False-Path-Aware Statistical Timing Analysis and Efficient Path Selection for Delay Testing and Timing Validation*

Jing-Jia Liou, Angela Krstic, Li-C. Wang and Kwang-Ting Cheng
Electrical and Computer Engineering Department,
University of California, Santa Barbara,
{jjliou,angela,licwang,timcheng}@windcave.ece.ucsb.edu

ABSTRACT

We propose a false-path-aware statistical timing analysis framework. In our framework, cell as well as interconnect delays are assumed to be correlated random variables. Our tool can characterize statistical circuit delay distribution for the entire circuit and produce a set of true critical paths.

Categories and Subject Descriptors

B.8.2 [**Hardware**]: Performance Analysis and Design Aids

General Terms

Algorithm, Performance, Reliability

Keywords

Critical path selection, false path, statistical timing analysis

1. INTRODUCTION

With the continuous shrinking of VLSI devices, accurate prediction of circuit performance is becoming an increasingly difficult problem. Factors such as process variations, noise sources, modeling errors can significantly affect the performance of deep-submicron designs, and these factors are all statistical in nature. Therefore, the use of statistical methods for timing analysis [10, 11] seems to be inevitable in the near future. The use of statistical models is equally important for path selection tools for maximizing the probability of covering critical paths and for detecting defects caused by delay disturbances [1, 2, 3]. These models allow considering various sources of delay variations and can result in a more realistic set of critical paths for delay testing and timing validation. The importance of including power supply noise induced delay variations into the statistical timing models and into the critical path selection process has been demonstrated in [4].

Even though the potential benefits of using statistical models are clear, a significant effort is required to develop models, tools and algorithms for statistical analysis. One of the main difficulties is to include false path analysis. The traditional timing analysis methodologies with false path analysis [5, 6, 7, 8, 9] are not easily extendable into the statistical domain. This is because, with statistical timing models, a path can be true in some circuit instances and false in other instances. By definition, *false paths* are paths that cannot be sensitized under any input vector pair and hence including false paths in circuit timing calculation may lead to unrealistic results. The false paths can also result in waste of engineering resources during the path selection process. In this paper, we present an efficient false-path-aware statistical framework that can minimize the negative effects of false paths on circuit performance estimation and critical path selection for delay testing and timing validation.

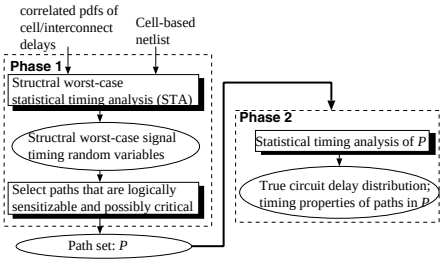

Figure 1: False-path-aware statistical timing analysis.

Given the correlated delay pdfs (pin-pin cell or interconnect delays), our goal is to estimate the true circuit delay distribution and deliver a set of *logically sensitizable* (or simply "sensitizable"), *timingly critical* (or simply "critical"), and *timingly true* (or simply "true") paths. We differentiate between a path being sensitizable and being true because the former considers only the functional (logic) criteria for sensitizing a path while the later considers both logic and timing (statistical timing) criteria for a path to be a long path. For each path, our tool can report its probability of being critical, being true as well as being true *and* critical. This information can be used to improve path selection for timing optimization and delay testing. It also allows ranking the selected paths according to their critical and true probabilities.

2. THE FRAMEWORK

The complete flow of our false-path-aware statistical timimg analysis and path selection framework is shown in Figure 1. It consists of two phases. In the first phase, the objective is to select all logically sensitizable long paths. Worst-case statistical timing information is used to select the long paths [1]. In the second phase, we process the paths in P to obtain their true timing information. With this information, we can determine the circuit delay distribution and further eliminate timingly false paths The outputs of our framework are the true signal timings of the refined set P and the circuit delay distribution.

2.1 First Phase: Identifying Logically Sensitizable Critical Paths

Critical paths are defined as paths whose delays exceed a certain percentage of the longest path delay (or exceed a predefined

*This work was supported in part by the MARCO/DARPA Gigascale Silicon Research Center (http://www.gigascale.org). Their support is gratefully acknowledged.

DAC 2002, June 10-14, 2002, New Orleans, Louisiana, USA.

cutoff period). Although there are several path selection tools [13, 14] available, these methods are based upon nominal or worst-case cell/interconnect delays and cannot capture the statistical factors. As a result, they may fail to select the right paths for devices whose cells/interconnects are subject to delay variations.

To find the critical paths for statistical timing models, we first find the critical nodes. In the beginning, our statistical timing analysis produces the worst-case signal timing random variables including arrival times, required times and slacks (Figure 1). The slack is used to determine if the node is critical or not. Since in the statistical framework, the arrival times, required times and slacks are all random variables, a node could be critical in some circuit instances but non-critical in others. Actually, the area of the slack pdf in the negative region is the probability of the node being a critical node. If the probability of being critical is 1 (with a certain confidence level), this node has a negative slack. It means for all sampled circuit instances, this node is critical and hence it has to be considered when selecting structurally long paths. On the other hand, if the probability is 0, the node has a positive slack. There is also a mixed-type node whose probability can be a non-zero value (other than 1). Mixed-type nodes with slacks of higher critical probabilities than the threshold set by the user will be included in the path selection. Once slack distributions are produced by the structurally worst-case statistical timing analysis tool, we can construct paths from critical nodes such that all nodes on the paths have non-zero probabilities of being critical. The above analysis finds all structurally long paths in the statistical domain. Next, we want to find the logically true paths by checking the functional sensitizability condition [12] for these paths. The outcomes of the first phase are the set of logically sensitizable and structurally long (critical) paths. However, a logically sensitizable path might not be sensitizable in the timing sense, i.e., no vector can sensitize this path such that it always dominates all on-path cell's timing behavior. Furthermore, the true delay of a structurally long path might not be critical because the path is constructed from the worst-case slack distributions. It is the responsibility for the second phase to handle these two problems. In the following, we will discuss the implementation details of the first phase process.

2.1.1 Statistical Timing Analysis

The statistical timing analysis is the core engine of our framework.We adopt a cell-based approach in building our framework. It requires pre-characterization of cells, i.e., extracting random variables for cell delays and output transition times. The input transition time and output loading of the cells are used as indices for building these libraries. Since the goal of statistical timing analysis is to describe the timing behavior of the circuit regardless of the applied input patterns, we use the worst-case analysis in building the cell libraries. The interconnect delay is also modeled as a random variable and is pre-characterized once the RCs are extracted. The random variables of the signal arrival times at cell/interconnect outputs are computed using the information on the arrival and transition times of the cell fanins as well as the information on the cell/interconnect delays [15]. Our framework uses Monte-Carlo-based techniques to approximate the pdfs of the signal arrival times for all internal signals and primary outputs. The convergence criteria are decided based on desired accuracy of results.

2.1.2 Handling Correlations

In Monte Carlo sampling, we need to deal with the fact that the cell/interconnect delay random variables are strongly correlated. Our technique [1] to support correlations is to partition circuit into several groups. We assume a common correlation factor among random variables of each group. Our method is to add an independent random variable into the group of variables which share the same correlation factor r. In the following, we use the case of two random variables as an example. Let $X = A + t_1 C$ and $Y = B + t_2 C$, where A, B, C (C is standard normal pdf) are independent random variables. It can be proved that $t_1 = \sqrt{r}\sqrt{X^2}$ and $t_2 = \sqrt{r}\sqrt{Y^2}$. In the sampling process, we can simply sample values from A, B and C (using sampling techniques for independent random variables) and hence a sample of X and Y can be obtained.

```
SearchTrueCriticalPaths()
    for each PO with a falling/rising transition
        if (the prob. of the negative area of the PO slack > threshold
            push the gate connected to PO into the stack
        while (stack is not empty)
            pop a gate from the stack
            if (all fanins of the gate have been searched)
                Backtrack to the previous gate
            else // Extend a partial path
                bool Successful = ExtendBySensitization(gate);
                if (Successful) Successful=Implications();
                if (Successful)
                    if (the current gate is a PI)
                        Print out the current found path
                        and its probability of being true and critical
                        Backtrack to the previous gate
                    else
                        for each fanin of the current gate:
                            if (the negative area of the slack > threshold)
                                push this fanin into the stack
                else // Extend a partial path unsuccessfully
                    Backtrack to the previous gate
```

Figure 2: Identifying logically true and critical paths.

2.1.3 Selection Algorithm for Logically True Paths

In this algorithm, The circuit is processed in a depth-first manner from POs towards PIs level-by-level. At each level, the current partial path is extended (if contains mixed-type slacks) and the sensitization criteria (functional sensitization [12]) are checked. If the criteria are satisfied, the process continues by further extending the current partial path. If they are not satisfied, it means that none of the paths that include the current partial path are true and the algorithm backtracks to the previous level. The process, then, continues by trying a different partial path. Once a primary input has been reached, a sensitizable and structurally true path is found. The pseudo code for our path search algorithm is shown in Figure 2.

2.2 Second Phase: Identifying Timingly True Critical Paths

A path is defined as a timingly true path for a circuit instance if the transitions on the path cannot be invalidated by any other off-input paths. In the statistical environment, the status of timing conditions could be different from instance to instance. Therefore, a path could be a true path in some instances and a false path in others. Such a path would have a probability of being true between 0 and 1. Then the goal of this phase is to derive various probablility values by performing statistical analysis on P set (Figure 1): (1) the probability of a path being critical, i.e., the critical probability, p_c, (2) the probability of a path being true (functionally and timingly sensitizable), i.e., true probability, p_t and (3) the probability of a path being true *and* critical, p_{tc}. One possible method for timing analysis in Phase 2 is to modify the statistical analysis method in [16] to obtain the signal timing distributions (in the paper the analysis is used to obtain diagnostic infomation). However, probability values, p_t and p_{tc}, are not available in this scheme.

It is often useful to know what are the most critical true paths among the selected paths, i.e., orders of long paths. Traditionally, long paths are defined according to a fixed path delay (one circuit instance). In our framework, the probability values p_c, p_t and p_{tc}

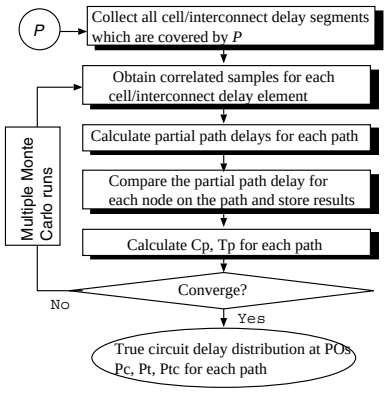

Figure 3: Statistical analysis of P.

can be used to rank the selected paths among themselves in a statistical sense. The longest path in the statistical domain can be defined as a path with the largest p_c and nonzero p_t. That means that it is statistically a true longest path. In the case that there are paths with the same p_c, we propose to use p_{tc} for further ordering of the paths. Choosing a path with higher p_{tc} will yield a higher probability to sensitize a critical path.

2.2.1 *Statistical Analysis of P*

Details of the second phase algorithm is illustrated with the flow chart in Figure 3. It is basically a Monte Carlo sampling technique. In the beginning, the procedure collects all covered segments (cell and interconnect delay elements) in P. This is to avoid getting two samples for the same segment in the Monte Carlo run and also help to maintain the correlations between segments using techniques described in section *2.1.2*. Maintaining the correlations is particularly important because the relative comparison of arrival times will be incorrect if the correlations are not kept during calculation.

For a circuit instance, the delay of partial paths in P that converges at a node will be compared and both the minimum and maximum of the partial delays will be stored for each node. Also, we can obtain a true circuit delay sample using the same technique for POs. It is to compare and store all complete path delays in P. By the method described in the next section, all true timing properties (p_c,p_t and p_{tc}) can also be calculated.

2.2.2 *Calculation of Probability Values*

First, we define two events C_P and T_P for a circuit instance. The C_P event of which a path is critical is defined as $C_P = \{\sum D_i - \textit{cutoff time}\}$, where i is the index running over all delay elements on P, and D_i is a sample of the distribution of the delay element i. The event of T_P of which a path being true for an instance is also defined as follows. Assume that a and b are inputs to gate g in path P and both a and b have transitions. And a is the on-path input (b is a side-input to g). T_P is the event that all gates on P satisfy each of the following condition:

$T_g = \{\sum D_i < \textit{max_arrival_time}(b)\}$if a is a controlling input,

$T_g = \{\sum D_i > \textit{min_arrival_time}(b)\}$if a is a non-controlling input.

where i is the index running over all delay elements of the partial path from PI to g on P. For gates that have more than two inputs, this method is applied for each side-input. Counting these event occurrence for paths, we can derive the probability values for a Monte Carlo sampling scheme. Note that for the event of a path being true and critical is the conjuncture of the two events: $\{C_P, T_P\}$.

3. EXPERIMENTAL RESULTS

As described in the section 2, the framework (Figure 1) is a cell-based statistical timing analysis. It requires pre-characterization of

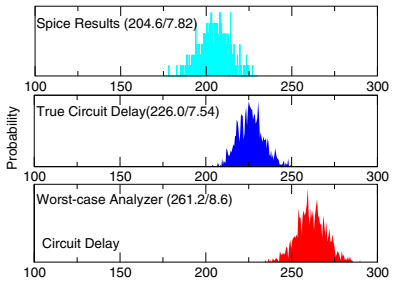

Figure 4: Delay distributions.

cells. We use a Monte-Carlo-based SPICE (ELDO) [17] to extract cell delays for a 0.25μm, 2.5V CMOS technology. The input transition time and output loading of the cells are used as indices for building these libraries. The interconnect delay is also modeled as a random variable. We perform 1000 runs of Monte Carlo analysis for each phase.

Figure 4 demonstrates the results of our algorithm comparing to a Monte-Carlo-based SPICE on a benchmark circuit, s1196. In the Figure, the random variables are expressed as mean/σ, e.g., 204.6 is the mean and 7.82 is the σ for the spice results. The true circuit delay distribution obtained by our algorithm (the second frame) represents a better statistical upper bounds of the results of the SPICE. On the other hand, the results of the statistical worst case analysis (third frame) give a pessimistic estimation of the circuit performance. It shows that our algorithm has a better estimation for this circuit than the traditional statistical timing analysis tools which use only structural information. Note also that typical static timing analysis tool will predict the circuit delays beyond the 3σ bound (261+3*8.6=286.8) of the structural statistical worst case in the figure because these tools only use fixed worst-case cell/interconnect delays. Even enhancing traditional tools with false path pruning, the best figure can be obtained is still beyond the 3σ bound (226+7.54*3=248.6) of our "True Circuit Delay". Moreover, traditional timing analysis tools suffer from inability to target the correct paths [2, 4]. Still the results in Figure 4 is about 12% behide the SPICE results. It is because "static" analysis is applied and thus only "worst-case" random variables for cells can be used. If test vectors are generated for select paths and dynamic simulation is performed, it is possible to match SPICE results more closely.

Circuit delays estimated by our algorithm and traditional worst-case analysis under different correlation assumptions for a set of ISCAS89 circuits are shown in Table 1. There is notable reduction of the estimated circuit delays for most circuits, especially for s1196 and s38417. The mean of the true circuit delay for s5378 is slightly larger than the one obtained by structural worst-case analysis, but on the other hand the σ is smaller. This means that we still have better bound estimation for the circuit delay. It is interesting to note that when correlation factors increase, the standard deviations (σ) increase accordingly while the means decreases. This kind of analysis and comparison is only available by our new algorithm.

From Table 1, one interesting observation can be made: some σ values of true circuit delay are significantly smaller than those of structural worst-case, especially when assuming independent delay random variables. For s38417, σ=0.9 in the case of true circuit delay is smaller than σ=2.1 for the structural worst case. Further analysis reveals that the true circuit delay in this example is actually the collection distribution of 899 paths. All 899 paths have non-zero probabilities of becoming the longest path and they are almost equally long. On the other hand, for structural worst case the circuit delay is only the composite of 64 paths. A larger number of equally long paths means a higher probability for circuit delay to concentrate in a smaller range and hence, a smaller σ.

Table 1: Comparison of random variables for circuit delays with three different correlation factors.

	independent cell/interconnect delays				cell/interconnect delays with a correlation factor of 0.5				cell/interconnect delays with a correlation factor of 0.9			
	true circuit delay		structural worst-case circuit delay		true circuit delay		structural worst-case circuit delay		true circuit delay		structural worst-case circuit delay	
circuits	mean	σ	mean	σ	mean	σ	mean	σ	mean	σ	mean	σ
s1196	226.1	2.7	261.6	2.8	226.0	5.7	261.3	6.7	226.0	7.5	261.2	8.6
s5378	242.8	1.8	242.3	2.2	241.0	7.3	241.3	7.1	239.2	9.0	239.7	9.3
s9234	180.4	1.1	179.4	1.8	178.2	4.6	179.1	4.3	178.3	5.9	179.0	5.6
s13207	482.0	2.6	483.4	2.5	481.3	11.2	482.4	11.7	480.2	14.9	481.3	15.6
s15850	372.3	1.8	374.2	2.4	370.7	7.7	373.6	7.8	370.0	10.4	373.0	10.3
s35932	193.5	0.8	209.3	0.6	192.7	4.2	208.1	4.6	190.7	5.6	206.3	6.2
s38417	299.0	0.9	363.7	2.1	297.3	7.1	363.4	8.5	296.0	10.0	362.7	11.3
s38584	559.2	4.2	566.7	4.4	559.2	12.6	566.6	12.6	558.7	16.1	566.6	16.4

Our framework (Figure 1) can also be used to select paths for performance validation and delay testing. It will output comprehensive timing information associated with each selected path. We compiled the number of selected paths in Table 2 for the outputs of both phases in our framework. In the table, the third column (”logically sensitizable path”) is the number of paths (P in Figure 1) after the first path selection phase. This is the number of paths which are functionally sensitizable and possibly critical. And the fourth column (”true and critical path”) is the number of paths after the complete process is finished. This is the number of paths that satisfy the following timing conditions: $p_{tc}(P) > 0$ for each path in P. It means that these paths are both timingly true and critical with respect to the cutoff period. We also listed in the second column (”structurally long paths”) the number of paths which will be outputed at the end of the first phase in Figure 1 without any functional analysis to eliminate functionally unsensitizable paths, i.e., they are only possibly critical paths. As it can be seen, while the number of the logically sensitizable paths (column 3) is only a fraction of the total possibly critical paths (column 2), the number of the true critical path (column 4) is even smaller. This emphasizes the importance of eliminating the false paths. This table also shows that our algorithm runs efficiently for the benchmark circuits.

4. CONCLUSIONS

We have proposed an efficient statistical timing analysis algorithm to analyze the true timing behavior of a circuit and select true critical paths based on statistical delay modeling. Our framework will provide a better estimation of the circuit delay by considering the statistical cell/interconnect delay models and also the correlations between the pdfs of the delay elements. Furthermore, for applications of path selection the framework can eliminate false paths which are functionally unsensitizable and/or timingly invalidated by other paths. The experimental results show that there is significant reduction of paths if these false paths are eliminated. This tool also provides the capability to order paths according to their probabilities of being true and critical. For deep submicron designs with large variations of different circuit parameters, this information is crucial for circuit optimization, test generation for delay faults and dynamic timing simulation.

Table 2: The number of selected paths by different methods and CPU/MEM consumption of our algorithm.

circuits	# structurally long paths	# logically sensitizable path	# true and critical path	cutoff time	CPU time (s)	MEM (Mbytes)
s1196	534	255	98	200	12.7	2.8
s5378	144	144	114	240	11.8	6.8
s9234	14	7	5	179	8.8	5.4
s13207	3571	926	221	470	55.9	19.8
s15850	8730	1728	281	370	96.9	11
s35932	56339	1664	576	190	207	39.5
s38417	29622	6506	1235	298	181	50
s38584	110	97	66	500	102	41

Note that the CPU time includes the time for both the path selection and and analysis phase. The benchmarks are done on a PentiumIII 733MHz with 256M RAM.

5. REFERENCES

[1] J.-J. Liou, A. Krstić, K.-T. Cheng, D. Mukherjee, and S. Kundu. Performance Sensitivity Analysis Using Statistical Methods and Its Applications to Delay Testing. *Proc. ASP-DAC*, pp 587–592, Jan 2000.

[2] J.-J. Liou, K.-T. Cheng, and D. Mukherjee. Path Selection for Delay Testing of Deep Sub-Micron Devices Using Statistical Performance Sensitivity Analysis. *Proc. of VTS*, pp 97–104, Apr 2000.

[3] J.-J. Liou, K.-T. Cheng, S. Kundu, and A. Krstic. Fast Statistical Timing Analysis by Probabilistic Event Propagation. *Proc. of DAC*, Jun 2001.

[4] J.-J. Liou, A. Krstić, Y.-M. Jiang, and K.-T. Cheng. Path selection and pattern generation for dynamic timing analysis considering power supply noise effects. *Proc. of ICCAD*, pp 493–496, Nov 2000.

[5] D. Brand and V. Iyengar. Timing Analysis Using Functional Analysis. *IBM Thomas J. Watson Center, Technical Report*, 1986.

[6] P. McGeer and R. Brayton. Efficient Algorithms for Computing The Longest Viable Path in A Combinational Network. *Proc. of DAC*, pp 561–567, Jun 1989.

[7] S. Devadas, K. Keutzer, and S. Malik. Computation of Floating Mode Delay in Combinational Circuits: Theory and Algorithms. *IEEE Trans. on CAD*, 12(12):1913–1923, Dec 1993.

[8] H.-C. Chen and D. H. C. Du. Path Sensitization in Critical Path Problem. *IEEE Trans. on CAD*, 12(2):196–207, Feb 1993.

[9] H. Chang and J. A. Abraham. An Efficient Critical Path Tracing Algorithm for Designing High Performance VLSI Systems. *JETTA*, 11:119–129, 1997.

[10] D. R. Tryon, F. M. Armstrong, and M. R. Reiter. Statistical Failure Analysis of System Timimg. *IBM Journal of Research and Development*, 28(4):340–355, Jul 1984.

[11] H.-F. Jyu, S. Malik, S. Devadas, and K. Keutzer. Statistical Timing Analysis of Combinational Logic Circuits. *IEEE Trans. on VLSI*, 1(2):126–137, Jun 1993.

[12] A. Krstić and K.-T. Cheng. *Delay Fault Testing for VLSI Circuits*. Kluwer Academic Publishers, Boston, MA, 1998.

[13] W.-N. Li, S. M. Reddy, and S. K. Sahni. Long and Short Covering Edges in Combinational Logic Circuits. *IEEE Trans. on CAD*, 9(12):1245–1253, Dec 1990.

[14] S. Tani, M. Teramoto, T. Fukazawa, and K. Matsuhiro. Efficient Path Selection for Delay Testing Based on Partial Path Evaluation. *Proc. of VTS*, pp 188–193, May 1998.

[15] H. Edamatsu, K. Homma, M. Kakimoto, Y. Koike, and K. Tabuchi. Pre-Layout Delay Calculation Specification for CMOS ASIC Libraries. *Proc. of ASP-DAC*, pp 241–248, Feb 1998.

[16] M. Sivaraman and A. J. Strojwas. Path delay fault diagnosis and coverage-a metric and an estimation technique. *IEEE Trans. on CAD*, 20(3):440–457, Mar 2001.

[17] Anacad. Eldo v4.4.x User’s Manual. 1996.