

ERIF Mid-Term Report

Analysis and Optimization of Sequential Circuits under Process Variations

Jia Wang, Dept. of ECE, IIT

1 Summary of Progresses

The proposed approach for the statistical static timing analysis (SSTA) problem of latch-based sequential circuits under process variations was investigated. The proposed statistical positive cycle detection algorithm is developed in Section 2. Practical implementation considerations are presented in Section 3. Experimental results are shown in Section 4.

Based on the current progress, the future research work will be focused on risk aversion statistical optimizations as proposed in the original proposal.

2 Statistical Positive Cycle Detection Through Structural Graph Traversal

According to the algorithmic idea presented in the original proposal, the following SGT-NP algorithm as shown in Fig. 1 is designed for the statistical positive cycle detection problem.

| Algorithm SGT-NP | |
|-------------------------|---|
| Inputs | $G = (V, E)$: latch timing graph. d : edge delays. |
| Outputs | Y : probability that G has no positive cycle. |
| 1 | $V^* \leftarrow V$ |
| 2 | While G is not empty: |
| 3 | Get a vertex u from V |
| 4 | $\forall v \in V, D_{uv}^1 \leftarrow \begin{cases} d_{uv}, & (u, v) \in E \\ -\infty, & (u, v) \notin E \end{cases}$ |
| 5 | For k from 2 to $ V + 1$: |
| 6 | $D_{uv}^k \leftarrow \max_{(w,v) \in E} (D_{uw}^{k-1} + d_{wv})$ |
| 7 | $D(u) \leftarrow \max_{1 \leq k \leq V +1} D_{uu}^k$ |
| 8 | Remove u from G |
| 9 | $Y \leftarrow Pr\{\max_{u \in V^*} D(u) \leq 0\}$ |

Figure 1: The SGT-NP algorithm.

Given a latch timing graph G and the edge delays d , the SGT-NP algorithm computes the probability Y that G has no positive cycle. Note that since the edge delays d are random numbers, all the sum and maximum operations, i.e. on line 6, 7, and 9, are

performed statistically as in [1].

3 Practical Implementation Considerations

3.1 Graph Decomposition

According to the SGT-NP algorithm, only paths traveling back to the starting points, i.e. D_{uu}^k , contribute to the probability Y . Therefore, if a path will not travel back to its starting point, it can be excluded from the computation to speed-up the algorithm practically without affecting the correctness of the algorithm.

Since a path will travel back to its starting point if and only if all the vertices along the path belong to a strongly connected component (SCC), we propose to decompose the graph into SCCs before performing traversal. Therefore, paths that will not travel back to its starting point are eliminated from computation. Moreover, since paths are limited to each SCC, the number of required iterations, which is $|V| + 1$ as shown on line 5, can be reduced to the number of the vertices in the SCC plus 1, which further reduces the practical running time.

3.2 Limiting Number of Iterations

On the other hand, when the circuit is large, it is worthwhile to trade-off running-time with solution accuracy. An observation of the latch timing graph is that most simple cycles in the graph correspond to the time borrowing among the latches along the cycle. A positive cycle emerges if the time borrowing cannot be realized without violating timing constraints. As a designer will generally avoid time borrowing across a large number of latches, long cycles are less likely becoming positive due to process variations. Such observation motivates us to limit the number of iterations on line 5 in order to speed-up the analysis at the cost of solution accuracy. A designer can choose a proper bound to trade-off running-time with accuracy.

Table 1: Comparison of Algorithms

| circuit | | | NPCycle [1] | | | SGT-NP | | | Monte Carlo | |
|---------|-------|-------|-------------|---------|--------|--------|---------|--------|-------------|----------|
| name | $ V $ | $ E $ | yield% | time(s) | error% | yield% | time(s) | error% | yield% | time(s) |
| s27 | 4 | 14 | 95.78 | 0.01 | -0.08 | 95.78 | 0.007 | -0.08 | 95.86 | 0.19 |
| s208.1 | 9 | 53 | 94.50 | 0.04 | 1.12 | 94.01 | 0.013 | 0.63 | 93.38 | 0.64 |
| s298 | 15 | 84 | 91.50 | 0.05 | 1.14 | 91.05 | 0.022 | 0.69 | 90.36 | 1.38 |
| s344 | 16 | 115 | 97.28 | 0.07 | -1.78 | 98.94 | 0.038 | -0.12 | 99.06 | 1.75 |
| s444 | 22 | 173 | 96.03 | 0.15 | -1.88 | 97.37 | 0.049 | -0.54 | 97.91 | 3.35 |
| s526 | 22 | 165 | 95.38 | 0.09 | -1.14 | 94.81 | 0.037 | 0.29 | 94.52 | 3.28 |
| s832 | 6 | 36 | 97.99 | 0.03 | -0.02 | 97.99 | 0.009 | -0.02 | 98.01 | 0.40 |
| s1196 | 19 | 57 | 94.66 | 0.05 | -2.01 | 96.49 | 0.034 | -0.18 | 96.67 | 1.51 |
| s1238 | 19 | 57 | 95.91 | 0.05 | -2.05 | 97.70 | 0.032 | -0.26 | 97.96 | 1.54 |
| s1494 | 7 | 49 | 99.98 | 0.03 | 1.19 | 98.19 | 0.013 | -0.60 | 98.79 | 0.48 |
| s5378 | 180 | 1423 | 97.01 | 4.98 | -2.19 | 99.36 | 0.723 | 0.16 | 99.20 | 200.12 |
| s9234 | 229 | 2923 | 94.66 | 19.64 | -5.24 | 99.82 | 0.751 | -0.08 | 99.90 | 488.90 |
| s13207 | 670 | 4052 | 95.52 | 77.15 | -1.28 | 97.19 | 6.742 | 0.39 | 96.80 | 2669.81 |
| s15850 | 598 | 15891 | 92.34 | 567.36 | -1.95 | 93.67 | 11.879 | -0.62 | 94.29 | 7399.26 |
| s35932 | 1729 | 6940 | 97.54 | 235.92 | -2.16 | 98.48 | 100.139 | -1.22 | 99.70 | 11554.41 |

4 Experimental Results

The SGT-NP algorithm is implemented in C++ and tested on the ISCAS89 benchmark circuits. For comparison, we also implemented the Monte Carlo analysis algorithm, which will generate the most accurate results, and the NPCycle algorithm in [1], which is the most recent work on the same problem. The experiments are performed on a Linux PC with a 2.4GHz Core Due processor and 3GB memory.

The experimental results are shown in Table 1, it can be seen that the proposed approach can generate more accurate results in less running-time for all the benchmarks.

References

- [1] R. Chen and H. Zhou. *Statistical Timing Verification for Transparently Latched Circuits*. IEEE Transactions on Computer-Aided Design, 25(9):1847–1855, Sep. 2006.