Handling Inverted Temperature Dependence in Static Timing Analysis

ALI DASDAN and IVAN HOM
Synopsys, Inc.

In digital circuit design, it is typically assumed that cell delay increases with decreasing voltage and increasing temperature. This assumption is the basis of the cornering approach with cell libraries in static timing analysis (STA). However, this assumption breaks down at low supply voltages because cell delay can decrease with increasing temperature. This phenomenon is caused by a competition between mobility and threshold voltage to dominate cell delay. We refer to this phenomenon as the *inverted temperature dependence* (ITD). Due to ITD, it becomes very difficult to analytically determine the temperatures that maximize or minimize the delay of a cell or a path. As such, ITD has profound consequences for STA: (1) ITD essentially invalidates the approach of defining corners by independently varying voltage and temperature; (2) ITD makes it more difficult to find short paths, leading to difficulties in detecting hold time violations; and (3) the effect of ITD will worsen as supply voltages decrease and threshold voltage variations increase. This article analyzes the consequences of ITD in STA and proposes a proper handling of ITD in an industrial sign-off STA tool. To the best of our knowledge, this article is the first such work.

Categories and Subject Descriptors: J.6 [**Computer-Aided Engineering**]: *Computer-aided design (CAD)*; B.5.2 [**Register-Transfer-Level Implementation**]: Design Aids—*Automatic synthesis*; *optimization*; B.6.3 [**Logic Design**]: Design Aids—*Automatic synthesis*; *optimization*

General Terms: Algorithms, Performance

Additional Key Words and Phrases: Static timing analysis, temperature dependence, timing corners, voltage dependence

1. INTRODUCTION

Static timing analysis (STA) is a primary component of both implementation and verification flows for any digital design. Roughly speaking, an STA tool takes in a circuit and its timing constraints, computes its performance bounds, compares them against its timing constraints, and outputs a pass or a fail.

The performance of a circuit depends on the delays of the paths in the circuit. The delay of a path in turn depends on the delays of its cells and nets. Net delay

This work was done while both authors were employed at Synopsys, Inc.
Authors' current addresses: A. Dasdan: Yahoo Inc., 701 First Ave., Sunnyvale, CA 94089; email: ali_dasdan@yahoo.com; I. Hom: Transmeta Corp., 3990 Freedom Circle, Santa Clara, CA 95054; email: ihom@transmeta.com.

is out of the scope of this article. Cell (arc) delay is a nonlinear function of many parameters. To guarantee a pass under any parameter settings in the high-dimensional parameter space, cell delay is usually measured (or characterized) at the "corners" of the parameter space. Then, these measurements are provided to the STA tool in cell (timing) libraries. These corners are expected to bound the performance.

The actual number of corners can be large, sometimes more than a hundred; however, their number has traditionally been restricted to three (which we will use to simplify our exposition without loss of generality): a fast corner, a slow corner, and a typical corner. These corners are determined by three parameters: process (P), voltage (V), and temperature (T). These parameters and corners are referred to as the *PVT parameters* and *corners*. Together with these three parameters, two other parameters, input transition time or slew (S_{inp}) and output load capacitance (C_{out}), are needed to compute the delay of any cell.

1.1 ITD and Its Effects

Traditionally, in terms of voltage and temperature, the slow corner of a design is verified at low voltage and high temperature, and the fast corner is verified at high voltage and high temperature. This is due to *the "normal" pattern* that cell delay increases with decreasing voltage and increasing temperature. However, when voltage is low, *a reversal of this normal pattern* occurs in that at low voltages, cell delay can increase with decreasing temperature. We will refer to this phenomenon as the *inverted temperature dependence (ITD)*. As a result of this inversion, there is one or more crossover voltages at which the inversion occurs and delay is independent of temperature.

The normal and inverted temperature dependences can be seen from the top and bottom plots in Figure 1, respectively. Each plot shows how the delay of a NAND cell from a 90-nm library changes as a function of voltage and temperature. The same 90-nm library is used for all our results. For voltage dependence, these plots agree: delay increases as voltage decreases. For temperature dependence, these plots disagree due to ITD: the top plot indicates that delay increases with increasing temperature independent of voltage whereas the bottom plot indicates that delay increases with increasing or decreasing temperature depending on whether voltage is larger than or smaller than the crossover voltage of nearly 1.12 V.

The effect of ITD on path delays is more serious, and can be seen from Figure 2. If the delay of a path follows the normal temperature dependence, it is expected to increase monotonically with increasing temperature. For example, the delay curve for $V_{dd} = 1.32$ follows this behavior. However, due to ITD, the delay of a path may *increase or decrease* with increasing temperature. For example, the delay curve for $V_{dd} = 1.08$ decreases with increasing temperature whereas the delay curve for $V_{dd} = 1.20$ first decreases and then increases with increasing temperature. The latter delay curve especially shows that it is nontrivial to determine the temperature that minimizes a path delay. This, of course, leads to difficulties in hold time analysis due to its reliance on such paths.

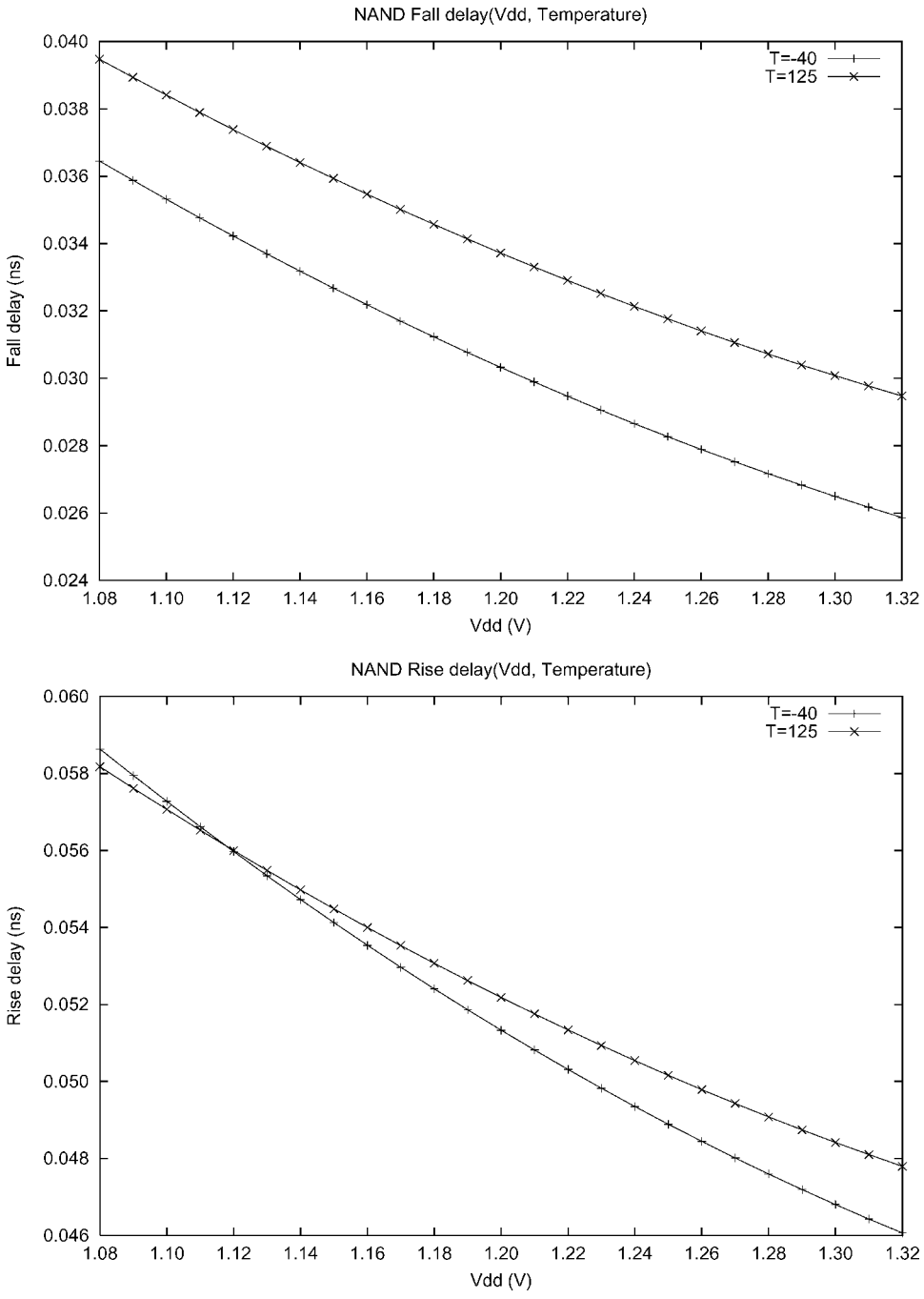

Fig. 1. The fall (top) and rise (bottom) delay of a NAND cell as a function of voltage and temperature at $S_{inp} = 0.015$ ns, $C_{out} = 0.003$ pF. The top (bottom) plot shows the normal (inverted) temperature dependence. The ITD crossover voltage is around 1.12 V.

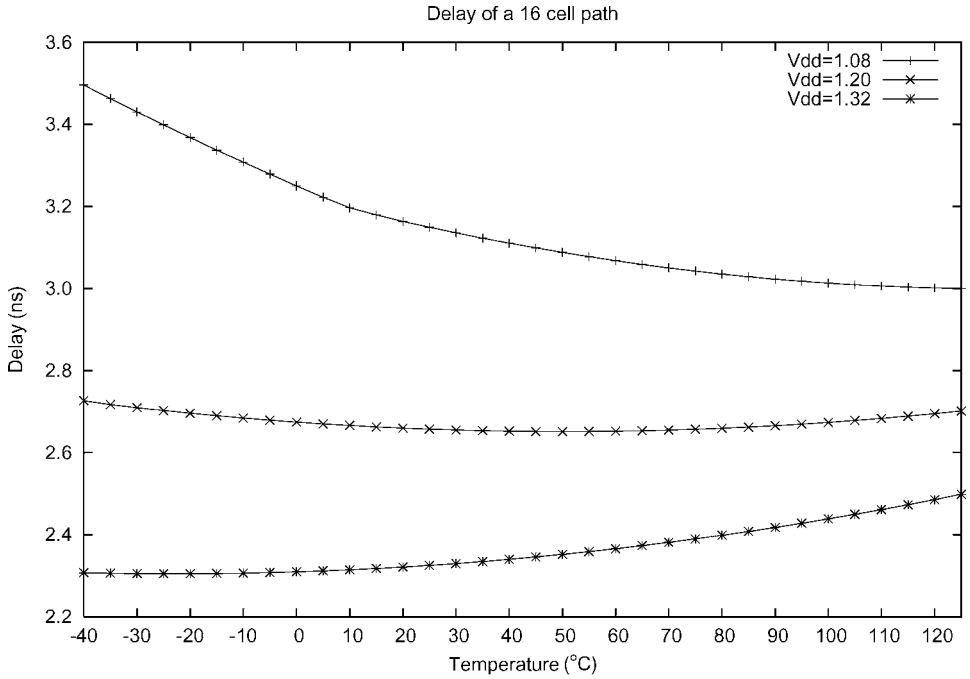

Fig. 2. The delay of a path of 16 cells as a function of voltage and temperature. The plot shows how the path delay changes as temperature changes.

1.2 Contributions and Related Work

Since ITD breaks the assumption of how delay changes with temperature, it has important consequences for static analysis (STA as well as other static analyses like power). The chief consequence is that ITD makes it very difficult to determine the temperatures that maximize or minimize cell delays as well as path delays. This also points out the chief contribution of this article: we show how to bound cell and path delays correctly for STA.

ITD is not new as it has been well known in the analog design literature for quite some decades [Filanovsky and Allam 2001; Kanda et al. 2001]. For example, the analog design literature refers to the crossover voltage as the zero temperature coefficient (ZTC) voltage. Surprisingly, it was relatively unknown outside of that literature: it was first mentioned in Park et al. [1995], and there have been a few publications since then such as Bellaouar et al. [1998], Daga et al. [1998], Gerousis [2003], Long et al. [2004], and Kanda et al. [2001] (under the names "positive/negative temperature dependence" and "temperature inversion"). Almost all of these articles (and those in the analog design literature) seem to have focused mostly on how to take advantage of the (existence of) crossover voltages, for example, to build voltage and current references that are stable over a wider range of temperature.

We are not aware of any previous work both in the analog design literature and among the articles mentioned here that address the STA aspects of ITD. The only articles that address the STA aspects of ITD to some limited extent

have been Daga et al. [1998] and Gerousis [2003]. The first article showed that derating delays independently using voltage and temperature results in error. This is an important finding and also one of our conclusions. Using the alpha-power law [Sakurai and Newton 1990], it instead proposed a new derating factor expression that depends on both voltage and temperature simultaneously. This expression cannot be applied in cases where delay is not modeled using the alpha-power law, which is the case in industrial sign-off STA. Our solution does not use this law either. The second article drew attention to the challenges for the 90- and 50-nm technologies. It mentioned ITD as one of the challenges and pointed out that any proper handling of ITD must deal with the fact that it depends on such factors as cell type, arc type, input slew, output load, logic style, and process corners without offering a solution.

We want to emphasize that ITD is a *real problem* (e.g., see Gerousis [2003] about its importance) and the users of the current industrial STA tools insist on its solution. Since none of these tools solve this problem yet, the users usually resort to a temporary solution in which each existing PV corner is combined with two or more temperatures to create new PVT corners. For example, without ITD, a slow PVT corner occurs at the maximum temperature; however, with ITD, a slow PVT corner can occur at any temperature. Consequently, the users usually create two new slow PVT corners: one at the maximum temperature and another at the minimum temperature. In Section 4.1, we discuss this temporary solution in more detail and show that is not guaranteed to correctly bound the cell and path delays.

We believe that this article is the first work that correctly addresses the impact and a proper handling of ITD in STA. We observe that it is very difficult to analytically determine the temperature at which a cell or path delay is maximized or minimized. As such, we show that a proper approach to handle ITD is the bounding approach in which the delay of each cell is tightly bounded from above and below. This approach also guarantees that any path delays will be bounded correctly.

1.3 Article Organization

The rest of the article is organized as follows. In Section 2, we provide an intuitive explanation of why and how ITD happens. This is followed in Section 3 by a short review of delay modeling in cell libraries. Next, in Section 4, we discuss two approaches to handling ITD in STA; an existing approach, which does not always work, and our approach. We then look at the algorithmic issues raised by ITD in Section 5. In Section 6, we show the implications of our proposal for delay modeling in cell libraries. We discuss our experimental setup and experimental results in Section 7, and conclude in Section 8.

2. INVERTED TEMPERATURE DEPENDENCE (ITD)

For the sake of simplicity, let us assume a simple delay model like the alpha-power law [Sakurai and Newton 1990]. Note that we use this assumption only to intuitively explain ITD in this section; our solution does not assume or need to use this law at all.

By the alpha-power law, the delay of a cell is expressed as

$$Delay \propto \frac{C_{out} V_{dd}}{I_d}, \tag{1}$$

where C_{out} is the output load capacitance, V_{dd} is the supply voltage, and I_d is the drain current. According to this equation, the delay is inversely proportional to the drain current.

The drain current is in turn expressed as

$$I_d \propto \mu(T)(V_{dd} - V_{th}(T))^{\alpha}, \tag{2}$$

where μ is the mobility, V_{th} is the threshold voltage, and α is a small positive constant.

The temperature dependence of I_d is due to those of the mobility μ and the threshold voltage V_{th}. These parameters depend on temperature as

$$\mu(T) = \mu(300)\left(\frac{300}{T}\right)^{m} \tag{3}$$

and

$$V_{th}(T) = V_{th}(300) - \kappa(T - 300), \tag{4}$$

where 300 is the room temperature in Kelvin, m and κ are small positive constants.

Now, Equations (3) and (4) show that both the mobility and the threshold voltage decrease with increasing temperature. However, as Equation (2) shows, they affect the drain current in opposite ways: lower mobility decreases the drain current, but lower threshold voltage increases the drain current. The final drain current is determined by which trend dominates the drain current at a given voltage and temperature pair.

At high voltages, the mobility determines the drain current but at low voltages, the threshold voltage determines the drain current. In other words, at high voltages, higher temperature increases the delay but at low voltages, higher temperature decreases the delay. Thus, the delay increases or decreases with increasing temperature depending on the magnitude of V_{dd}. This is the inverted temperature dependence (ITD) phenomenon. The voltage where temperature dependence reverses (or inverts) is called the *crossover voltage*, the *zero-temperature coefficient (ZTC) voltage*, or the *inversion voltage*.

The exact values of the small constants α, m, and κ are largely immaterial to our discussion above. They are also out of our scope. However, this is not to say that their values are unimportant for ITD. Interested readers can look at the lengthy discussion in Filanovsky and Allam [2001] on the importance of their values on observing ITD at all or observing one or more crossover voltages.

Figures 3, 4, and 5 show how ITD affects the delay characterization of a NAND cell. In particular, Figure 3 shows the delay surface as a function of voltage and temperature. Note that the negative gradients of the surfaces in Figure 4 are a result of ITD. Also note that the crossovers (or intersecting curves) in the temperature plots in Figure 5 are a result of ITD.

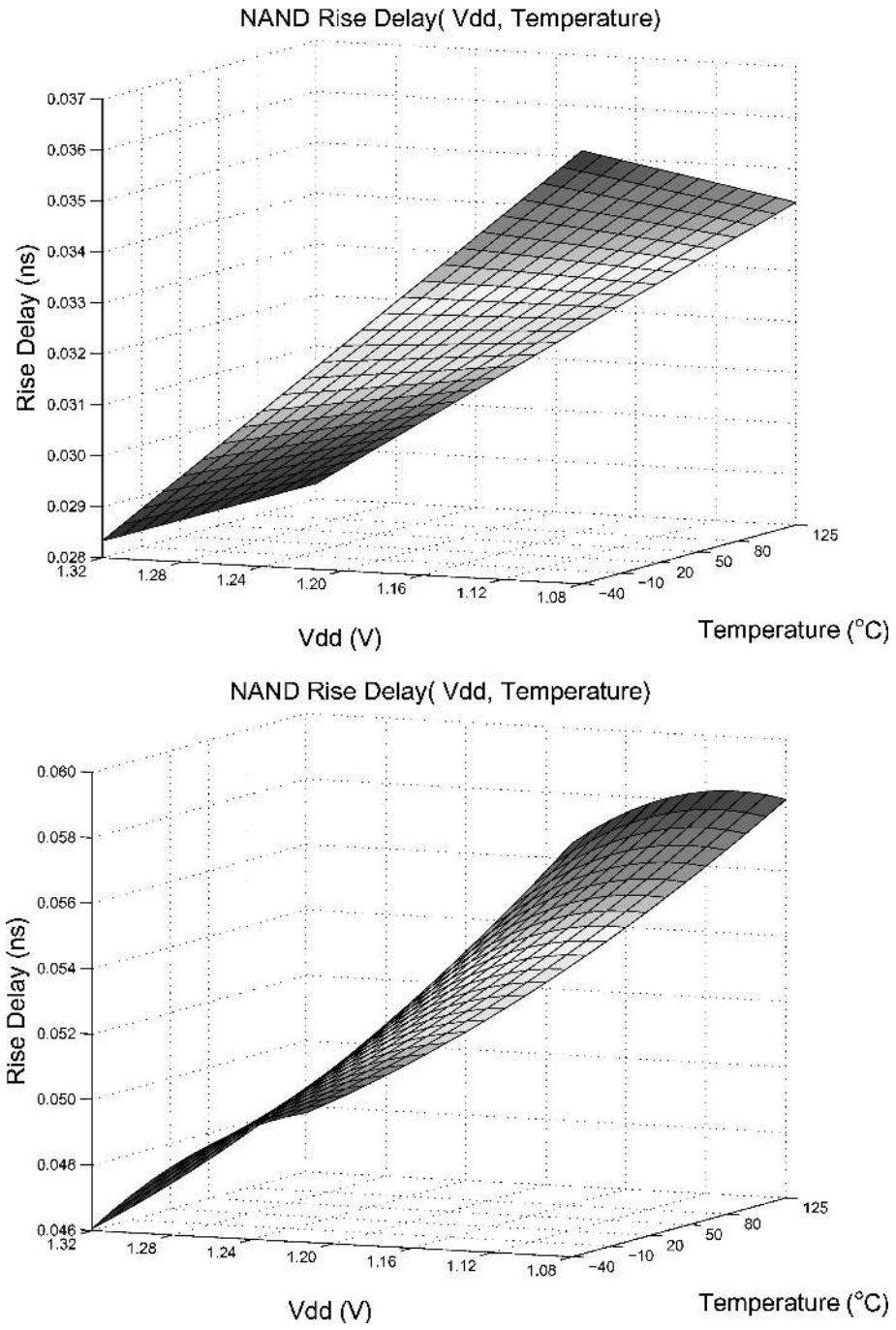

Fig. 3. The three-dimensional (3D) surface plots for the rise delay of a NAND cell as a function of voltage and temperature at $S_{inp} = 0.015$ ns, $C_{out} = 0.003$ pF. The top (bottom) one is for a strong (weak) NAND cell. The voltage and temperature ranges are $[V_{\min}, V_{\max}] = [1.08, 1.32]$ V and $[T_{\min}, T_{\max}] = [-40, 125]$ C.

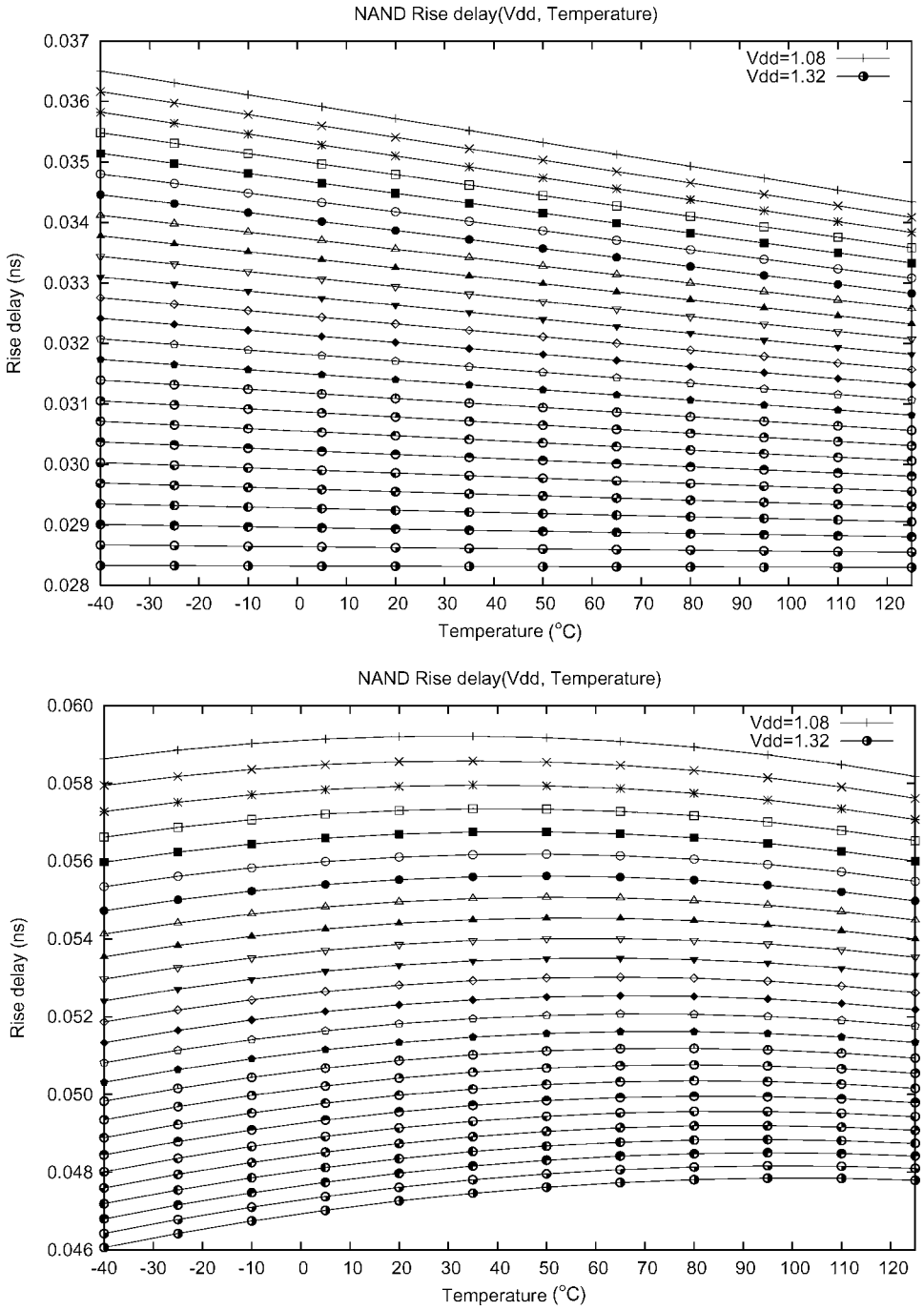

Fig. 4. Slices of Figure 3 from the temperature axis for 0.01-V increments. Voltage increases from the top of the plot to its bottom. Note that negative slope implies ITD.

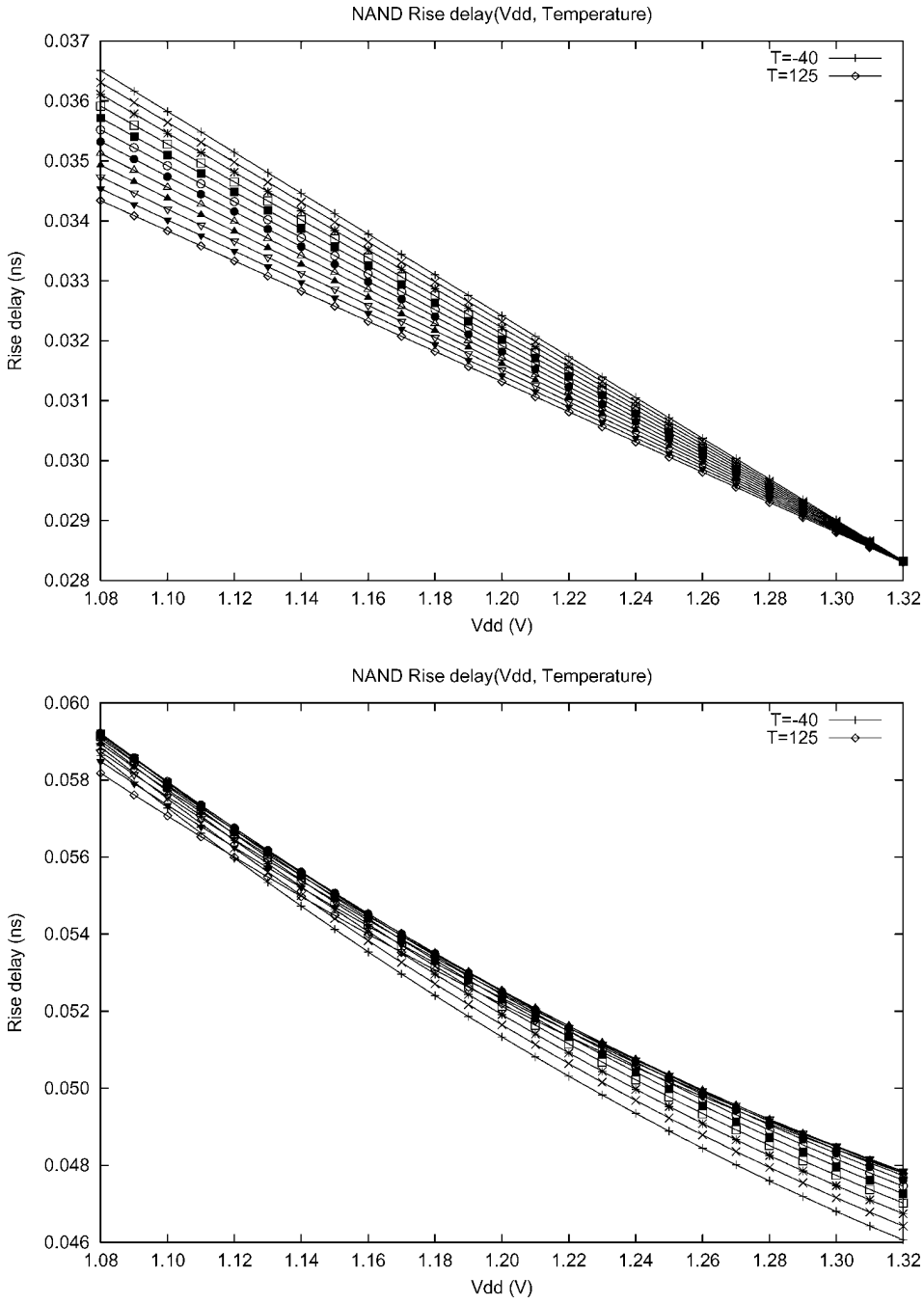

Fig. 5. Slices of Figure 3 from the voltage axis. The curves in the top plot crossover at $V_{\max}$ only whereas those in the bottom plot crossover at multiple voltages. Moreover, the curve of $T_{\min}$ produces the maximum delay in the top plot whereas the curve of 35°C, an intermediate temperature, produces the maximum delay in the bottom plot. Note that crossovers and order reversals imply ITD.

3. PVT CORNERS

Cell libraries usually model delay and output transition time directly although they have recently started to model them indirectly through current. Without loss of generality, we will focus only on delay. The delay model uses either tables or polynomials. Both models have limitations. In table-based modeling, the PVT parameters cannot be used as indices into tables (which have at most three dimensions). As a result, such modeling requires a separate library for each PVT corner and uses scaling (or derating) to consider variations around these corners. In polynomial-based modeling, polynomials can have six parameters and two of them can be voltage and temperature. As a result, such modeling does not need voltage and temperature scaling.

Here is how scaling works. Let $< P, V, T >$ denote a corner with the PVT parameter values P, V, and T. For each PVT parameter X, let $X_{\max}$ and $X_{\min}$ denote its maximum and minimum values. Then, for example, the traditional slow and fast corners are $< P_{\max}, V_{\min}, T_{\max} >$ and $< P_{\min}, V_{\max}, T_{\min} >$, respectively. Also, let D denote the delay function and $D_{nom} = D(P_{nom}, V_{nom}, T_{nom})$ be its nominal value at some corner. Then, $D(P, V, T)$ can be computed from D_{nom} using

$$D(P, V, T) = D_{nom} * S_P * S_V * S_T, \tag{5}$$

where, for any PVT parameter X,

$$S_X = 1 + K_X * \Delta X = 1 + K_X * (X - X_{nom}) \tag{6}$$

is the scaling done with respect to X.

In this scaling equation, K_X is called the scaling factor, the derating factor, or k-factor. Note that delay is scaled independently using the PVT parameters, and as stated in Daga et al. [1998], this in general cannot bound delay properly when ITD occurs.

4. HANDLING ITD IN STA

We discuss two approaches to handle ITD in STA: an existing approach called *corner doubling* and our proposed approach called *bounding*.

A note on the notation. Although ITD depends on process, voltage, and temperature, we will, for simplicity of exposition, drop process from the parameter list of delay.

4.1 Corner Doubling

As mentioned in Section 1, the corner doubling approach has already been employed in practice by the some users of STA tools as a temporary solution until these tools can support ITD [Gerousis 2003].

From observing ITD behavior in Figure 1 (bottom), the following assumption seems plausible: all the other delay curves are sandwiched between these two curves passing through the same inversion point. This implies that the slow corner can happen at $T_{\max}$ or $T_{\min}$ only, so we should include both in our corner definitions. For example, the new slow corners are $< P_{\max}, V_{\min}, T_{\max} >$ and $< P_{\max}, V_{\min}, T_{\min} >$. This similarly applies to other corners.

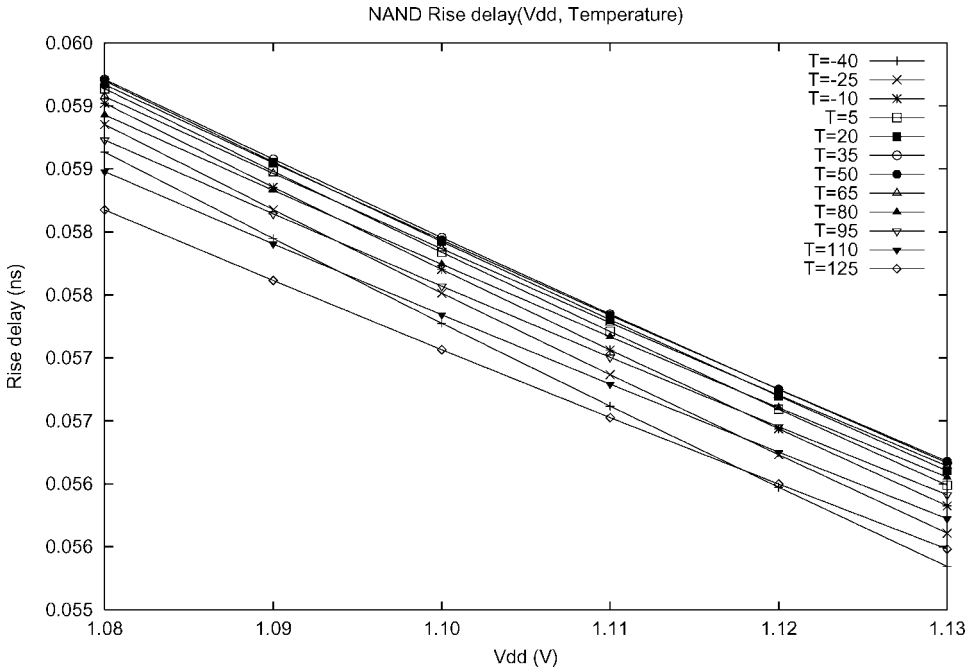

Fig. 6. A magnified view of Figure 5 (bottom).

The advantage of the corner-doubling approach is its simplicity, for example, it does not incur changes to the library characterization scripts. Its disadvantage is that it doubles the number of corners (hence, its name). This, of course, increases library characterization time, library size, timing verification using STA, etc. The overhead is acceptable if we can prove that this approach really bounds delay. Unfortunately, we will show below that this approach does not work because it can underestimate the worst case delay.

The assumption that the corner-doubling approach is based on can be written mathematically as

$$D(V_{\min}) \leq \max\{D(V_{\min}, T_{\max}), D(V_{\min}, T_{\min})\}, \tag{7}$$

where $D(V_{\min})$ is the delay over all temperatures. The fact that this assumption is incorrect can easily be seen experimentally from Figure 6. This plot is a magnified view of Figure 1 (bottom) and Figure 5 (bottom). In Figure 6, the maximum value of $D(V_{\min}, T)$ occurs at $T = 35^{\circ}$C. This plot and Figure 5 (bottom) show that multiple inversion points do exist and are actually very likely; hence, the corner-doubling approach does not bound the worst-case delay. Interested readers should consult Filanovsky and Allam [2001] for a detailed discussion of the reasons behind multiple inversion points.

In practice, the corner-doubling approach can use more than two temperatures, that is, it can multiply the number of corners by a factor greater than 2. These temperatures are usually determined so that their delay curves sandwich many other curves for some voltage interval. In the limit (when there are many crossover points close to each other), this approach converges to the

bounding approach described in the Section 4.2, which is our solution to the ITD problem.

4.2 Bounding

Any approach to handle ITD correctly needs to bound $D(V, T)$ correctly for industrial sign-off STA tools; hence, we propose the bounding approach. This approach revises Equation (7) to obtain

$$D(V_{\min}) \leq \max_{T_{\min} \leq T \leq T_{\max}} \{D(V_{\min}, T)\}, \tag{8}$$

where $D(V_{\min})$ is the delay over all temperatures (as in Equation (7)).

This equation means that the bound has to be computed over all temperatures or by sweeping over all temperatures. The temperature sweeping actually guarantees the correctness of this equation. The need for it can again be seen from Figure 5 (bottom); it is very difficult to analytically determine what temperatures bound delay.

The advantage of this approach is its correctness in bounding delay. Its main disadvantage is that library characterization time increases due to the need for the extra temperature sweep. However, the temperature sweep does not have to be done at library characterization time. Library can still be characterized in the usual way, taking temperature as a parameter of delay. At runtime, an STA tool can easily do this temperature sweep; it can also get a considerable speedup by focusing only on the temperature and voltage ranges of the cells of the input design (as discussed in Section 5.5). This focus can even tighten the delay bounds.

5. ALGORITHMS

The existence of ITD behavior and its correct handling by the bounding approach give rise to a number of algorithmic questions. Here we examine these questions and propose algorithms. The pseudocode for these algorithms is presented in Figure 7. Each algorithm takes in nonempty voltage and temperature intervals, and returns a vector (denoted by <>). The first element of the vector is always a flag indicating whether or not the algorithm was successful in its search. The other elements are the values that the algorithm was searching for.

The algorithms below are devised to run in an STA tool. To run them during library characterization, they need to be converted to a form in which they iterate over all voltages and temperatures until they find the answer.

Moreover, we assume continuous voltage and temperature intervals. Our algorithms are easy to adapt to discrete intervals.

5.1 Detecting ITD

How can we detect whether or not ITD occurs within a voltage interval? Any algorithm to decide this question must rely on the following fact, which follows by the definition of ITD: given two temperature curves at T_l and T_u, if ITD occurs at one or possibly more unknown crossover voltages V_x in a voltage interval of $[V_l, V_u]$, then the order of $D(V, T_l)$ and $D(V, T_u)$ changes depending

```
FIND-ITD(V_l, V_u, T_l, T_u)
1.  if V_l ≥ V_u or T_l ≥ T_u then
2.    return < false >
3.  diff1 = D(V_l, T_l) − D(V_u, T_l)
4.  diff2 = D(V_l, T_u) − D(V_u, T_u)
5.  if diff1 * diff2 < 0 then
6.    return < true >
7.  else
8.    V_mid = (V_l + V_u)/2
9.    < found > = FIND-ITD(V_l, V_mid, T_l, T_u)
10.   if not found then
11.     < found > = FIND-ITD(V_mid, V_u, T_l, T_u)
12.   return < found >

FIND-XOVER-VOLT(V_l, V_u, T_l, T_u)
13. if V_l ≥ V_u or T_l ≥ T_u then
14.   return < false, −∞ >
15. V_mid = (V_l + V_u)/2
16. if D(V_mid, T_l) == D(V_mid, T_u) then
17.   return < true, V_mid >
18. else
19.   < found, V_x > = FIND-XOVER-VOLT(V_l, V_mid, T_l, T_u)
20.   if not found then
21.     < found, V_x > = FIND-XOVER-VOLT(V_mid, V_u, T_l, T_u)
22.   return < found, V_x >

FIND-DELAYS(V_l, V_u, T_l, T_u)
23. if V_l ≥ V_u or T_l ≥ T_u then
24.   return < false, +∞, −∞ >
25. D_min = +∞; D_max = −∞
26. for each V from V_l to V_u do
27.   for each T from T_l to T_u do
28.     D_min = min{D_min, D(V, T)}
29.     D_max = max{D_max, D(V, T)}
30. return < true, D_min, D_max >
```

Fig. 7. Algorithms to detect ITD, to find a crossover voltage, and to find min and max delay bounds.

on the order of V_x and V. This fact leads to the algorithm FIND-ITD in Figure 7. Doing a sequential search over the voltage interval suffices to detect ITD, and this leads to a nonrecursive algorithm. However, as Figure 1 (bottom) shows, checking the order at the interval ends may lead to earlier detection in practice, so FIND-ITD is written recursively using the divide-and-conquer paradigm.

FIND-ITD works as follows. Lines 1–2 checks for empty intervals. Lines 3–4 compute the delay differences at the ends of the voltage interval. Line 5 checks to see if there is any crossover. If so, line 6 exits indicating a success. If not, line 8 computes the midpoint of the voltage interval, and FIND-ITD first recurs in one side of the midpoint. If the result from this side is negative, then FIND-ITD recurses in the other side of the midpoint.

In the worst case, FIND-ITD will check every voltage in the voltage interval. This guarantees its correctness. It also implies its time complexity; hence, FIND-ITD runs in $O(1 + V_u - V_l)$ time. A dramatic speedup is possible if the input voltage interval is known to have one or, more generally, an odd number of crossover voltages. If this assumption, called the *odd-number assumption*, holds, one order check at the ends of the voltage interval suffices. This

reduces the time complexity to constant time and obviates the need to have lines 7–12.

5.2 Finding Crossover Points

Given any two delay curves at two constant temperatures T_l and T_u, how can we find a crossover voltage? Before answering this question, let us formulate it mathematically.

We are given $D(V, T_l)$ and $D(V, T_u)$, where V is a variable. We are required to find the crossover voltage V_x such that $D(V_x, T_l) = D(V_x, T_u)$. If we rewrite the last equality as $D(V_x, T_l) - D(V_x, T_u) = 0$, then we can reduce the original problem to a root finding problem. If we have an analytical formula for the D function, then we can find all its roots [Press et al. 1999]. Unfortunately, no analytical formula general enough to support industrial sign-off STA seem to exist. As such, we instead reformulate the problem as a search problem and propose the algorithm FIND-XOVER-VOLT in Figure 7. Similar to FIND-ITD, this algorithm is also written recursively using the divide-and-conquer paradigm.

FIND-XOVER-VOLT works like FIND-ITD. The main difference is the test at line 16. This line tests for equality of the delays as they need to be equal at the crossover voltage. As a result of this similarity, the algorithms also have the same runtime.

Like FIND-ITD, FIND-XOVER-VOLT gets faster with the odd-number assumption. If it holds, FIND-XOVER-VOLT reduces to a bisection or binary search [Press et al. 1999], eliminating half of of the voltage interval at every invocation. This, of course, speeds up FIND-XOVER-VOLT to run in $O(1 + \lg(1 + V_{\max} - V_{\min}))$ time.

Here is how this speedup can be achieved. At line 18, if $D(V_{mid}, T_l) < D(V_{mid}, T_u)$ and $D(V_l, T_l) < D(V_l, T_u)$, then the $[V_l, V_{mid}]$ interval has an even number, including zero, of crossovers. Since there needs to be an odd number of changes, the other half of the voltage interval, namely, $[V_{mid}, V_u]$, must have at least one crossover. Hence, the search should focus on the latter interval only. The case for the $[V_l, V_{mid}]$ interval works similarly.

Note that FIND-XOVER-VOLT finds only one crossover voltage. To find all of them, this algorithm needs to be written iteratively to sweep over all voltage and temperature pairs and performing the check of line 16.

Tables I and II show how the crossover voltages for the $T_{\min}$ and $T_{\max}$ curves change as a function of input slew and output load. The cell arc is the same one as in Figure 5 (top) and (bottom). The absence of a crossover voltage is indicated by $-\infty$.

5.3 Bounding Cell Arc Delays

How can we bound from below and above the delay of any cell arc? The answer to this question relies on Equation (8). This equation and its min version simply state that, in the absence of an analytical delay formula, the maximum and minimum bounds can be obtained by sweeping over the temperature and

Table I. Crossover Voltages for $T_{\min}$ and $T_{\max}$ Curves from Figure 5 (top) at Each Pair of Input Slew and Output Load

Output Loads (pF)	Input Slews (ns)						
	0.010	0.015	0.036	0.080	0.166	0.338	0.500
0.001	1.27	1.32	1.32	1.32	1.32	1.32	1.32
0.003	1.26	1.32	1.32	1.32	1.32	1.32	1.32
0.007	1.23	1.31	1.32	1.32	1.32	1.32	1.32
0.016	1.17	1.28	1.32	1.32	1.32	1.32	1.32

Table II. Crossover Voltages for $T_{\min}$ and $T_{\max}$ Curves from Figure 5 (bottom) at Each Pair of Input Slew and Output Load

Output Loads (pF)	Input Slews (ns)						
	0.010	0.015	0.036	0.080	0.166	0.338	0.500
0.001	1.09	1.19	1.26	1.28	1.27	1.22	1.15
0.003	$-\infty$	1.11	1.24	1.29	1.30	1.24	1.17
0.007	$-\infty$	$-\infty$	1.20	1.30	1.32	1.29	1.21
0.016	$-\infty$	$-\infty$	1.14	1.30	1.32	1.32	1.26

voltage ranges. That is, the algorithm computes the lower and upper bounds in

$$\min_{T_{\min} \le T \le T_{\max}} \{D(V,T)\} \le D(V) \le \max_{T_{\min} \le T \le T_{\max}} \{D(V,T)\} \tag{9}$$

at each voltage V, where $D(V)$ is the delay over all temperatures. The time complexity of this algorithm is $O((V_{\max} - V_{\min})(T_{\max} - T_{\min}))$.

Note that this algorithm finds the exact bounding in that, at every sampling point, it has zero approximation error. One way to speed this up is to use one of the many algorithms on determining a piecewise linear (PWL) representation of curves (like delay curves). These algorithms allow different optimization criteria such as minimizing approximation error under a bounded number of chords [Goodrich 1994] or minimizing the number of chords under a bounded approximation error [Hakimi and Schmeichel 1991].

On the extreme case, a linear upper bound can be found in $O(1+T_{\max}-T_{\min})$ time by computing temperature sweeps at $V_{\min}$ and $V_{\max}$ only. This, of course, relies on the convexity of the upper bound. Another way to speed this up is to note that the lower bound traces the curves for temperatures $T_{\max}$ and $T_{\min}$. Thus, there is no need to sweep over all temperatures. Both of these speedup techniques are empirical observations that need further validation.

5.4 Bounding Path Delays

How can we bound from below and above the delay of any path? Without loss of generality, assume that the net delays in the path are already bounded. Then, bounding needs to consider only the cell arc delays. As a result, the answer to this question follows directly from Equation (9). That is, correct bounds for path delays follow from correct bounds for cell delays.

Figure 8 (top) shows the delay of a path of 16 cells as a function of voltage and temperature. This plot clearly shows the effect of ITD on path delays as well as the need for the bounding approach, for example, the minimum path delay happens at a temperature other than $T_{\min}$ or $T_{\max}$.

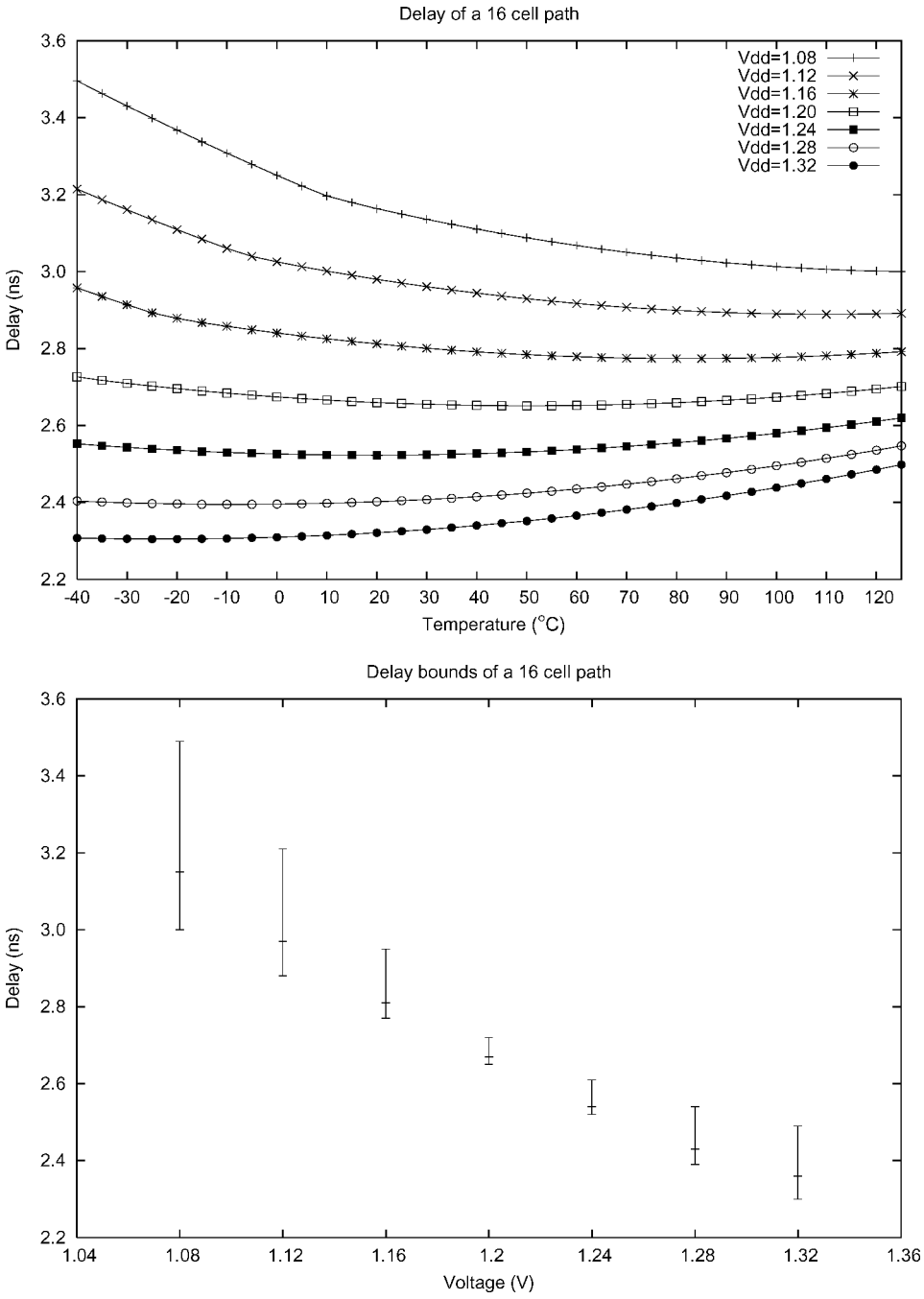

Fig. 8. The delay of a path of 16 cells as a function of voltage and temperature. The top plot shows how the delay changes as temperature changes, and the bottom plot shows how the delay is bounded by an interval and where its average value falls in the interval.

Figure 8 (bottom) shows how the bounding approach can tightly bound the path delays. It also shows how much variability in path delays ITD can cause.

5.5 Bounding Path Delays with Known Voltage and Temperature Ranges

The voltage and temperature ranges of a cell can fall into one of the following four cases: (1) one voltage and one temperature; (2) one voltage and multiple temperatures; (3) multiple voltages and one temperature; and (4) multiple voltages and multiple temperatures. These cases, respectively, reduce to the following geometric objects: (1) one point; (2) one voltage curve as in Figure 4; (3) one temperature curve as in Figure 5; and (4) a surface as in Figure 3.

It may be claimed that our bounding approach is unnecessarily pessimistic in estimating path delays; however, this is not the case. First, our approach does not use bounding for a path if its cells are in cases 1 or 3. Second, when our approach computes a path delay bounds in cases 2 or 4, the bounds are *exact* in that there is at least one voltage-temperature setting to each cell at which the path delay attains the bounds. These arguments show that our bounding approach is not pessimistic.

On the other hand, it is true that the setting mentioned above might include a scenario in which a high-temperature cell can happen right next to a low-temperature cell. This temperature pair is, of course, highly unlikely because temperature changes slowly across a die even though the die can have regions with very high and low temperatures. A similar scenario can be derived for voltages. To avoid such scenarios, users need to use voltage and temperature correlation data such as voltage and temperature maps (with voltage drops and temperature gradients).

6. LIBRARY EXTENSIONS

In the current cell libraries, the corners are defined using $T_{\min}$ and $T_{\max}$. As we have shown, this results in errors in presence of ITD. Since the previous section has shown that we can correctly bound cell delays, we now describe how to include these bounds in the cell libraries as follows: let T_{up} and T_{lo} correspond to the set of temperatures that produce the upper and lower bounds of delay, respectively. Then, we can replace $T_{\max}$ and $T_{\min}$ with T_{up} and T_{lo} as if ITD never happened. For example, the slow corner $< P_{\max}, V_{\min}, T_{\max} >$ needs to be changed to $< P_{\max}, V_{\min}, T_{up} >$.

The disadvantage of this proposal, if chosen, is that it increases library characterization time, but this is unavoidable due to the need for correct bounds. Its advantage is that it keeps the number of libraries and corners the same and it also correctly bounds delays.

If voltage and temperature maps (with correlations and distributions) are available, we actually do not recommend this approach. Instead, we think the users should use the existing libraries and allow their STA tool to handle ITD. The users should, however, ensure that their libraries allow the accurate calculation of delay at any temperature.

7. EXPERIMENTS AND RESULTS

For our experiments, we used a 90-nm industrial cell library and PrimeTime, a leading industrial sign-off STA tool. The library had more than 500 cells, and it used polynomial-based delay modeling in which the voltage and temperature parameters were inputs of the polynomials and the process parameter was a scaling factor.

We characterized four combinational cells: NAND, NOR, INVerter, and BUFfer. For each cell, we used the minimum- and maximum-sized versions, which we refer to as the *weak* and *strong versions*, respectively.

For cell delay, we ran experiments involving all possible combinations of the following parameters: input transition sense (two choices for rise or fall), cell strength (two choices for weak or strong), input slews (seven choices), output loads (four choices), process (three choices for fast, slow, and typical corners), voltage (25 choices in [1.08, 1.32]), and temperature (12 choices in [−40, 125]). This amounted to more than 100,800 runs for each cell.

For path delay, we extracted critical paths from industrial designs. We remapped the paths to use cells from our library. We also set side input sensitization to noncontrolling values.

We have already provided some of our observations when we explained our plots and tables. We now list our remaining observations: (1) ITD occurred at all process corners; (2) ITD occurred with both weak and strong cells yet crossover voltages seemed to be lower with weak cells; (3) ITD occurred with both rising and falling input transitions (see Filanovsky and Allam [2001] and Park et al. [1995] for some discussion of differences); (4) ITD seemed to occur more often with large input slew and smaller output load combinations; finally, (5) ITD occurred at all path lengths yet larger path lengths could show a larger effect, for example, the 16-cell path in Figure 8 has nearly 17% difference in delay due to ITD.

How much error is introduced if ITD is ignored? For path delays, we observed an error of nearly 17%. For cell delays, we observed an average error of nearly 10% (with a standard deviation of nearly 10%) and a maximum error of nearly 63%. These figures obviously depend on the cells and libraries we used, and it is quite possible to observe larger or smaller errors with other cells and libraries; however, these figures show that ITD cannot be ignored.

These observations can provide starting points for future research. More notably, these and earlier observations show that ITD seems to depend on many factors, as also well pointed out in Gerousis [2003]. Characterizing this dependence exactly (or analytically) seems very difficult in general. These findings strengthen our bounding approach.

8. CONCLUSIONS

ITD breaks down the usual assumption that cell delay increases with decreasing voltage and increasing temperature. As such, ITD makes it very difficult to analytically determine the temperatures that maximizes or minimizes the delay of a cell or a path. This, of course, creates problems with the usual way of doing timing verification and STA. Since supply voltage is expected to go down

with 65- and 45-nm technology nodes, the effect of ITD will be more pronounced. This article analyzes these problems and proposes a proper handling of ITD in STA via a bounding approach. It also shows that the bounding approach is an appropriate and safe solution given the fact that ITD depends on many factors.

ACKNOWLEDGMENTS

We thank the anonymous reviewers as well as Dr. Halim Damerdji, Dr. Rachid Helaihel, Jamil Kawa, and Dr. Kayhan Kucukcakar for their helpful comments.

REFERENCES

BELLAOUAR, A., FRIDI, A., ELMASRY, M. I., AND ITOH, K. 1998. Supply voltage scaling for temperature insensitive CMOS circuit operation. *IEEE Trans. Circ. Syst.-II: Analog Digital Signal Process. 45*, 3 (Mar.), 415–7.

DAGA, J. M., OTTAVIANO, E., AND AUVERGNE, D. 1998. Temperature effect on delay for low voltage applications. In *Proceedings of the Conference on Design, Automation and Test in Europe*. 680–685.

FILANOVSKY, I. M. AND ALLAM, A. 2001. Mutual compensation of mobility and threshold voltage temperature effects with applications in CMOS circuits. *IEEE Trans. Circ. Syst.-I: Fundament. Theor. Appl. 48*, 7 (July), 876–884.

GEROUSIS, V. 2003. Design and modeling challenges for 90 nm and 50 nm (invited paper). In *Proc. Custom Integrated Circuits Conf.* IEEE, 353–360.

GOODRICH, M. T. 1994. Efficient piecewise-linear function approximation using the uniform metric. In *Proceedings of the 10th Symposium on Computational Geometry*. ACM Press, New York, NY, 322–331.

HAKIMI, S. L. AND SCHMEICHEL, E. F. 1991. Fitting polygonal functions to a set of points in the plane. *CVGIP: Graph. Models Image Process. 53*, 2 (Mar.), 132–136.

KANDA, K., NOSE, K., KAWAGUCHI, H., AND SAKURAI, T. 2001. Design impact of positive temperature dependence on drain current in sub-1-V CMOS VLSIs. *IEEE J. Solid-State Circ. 36*, 10 (Oct.), 1559–1564.

LONG, E., DAASCH, W. R., MADGE, R., AND BENWARE, B. 2004. Detection of temperature sensitive defects using ZTC. In *Proceedings of the 22nd VLSI Test Symposium*. IEEE Computer Society Press, Los Alamitos, CA, 185–190.

PARK, C., JOHN, J. P., KLEIN, K., TEPLIK, J., CARAVELLA, J., WHITFIELD, J., PAPWORTH, K., AND CHENG, S. 1995. Reversal of temperature dependence on integrated circuits operating at very low voltages. In *Proceedings of the International Electron Devices Meeting*. IEEE Computer Society Press, Los Alamitos, CA, 71–74.

PRESS, W. H., TEUKOLSKY, S. A., VETTERLING, W. T., AND FLANNERY, B. P. 1999. *Numerical Recipes in C*. Cambridge University Press, Cambridge, U.K.

SAKURAI, T. AND NEWTON, A. R. 1990. Alpha-power law MOSFET model and its applications to CMOS inverter delay and other formulas. *IEEE J. Solid-State Circ. 25*, 2 (Apr.), 584–594.

Received August 2005; revised November 2005; accepted December 2005