Peak Power Estimation Using Genetic Spot Optimization for Large VLSI Circuits

Michael S. Hsiao (mhsiao@ece.rutgers.edu)
Department of Electrical and Computer Engineering, Rutgers University, Piscataway, NJ

Abstract

Estimating peak power involves optimization of the circuit's switching function. We propose genetic spot expansion and optimization in this paper to estimate tight peak power bounds for large sequential circuits. The optimization spot shifts and expands dynamically based on the maximum power potential (MPP) of the nodes under optimization. Four genetic spot optimization heuristics are studied for sequential circuits. Experimental results showed an average of 70.7% tighter peak power bounds for large sequential benchmark circuits was achieved in short execution times.

I Introduction

The continuing decrease in feature size and increase in chip density in recent years have given rise to concerns about excessive power dissipation in VLSI chips. Circuits become less reliable as large instantaneous power dissipation can cause overheating (local hot spots), and the failure rate for components roughly doubles for every $10^{\circ}C$ increase in operating temperature [1]. Furthermore, the growing market of portable computing products such as cellular phones and portable computers demands low-power consumption for long operational lifetime.

It has been shown in [2, 3, 4] that power estimation can be extremely sensitive to different gate delays, since multiple toggles at internal nodes can result due to uneven circuit delay paths. Both [2] and [3] computed the *upper bound* of maximum transition (or switching) density of individual nodes of a combinational circuit via propagation of uncertainty waveforms, while [4] computed the sensitivity of internal nodes due to a switch on the primary input. While these measures are useful to compute the bounds for individual signals, they cannot be used to compute a tight peak power bound on the entire circuit.

Unlike average power estimations [14-16] in which signal switching probabilities are sufficient to compute the average power, *peak* power is associated with a specific starting circuit state and a specific sequence of vectors that produce the power. Several approaches to measuring maximum power in CMOS VLSI circuits have been addressed in the recent years. The problem of worst-case power computation was transformed to a weighted max-satisfiability problem on a set of multi-output boolean functions for *combinational* circuits [5]. Peak current estimation for combinational circuits was addressed in [6, 7] where the goal was to find the time window during which a gate in the circuit could switch. A third approach using symbolic transition counts to compute maximum power cycles in the circuit's state transition graph (STG) was introduced in [8]. An automatic-test-generation (ATG)-based estimation technique for sequential circuits was proposed in [9] in which the aim is to create toggles in the circuit for gates with the greatest numbers of fan-outs. [10] extended the ATG approach to handle gate delays by adding multiple copies of internal gates at various propagation times for each gate. Finally, a genetic-algorithm (GA) based approach was proposed in [12, 13] in which the GA is used to maximize switching activity in both combinational and sequential circuits over various sequence lengths. The GA-based technique [12] gave very tight lower-bounds on peak power. Comparisons with 100 millions random state-vector tuples (more than 23 CPU hours of computation on the largest of the eight circuits) were made, and the best peak power estimates from 100 million tuples still lagged 4.4% behind the GA-based approach, which took less than 1 minute for all eight circuits.

Two major obstacles are faced in large circuits. The first is in avoiding local maxima in the huge search space, and the second is in getting out of local maxima once the search is stuck there. Moreover, switching activity of a given node in the circuit is also heavily dependent on the gate delays in the circuit. These added factors make peak power very difficult to estimate.

This paper proposes various *genetic spot optimization* techniques in which we try to optimize activity on a spot (group of nodes) at a time. The optimization spot can shift, shrink, and expand based on the maximum power potential (**MPP**) of the spot. Unlike maximum switching density, which is based on signal probabilities in the circuit, MPP takes both the number of possible transitions and output capacitance into consideration. Four genetic spot-optimization techniques are proposed and studied: (1) node-based, (2) path-based, (3) cone-based, and (4) Distance-based. The first three techniques aim to maximize switching on nodes, paths, or cones with the highest MPP, respectively. The fourth technique follows from the observation that the initial and intermediate states play important roles and aims to exploit weighted distances of these two states. All four genetic spots can change and grow dynamically during the optimization process. Experiments show that genetic spot-optimization techniques are very effective for obtaining tighter bounds for larger circuits.

The remainder of the paper is organized as follows. Section II explains the delay model, peak power model, and genetic algorithm used in this work; Section III describes the maximum power potential (MPP) and the various heuristics used for estimating peak power; experimental results are reported and discussed in Section IV; finally, Section V concludes the paper.

II Preliminaries

A Delay model

Since glitches and hazards are not taken into account in a zero-delay framework, the power dissipation measures may be off greatly from the actual powers [13]. In this work, variable delay model is chosen. Traditionally, a simple model based on the number of fan-outs has been used [10]. Though more accurate than the unit-delay model (e.g., every gate is assigned identical delay of one unit), fan-outs that feed larger gates are not taken into account, resulting in inaccuracies. A different variable delay

model based on fan-outs of a given node as well as fan-ins of its successor nodes is used [13]. The gate delay data for various types and sizes of gates are obtained from a VLSI library. Since traditional variable delay of a gate does not consider the sizes of its succeeding gates, the delay calculations may be less accurate.

B Peak power model

The unit of power used throughout the paper is energy per clock cycle and will simply be referred to as power. In a typical sequential circuit, the switching activity is largely controlled by the state vectors and less influenced by input vectors, because the number of flip-flops far outweighs the number of primary inputs. As illustrated in Figure 1, the power is controlled by both initial state S_1 and input vectors V_1 and V_2. The state S_1 and input vector V_1 initialize all gate outputs and determine the next state S_2. Then, vector V_2 and state S_2 switch some of the gates, which accounts for the power dissipation. We will obtain a three-tuple (S_1, V_1, V_2) that maximizes this power under the new variable delay model. In fully-scanned circuits, this bound is attainable because the state S_1 can be initialized to any arbitrary value. However, in cases where the initial state is not fully controllable, we can only speculate that during the operation of the circuit, the machine may reach state S_1, and only then can we be assured that the bound is attainable.

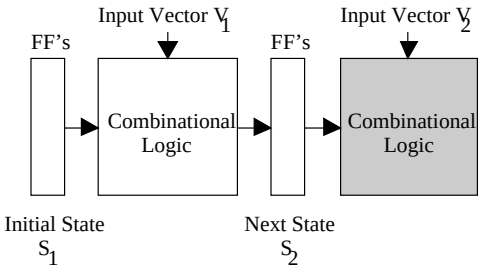

Figure 1: Power Model for Sequential Circuits.

The power dissipated in the combinational portion of the sequential circuit can be computed as

$$P = \frac{V_{dd}^2}{2 \times cycle\ period} \times \sum_{for\ all\ gates\ g} [toggles(g) \times C(g)], \qquad (1)$$

where the summation is performed over all gates g, and $toggles(g)$ is the number of times gate g has switched from 0 to 1 or vice versa within a given clock cycle; $C(g)$ is the output capacitance of gate g. Switching frequency (SF) per node is reported instead of total power in this paper, and it is computed simply as the second portion of equation (1) divided by the total number of capacitive nodes in the circuit (computed as the total number of gate inputs in the circuit),

$$SF = \frac{\sum_{for\ all\ gates\ g} [toggles(g) \times C(g)]}{total\ number\ of\ capcacitive\ nodes}. \qquad (2)$$

In this work, we made the assumption that the output capacitance for each gate is equal to the number of fan-outs; however, assigned gate output capacitances can be handled by our optimization technique as well.

C Genetic Algorithms

The GA framework used in this work is similar to the simple GA described by Goldberg [17]. The GA contains a population of *strings*, also called *chromosomes* or *individuals*, in which each individual represents a state-vector tuple. Peak power estimation requires a search for the 3-tuple (S_1, V_1, V_2) that maximizes power dissipation. This 3-tuple is encoded as a single binary string, as illustrated in Figure 2. The population size depends on the string length, which depends on the number of primary

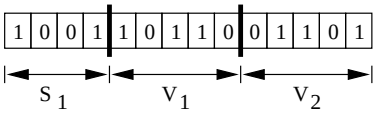

Figure 2: Encoding of an individual.

inputs and flip-flops. Larger populations are needed to accommodate longer individuals in order to maintain diversity. The population size is set equal to $128 \times \sqrt{string\ length}$.

Each individual has an associated *fitness*, which measures the quality of the vector sequence in terms of switching activity. The population is first initialized with random strings. A variable-delay logic simulator is then used to compute the fitness of each individual. The evolutionary processes of *selection*, *crossover*, and *mutation* are used to generate an entirely new population from the existing population. Evolution from one generation to the next is continued until a maximum number of generations is reached. In this work, a maximum of 32 generations is allowed. A mutation probability of 0.01 is used in this work, and since a binary coding is used, mutation is done by simply flipping the bit. Because selection is biased toward more highly fit individuals, the average fitness is expected to increase from one generation to the next. However, the best individual may appear in any generation, so we save the best individual found.

III Genetic Spot Optimization

Peak power estimation must consider both the number of toggles and the number of nodes with large output capacitances simultaneously during the maximization process. Maximization on either aspect alone is insufficient to ensure a tight bound on peak power measures; this is even more so with large circuits. Optimizations which merely try to generate one transition on as many gates with large output capacitances as possible may miss the opportunity to produce larger power consumption by generating more transitions on fewer gates. Conversely, maximizing merely the number of transitions may overlook the cases where greater power dissipations may result from fewer transitions on more nodes with larger output capacitances.

Four genetic spot-optimization heuristics are used in large circuits. All four heuristics try to maximize the circuit's switching activity by dynamically expanding localized spots. Before the explanations for each technique, the maximum power potential with which genetic expansion is based on will be discussed first.

A Maximum Power Potential

The knowledge of which nodes have the maximum potential for switching is critical to guiding the search. Instead of merely relying on output capacitance or signal probability, a node's maximum power potential (MPP) takes into consideration both the output capacitance and the number of possible transitions. The delays in various paths that lead to a node contribute to different possible times on which the node may toggle. For instance, Figure 3 illustrates the lists of possible transition times for each gate in the circuit. Variable delay model is used and gate delays are indicated inside each gate.

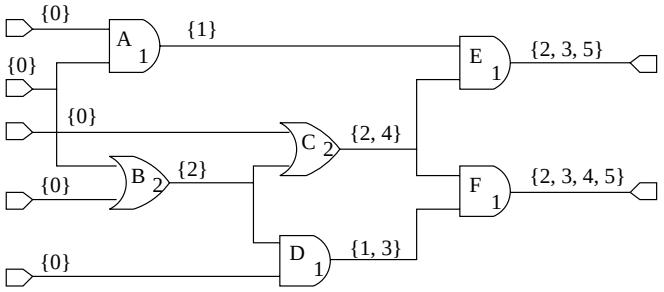

Figure 3: Computation of possible transition times.

Enumeration of all possible transition times for each gate can be done in a levelized fashion in $O(n)$ time, where n is the number of gates in the circuit. All inputs to the combinational portion of the circuit (i.e., primary inputs and flip-flop outputs) have only one possible transition time, namely 0. The set of transition times for any internal node can simply be obtained from a two-step process. First, we form a set union of the sets of transition times from all the immediate predecessors. Next, we offset each element in the set union by the gate delay of the current node. For example, in order to compute the set of possible transition times for node **E**, we first union the transition time sets of its inputs **A** and **C**: $\bigcup\{\{1\},\{2,4\}\} = \{1,2,4\}$. Then, we add the gate delay of node **E**, 1, to each element in the union, resulting in the set $\{2,3,5\}$. It should be noted that the set of possible transition times for each node varies with the underlying delay model, but all can be computed in the same manner. Although transitions may not be possible to occur at all the possible times in set S_n for a given node n, the MPP values can be computed very quickly by equation (3).

Once the transition times have been determined for all gates in the circuit, the maximum power potential for a given node is computed as the **product** of the *cardinality of its set of transition* and and its *output capacitance*:

$$MPP(n) = |S_n| \times C_n. \tag{3}$$

Note that under the zero-delay assumption, $|S_n| = 1$ since a gate can transition at most once in a time frame, so MPP of a gate is simply the output capacitance of that gate.

B Node-based heuristic

This is the most simplistic heuristic in which the optimization spot begins at the node with the greatest MPP. This approach is based on the assumption that peak power consumed in the circuit will likely include toggles on nodes with greatest MPP.

Even though delay is considered in our case, by considering only one or a few nodes with greatest MPP there may still remain shortcomings during the optimization process, particularly if many switches on a high-capacitive node may be less favorable than having more lower-capacitive nodes switch multiple times in the circuit. To remedy this problem, the spot on which the optimization focuses on expands steadily to include more nodes that can potentially increase the peak power. Genetic spot expansion is done by bringing additional nodes of greatest MPP that aren't currently in the optimization spot. Note that additional nodes may not be neighboring nodes. In doing so, the GA no longer works on a static spot. Instead, the spot changes whenever needed, as in the **dynamic fitness objectives** developed in [18].

C Path-based heuristic

It is observed that switching activity on a given node relies heavily on the switching activity of its predecessor nodes. If a gate n has two or more immediate predecessor gates, the predecessor p_i with higher MPP has a greater influence on n since p_i has potential for producing more transitions that can propagate to its successor gates. Thus, by induction, one would prefer to form a path from node n to a primary input or flip-flop via a set of nodes with greater MPP. We call this path a *maximal MPP path*. A maximal MPP path from n can be constructed simply by a depth-first search of nodes with greatest MPP, starting from node n. Figure 4 illustrates construction of a segment on

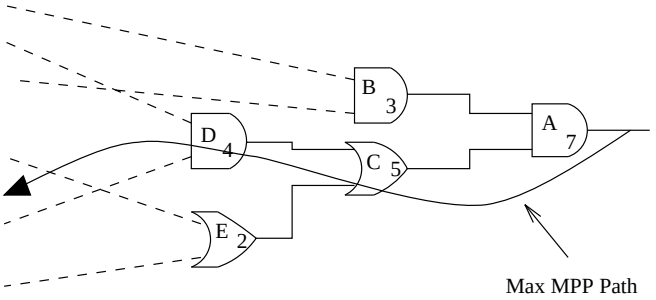

Figure 4: Construction of Max MPP Path.

the maximal MPP path. The number inside each gate indicates the MPP value for that gate. If we start constructing the MPP path from gate **A**, the first node to be added is gate **C** since it has a higher MPP than gate **B**. The construction continues in a depth-first manner by adding gate **D** to the path, and so on.

The initial genetic spot consists of the maximal MPP path containing the node with the greatest MPP. The spot expands by adding other MPP nodes on the path(s) into the spot. The construction of other MPP paths are done in a similar fashion except that we made a restriction that no two MPP paths should share any common segments.

D Cone-based heuristic

A tighter bound on peak power may require peak activity along several paths to a given node, which was not accounted for in the path-based heuristic. To remedy this problem, the cone-based heuristic is proposed in which all paths which lead up to an MPP node are considered, and the optimization is focused on producing maximum power inside the MPP cone.

Generally, the initial MPP cone includes less than 15% of the entire circuit for most circuits. And each addition of another MPP cone during expansion adds less than 15% additional nodes since cones usually share a portion of common nodes.

E Distance-based heuristic

The initial state plays an important role in determining power consumption. Indeed, results from the first three heuristics suggest that greater Hamming distances between the initial and intermediate states are favored when estimating peak power, where Hamming distance is the number of different flip-flop values between two states. For instance, states **101101** and **011001** have a Hamming distance of 3 (with the flip-flop values differ in the first, second, and the fourth bit positions).

Accounting for only the Hamming distance may be misleading, however, since toggles on different flip-flops in the circuit

contribute differently to power consumption. A toggle on a flip-flop may induce a controlling value to some paths that block hyperactive activity from further propagation. Moreover, vector pairs (**000000**, **111111**) and (**010101**, **101010**) both have the maximum Hamming distance, but they may not generate the same power. So instead of merely counting the hamming distance, different weights are placed on each flip-flop based on the favorability of a transition on the flip-flop.

The optimization in this heuristic tries to maximize the following function that computes a weighted-distance function:

$$\sum_{for\ all\ flip-flops} b_i \times w_i, \tag{4}$$

where b_i is a boolean variable and is equal to 1 when the flip-flop values on bit position i differ between the initial and intermediate states, and w_i is the weight for flip-flop i. The weight of a flip-flop i indicates the potential influence on total power dissipation from a toggle on flip-flop i; it is computed as the sum of MPP values on the reverse maximal MPP path starting from flip-flop i. A reverse maximal MPP path is similar to a maximal MPP path except that construction of a reverse MPP path is done in a forward manner, i.e., starting from a flip-flop and moves forward to a primary output or flip-flop.

The advantage of this weighted distance-based heuristic is that a zero-delay simulator is sufficient to evaluate the maximization function, since we only need to know if a transition can occur at each flip-flop. When the optimized state-vector tuple is obtained, a variable-delay simulator is then used to calculate the power consumed by the tuple.

IV Experimental Results

Peak powers under the new variable delay model were estimated for large ISCAS89 sequential benchmark circuits [19] and two synthesized circuits [20]. All computations were performed on a Sun Ultra-I with 64 MB RAM.

All power estimates are compared against the estimates obtained from 262,000 randomly generated state-vector tuples as well as those obtained using algorithms presented in [12]. All powers are expressed in **peak switching frequency per node (PSF)**, which is the average frequency of peak switching activity of the nodes (ratio of the weighted 0-to-1 and 1-to-0 transitions on all nodes to the total number of capacitive nodes) in the circuit. No zero-width spikes are considered in our approach.

As indicated in [12, 13], GA-based technique obtains very tight lower-bounds on peak power for many circuits, especially the smaller ones. Results for 100 million random state-vector tuples were compared in [12], and the GA technique outperformed the near-exhaustive search for eight small circuits. There is little difference between our results and [12] for most *small* circuits, indicating that simple genetic technique alone is adequate to compute tight bounds for peak power in small circuits.

Results are much more significant for larger circuits. Every large circuit studied has at least 55 flip-flops, with circuits s35932 and s38417 being the largest, each with more than **1600** flip-flops. The peak power estimates for large sequential circuits are shown in Table 1 using various approaches. For each circuit, the number of flip-flops is first given in parenthesis next to the circuit name. Then, the total number of capacitive nodes (computed as the total number of gate inputs in the circuit) is given, followed by the results of the random approach (best of 262,000 random simulations). Next, the peak powers obtained from [12] are shown along with the improvements [12] has over the random approach. Finally, the peak powers and their corresponding improvements over the random approach using the four proposed heuristics are reported in the table. The highest power estimates are highlighted in **bold**. The number of GA generations used for [12] was extended to be four times the original number in order to match the number of random simulations used in the random approach. Unlike the algorithm described in [12], no seeding of the best of random is used in our genetic spot optimization. Results from the ATG-based approach used in [10] (based on expanded combinational circuit) are not included, since bounds obtained from [12] are already better and [10] did not report results for large sequential circuits.

For all circuits, genetic spot-optimization heuristics surpassed both the random and the original GA-based [12] approaches. Among the four dynamic heuristics, the cone-based technique gave the tightest bounds most consistently. For instance, in circuits s1423 and am2910, the cone-based technique obtained 309% and 53% improvements, respectively, over the random approach. However, occasionally path-based and distance-based heuristics outperformed the cone-based heuristic. On the average, the node-based heuristic performed slightly better than the algorithm proposed in [12]. Path-based heuristic achieved an average of 61.9% improvement over the random approach, an average 70.7% improvement for cone-based heuristic, and 34.4% for the distance-based heuristic.

Even though the genetic spot-optimization heuristics performed better than the the algorithm presented in [12] for most circuits, there are a few cases where [12] was able to achieve tighter peak power bounds than one or two genetic spot-optimization heuristics. For instance, in the circuit s1423, the PSF obtained by [12] was higher than both node-based and distance-based heuristics, etc. In such cases, node-based and distance-based heuristics narrowed the search too quickly, resulting in a local maximum, and spot expansion did little to help the search get out of the local maximum.

The execution times for various techniques are shown in Table 2 for each circuit. The execution times required for the random simulations are very close to the original GA-based technique [12] since similar numbers of simulations were evaluated. However, [12] sometimes reaches a local maximum very quickly, requiring only one-third to one-half of the time. The execution times for the distance-based heuristic are consistently lower because zero-delay simulation is sufficient during the optimization process; significantly fewer event evaluations are needed since a gate can switch at most once in a time frame under this delay model. Execution times for the other three heuristics are, in contrast, much higher. The extra times needed are due to much more event evaluations, and execution times are directly proportional to the total number of events.

V Conclusions

Getting tight bounds on peak power requires efficient search algorithms in enormous search spaces. In this paper, four genetic spot-optimization heuristics are proposed to avoid local maximums during the search process by shifting and expanding the optimization spots dynamically. The proposed heuristics have been shown to be very effective in large sequential circuits. When compared to the results of 262,000 random simulations, the genetic spot-optimization heuristics achieved up to an av-

Table 1: Peak Power Estimates for Large Sequential Circuits

Circuit (FF's)	Cap Nodes	Random	[12]		Node-Based		Path-Based		Cone-Based		Distance-Based	
		PSF	PSF	Impr	PSF	Impr	PSF	Impr	PSF	Impr	PSF	Impr
s1423 (74)	1243	1.081	1.432	32.5	1.404	29.9	4.064	275.9	**4.427**	**309.5**	1.406	30.1
s5378 (179)	4440	0.856	1.213	41.7	1.249	45.9	1.263	47.5	**1.271**	**48.5**	1.261	47.3
s9234 (228)	8221	0.693	1.286	85.6	1.229	77.3	1.268	83.0	**1.316**	**89.9**	1.264	82.4
s13207 (669)	12031	0.919	1.261	37.2	1.256	36.7	1.262	37.3	1.208	31.4	**1.265**	**37.6**
s15850 (597)	14343	0.592	0.693	17.1	0.722	22.0	0.765	29.2	**0.786**	**32.8**	0.734	24.0
s35932 (1728)	30317	1.396	1.396	0.0	1.410	1.0	**1.417**	**1.5**	1.408	0.9	1.415	1.4
s38417 (1636)	33988	0.636	0.637	0.2	0.737	15.9	0.690	8.5	**0.756**	**18.9**	0.738	16.0
am2910 (87)	1998	4.798	5.688	18.5	5.586	16.4	5.720	19.2	**7.354**	**53.3**	5.691	18.6
mult16 (55)	1323	1.861	2.410	29.5	2.601	39.8	**2.892**	**55.4**	2.815	51.3	2.828	52.0
Impr				29.2		31.6		61.9		**70.7**		34.4

All peak powers expressed in Peak Switching Frequency per Node (PSF)
Impr: % Improvement over the best random estimate

erage of 70.7% improvement in large sequential benchmark circuits. Significant improvements were also observed when compared to the results using the algorithm in [12]. Future extensions include study of correlation between circuit structure and the effectiveness of various heuristics, mixture of various heuristics, as well as reducing execution time further during the genetic spot-optimization process.

Table 2: Execution Times

Ckt	Random	[12]	Node-Based	Path-Based	Cone-Based	Dist-Based
s1423	6.44	7.52	4.28	8.03	8.00	1.08
s5378	44.44	44.76	31.77	42.78	44.98	7.18
s9234	44.00	49.20	39.88	51.78	57.03	10.04
s13207	57.40	67.52	57.78	72.68	88.73	12.58
s15850	52.72	60.68	54.10	68.75	101.77	13.13
s35932	239.68	240.68	268.75	312.35	454.38	55.03
s38417	169.68	186.12	196.47	223.82	365.37	61.72
am2910	34.48	34.52	25.23	29.88	30.60	5.10
mult16	6.88	7.32	7.65	8.92	7.31	1.56

All times expressed in **minutes**

References

[1] C. Small, "Shrinking devices put the squeeze on system packaging," *EDN.*, pp. 41-46, February 1994.

[2] F. N. Najm and M. Y. Zhang, "Extreme delay sensitivity and the worst-case switching activity in VLSI circuits," *Proc. Design Automation Conf.*, 1995, pp. 623-627.

[3] C. Teng, A. M. Hill, and S. Kang, "Estimation of maximum transition counts at internal nodes in CMOS VLSI circuits," *Proc. Int. Conf. CAD*, 1995, pp. 366-370.

[4] Z. Chen, K. Roy, and T.-L. Chou, "Power sensitivity - a new method to estimate power dissipation considering uncertain specifications of primary inputs," *Proc. Int. Conf. CAD*, 1997, pp. 40-44.

[5] S. Devadas, K. Keutzer, J. White, "Estimation of power dissipation in CMOS combinational circuits using boolean function manipulation," *IEEE Trans. on CAD.*, pp. 373-383, March 1992.

[6] H. Kriplani, "Worst case voltage drops in power and ground busses of CMOS VLSI circuits," *Ph. D. Thesis, Univ. of Illinois*, November 1993.

[7] H. Kriplani, F. Najm, P. Yang, and I. Hajj, "Resolving signal correlations for estimating maximum currents in CMOS combinational circuits," *Proc. Des. Aut. Conf.*, 1993, pp. 384-388.

[8] S. Manne, A. Pardo, R. I. Bahar, G. D. Hachtel, F. Somenzi, E. Macii, and M. Poncino, "Computing the maximum power cycles of a sequential circuit," *Proc. Des. Aut. Conf.*, 1995, pp. 23-28.

[9] C. Wang, K. Roy, and T. Chou, "Maximum power estimation for sequential circuits using a test generation based technique," *Proc. Custom Integrated Circuits Conf.*, 1996.

[10] S. Manich and J. Figueras, "Maximizing the weighted switching activity in combinational CMOS circuits under the variable delay model," *Proc. European Design & Test Conf.*, 1997, pp. 597-602.

[11] C.-Y. Wang and K. Roy, "COSMOS: A continuous optimization approach for maximum power estimation of CMOS circuits," *Proc. Int. Conf. CAD*, 1997, pp. 52-55.

[12] M. S. Hsiao, E. M. Rudnick, and J. H. Patel, "K2: An estimator for peak sustainable power of VLSI circuits," *Proc. Int. Symp. Low Power Electr. & Des.*, 1997, pp. 178-183.

[13] M. S. Hsiao, E. M. Rudnick, and J. H. Patel, "Effects of delay model in peak power estimation of VLSI sequential circuits," *Proc. Int. Conf. CAD*, 1997, pp. 45-51.

[14] A. Ghosh, S. Devadas, K. Keutzer, and J. White, "Estimation of average switching activity in combinational and sequential circuits," *Proc. Des. Aut. Conf.*, 1992, pp. 253-259.

[15] F. N. Najm, "A survey of power estimation techniques in VLSI circuits," *IEEE Trans. on VLSI Systems*, vol. 2, no. 4, pp. 446-455, December 1994.

[16] T. Chou and K. Roy, "Accurate power estimation of CMOS sequential circuits," *IEEE Trans. on VLSI Systems*, vol. 4, no. 3, pp. 369-380, September 1996.

[17] D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Reading, MA: Addison-Wesley, 1989.

[18] M. S. Hsiao, E. M. Rudnick, and J. H. Patel, "Sequential circuit test generation using dynamic state traversal," *Proc. European Design & Test Conf.*, 1997, pp. 188-195.

[19] F. Brglez, D. Bryan, and K. Kozminski, "Combinational profiles of sequential benchmark circuits," *Int. Symp. on Circuits and Systems*, 1989, pp. 1929-1934.

[20] M. S. Hsiao, E. M. Rudnick, and J. H. Patel, "Automatic test generation using genetically-engineered distinguishing sequences," *Proc. VLSI Test Symp.*, pp. 216-223, 1996.