Regression Simulation: Applying Path-based Learning
In Delay Test and Post-Silicon Validation

Li-C. Wang
Department of ECE, UC-Santa Barbara, USA
licwang@ece.ucsb.edu

Abstract
This paper presents a novel path-based learning methodology to achieve timing Regression Simulation. The methodology can be applied for two purposes: (1) In pre-silicon phase, regression simulation can be used to produce a fast and approximate timing simulator to avoid the high cost associated with statistical timing simulation. (2) In post-silicon phase, regression simulation can be used as a vehicle to deduce critical paths from the pass/fail behavior observed on the test chips. Our path-based learning methodology consists of four major components: a delay test pattern set, a logic simulator, a set of selected paths as the basis for learning, and a machine learner. We summarize the key concepts in our regression simulation approach and present experimental results.

1. Introduction
With the advance to nanometer technologies (<130nm), circuit timing reflects many important sources of effects such as process variations, power noise, crosstalk, small defects, thermal effects, etc. [1, 2]. These effects are hard to predict and model deterministically. For these effects, traditional discrete-value timing models can be ineffective. Statistical timing analysis and timing simulation approaches are among the many that promise to better handle these deep sub-micron (DSM) timing effects for delay testing [3]. This work was motivated by two fundamental issues in the development of a simulation methodology for delay testing: (1) In a pre-silicon design environment, timing models may not be correct and 100% complete. Without an accurate timing model, results from timing analysis and simulation may be misleading. (2) Even with a reasonably accurate timing model, the timing models for DSM effects can be quite complex, resulting in high timing simulation cost.

Our objective was to provide an alternative methodology that could complement the existing statistical timing analysis and simulation approaches proposed so far. The core idea of this work is to utilize machine learning techniques [4, 5] to accomplish path-based learning where timing behavior and timing information can be learned either from an accurate but slow statistical timing simulator, or from the behavior of a collection of test chips.

2. Path-based learning scheme
In a typical Machine (statistical) Learning problem, we are given a collection of samples, each of the form \((X, y)\) where \(X = [x_1, x_2, \ldots, x_n]\). "\(n\)" is called the dimension, and \((X, y)\) is called a sample point (or a training sample). The relationship between \(X\) and \(y\) is through an unknown function \(f\) such that \(y = f(X)\). The job of learning is to learn from a given \(m\) sample points: \((X_1, y_1), (X_2, y_2), \ldots, (X_m, y_m)\) in order to statistically deduce an estimation \(f_{est}\) for \(f\). This is also called Supervised Learning [4].

In Classification, \(f(X) \in G\) where \(G\) is a set of finite elements. In recent years, Support Vector Machine (SVM) has been demonstrated as a powerful learning technique (classifier) for problems whose dimensions are very large [4, 5].

In the pre-silicon phase, we assume that a slow and reasonably accurate timing simulator is available. In our work, we use a statistical timing simulator developed in the past [7]. Given a set of paths and a set of training patterns, the goal is to learn from the behavior of the timing simulator to derive a regression simulator that can produce approximate results as the timing simulator for the same pattern or other pattern sets. In the post-silicon phase, we assume that a set of test chips is available. By testing them with a pattern set on a given test clock, we can obtain their pass/fail behavior. Then, given a set of potential critical paths, our goal is to deduce the most important ones that are sufficient to explain the pass/fail behavior. In this process, machine learning is treated as an explanation tool. This problem is known as feature reduction in machine learning literature [6].
Our path-based learning scheme has two phases. In the
Training Phase, a set of \( m \) sample points are created based
on a given training set of \( m \) patterns. The \( Y_i \) values
correspond to the probabilities of failing a test clock. \( X_j = \left[x_{j1}, \ldots, x_{jm}\right] \) where each \( x_{jk} \) indicates if it is possible for an
UR path \( k \) to decide the delay of pattern \( j \). Since we utilize
SVM [9] as a classifier, we treat probabilities falling into
the range \([0.5, 0.5(1 + 1)]\) as the same class. Hence, \( Y_j = 3 \) for
a pattern whose failing probability is 0.34. The UR path set
can be derived from our statistical timing analyzer (SAT) by
given a cut-off clock [8]. Then, in the Evaluation Phase, the
regression simulator relies on three components to approximate
the desired answers: the set of UR paths, the logic simulator
to decide how a pattern sensitizes UR paths, and the
SVM learned model. If the objective is to approximate the
statistical timing simulator, then the evaluation pattern set
can be different. If the objective is to explain the failing behavior of the test chips, the pattern set stays the same.

3. Summary of results

Table 1 shows the accuracy results by comparing the re-
gression simulator to the statistical timing simulation in the
evaluation phase. \( T_1 \) are path delay patterns. \( T_2 \) are patterns
for transition faults through their longest propagation paths.
\( TR_{15} \) and \( TR_{10} \) are 15-detection and 10-detection transition
fault patterns produced by a commercial ATPG tool. The
"STA" column denotes the average worst-case delays pro-
duced by the statistical timing analysis [8] for constructing the UR path sets. The "clock" column shows test clocks to derive the failing probabilities. In these experiments, we intentionally used, in the timing analysis, a timing model different from the one used in the statistical timing simulator (up to 15% difference on each pin-to-pin delay). We note that in the evaluation phase, the regression simulator usually can run about 1000X faster than the statistical simulator. In other words, it can be a fast and approximate simulator for the statistical timing simulator with high accuracy.

Figure 3 shows results by using path-based learning as a feature reduction tool [6]. The "reduced UR path set" contains patterns that are critical in SVM learning for deriving its statistical learned model. In other words, other paths do not provide useful statistical information. As it can be seen, although statistical timing analysis may give a large path set (with a smaller clock), the number of useful paths to explain

the failing results in our learning scheme does not grow much. If we allow 4% of errors, the learning process can identify 137 useful paths. In other words, these are 137 statistically significant paths sufficient to explain the observed results to the desired accuracy. Due to space limitation, we omit showing similar results for other examples.

References

VLSI Test Symposium, April 1997.
[4] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. The Elements of Sta-
tistical Learning - Data Mining, Inference, and Prediction. Springer Series in
Statistics, 2001
[5] Nello Cristianini, John Shawe-Taylor. An Introduction to Support Vector Mo-
tine and other kernel-induced-based learning methods. Cambridge Univer-
sity Press, 2002
issue on relevance, 97, pp 273-324, 1995.
[7] Angela Krstic, Li-C. Wang, Kwang-Ting Cheng, Jing-Jia Liou, Magdy S.
Abadir. Delay Defect Diagnosis Based Upon Statistical Timing Models –
Timing Analysis and Efficient Path Selection for Delay Testing and Tim-
ing Validation. in Proc. DAC, June 2002.