Gate Sizing Using Incremental Parameterized Statistical Timing Analysis

M. R. Guthaus*, N. Venkateswaran†, C. Visweswariah‡ and V. Zolotov‡
*Department of EECS, University of Michigan, Ann Arbor, MI 48109
mguthaus@eecs.umich.edu
†IBM Systems & Technology Group, Hopewell Junction, NY 12533
natesan@us.ibm.com
‡IBM T.J. Watson Research Center, Yorktown Heights, NY 10598
{chandu,zolotov}@us.ibm.com

***Abstract*— As technology scales into the sub-90nm domain, manufacturing variations become an increasingly significant portion of circuit delay. As a result, delays must be modeled as statistical distributions during both analysis and optimization. This paper uses incremental, parametric statistical static timing analysis (SSTA) to perform gate sizing with a required yield target. Both correlated and uncorrelated process parameters are considered by using a first-order linear delay model with fitted process sensitivities. The fitted sensitivities are verified to be accurate with circuit simulations. Statistical information in the form of criticality probabilities are used to actively guide the optimization process which reduces run-time and improves area and performance. The gate sizing results show a significant improvement in worst slack at 99.86% yield over deterministic optimization.**

I. Introduction

As technology scales into the sub-90nm domain, manufacturing variations have become an increasingly significant portion of circuit delay. As a result, delays are best modeled by statistical distributions rather than deterministic values. The resulting distribution of this performance indicates the fraction of chips that will suffer from parametric yield loss. Parametric yield loss occurs when a chip is functional, but does not meet performance specifications such as a required clock cycle.

To improve parametric yield, it must be considered during design optimization using statistical static timing analysis (SSTA). Continuing to use deterministic static timing analysis (DSTA) as in current methodologies can be both pessimistic and misleading for optimization tools. By using SSTA, the optimization tool is assured that the entire relevant process space is examined during a single timing analysis. Parametric SSTA further allows the optimizer to deduce the impact of multiple sources of variation on timing simultaneously. Due to the necessary additional work in SSTA, run-times of SSTA are larger than DSTA. To reduce this overhead, incremental SSTA is ever more important. Incremental SSTA assures that run-time overhead is minimized by requiring only the necessary timing updates to satisfy a timing answer after a perturbation of the design.

Recently, several methods for SSTA have been proposed [1]–[4]. The path-based methods, while they consider correlation due to re-convergent fanout and shared process parameters, are inherently not incremental. They require a run-time intensive integration over the process space after every change. On the other hand, the block-based methods, while they are incremental, have not considered correlation. The block-based methods recently presented in [3] consider correlation yet retain incremental capabilities. Complementing the previous work on SSTA, some work has started to examine statistical optimization using non-incremental, non-parameterized SSTA. These works tend to only address random variation [5]–[9], use simplified heuristics [5], [6], [8], [9], ignore path correlations [8], [9], or are too complex for large-scale optimization [10]. The contributions of this work can be summarized as the following:

- This paper uses incremental SSTA for circuit optimization while considering multiple correlated and independent sources of variation.
- This is the first paper that actively uses SSTA to guide gate sizing. Statistical information, in the form of criticality probabilities, is used to select candidate gates that guide the optimizer to a better solution more quickly than passive SSTA optimization.

In Section II, a brief description and some examples show how DSTA cannot consider optimization effects on parametric yield. The benefits of using SSTA in the same situations are also presented. In Section III, the deterministic sizing problem is formulated and a sensitivity-based sizing algorithm is presented. Then in Section IV a brief introduction to parameterized, block-based SSTA is given. In Section V, the deterministic sizing algorithm is extended to use a SSTA tool and improvement over worst-case (WC) DSTA-based optimization is demonstrated. In Section VI, the baseline statistical sizing algorithm is improved using criticality information to actively guide the optimization. Results comparing the deterministic sizing, statistical sizing, and criticality-guided statistical sizing are presented and discussed. Finally, conclusions and future work are presented in Section VII.

II. DSTA vs. SSTA

Traditional DSTA has used process corner case files for performing timing analysis. A *process corner case file* (or corner for short) is a set of values assigned to all process parameters

to bound the circuit performance. Past methodologies have performed late mode optimization while considering DSTA in a single WC corner[1]. The *WC corner* is defined as the corner with every parameter at a $\mu \pm 3\sigma$ value such that a typical circuit has the least slack. Similarly, a best-case (BC) corner is defined for early mode timing analysis. Corner-based DSTA is both pessimistic and misleading for circuit optimization.

First, the WC corner (or a worse corner) has a very small likelihood of occurring in any manufactured part. This results in a pessimistic view of circuit performance if manufacturing has a true goal of 3σ yield. For example, consider a design that has equal sensitivity to two independent sources of variation. In the traditional WC corner, each of these sources is set at its 3σ value. For every parameter to assume a value greater than or equal to 3σ in a single manufactured part has a probability of $(1 - F(\mu + 3\sigma))^2 \approx 1.83 \times 10^{-6}$ where F is the cumulative distribution function (CDF). Therefore, optimization is performed at an effective corner of $F^{-1}(1.83 \times 10^{-6}) = 4.24\sigma$ where F^{-1} is the inverse CDF. Thus, the corner is pessimistic by a factor of $\frac{4.24\sigma}{3\sigma} = \sqrt{2}$. If we generalize this to n sources of variation, we can potentially reduce the pessimism by a factor of $\sqrt{n}$. SSTA avoids this pessimism by reporting the true 3σ circuit performance. This extreme pessimism can potentially mask the likely critical paths in a design with paths that are extremely unlikely to be limiting in any manufactured chip. The additional optimization to fix these paths can consume extra power and prevent finding a good solution.

In addition to using an extremely unlikely corner, DSTA could miss the performance-limiting process corner entirely since it only uses a single WC corner. The true performance-limiting corner depends on timing constraints between clock and data-path arrival times and their sensitivity to the specific source of variation. For example, suppose that both the data-path and clock are sensitive to L_{eff} variation. If the data-path is logic dominated, it will be more sensitive (i.e., it becomes slower more quickly) than the clock as L_{eff} increases. In this case, the WC setting for L_{eff} is the largest $(\mu + 3\sigma)$ value. However, if the data-path is mostly wire-dominated delay, it will be less sensitive to L_{eff} than the clock path. In this case, the smallest $(\mu - 3\sigma)$ value of L_{eff} is performance limiting. To find the true WC corner, every combination of process parameters that are shared between the clock and data-path must be enumerated. This results in an exponential number of process corners with the true WC corner being design specific. Examining all corners during optimization is prohibitive. SSTA avoids enumerating all process corners by propagating probability distributions that represent all corners simultaneously.

In addition to potentially missing the true WC corner, DSTA cannot accurately assess the effects of many critical paths on yield. If there are many equally criticality paths on a chip that are not perfectly correlated, the maximum circuit delay is always worse than if there is a single worst path. This is termed a "slack wall." Having more paths increases the probability that one of the paths will be critical in some portion of the process space. WC design cannot formally quantify this effect, but heuristic figures-of-merit as in [8] often do. However, these heuristics do not consider factors such as correlation among the paths. In addition, they also fail to be truly incremental. In WC optimization, the critical paths presented may be misleading to the optimizer due to inadequate correlation information. For example, if we have two completely correlated paths with one being nominally more critical, the less critical path will not be limiting in any manufactured chip. This is because the more critical path will always dominate it in every portion of the process space. WC design, in the absence of correlation information, assumes that the single worst path is always the most critical or any near critical path could potentially be the critical path in a manufactured part. Optimizing the second most critical path may result in wasted area and power. In addition, optimizing this path could potentially prevent the optimizer from finding a solution that meets timing. SSTA can quantify the effects of a slack wall and thereby avoid unnecessary yield loss due to it.

[1]The nominal corner is sometimes used during optimization for aggressive, speed-binned designs.

III. Deterministic Gate Sizing

The objective of deterministic gate sizing is to compute the sizes of logic gates such that a timing constraint is met while consuming the smallest amount of total cell area or power. Traditional deterministic gate sizing can be formally defined as,

$$\begin{aligned} \min \quad & \sum_{\forall i \in gates} C(i) \\ s.t. \quad & \tau \leq T_{target}, \end{aligned}$$

where $C(i)$ is the cost (power, area, etc.) of gate i, τ is the circuit delay, and T_{target} is a target cycle delay. If the timing constraint cannot be met, which is often the case, it is best to minimize the amount by which it fails.

Many exact and heuristic approaches to deterministic gate sizing have been proposed [11]–[14]. Algorithm 1 shows the implementation of a generic sensitivity-based sizing algorithm that is similar in spirit to TILOS [12]. The general idea of the algorithm is to start from minimum-sized gates, pick a set of candidate gates and try increasing each candidate gate one-at-a-time. For each candidate, the circuit slack change and added gate capacitance are measured. The gate with the largest slack improvement per increase in gate capacitance is then selected as the best move. It is sized up and the whole procedure is repeated until the constraints are met or no improvement is seen. Traditional algorithms [11]–[14] only consider a single (often WC) process corner and, therefore, are incapable of guaranteeing a specific yield.

IV. Parameterized SSTA

SSTA performs probabilistic timing analysis of a circuit considering process variations. Parameterized SSTA performs statistical timing while retaining the contributions of individual

Algorithm 1 Generic sensitivity-based sizing algorithm.

```
Minimum size all gates.
repeat
    Γ ⇐Generate list of candidate gates
    Initialize best solution, S_best = 0, g_best = 0
    for g_cur ∈ Γ do
        Increment size of g_cur
        Compute change in slack, Δ_slack
        Compute change in cost, Δ_cost
        S_cur ⇐ Δ_slack / Δ_cost
        Restore size of g_cur
        if S_cur > S_best then
            g_best ⇐ g_cur
            S_best ⇐ S_cur
        end if
    end for
    Increase size of best gate, g_best
until S_best = 0 or Slack ≥ 0
```

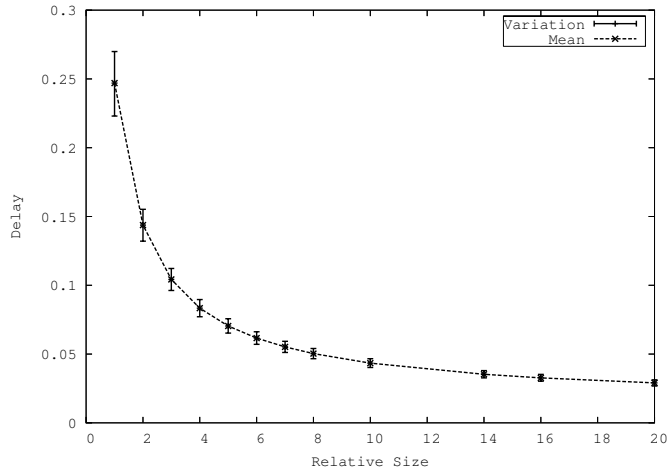

Fig. 1. INV delay with L_{eff} variation.

process parameters to the variation. The fundamental delay model used in our block-based SSTA is a weighted sum of normal distributions. The parameterized first-order expansion is expressed as

$$d = d_0 + \sum_i s_i x_i + s_r x_r, \tag{1}$$

where d_0 is the mean value of delay. x_r and the x_i's are random variables normally distributed with mean zero and variance one, $N(0,1)$, that represent individual process parameters. The x_i random variables represent process parameters that are globally shared among all gates or wires while the x_r random variable represents independent (uncorrelated) variation. The magnitude of the variation is given by the absolute value of coefficients, s_i and s_r, which can differ for each timing quantity. Equation 1 is the statistical canonical form for all delays, required arrival times (RAT), arrival times (AT), and slacks.

A. Statistical Propagation

The SSTA engine traverses the circuit network in breadth-first topological order like DSTA. At each node, the mean and variance of the statistical maximum AT is efficiently calculated using analytic formulas [15], [16]. The correlation coefficient is calculated from the covariance of the two canonical delays. Each output parameter sensitivity, s_i, is computed by linearly combining the delay sensitivities of the ATs using tightness probabilities. A *tightness probability*, T_a, is the probability that the random variable (a) is the greatest of a set. Thus, for signals a and b, the output parameter sensitivity is calculated as $s_i = T_a s_{i,a} + T_b s_{i,b}$ for each parameter, i, where T_a and T_b are the tightness probabilities. The first and second moments of the resulting distribution are matched by selecting an appropriate mean and random sensitivity, s_r. Backwards propagation is done in a similar manner with RATs and a statistical minimum operation. Addition and subtraction are done by adding or subtracting the correlated terms element-by-element and using the square root of the sum of the squares for the independent term. All primary outputs and latch inputs are connected to a virtual sink and annotated with the appropriate negative RAT or setup time. The slack of the entire circuit is then measured at the virtual sink. For detailed explanation of the SSTA methodology and verification of the accuracy with monte carlo, please refer to [3], [17], [18].

B. Process Sensitivities

The model for the delay variation coefficients, s_i, of the first-order parameterized model are now experimentally verified using Spice as in [19]. The variability parameters are assumed to be fully correlated within a gate. In general, every gate arc could have an arbitrary sensitivity that is derived from the transistor widths, gate load, and input slew. However, the experiments in this section show that this dependency can be simplified by substituting the nominal gate delay itself into the variability equation. From this observation, the coefficients of the canonical delay model (Equation 1) are easily derived from the nominal delay as

$$s_i = f(w, C_L, S_{in}) = cd_0. \tag{2}$$

where w is the set of transistor widths, C_L is the load capacitance, and S_{in} is the input slew rate. c is a fitted constant for a timing arc of a specific gate type and d_0 is the delay in the absence of variation. First, to motivate the model, the delay of an inverter with an output load of 100 fF and input slew of 25 ps is simulated with a range of sizes $1\times$ to $20\times$ and the best and worst delays are measured. Figure 1 shows the resulting delay and its variation due to L_{eff}. Qualitatively, it can be seen that the variation bars are proportional to the nominal delay. Reducing the gate delay (by increasing the size of the gate or decreasing the load) decreases the delay variation. Similarly, increasing the gate delay (by decreasing the size or increasing the load) increases the delay variation. Therefore, expressing the variability as a constant percentage of nominal delay is

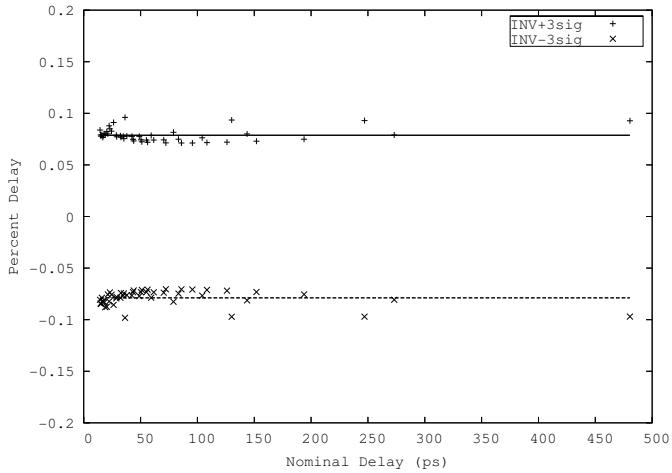

Fig. 2. Percent delay change ($\frac{d_0 \pm 3\sigma}{d_0}$) plotted against nominal delay (d_0) for L_{eff} variation.

reasonable. Figure 2 is a plot of the nominal delay (d_0) of the inverter against the percentage change of the nominal delay ($\frac{d_0 \pm 3\sigma}{d_0}$) for the entire L_{eff} data set with a variety of loads and input slews. The data is approximately constant for the simulated ranges. In addition, the results show a constant for many other variation parameters and different gate types[2].

In the following experiments, there are three correlated and one independent sources of variation. Each parameter has an asserted value $\pm 10\%$ of nominal delay as the sensitivity. These sources represent negative-bias temperature instability (NBTI) degradation[3], temperature, a single V_{th}, and an independent term to represent intra-die variation of all three parameters. Each of the first three terms is fully correlated among all gates whereas the last term is independent among all gates. The independent variable for intra-die modeling can be made less pessimistic as shown in previous publications [4], [20], but that is not done in this paper. The nominal gate delay model considers the effects of both load capacitance and input slew rates on delay.

V. Statistical Gate Sizing

Similar to deterministic gate sizing in Section III, the objective of statistical gate sizing is to meet a timing constraint while minimizing total cell area or power. However, since the circuit delay is now a probability distribution, the timing constraint is probabilistic. The statistical sizing problem is supplemented with a user-defined yield target which specifies that a fixed percentage of manufactured parts are required to work in order for the timing constraint to be met. The statistical gate sizing problem can be formally defined as,

$$\begin{aligned} \min \quad & \sum_{\forall i \in gates} C(i) \\ s.t. \quad & Y(T_{target}) \geq Y_{target}, \end{aligned}$$

where Y_{target} is a yield target and $Y(\tau)$ is the yield of a given circuit delay, τ. Specifically, $Y(\tau) = 1 - CDF(\tau)$, since a larger slack has a lower yield. T_{target} is the same delay constraint as in the deterministic formulation.

The baseline statistical optimization is similar to Algorithm 1, but uses statistical slack instead of deterministic slack. *Statistical slack* is the slack at which the designer specified yield target is met. Formally, this is the inverse yield function evaluated at the yield target, $Y^{-1}(Y_{target})$. The statistical slack is a convenient way to reuse existing circuit optimization algorithms that rely on a single (deterministic) timing quantity. All AT and RAT values are propagated as distributions and slack values are calculated as distributions. For this paper, all statistical slacks are measured at $Y_{target} = 99.86\%$ (3σ). It should be noted that this concept depends on the first two moments of the distribution and is therefore not dependent on any specific distribution. For the purposes of this paper, however, all distributions are approximated as normal. Methods for analyzing non-linear and non-Gaussian parameters in the same SSTA framework were demonstrated in [18] and could easily be used in all algorithms presented.

In this section, optimizations are performed with either DSTA or SSTA, but all results are measured with the same SSTA setup. The DSTA-based optimization uses the gates on the most critical path in the WC corner as candidates for up-sizing. Among the candidate gates, it up-sizes the one that shows the most improvement in deterministic slack per unit gate capacitance. SSTA, however, can find critical paths in different portions of the process space. Therefore, in statistical optimization, the worst statistical slack gates are used as candidates for up-sizing. This allows the optimizer to consider up-sizing gates that are sub-critical in the WC corner but are on the most critical path in other portions of the process space. Statistical sizing was performed with a maximum candidate list size of 10, 20, 50 or all gates in the circuit. Statistical sizing up-sizes the candidate gate with the largest improvement in statistical slack per unit gate capacitance.

Table I shows the final statistical slack, cell area, and run-time for sizing a small industrial circuit. The best solution considers all gates in the circuit during every iteration and has1 37.6% better worst slack than the DSTA-optimized solution. Obviously, the run-time of this method is excessive. Examining more candidate gates in statistical optimization has an obvious run-time penalty, because each gate requires that the circuit slack be evaluated through incremental SSTA. The run-time with 20 candidates is nearly $3\times$ longer than the deterministic case and 50 gates is almost $7\times$ longer. For larger circuits, examining all of the gates is impractical. If we decrease the number of gates to 10 so that the run-time is comparable to deterministic optimization, the results are much

[2]For very slow input slew rates with low gain gates (large size and low load), the constant phenomena does not quite hold, because the above model assumes that the time spent charging or discharging the load capacitance dominates the gate delay. If this is not true, then the input slew and the switching speed of the transistors within the gate determine the delay variation. This case is not considered further since it involves severely over-sizing the current stage of logic and under-sizing the previous stage, which is not realistic in an optimized circuit.

[3]NBTI is a time-related fatigue of PFET drive strength.

TABLE I

GATE SIZING RESULTS FOR A 1200 GATE INDUSTRIAL DESIGN.

Statistical?	Candidates	Slack (ns)	Area	Time (s)
No	-	-1.569	9805	94
Yes	10	-2.906	9823	54
Yes	20	-1.523	9825	263
Yes	50	-1.501	9836	720
Yes	All	-0.978	9868	20792

worse than deterministic optimization. This is due to the fact that the optimization cannot even consider the entire critical path at each move. In the next section, a heuristic to replace statistical slack for selecting candidate gates is presented.

VI. CRITICALITY GUIDANCE

In the previous section, significant improvements were made with statistical gate sizing by up-sizing gates that are sub-critical in the WC corner. These sub-critical gates can contribute to the mean and variance of the critical path but do not lie directly on the WC critical path. The run-time when doing this exhaustively during statistical sizing is exorbitant. This section presents the criticality probability metric and demonstrates how it can improve candidate selection in statistical sizing. The criticality probability of a gate is the likelihood that the critical path of a manufactured chip goes through the gate. A high criticality probability indicates that a gate is a good candidate for improving the critical path in many manufactured parts. There are three properties that affect the criticality probability of a gate: the slack magnitude, the circuit topology and the slack correlation with other critical structures.

The easiest property to understand is the slack magnitude. Consider the slack PDFs of three gates in Figure 3. For simplicity, assume that the slacks are uncorrelated. Gate A is always more critical than gate B. The PDFs of A and B do not overlap in the region of -3σ to $+3\sigma$ for either PDF. Therefore, signal A is more negative in every manufactured part. However, if we compare A and C, the situation is mixed. Sometimes, C can be more critical than A, but usually, A is more critical than C. The exact probability of this outcome is given by the tightness probability of the statistical minimum as described in Section IV.

The second property that can affect the criticality probability of a gate is the topology of the circuit. As an example, consider Figure 4 where all paths have equal slack and, according to traditional DSTA, are equally critical. According to the slack, each of these gates is an equally viable candidate for sizing. However, it is much more efficient to increase the size of gate A because it improves the delay of all paths simultaneously. It is topologically preferred over the other gates[4].

Lastly, the correlation of gates can affect their criticality. Again, consider Figure 4. As before, the best candidate for sizing would be gate A since all paths converge through it, but this might not offer improvement due to loading of the previous stages. The next best candidate is gate D, the 2-input NOR, because it is the least correlated with the other paths. NOR gates are more sensitive to PFET variations because they have a taller PFET stack than NAND gates. Therefore, the overall circuit variability (and 3σ delay) can be improved the most by up-sizing the single NOR gate. This NOR gate is preferred, since it will make the output delay less dependent on NBTI and improves the correlation of the critical paths as a whole.

However, the effects of correlation on circuit performance can be more complicated than this. In another example, consider a circuit with two outputs A and B. Signal A has a mean delay of 3 and standard deviation 0.101. Signal B has a mean delay of 3 and standard deviation 0.1. The 3σ delay of A is 3.303 which is obviously greater than the delay of B at 3.3. In the DSTA, the logic driving A is the only one that can have a positive sensitivity because infinitesimally improving the delay of signal B will not improve overall timing. However, in the statistical regime, signal B can contribute almost equally to the variability of the statistical maximum of A and B. The amount of this contribution depends on the correlation between

[4]It should be noted that past works have used critical path counts as a weighting factor for gates. However, this is topological-only and doesn't consider the magnitude and correlations of the slacks.

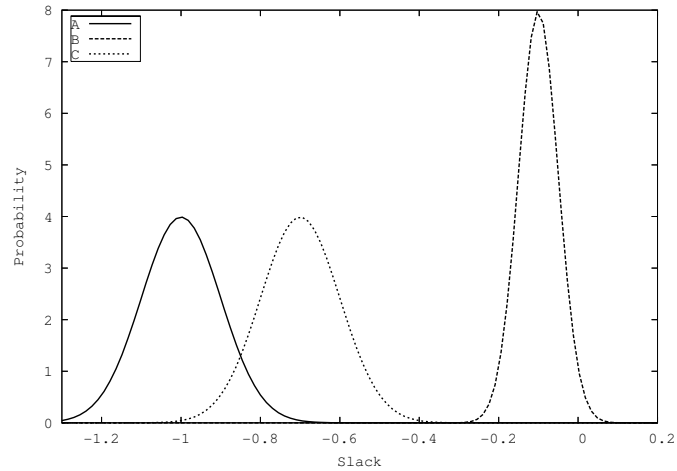

Fig. 3. Three uncorrelated PDFs to illustrate how the slack magnitude effects criticality.

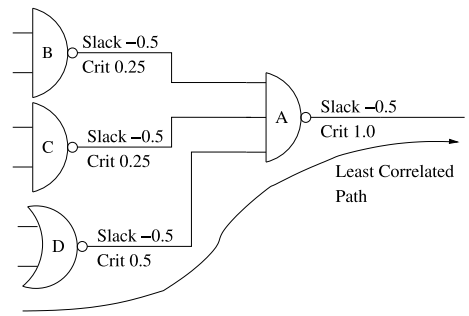

Fig. 4. Criticality probability and slack as selection metrics.

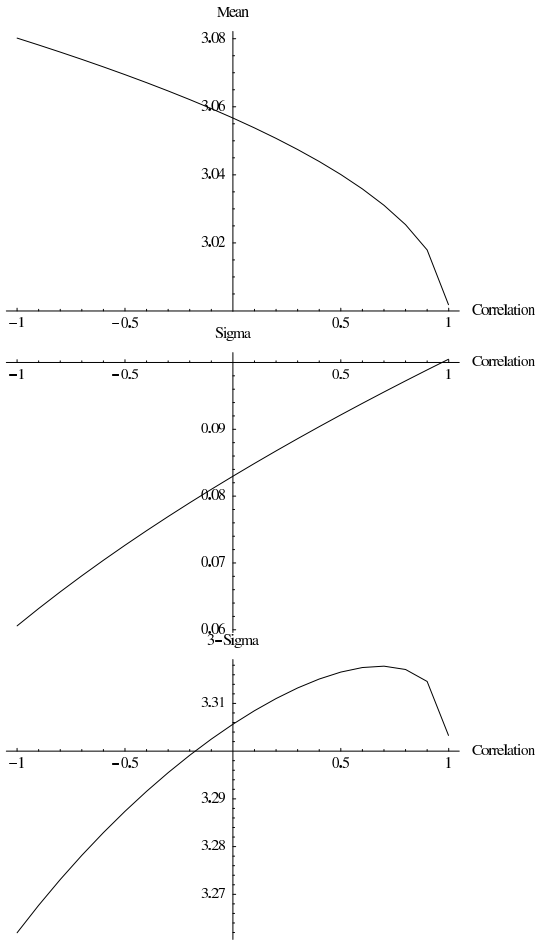

Fig. 5. The effect of correlation on the (a) mean (μ), (b) standard deviation (σ), and (c) $\mu + 3\sigma$ of the maximum of two random variables.

signals A and B. Figure 5 shows the mean and standard deviation of the statistical maximum of the previous two paths for different correlation coefficients. There is a penalty when the sub-critical path is not fully correlated because the mean of the maximum of two non-perfectly correlated variables is worse than the maximum of two perfectly correlated variables (see Figure 5a). There is a sharp increase in the mean as random variables become slightly uncorrelated ($\rho < 1.0$) while the standard deviation has an approximately linear decrease in Figure 5b. The resulting effect on the $\mu + 3\sigma$ point is shown in Figure 5c. This phenomenon can increase the $\mu + 3\sigma$ delay at the output of a critical path gate and demonstrates that sub-critical paths must be considered. By speeding up the sub-critical path so that the resulting delay distributions overlap less, one can minimize the contribution of these paths to the resulting output distribution. Improvement could also be made by making the delays more (or much less) correlated. Our algorithm considers improving the side paths as we consider more candidate gates for up-sizing. However, the candidate list must be efficiently generated so that the run-time penalty is minimized. This problem is addressed in this section.

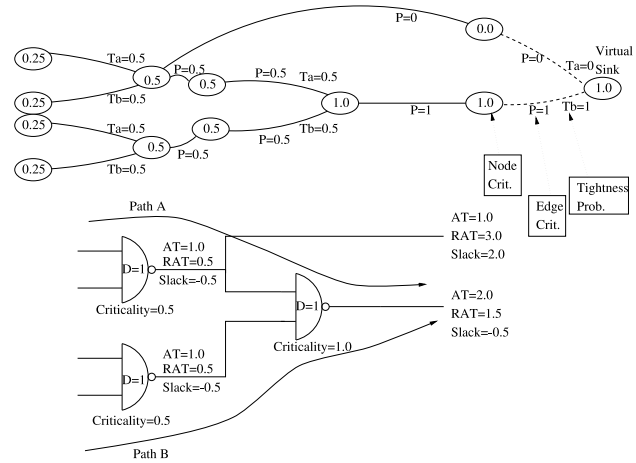

Fig. 6. Criticality probability computation example.

A. Criticality Probability

After propagating statistical AT and RAT distributions in block-based SSTA, each edge retains the tightness probabilities as they merge into their respective sink timing points. Many paths in a circuit can have some non-zero probability of being the critical path when using SSTA (i.e., they are critical in some portion of the process space). Other paths have zero probability due to being dominated by other paths. Given the tightness probabilities that include the effects of correlation, the global node and edge criticalities can be calculated with a backward breadth-first calculation as explained in [3]. Consider the timing graph in the top of Figure 6. Each pin-to-pin arc of every gate and net is represented by an edge in the graph. An *edge criticality* is the probability that a timing arc is on the critical path. A *node criticality* is the probability that a pin of a gate or net is on the critical path. The virtual sink is given a probability of 1 since it is the sink of all paths and is always on the critical path. An edge criticality is formally defined as the product of its output node criticality and the edge's tightness probability,

$$P_e = T_e P_{n,sink}. \tag{3}$$

Each node has a criticality that is the sum of all outgoing edge criticalities,

$$P_n = \sum_{e \in FO} P_e. \tag{4}$$

The probability of a gate being on a critical path is given by the sum of all of its edge criticalities,

$$P_{gate} = \sum_{e \in gate} P_e = P_{outnode}. \tag{5}$$

Since the sum of the tightness probabilities entering a node is equal to one, the probability of the gate is equal to the output node criticality assuming a single output. The tightness probabilities are calculated while considering correlation, but the criticality probabilities are assumed to be independent during criticality computation.

B. Candidate Selection

The metric for candidate generation that can efficiently select candidate gates for statistical gate sizing is the *criticality probability* as described in Section VI-A. The criticality probability is desirable as a candidate selection metric because it considers the slack magnitude, topology, and slack correlation while statistical slack in Section V and III only considers the magnitude of the slack.

Algorithm 2 shows the entire candidate selection algorithm that replaces line 3 in Algorithm 1. The algorithm first updates the criticality probabilities and then considers each gate in the design as a potential candidate. If the criticality-probability slack is one of the k largest seen, we add it to the candidate list. If the candidate list exceeds the k gate limit, the worst item is removed. Algorithm 2 requires linear time, $O(n)$, to compute the criticality probabilities where n is the number of gates. During a single pass of all gates, the algorithm can keep track of the best candidates in a priority queue implementation of the candidate list. Whenever a new largest element is found, the smallest element can be removed in $O(\log k)$ where k candidates are saved. The worst case requires that all but the final candidates be inserted and removed from the queue. Therefore, the overall complexity of Algorithm 2 is $O(n \log k)$. If k is constant, this reduces to $O(n)$.

Algorithm 2 Improved candidate generation algorithm.

```
Calculate criticality probabilities.
Reset candidate list, Γ
for gates ∈ entire circuit do
    if P_gate > Min(Γ) or Size(Γ) < k then
        Add to candidate list, Γ
    end if
    if Size(Γ) > k then
        Remove minimum P_gate from candidate list, Γ
    end if
end for
```

C. Criticality-Guided Results

In this experiment, four variants of sizing are compared: nominal deterministic, WC deterministic, baseline statistical and criticality-guided statistical. All variants follow the form of Algorithm 1, but they differ on how candidate gates are identified in line 3. In each variant, we allow the optimizer to calculate sensitivities for k candidate gates. k is fixed at 15% of the number of gates in the design which should easily subsume the critical path[5].

In the deterministic sizing variants, we use the sum of negative slacks, instead of minimum slack, as the optimization figure-of-merit (FOM) to approximate the behavior of the statistical virtual sink. This gives DSTA a global view of all circuit outputs and prevents it from prematurely getting stuck at a local minimum. Deterministic sensitivities are calculated as improvement in FOM divided by the additional gate input capacitance.

Statistical sensitivities are computed as the improvement in 3σ slack at the virtual sink divided by the additional gate input capacitance. In the baseline statistical variant, statistical optimization uses the k most negative statistical slack gates in the same manner as the experiment of Section V. The criticality-guided variant is the improved algorithm that uses criticality probability information to select candidate gates for sensitivity analysis as described in Section VI-A.

The results of the four sizing variants are shown in Table II. The netlist topologies are taken from the ISCAS benchmarks and synthesized with an industrial logic synthesis tool. The final gate sizes are from a large 130nm industrial library (14 inverter sizes, 12 buffer sizes, 8 NAND2 sizes, etc.). As before, all final measurements are done with the same SSTA setup, but the deterministic cases use DSTA during optimization only. The statistical slacks are up to several hundred picoseconds better than WC deterministic optimization and improvements are seen in most cases. Sometimes the extra robustness requires additional power (e.g., C17, C432, C499, C880), but not always (e.g., C1908). Only one case (C1355) had a slack significantly worse than the WC optimization. The criticality probability is usually more efficient in guiding us to a final solution than the statistical slack approach which can reduce the area or the overall run-time of the algorithm. The improvement can be seen in a power-delay plot of Figure 7 where the criticality-guided algorithm clearly is more efficient than the baseline sigma-sampled algorithm. The run-time overhead compared to the deterministic cases is still typically $5-10\times$, but it should be noted that the WC deterministic case may miss the true WC corner completely. These experiments assume that the WC corner is known *a priori*. Interestingly, the nominal optimization sometimes obtains better results than the WC optimization (e.g., C17, C432, C3540) as observed in other works [19].

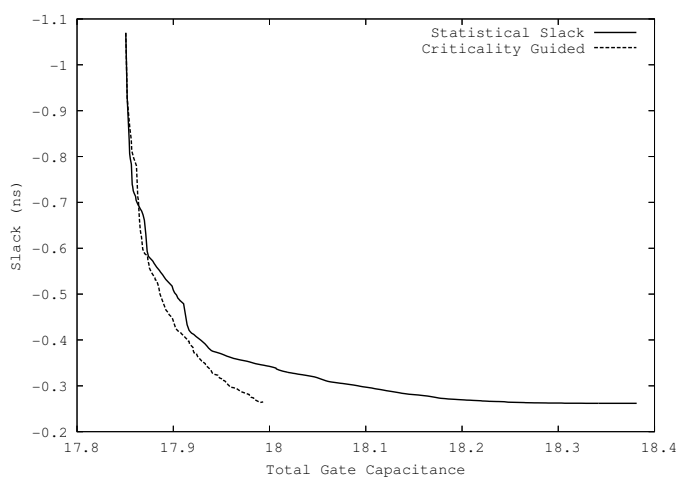

Fig. 7. Power-delay plot for C3540 with statistical slack and criticality probability guided statistical optimization.

[5]Making k a percentage of the total gate count no longer guarantees linear run-time, but run-time is not significantly effected in practice.

TABLE II

NOMINAL DETERMINISTIC (NOM), WC DETERMINISTIC (WC), STATISTICAL SLACK (SS) AND CRITICALITY PROBABILITY (CP) GUIDED STATISTICAL RESULTS (SLACK IN PS, CAP IN PF, AND TIME IN SEC).

Bench.	Mode	Slack	Cap	Time	ΔSlack
C17	WC	-269	1.438	0.0	-
	Nom	-225	1.464	0.0	44
	SS	-315	1.436	0.0	-46
	CP	-217	1.501	0.2	52
C432	WC	-104	9.686	23.0	-
	Nom	-77	9.758	27.2	27
	SS	-75	9.883	153.7	29
	CP	-1	10.177	182.6	103
C499	WC	-190	15.806	21.7	-
	Nom	-190	15.803	19.9	0
	SS	-153	16.015	237.3	37
	CP	-157	15.999	166.4	33
C880	WC	-289	18.476	13.7	-
	Nom	-289	18.476	12.5	0
	SS	-274	18.478	60.5	15
	CP	-268	18.505	41.6	21
C1355	WC	-60	18.139	421.1	-
	Nom	-65	17.912	366.1	-5
	SS	-91	17.710	1679.4	-31
	CP	-93	17.741	1379.2	-33
C1908	WC	-545	13.643	68.2	-
	Nom	-544	13.670	65.4	1
	SS	-544	13.640	331.6	1
	CP	-546	13.627	278.6	-1
C3540	WC	-282	18.295	200.2	-
	Nom	-279	18.240	188.9	3
	SS	-265	18.354	1022.5	17
	CP	-264	18.356	754.1	16
C5315	WC	-67	66.654	473.2	-
	Nom	-75	66.567	422.6	-8
	SS	-3	67.226	3801.2	64
	CP	-8	67.130	2474.8	59
C7552	WC	-50	68.898	367.8	-
	Nom	-78	68.663	184.5	-28
	SS	-38	68.900	1474.6	12
	CP	-39	68.846	1167.7	11

VII. CONCLUSIONS

Statistical optimization proves to be crucial to obtain adequate performance with a yield constraint. Worst-case DSTA can be both pessimistic and misleading for optimization. This produces sub-par results on many circuits. This work has shown that incremental, parameterized SSTA can be used during gate-sizing to optimize circuits while considering multiple sources of both correlated and uncorrelated variation. The improved accuracy of SSTA allows gate sizing to obtain superior results when compared to DSTA optimization. However, since the run-time of SSTA is larger than DSTA, the run-time of gate sizing algorithms can become exorbitant if not used wisely. Subsequently, the active use of statistical criticality probabilities are shown to guide gate sizing to better results than statistical optimization alone. This sometimes requires extra area/power, but is often compensated for with more efficient use of the area/power.

ACKNOWLEDGMENTS

The authors wish to thank Richard Brown, Dennis Sylvester, Ruchir Puri and David Kung for many insightful conversations on the subject matter.

REFERENCES

[1] A. Agarwal, V. Zolotov, and D. T. Blaauw, "Statistical timing analysis using bounds and selective enumeration," *TCAD*, vol. 22, no. 9, pp. 1243–1260, September 2003.

[2] J. A. G. Jess, K. Kalafala, S. R. Naidu, R. H. J. M. Otten, and C. Visweswariah, "Statistical timing for parametric yield prediction of integrated circuits," in *DAC*, June 2003, pp. 932–937.

[3] C. Visweswariah, K. Ravindran, K. Kalafala, S. G. Walker, and S. Narayan, "First-order incremental block-based statistical timing analysis," in *DAC*, 2004, pp. 331–336.

[4] H. Chang and S. Sapatnekar, "Statistical timing analysis considering spatial correlations using a single PERT-like traversal," in *ICCAD*, 2003, pp. 621–625.

[5] A. Davoodi, V. Khandelwal, and A. Srivastava, "Variability inspired implementation selection problem," in *ICCAD*, 2004, pp. 423–427.

[6] E. T. A. F. Jacobs and M. R. C. M. Berkelaar, "Gate sizing using a statistical delay model," in *DATE*, March 2000, pp. 283–290.

[7] A. Srivastava, D. Sylvester, and D. Blaauw, "Statistical optimization of leakage power considering process variations using dual-Vth and sizing," in *DAC*, June 2004, pp. 773–779.

[8] X. Bai, C. Visweswariah, P. N. Strenski, and D. J. Hathaway, "Uncertainty-aware circuit optimization," in *DAC*, June 2002, pp. 58–63.

[9] M. Mani and M. Orshansky, "A new statistical optimization algorithm for gate sizing," in *ICCD*, October 2004, pp. 272–277.

[10] S. Director, P. Feldmann, and K. Krishnan, "Statistical integrated circuit design," *JSSC*, vol. 28, no. 3, pp. 193–202, March 1993.

[11] I. Sutherland, B. Sproull, and D. Harris, *Logical Effort: Designing Fast CMOS Circuits*. Morgan Kaufmann Publisher, 1999.

[12] J. P. Fishburn and A. E. Dunlop, "TILOS: a posynomial programming approach to transistor sizing," in *ICCAD*, 1985, pp. 326–328.

[13] J. Shyu, A. Sangiovanni-Vincentelli, J. Fishburn, and A. Dunlop, "Optimization-based transistor sizing," *JSSC*, vol. 23, no. 2, April 1988.

[14] S. Sapatnekar, V. Rao, P. Vaidya, and S. Kang, "An exact solution to the transistor sizing problem for cmos circuits using convex optimization," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 12, no. 11, pp. 1621–1632, November 1993.

[15] C. E. Clark, "The greatest of a finite set of random variables," in *Operations Research*, March-April 1961, pp. 145–162.

[16] M. Cain, "The moment-generating function of the minimum of bivariate normal random variables," *The American Statistician*, vol. 48, no. 2, pp. 124–125, May 1994.

[17] C. Visweswariah, Dockets YOR9-2003-401 &YOR9-2003-402 & YOR9-2003-403, Filed with the U.S. Patent Office, September 2003.

[18] H. Chang, V. Zolotov, S. Narayan, and C. Visweswariah, "Parameterized block-based statistical timing analysis with non-gaussian parameters and non-linear delay functions," in *Tau*, 2005, pp. 99–104.

[19] M. R. Guthaus, N. Venkateswaran, V. Zolotov, D. Sylvester, and R. B. Brown, "Optimization objectives and models of variation for statistical gate sizing," in *GLSVLSI*, April 2005, pp. 312–316.

[20] A. Agarwal, D. Blaauw, and V. Zolotov, "Statistical timing analysis for intra-die process variations with spatial correlations," in *ICCAD*, 2003, pp. 900–907.