VIP - An Input Pattern Generator for Identifying Critical Voltage Drop for Deep Sub-Micron Designs

Yi-Min Jiang,* Tak K. Young,** Kwang-Ting Cheng*

Dept. of Electrical & Computer Engineering*
University of California
Santa Barbara, CA 93106
e-mail: {jym, timcheng}@bigbend.ece.ucsb.edu

Synopsys, Inc.**
700 East Middlefield Road
Mountain View, CA 94043
e-mail: tak@synopsys.com

Abstract

We present a novel input pattern generator for dynamic power network simulation. The obtained patterns successfully identify critical voltage drop areas for a set of industrial designs, which are difficult to be found using functional vectors. The search engine of the pattern generator for worst-case IR voltage drop is based on the multi-objective genetic algorithm. To achieve high coverage for critical voltage drop cells, we propose to model the search criteria into the maximum weighted matching of a bipartite graph, and guide the search direction according to the matching results. Experimental results show that, compared with the other approaches, our patterns give a higher coverage of critical voltage drop cells.

1. Introduction

For designers of today's high performance and complexity ICs, the accurate and efficient analysis for power net voltage drop is very important. Excessive voltage drop increases the transistor and gate delays, which results in unpredictable performance or performance failing to meet original design goal. To identify this problem, dynamic simulation is needed for providing the profile of the voltage drop. Therefore, generating high-quality input patterns for simulation of the voltage drops has become a necessary step in the entire design cycle.

Recently, several Genetic-Algorithm-based (GA-based) techniques have been proposed to generate the input patterns for identifying the maximum instantaneous current [4], maximum power dissipation [1], and maximum voltage drop [5]. Through iteratively generating the new test patterns for simulation based on the "good" property of the current patterns, they produce tight lower bounds for these problems. In such a way, however, certain functional blocks whose current has little contribution to the maximum total current or maximum voltage drop at the V_{dd} pin of the chip may *not* be *actively* exercised during the pattern generation process. Using patterns derived in such ways may not identify the voltage drop problems in these blocks. On the other hand, if the blocks are targeted one by one, the overall process could be too time consuming.

In this paper, we address the problem of generating a set of patterns that would cause high voltage drops at the V_{dd} ports of all the cells whose worst-case drop could exceed a given threshold. This problem is referred to as the critical voltage drop problem. We propose a heuristic procedure for this problem and attempt to maximize the coverage of these cells. Our approach extends the technique of [4] to generate input patterns for identifying the voltage drop problem of all potential problematic blocks. We propose to induce the maximum current drawn from *each* individual block using GA. Without losing the functional correlation between different blocks, we transform the single-objective Genetic Algorithm to satisfy correlated multi-objectives simultaneously, where each objective denotes the maximum current associated with a specific block. To achieve this, we model the search criteria in GA as the maximum weighted matching of a bipartite graph, which can be efficiently solved by the Hungarian method [7].

We have implemented the proposed algorithm as an input pattern generator, named *VIP.* The obtained patterns, which will cause high voltage drops for all potential problematic blocks, can be used by any power network simulator to analyze the critical voltage drop. *VIP* has been tested on a set of benchmarks with completed physical designs, which is comprised of twelve large industrial designs with a wide variety of applications such as microprocessors, DSP processors, and large memory banks. Experimental results show that, compared with the other approaches, this novel technique identifies the voltage drop sensitive portions of the designs with high accuracy in terms of identified critical cells.

The rest of this paper is organized as follows. In Section 2, we give an introduction to Genetic-Algorithm-based approach described in [4]. Section 3 describes the details of our approach. The experimental results are shown in Section 4. Section 5 concludes the paper.

2. The Genetic-Algorithm-based Technique for Generating Input Patterns Causing Maximum Instantaneous Current

Genetic Algorithm (GA) [3] is a robust search algorithm which has been applied to solve many problems efficiently. The key idea is that, if solutions are represented by strings, the string associated with the optimal solution can eventually be found through "evolution-like" string operations. The search engine is an iterative process which employs three operations: selection, crossover, and mutation. The objective of these operations is to remove "poor" strings and produce new strings which is comprised of parts of "healthy" strings.

To use GA, the elements in the solution space need to be coded into finite length strings. Each string has an associated fitness which depends on the application. An initial population needs to be specified as the input of GA. The initial population contains N random strings of length L, where N and L are parameters used in GA. Generation of a new population is found by (1) evaluating the fitness for each string, (2) selecting two individuals from the current population, (3) crossing the two selected strings to generate two child-strings from two parent-strings, and (4) mutating the elements of the new strings with a given mutation probability. The selection process is biased towards individuals with higher fitness values. The next population is generated based on the current population using the same procedure. During the string generation process, the strings with the highest fitness would be recorded.

Under this scenario, the technique in [4] transforms the solution space of two-vector sequences causing maximum instantaneous current into the GA search domain, and drive the search engine to find the solution. In the transformation, each input sequence is coded into a string, and the associated peak current corresponds to the fitness of this string. According to this, GA starts with a population of strings and iteratively generates successive population with likely higher fitness. The procedure is shown in Figure 1. In this approach, the initial set can be either generated randomly or specified by users. To ensure high accuracy, a transistor-level power/current simulator PowerMill [9] is used to simulate each sequence and report the peak current as the fitness. The maximum instantaneous current is updated based on the fitness for each iteration. The selection and crossover schema used are tournament selection without replacement [6] and one-point crossover, respectively. The process continues until no further improvement is achieved or the number of iterations reaches a pre-defined value.

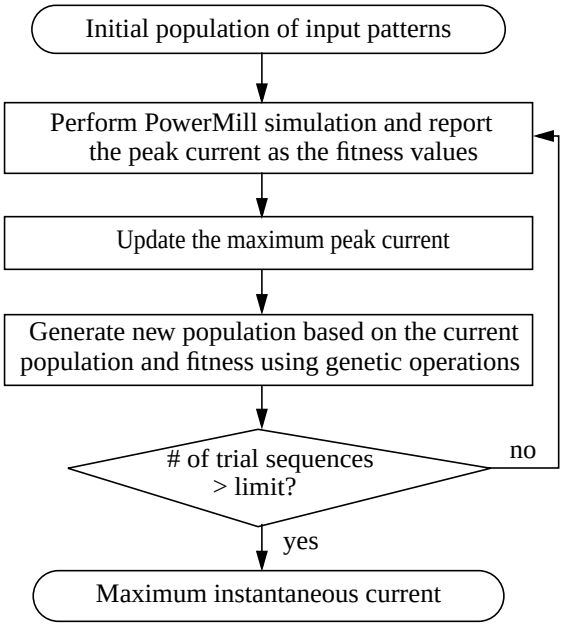

Figure 1: GA-based approach for maximum instantaneous current.

3. Critical Voltage Drop Identification

In this section we will define the critical voltage drop problem and propose a methodology for solving this problem. The term, an input pattern is regards to as a two-vector sequence, $V = (v_1,v_2)$, where the first vector v_1 is used to initialize the designs, and v_2 causes the switching in the internal circuitry. Without losing generality, in the paper, a cell is referred to a logic gate.

Definition 1 *A cell is called a* ***critical voltage drop cell*** *if there exists at least one input pattern which causes high voltage drop at the cell's V_{dd} port whose drop value exceeds a given threshold and the duration is longer than the user-specified value.*

Figure 2 shows the voltage waveform of the V_{dd} port of a cell after applying an input pattern. During the period from t_1 to t_2, the voltage level is less than the threshold and this time period is longer than the specified one. This cell is thus a critical voltage drop cell. Note that there may exist multiple input patterns which could identify the same critical voltage drop cell. On the other hand, an input pattern might be capable of identifying multiple critical voltage drop cells.

Definition 2 *The* ***critical voltage drop set*** *of an input pattern is the set of the critical voltage cells identified by this pattern.*

There is one corresponding critical voltage drop set for each input pattern. The union of the critical voltage drop sets of all input patterns represents all the critical voltage drop cells in a design.

Definition 3 *The* ***critical voltage drop problem*** *is defined as the problem of identifying all critical voltage drop cells in a design.*

One way to solve this problem would be to exhaus-

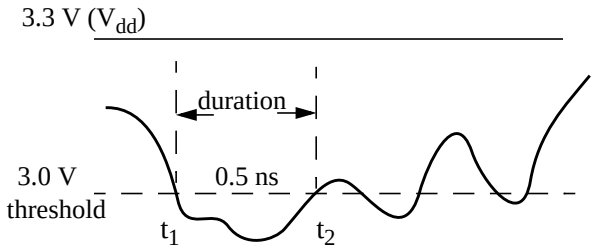

Figure 2: The voltage waveform for a cell's V_{dd} port w.r.t. an input pattern.

tively simulate all possible patterns, identify their critical voltage drop sets from simulation results, and obtain the union of these sets. For a circuit with *n* primary inputs, it would require simulation of 4^n patterns. This is impractical even for circuits with a small number of primary inputs.

In this paper we propose to find a small set of input patterns such that the union of their critical voltage drop sets would be identical to the union of the sets of all input patterns. We apply the Genetic-Algorithm-based technique proposed in [4] (introduced in Section 2) to generate such input patterns. The key issue here is the selection of a suitable fitness function for GA as the quality of the input patterns generated by Genetic Algorithm is strongly dependent on the fitness function used. The voltage drop computation requires accurate simulation of the power supply network together with the transistor netlist that drives them. If we use voltage drop as fitness, this comprehensive simulation needs to be performed once for each pattern. In such a way, the overall GA-based procedure will be prohibitively slow for most of today's large designs with million transistors and power net *RCs*. Therefore, we need to find an easy-to-compute metric as the fitness in which high fitness corresponds to the high coverage of critical voltage drop cells.

Peak current of a design is the maximum current which the design draws in response to an input pattern. This current distributes through the power network to transistors and capacitances. Higher peak current tends to cause higher voltage drop because more current flows in the resistive network. Similarly, we can define the peak current of a functional block as the maximum current the block draws in response to an input pattern. If we use the peak current of an entire design as fitness, the generated patterns may not activate all the critical voltage drop cells. This is because for different functional blocks in a design, the times that the peak current occurs may not coincide with each other. If that's the case, maximizing peak current of the whole design

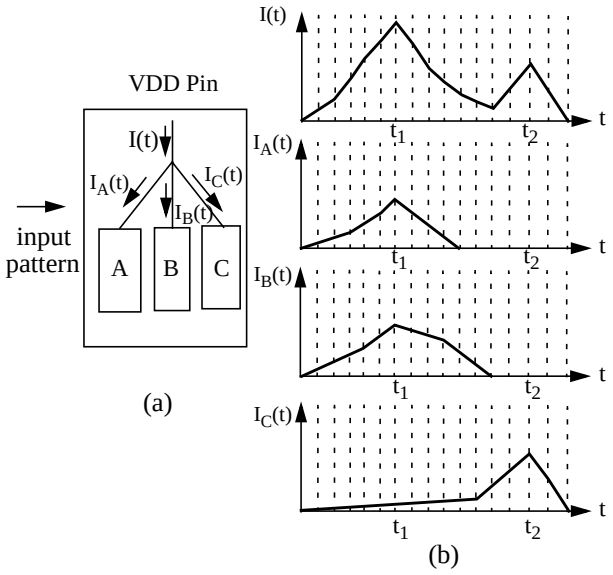

Figure 3: The current waveform for an entire design and each individual block w.r.t. an input pattern.

only results in exhibiting the worse cases for cells in some critical blocks, not all critical cells of the design. Consider a 3-block design in Figure 3(a). The current waveform with respect to an input pattern for the entire design and each block is shown in Figure 3(b). The peak current of the entire design occurs at time t_1. Note that the peak current of block *C* does not coincide with the entire design; on the contrary, the current drawn by block *C* at time t_1 is much lower than the one from the other blocks. For this case, peak current for block *C* cannot be maximized by the GA using entire design' peak current as the fitness.

Motivated by this, we propose to use the peak current of each individual block as a major factor of fitness. In the meantime, we also include the current of the entire design into fitness. This is because current from other blocks also contributes to the voltage drop of the target block. We perform transistor-level simulation to extract the peak current based on the simulation results. We use an approximate but efficient approach proposed in [11][1] to transistor-level simulation. During the transistor-level simulation, power supply network voltage drops are ignored to speed up the simulation. In other words, we simulate only the transistor netlist and assume constant voltages at power buses during the simulation. The model for this speed-up approach is shown in Figure 4. Note that the V_{DD} current obtained by this approach would be higher than the one obtained by simulating both transistor and power network netlists. This is because the voltage drop on power network will cause lower V_{DD} current compared to the one without voltage drop. It is important to note that this approach is used only to generate

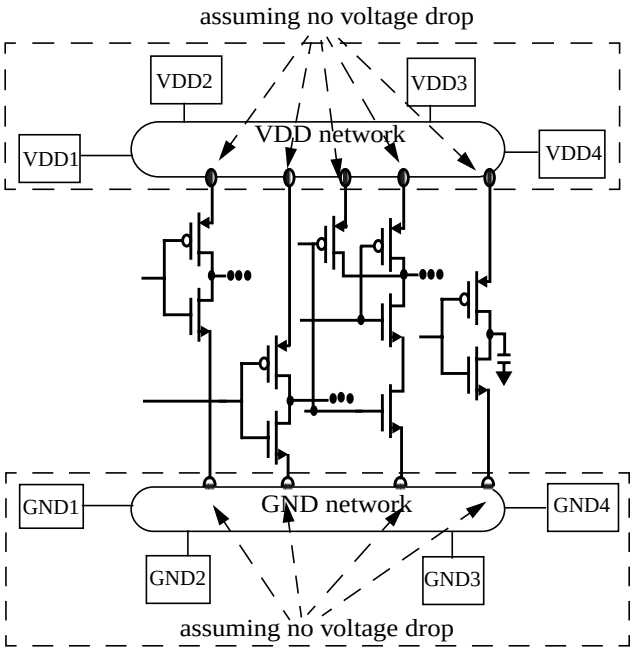

Figure 4: The model for transistor-level simulation speed-up approach.

the input patterns. After the patterns are generated, we simulate both transistor and power network netlists to these patterns to identify critical voltage drop cells based on the simulation results.

We define the voltage drop factor of an input pattern *v* to block *m* as follows:

$$voltage\_drop\_factor\ (v, m) = peak\_current\ (v, m) + \alpha * current\ (v, t) \quad (1)$$

where *peak_current (v, m)* denotes the peak current of block *m* with respect to input pattern *v*. The argument *t* in the term *current (v, t)* represents the time when *peak_current (v, m)* occurs. *Current (v, t)* is the current of the entire design at time *t* to *v*. The correlation factor α is defined as the reciprocal of the number of blocks.

We use this voltage drop factor as a measure of voltage drop. The objective, for each block, is to find the input pattern with the maximum factor value. Instead of performing GA one by one, we propose a novel approach, which utilizes the "group search" feature of GA, to maximize multi-objectives within a single GA run. The method proceeds as follows. To make GA maximization process covering each block, we set the population size as the number of blocks, and perform one-to-one mapping between input patterns and blocks. For each pattern, the fitness is referred to as the corresponding voltage drop factor of the mapped block. We propose to find the mapping with the maximum summation of the fitness, and then use the fitness to generate the new population of input patterns.

Consider an example shown in Figure 5 with four input patterns: 1, 2, 3, and 4, and four blocks: *a*, *b*, *c*, and *d*. The weight of edge V_{ij} between pattern *i* and block *j* is the voltage drop factor. Suppose the mapping (denoted by the thick lines) have the maximum summation. Then the fitness for

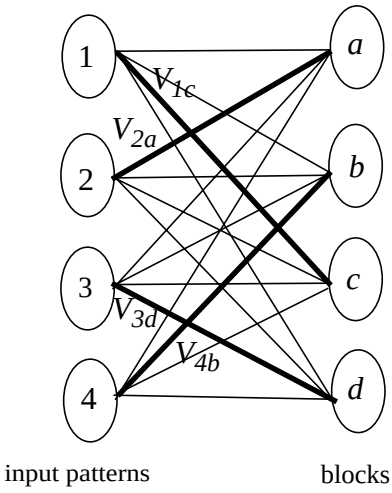

Figure 5: The fitness determination for input patterns.

patterns 1, 2, 3, and 4 are V_{1c}, V_{2a}, V_{3d}, and V_{4b}, respectively. The mapping is referred to as the maximum weighted matching of a bipartite graph, which can be efficiently solved by the Hungarian method [7].

For each iteration, we update the maximum voltage drop factor and the associated input pattern with respect to each block. The genetic operation schema and the termination condition of the process are the same as used in [4]. Finally, we perform the power network simulation to these recorded input patterns. The overall flow of our technique is shown in Figure 6.

4. Experimental Results

We perform the following experiment to validate the effectiveness of the *VIP*. For small benchmark circuits, we simulated the power network and the transistor netlist using HSPICE for all possible input patterns and then report all the critical voltage drop cells. These cells are used to compare with the ones derived by simulating only the patterns which are generated by *VIP* and [4], which uses to the peak current of the entire designs as the fitness of each pattern. Due to the large number of simulation runs needed for circuits with a large number of primary inputs, this validation experiment is only applied to circuits with a small number of inputs. Before performing the simulation, we partition each circuit into blocks based on the sizes and topology of circuits. The simulation are performed for three sets of input patterns: (1) patterns generated by *VIP* (the number is the same as the number of blocks for each circuit), (2) patterns generated by [4] (the number is the same as the *VIP* uses), and (3) all possible input patterns.Table 1 shows the number

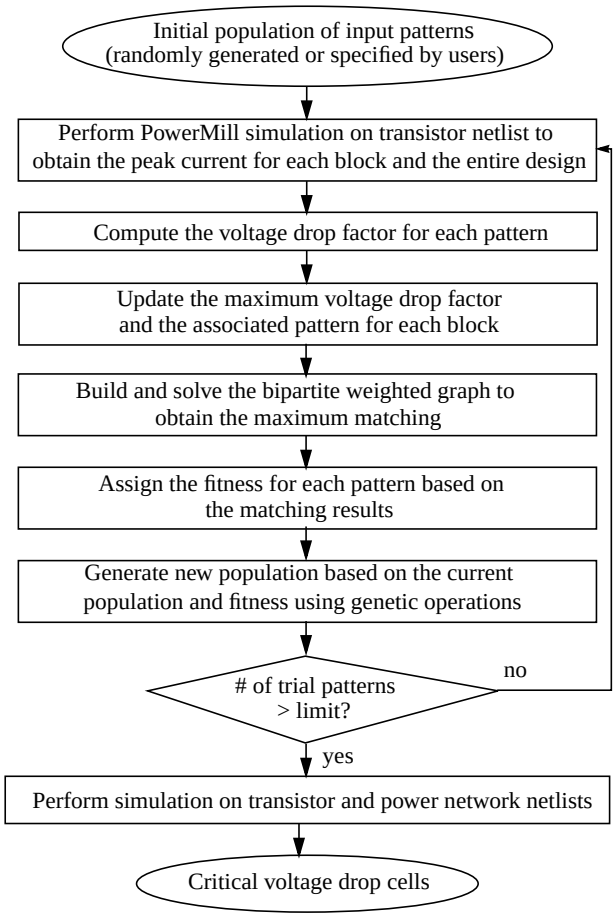

Figure 6: The overall flow of our technique for critical voltage drop problem.

of critical voltage drop cells identified by the three sets of patterns for the 7 small MCNC91 benchmark circuits. All the experimental results are based on a 0.25 μm with supply voltage 2.5 *V* library.

The number of critical voltage drop cells and normalized values by (1) *VIP*, (2) [4], and (3) all patterns are shown in Columns 2-3, 4-5, and 6-7, respectively. All normalized values are with respect to the values derived by all patterns. Column 8 gives the number of blocks for each circuit where this number is the same as the number of simulated patterns used by *VIP* and [4]. Columns 9 and 10 show the specified threshold voltage drop and threshold time period. The experimental results show that, on average, four patterns generated by *VIP* identify 96% of the critical voltage drop cells.

VIP is also tested to a set of industrial designs with a wide range of applications such as CPUs, DSP processors, and large memory banks. The number of transistors ranges from 61K to 1.14M, and the technology from 0.25μ to 0.8μ.

Table 1: The comparison of critical voltage drop cells for three sets of input patterns.

Ckt.	# of the identified critical voltage drop cells						#. of blocks	threshold voltage drop (mV)	threshold time period (ns)
	VIP		[4]		all possible patterns				
	#.	norm.	#.	norm.	#.	norm.			
cm42a	4	1.00	4	1.00	4	1.00	3	5.6	0.02
cm82a	5	1.00	4	0.80	5	1.00	3	4.1	0.02
cm85a	6	0.86	2	0.29	7	1.00	3	9.9	0.02
cm138a	5	1.00	4	0.80	5	1.00	4	4.7	0.02
cmb	0	1.00	0	1.00	0	1.00	4	5.7	0.02
cu	1	1.00	1	1.00	1	1.00	3	9.7	0.02
vda	18	0.86	11	0.52	21	1.00	8	53.1	0.04
Ave.	-	0.96	-	0.77	-	1.00	4	-	-

Table 2: Design statistics.

Designs	# of PIs	technology (μ)	supply voltage (V)	#. of transistors	#. of power net' RCs	threshold voltage drop (V)	threshold time period (ns)
1	35	0.25	2.5	1.0M	898K	0.09	0.03
2	194	0.28	2.2	27.9K	83K	0.10	0.04
3	112	0.5	3.3	1.01M	1.81M	0.60	0.06
4	165	0.5	3.3	472K	937K	0.45	0.05
5	108	0.5	3.3	508K	602K	0.55	0.06
6	109	0.6	3.3	1.14M	1.19M	0.50	0.03
7	135	0.8	4.0	628K	590K	1.05	0.1
8	97	0.8	4.0	329K	863K	1.50	0.15
9	112	0.8	5.0	208K	375K	1.25	0.07
10	175	0.8	5.0	62K	310K	0.80	0.08
11	14	1.0	5.0	61K	52K	0.75	0.1
12	98	1.0	5.0	73K	88.5K	0.10	0.1

An industrial *RC* extraction tool Arcadia [8] is used based on the layout database to extract the power/ground net *RCs*, and generate the power/ground netlists. PowerMill [9] is used as the embedded simulator to report the current for fitness computation. Table 2 shows the design statistics. Columns 2, 3, and 4 show the number of primary inputs, process technology, and power supply voltage. The numbers of transistors and power net *RCs* are shown in Columns 5 and 6, respectively. Columns 7 and 8 show the specified threshold voltage drop and time period.

A power network simulator RailMill [10] is used to simulate the designs by the obtained input patterns. To evaluate the quality of the generated patterns, we compare the results with those produced by using the same simulator but applying two different pattern sets: (1) generated by [4], (2) functional verification vectors given by the designers. Table 3 shows the comparison for the 12 tested industrial designs. The numbers of the critical voltage drop cells found by (1)

Table 3: The comparison for voltage drop analysis.

Designs	# of critical cells			max. voltage drop (V)			#. of patterns		
	VIP	[4]	func. vec.	VIP	[4]	func. vec.	VIP	[4]	func. vec.
D1	1976	1300	525	0.09	0.09	0.09	168	152	323
D2	0	0	0	0.05	0.05	0.02	1800	1200	720
D3	1179	0	0	0.70	0.55	0.57	108	180	551
D4	3953	1520	119	0.47	0.56	0.45	360	250	1400
D5	344	285	0	0.60	0.58	0.53	696	400	2000
D6	1322	0	0	0.62	0.48	0.35	120	140	500
D7	816	700	18	1.10	1.10	1.05	240	360	1431
D8	2059	0	0	1.74	1.40	1.05	130	110	488
D9	1191	760	58	1.30	1.30	1.26	252	200	160
D10	2280	1170	0	0.86	0.84	0.34	840	680	823
D11	64	52	35	0.90	0.90	0.90	100	120	983
D12	0	0	0	0.04	0.04	0.04	228	160	650
Ave.	1265	482	63	-	-	-	420	329	836

Table 4: The comparison of CPU time of three techniques for voltage drop analysis.

Designs	CPU time (min.)							
	VIP				[4]			func. vec.
	solving graph	Power-Mill	RailM-ill	total	Power-Mill	RailM-ill	total	
D1	5	174	121	300	157	129	286	554
D2	3	308	5	316	204	6	210	17
D3	6	391	153	550	414	136	550	830
D4	8	380	187	575	410	174	584	1612
D5	6	278	54	338	163	51	214	1784
D6	5	313	100	418	365	90	455	4096
D7	5	210	88	303	205	90	295	443
D8	4	254	106	364	200	110	310	389
D9	4	106	10	120	89	11	100	924
D10	5	153	178	336	102	170	272	840
D11	3	72	4	79	91	4	95	17
D12	3	94	9	106	68	10	78	207
Ave.	5	228	85	317	206	82	288	976

VIP, (2) the technique of [4], and (3) functional vectors are shown in Columns 2, 3, and 4, respectively. Columns 5-7 show the level of the obtained maximum voltage drop. The total number of patterns used for running both PowerMill and RailMill is shown in Columns 8-10. Table 4 gives the CPU time for each technique. The CPU time consumed by *VIP* for (1) solving the bipartite graph for the maximum weight matching, (2) PowerMill simulation, (3) RailMill simulation is shown in Columns 2, 3, and 4, respectively. Column 5 gives the overall CPU time *VIP* consumes. The CPU time for technique [4] is shown in Columns 6-8. Column 9 gives the time for functional vectors used by RailMill simulation.

For designs D3, D6, and D8, *VIP* identifies a large number of critical voltage drop cells and none of them can be found by the other two approaches. For designs D2 and D12, no critical voltage drop appears and these two designs qualify the voltage drop test. For all the other designs, the number of critical cells found by *VIP* is much higher than those by the others. *VIP* also obtains higher or equal maximum voltage drop compared to those by [4] for 11 out of the 12 tested designs. For design D4, *VIP* obtains lower maximum voltage drop than [4]. However, the number of critical nodes found is 2.6 times higher. For the largest design D6, simulating the functional vectors needs 68.3 hours, and *VIP* needs only 7.0 hours.

5. Conclusion

We propose a robust input pattern generator for verifying reliability of deep submicron designs. Experimental results show that the use of the generated input patterns successfully identify cells that encounter critical voltage drops, which cannot be found by other sources of vectors. This generator can be included in the design cycle for accurate reliability analysis.

References

[1] A.-C. Deng, "Power Analysis for CMOS/BiCMOS Circuits," *International Symposium on Low-Power Electronics and Design (ISLPED)*, pp. 3-8, 1994.

[2] M. S. Hsiao, E. M. Rudnick, and J. H. Patel, "K2: An Estimator for Peak Sustainable Power of VLSI Circuits," *International Symposium on Low-Power Electronics and Design (ISLPED)*, pp. 178-183, August, 1997.

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

[4] Y.-M. Jiang, K.-T. Cheng, and A. Krstic, "Estimation of Maximum Power and Instantaneous Current Using a Genetic Algorithm," *Proc. of IEEE Custom Integrated Circuits Conference*, pp. 135-138, May 1997.

[5] Y.-M. Jiang and K.-T. Cheng, "Estimation of Maximum Power Supply Noise for Deep Sub-Micron Designs," *International Symposium on Low-Power Electronics and Design (ISLPED)*, pp. 233-238, August, 1998.

[6] B. L. Miller and D. E. Goldberg, "Genetic Algorithms, Tournament Selection, and the Effects of Noise," *Complex Systems*, pp. 193-212, June 1995.

[7] C. H. Papadimitriou and K. Steiglitz, "Combinatorial Optimization, Algorithms and Complexity," Reading, Prentice-Hall, 1982.

[8] SYNOPSYS, "Arcadia User Guide," August 1998.

[9] SYNOPSYS, "PowerMill User Guide", August 1998.

[10] SYNOPSYS, "RailMill User Guide," August 1998.

[11] K. Yoshida, "Speed Up of An Evaluation Speed 5~15 Times Faster by Using Hierarchical Power Supply Network," NTT LSI Lab.