect of Crosstalk on Delay in Integrated Circuits”. In *Proc. IEEE of 8th Topical Meeting on Electrical Performance of Eletronic Package*, pp. 245–248, 1999.

[4] B. Franzini, *et al.* “Crosstalk Aware Static Timing Analysis:a Two Step Approach”. In *Prof. of ISQED*, pp. 499–503, Mar. 2000.

[5] P. F. Tehrani, *et al.* “Deep Sub-Micron Static Timing Analysis in Presence of Crosstalk”. In *Proc. of ISQED*, pp. 505–512, Mar. 2000.

[6] R. Arunachalam, *et al.* “TACO: timing analysis with coupling”. In *Proc. of DAC*, pp. 266–269, 2000.

[7] P. Chen, D. A. Kirkpatrick, and K. Keutzer. “Switching Window Computation for Static Timing Analysis in Presence of Crosstalk Noise’. In *Proc. of ICCAD*, pp. 331–337, 2000.

[8] T. Xiao, *et al.* “Efficient static timing analysis in presence of crosstalk”. In *Proc. of IEEE International ASIC/SOC Conference*, pp. 335–339, 2000.

[9] H. Zhou, *et al.* “Timing Analysis with Crosstalk as Fixpoints on Complete Lattice”. In *Proc. of DAC*, pp. 714–719, 2001.

[10] G. Engeln-Mülleges and F. Uhlig. *“Numerical Algorithms with C”*. Springer-Verlag, 1996.

[11] P. D. Gross, *et al.* “Determination of Worst-Case Aggressor Alignment for Delay Calculation”. In *Proc. of ICCAD*, pp. 212–219, Nov. 1998.

[12] S. S. Sapatnekar. “A timing model incorporating the effect of crosstalk on delay and its application to optimal channel routing”. *IEEE Trans. on Computer-Aided Design*, Vol.19:pp.550–559, May 2000.

[13] A. B. Kahng, *et al.* “On Switch Factor Based Analysis of Coupled RC Interconnects”. In *Proc. of DAC*, pp. 79–84, 2000.

[14] P. Chen, *et al.* “Miller Factor for Gate-Level Coupling Delay Calculation’. In *Proc. of ICCAD*, pp. 68–74, 2000.

[15] K. L. Shepard, *et al.* “Global Harmony: Coupled Noise Analysis for Full-Chip RC Interconnect Networks”. In *Proc. of ICCAD*, pp. 139–146, 1997.

[16] T. Xiao and M. Marek-Sadowska. “Worst delay estimation in crosstalk aware static timing analysis”. In *Proc. of IEEE ICCD*, pp. 115–120, 2000.

[17] L. H. Chen and M. Marek-Sadowska. “Aggressor Alignment for Worst-Case Crosstalk Noise”. *IEEE Trans. on Computer-Aided Design*, Vol.20:pp.612–621, May 2001.

[18] S. Devadas, *et al.* “Computation of floating mode delay in combinational networks: Theory and algorithms”. *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, Vol.12:pp.1913–1923, Dec. 1993.