Realistic Delay Modeling in Satisfiability-Based Timing Analysis

Luís Guerra e Silva, João P. Marques Silva, Luís Miguel Silveira and Karem A. Sakallah*

Cadence European Laboratories/INESC
Instituto Superior Técnico
R. Alves Redol, 9
1000 Lisboa, Portugal

*Electrical Engineering and Computer Science Dept.
Advanced Computer Architecture Lab.
University of Michigan
Ann Arbor, MI 48109-2122

Abstract

Circuit delay computation taking into account the existence of false paths represents a significant and computationally complex problem. Existing research work has focused mainly on path sensitization models and algorithms, and on gate and interconnect delay models. Nevertheless, work in these two main areas has evolved separately, and so most path sensitization models and algorithms assume very rudimentary gate and interconnect delay models. In this paper we propose a modeling framework for circuit delay computation as a sequence of instances of propositional satisfiability. This framework is used to capture several path sensitization models under the unit delay model. Moreover, several algorithms for propositional satisfiability are evaluated seeking to illustrate the computational challenges posed by the circuit delay computation problem. Finally, realistic delay models taking into account extracted interconnect delays and fanout data are incorporated into the proposed circuit delay computation framework in order to experimentally evaluate its applicability.

1 Introduction

Recent years have seen an ever increasing need for more accurate delay estimation methodologies in digital circuits, in particular due to the decisive role that delay estimation plays in determining limiting operating clock frequencies. A key problem associated with circuit delay estimation is the existence of false paths, which cause straightforward and efficient topological path analysis procedures to yield potentially conservative delay estimates. In contrast with topological delay estimation, solving the false path problem is computationally hard, being an NP-complete problem [15]. Research work on false paths has been extensive and, among others, several promising modeling and algorithmic approaches have been proposed [1, 2, 6, 9, 15, 15, 18, 22]. Despite this research effort, we believe that a comprehensive and unified computational study of different models and algorithms for solving the false path problem is still missing. In this paper we propose to partially solve this problem by studying a set of path sensitization criteria, under the assumption of floating mode circuit operation. This work is undertaken within a unified framework for solving the false path problem, which is based on propositional satisfiability models and algorithms. Furthermore, we explore more realistic delay modeling within the proposed framework, thus evaluating how SAT-based circuit delay computation is dependent upon the delay model considered. The computational study described in this paper is necessarily incomplete, since several relevant models and algorithms are not covered. Nevertheless, this study proposes an experimental procedure which can be generalized for those other models. Furthermore, we note that false paths can exist in both combinational and sequential circuits, even though in this paper we will exclusively consider combinational false paths.

The organization of the paper is as follows. We start with a few brief definitions, and then describe how to capture path sensitization using propositional satisfiability models. This section follows closely the work of [16], but a significantly simpler approach is used to derive the SAT models for the viability path sensitization criterion [15]. In addition, the SAT modeling approach of [16] is shown to be easily extended to other floating mode path sensitization criteria, namely static sensitization [5] and exact floating-mode sensitization [6]. Afterwards, in Section 4, the experimental procedure is described and experimental results are analyzed. Conclusions resulting from the proposed analysis are given in Section 5.

2 Definitions

In the following we shall assume a combinational circuit *M*, with *PI* primary inputs, *PO* primary outputs, composed of simple gates (AND, NAND, OR, NOR, NOT), where for a circuit node *f*, *c*(*f*) denotes the controlling logic value of *f* and *nc*(*f*) denotes the non-controlling logic value of *f*. For each circuit node *f*, *FI*(*f*) denotes the fanin nodes of *f* and *FO*(*f*) denotes the fanout nodes of *f*. The delay between the fanin node *g* of a circuit node *f* and *f* is denoted by *d*(*g*,*f*). A *complete path* (or simply a path) in a circuit is a sequence of nodes connecting a primary input to a primary output. A *partial path* denotes a connected sequence of nodes within a path.

The circuit delay computation problem consists of identifying the *largest* path delay value in a given circuit along which a signal transition is able to propagate from the primary input to the primary output of the path, under a chosen propagation model and for some primary input vector.

3 Path Sensitization Conditions

The conditions under which signals propagate from the primary inputs to the primary outputs in a combinational circuit are generally referred to as path sensitization conditions. Path sensitization conditions depend on the model of operation assumed for the circuit, in particular the different forms of stimuli on the primary inputs, and the waveform model assumed at each node in the circuit. Even though detailed and precise models can be considered, we shall restrict ourselves to floating mode operation, under which all nodes are assumed to undergo a single known transition, from an initial unknown value to a final *stable* known value. Most criteria defined under floating mode operation are conservative (e.g. viability [15] and the *exact* criterion under floating mode operation [6]), thus overestimating the circuit delay in some situations. Nevertheless, as shown in [15], viability and floating mode sensitization are *robust*, thus providing upper bounds on the circuit delay under the bounded gate delay model (i.e. assuming that each gate delay is within some interval [0, d_{max}]).

A characterization of different sensitization criteria for floating-mode operation for simple gates, under the assumption of single path sensitization, is illustrated in Figure 1, and identifies logical and temporal constraints on the side inputs to each node *x* in a path. $\tau(x)$ denotes the propagation delay of a signal transition to node *x* along a given path. The side inputs values can either be *controlling* (*c*) or *non-controlling* (*nc*). Symbol *C* indicates that a given circuit node value is unknown and may experience changes in time. For static sensitization, the side

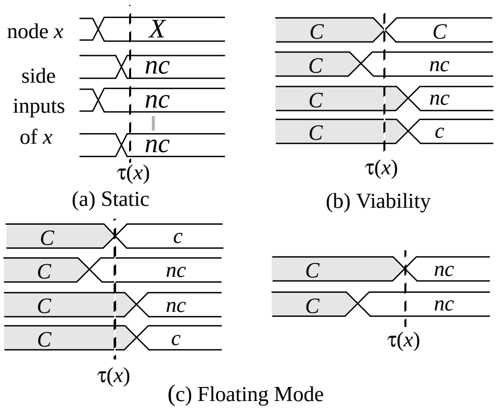

Figure 1: A characterization of path sensitization criteria

inputs are required to assume non-controlling values for propagation of a signal transition to occur. For viability, the side inputs are required to either be non-controlling or stabilize later that the node on the path. Finally, for floating-mode operation, it is assumed that the initial value of each primary input is unknown and changes to a known logic value at the specified arrival time. In the floating-mode sensitization criterion, a node *y* in the fanout of a node *x* stabilizes as a direct consequence of node *x* stabilizing if *x* is either the *earliest* controlling value to stabilize or all fanin nodes assume non-controlling values and *x* is the *latest* node to stabilize.

3.1 Satisfiability Models for Path Sensitization

In this section we show how to capture different path sensitization conditions using satisfiability models. Basically, the objective is to define conditions under which a given circuit node can stabilize at a given time instant.

Definition 1. We define the Boolean function $\chi^{f,t}(c)$ such that $\chi^{f,t}(c) = 1$ if and only if circuit node *f* stabilizes at a time greater than or equal to *t* when input vector *c* is applied to the primary inputs.

Clearly the definition of $\chi^{f,t}(c)$ leads naturally to the following observations.

Lemma 1. For a given input vector *c* and a circuit node *f*, the following conditions must hold:

1. $(\chi^{f,t}(c) = 1) \Rightarrow (\chi^{f,\tau}(c) = 1)$ for all $\tau \le t$.
2. $(\chi^{f,t}(c) = 0) \Rightarrow (\chi^{f,\tau}(c) = 0)$ for all $\tau \ge t$.

Moreover, for a given circuit delay Δ, and considering the set of primary outputs *PO*, we have the condition,

$$\sum_{g \in PO} \chi^{g,\Delta}(c) = 1 \qquad (1)$$

for some input vector *c*. This condition must be satisfiable to ensure that at least one path with delay Δ is sensitizable under the path sensitization model assumed. Furthermore, the definition of function $\chi^{f,t}(c)$ will differ for different sensitization conditions, as we will see in the following sections.

3.2 Viability

Given the interpretation of viability for simple gates in Figure 1-(b) and considering the generalization for multiple paths with the same delay values, we have the following conditions for a given circuit node *f* to stabilize at a time no earlier than a given delay *t* for some input vector *c*:

1. At least one fanin node *g* of *f*, with delay $d(g, f)$ between *g* and *f*, must stabilize at a time no earlier than $t - d(g, f)$. (This condition permits the existence of multiply sensitized partial paths.)
2. Furthermore, either a fanin node assumes a non-controlling value or it stabilizes at a time no earlier than $t - d(g, f)$, thus being passive regarding propagating a signal transition from *g* to *f*. Formally, we have,

$$\chi^{f,t}(c) = \sum_{g \in FI(f)} \chi^{g,t-d(g,f)}(c) \cdot \prod_{h \in FI(f)} (\chi^{h,t-d(h,f)}(c) + (h = nc(f))) \qquad (2)$$

which is basically equivalent to the viability condition proposed in [16]. Furthermore, observe that each function $\chi^{f,t}(c)$ can be viewed as a node in a combinational circuit. Given T. Larrabee's well-known mapping [14] from circuits into CNF formulas and from condition (1) it is straightforward to generate a CNF formula for capturing the sensitization conditions for all paths with delay no smaller than a given threshold delay Δ. It can easily be concluded that the CNF formula size is polynomial in the number of $\chi^{f,t}(c)$ functions considered.

3.3 Static Sensitization

For static path sensitization, using the model illustrated in Figure 1-(a), and again taking into account that multiple signal transitions can propagate from the fanin nodes to a given node *f*, we get the following definition of $\chi^{f,t}(c)$:

$$\chi^{f,t}(c) = \sum_{g \in FI(f)} \left(\chi^{g,t-d(g,f)}(c) \cdot \prod_{h \in FI(f) - \{g\}} (h = nc(f)) \right) \qquad (3)$$

which basically requires that at least one fanin node *g* of *f* to stabilize no earlier than $t - d(g, f)$ and such that the remaining fanin nodes assume non-controlling values. Clearly this condition must hold for any of the fanin nodes. Moreover, and as with viability, creating the CNF formula for static sensitization becomes immediate by using conditions (1) and (3).

3.4 Floating Mode Sensitization

In order to capture the exact path sensitization model under the floating mode of operation [6], the following observations are useful:

1. If the fanin node in any path being studied assumes a controlling value, then the floating mode condition is equivalent to viability.
2. Otherwise, all input nodes must be non-controlling. In this situation, propagation from any potential fanin node *g* only requires that a transition reaches that node, i.e. $\chi^{g,t-d(g,f)}(c) = 1$ and that all other inputs assume non-controlling values.

These observations lead to the following definition of $\chi^{f,t}(c)$:

$$\chi^{f,t}(c) = \sum_{g \in FI(f)} \chi^{g,t-d(g,f)}(c) \cdot \left((g = c(f)) \cdot \prod_{h \in FI(f)} (\chi^{h,t-d(h,f)}(c) + (h = nc(f))) + \prod_{h \in FI(f)} (h = nc(f)) \right) \qquad (4)$$

Observe that since a fanin node is required to satisfy $\chi^{g,t-d(g,f)}(c) = 1$, then at least one of these nodes will guarantee $\chi^{f,t}(c) = 1$ provided all inputs assume non-controlling values.

4 Experimental Results

The circuit delay computation algorithm consists solely of iteratively generating and solving instances of SAT for decreasing circuit delays starting from the largest topological path delay

Circuit [18]	LTP	Δ	Iterations	Viability		Floating-Mode	
				Clauses	Variables	Clauses	Variables
C432	17	17	1	942	330	1335	450
C499	11	11	1	585	226	661	247
C880	24	24	1	627	242	932	341
C1355	24	24	1	2053	704	3029	1025
C1908	40	37	7	5755	1932	8962	2936
C2670	32	30	5	6559	2255	10132	3402
C3540	47	46	3	5732	2027	7318	2532
C5315	49	47	5	5401	1887	8085	2773
C6288	124	123	2	13244	4239	19299	6257
C7552	43	42	3	2180	760	3386	1158
CBP.12.2	40	23	77	2081	707	3757	1261
CBP.16.4	44	27	89	1698	592	2957	1009
CLA.16	34	34	1	479	183	812	294
TAU92EX1	27	24	31	1213	450	2026	711
MULT-CSA	78	78	1	11415	3496	12309	3982

Table 1: Statistics for the benchmark circuits

and until a satisfiable instance of SAT is found, which corresponds to the circuit delay. All the satisfiability models described in the previous sections have been implemented and used for generating a large number of instances of SAT, each of which denotes the sensitization conditions for a given target circuit delay for a chosen circuit. In this section we provide results of a large number of satisfiability algorithms [3, 4, 7, 10-13, 19, 21][1] on these instances of SAT. For the results shown in Table 2 a SUN Sparc 5/85 machine, with 64 MByte of physical memory, was used. Furthermore, we also study the effects on computed circuit delay and algorithm execution time when a more realistic delay model is used. The results of this study are presented in Table 3 and were obtained on a SUN UltraSparc 1 with 384 MByte of physical memory.

4.1 Statistics for the Benchmark Circuits

One potential problem of SAT-based circuit delay computation algorithms is the size of the CNF formulas. In Table 1, we provide statistics for the different benchmark circuits, under the unit gate delay model. LTP denotes the largest topological path delay and Δ denotes the circuit delay under the viability and floating-mode path sensitization criteria. For the most significant path sensitization criteria, i.e. viability and floating-mode, Table 1 includes the largest number of clauses for any given iteration of the algorithm, as well as the number of variables in that situation. The number of iterations until a sensitizable path delay is found is also included. As can be concluded, the worst-case number of clauses is reasonable, given the original circuit sizes.

4.2 Results for Viability

The results for viability are shown in Table 2. (A more comprehensive set of results, involving other criteria can be found in [20].) Entries with a '*' indicate that the respective algorithm did not finish in less than 3,000 CPU seconds. It is interesting to observe that the vast majority of the generated instances of SAT are extremely easy to solve with most SAT algorithms. The exceptions to this rule are the less sophisticated SAT algorithms, in particular the Davis-Putnam procedure, which is unable to solve a large number of benchmarks. On the other hand, for a few benchmarks, only a few SAT algorithms are able to compute the circuit delay in a reasonable amount of time. In general, GRASP and rel_sat are by far the most efficient algorithms for solving this class of instances of SAT. Finally, we observe that

1. A description of SAT algorithms and an evaluation of their use for solving the circuit delay computation problem is given in [20].

Circuit	LTP/Δ	grasp	rel_sat	posit	tegus	h2r	csat	dpl
C432	17/17	0.03	0.03	0.02	0.13	0.18	0.10	0.45
C499	11/11	0.02	0.22	0.01	0.11	518.08	68.60	0.17
C880	24/24	0.04	0.02	0.01	0.11	0.08	0.10	0.15
C1355	24/24	0.12	0.09	0.19	0.32	2.27	0.50	*
C1908	40/37	0.26	1.43	0.48	3.29	4.37	5.40	107.37
C2670	32/30	2.83	3.21	*	*	*	*	*
C3540	47/46	0.54	0.29	0.69	4.72	2.81	3.10	519.63
C5315	49/47	1.27	0.54	1.03	*	6.44	5.90	*
C6288	124/123	11.19	42.79	*	86.73	206.50	203.90	*
C7552	43/42	0.17	0.44	0.05	1.02	0.63	0.80	3.27
CBP.12.2	40/23	1.53	5.63	0.73	23.95	17.67	15.40	1228.67
CBP.16.4	44/27	1.03	6.66	0.56	16.40	16.53	17.70	310.65
CLA.16	34/34	0.04	0.02	0.00	0.07	0.05	0.00	0.03
TAU92EX1	27/24	0.63	2.73	0.06	5.73	2.65	1.90	13.27
MULT-CSA	78/78	5.90	13.89	4.43	518.06	88.83	21.80	*

Table 2: CPU times for viability

for a few benchmarks and for viability using TEGUS we have been unable to reproduce the results of [16]. One possible justification is that the CNF formulas used in this paper and in [16] are necessarily different. Furthermore, the version of TEGUS used in [16] may have been optimized for the circuit delay computation problem, whereas the results of TEGUS included in the paper are obtained with the version that is available in SIS [21].

4.3 Realistic Delay Modeling

The previous results were obtained assuming a unit delay model for each gate. However, in general we need to consider more realistic delay models which should take into account the following constraints:

1. Different delay values for different types of gates.
2. Variation of delay with the number of fanouts/fanins.
3. Interconnect delay estimation (for circuits for which layout information is available).

Gate delays and delay variation with the number of fanouts/fanins can be easily modeled using information available from an IC library databook. Interconnect delay, however is hard to estimate for the benchmark circuits available, which are only described at the gate level. To obtain this information the benchmark circuits were mapped using the standard-cell library ECPD07 (ES2/Atmel)[2] [11], and the parasitic capacitances of the interconnect were extracted. For each gate, the (load-dependent) propagation delay (t_p) is given by:

$$t_p = t_{p_i} + dt_p \cdot C_l \qquad (5)$$

where t_{p_i} is the intrinsic propagation delay, dt_p is the differential (load-dependent) propagation delay and C_l is the load capacitance at the gate output. Further, this load capacitance is given by:

$$C_l = C_i + \sum C_g \qquad (6)$$

where C_i is the lumped interconnect capacitance and $\sum C_g$ is the sum of the input capacitances of all the fanouts. The interconnect resistance for this technology is very small, resulting in a negligible interconnect delay that has been discarded. However the interconnect capacitance is significant and was used to more realistically model the load-dependent propagation delay of each gate. We further note that all delays were computed with two-digit precision. Clearly, this gate delay model leads to a significantly larger number of path delays, which increases the number of iterations of the circuit delay computation algorithm.

2. Mapping to this library requires that each gate with more than 4 inputs has to be expanded into a sequence of gates each with no more than 4 inputs.

Circuit	Unit Delay				Realistic Delay			
	LTP/Δ	**grasp**	**rel_sat**	**tegus**	**LTP/Δ**	**grasp**	**rel_sat**	**tegus**
C432	20/20	0.02	0.02	0.04	20.20/19.90	1.26	0.75	25.31
C499	12/12	0.01	0.11	0.03	16.67/16.64	0.01	0.24	0.03
C880	24/24	0.02	0.01	0.02	18.59/18.59	0.01	0.01	0.03
C1355	25/25	0.06	0.04	0.08	22.38/21.97	0.21	3.87	2.33
C1908	42/39	0.17	0.95	0.91	32.44/29.68	75.35	#V	427.57
C2670	34/32	0.63	0.53	8.62	40.31/38.62	65.95	155.06	#B
C3540	47/46	0.33	0.15	6.07	45.19/43.10	1994.69	#V	#B
C5315	49/47	0.57	0.30	7.12	58.57/57.36	4.36	2.34	113.97
C6288	124/123	6.64	28.60	5.40	73.82/73.06	4672.90	#V	#B
C7552	43/42	0.11	0.28	0.26	38.57/36.39	142.44	50.85	285.34
CSA.16.4	41/22	0.08	11.25	2.27	36.00/20.10	0.36	24.92	14.16
CBP.12.2	40/23	0.71	3.29	7.72	22.65/13.94	25.10	67.45	301.74
CBP.16.4	44/27	0.44	15.20	7.05	25.84/16.52	4.31	39.29	70.74
CLA.16	34/34	0.02	0.00	0.01	21.68/21.65	0.08	0.25	0.20
MULT-CSA	78/78	3.35	3.35	4.82	81.10/80.87	3277.26	#V	#B

Table 3: CPU times for unit vs. realistic delay models, using viability

In Table 3 we present the CPU times for checking satisfiability for different SAT algorithms, obtained on the technology mapped circuits assuming both a unit delay model and a realistic delay model. For this experiment the path sensitization criterion used was viability. In the column for rel_sat, entries with “#V” indicate that the maximum number of variables (23,000) was exceeded. In the column for TEGUS, entries with “#B” indicate that the maximum number of backtracks was exceeded. As can be observed, the CPU times increase significantly because the number of iterations of the algorithm also increases accordingly. However, this added complexity signifies that we are now able to obtain much more accurate delay estimates for the circuit delay.

5 Conclusions

In this paper we propose a unified propositional satisfiability modeling and algorithmic framework for studying circuit delay computation methodologies. Different path sensitization models were considered and reasonably efficient results were obtained. Regarding the SAT algorithms used, one class of algorithms provides by far the most efficient and robust results. Both algorithms in this class (GRASP and rel_sat) use a number of search pruning techniques, which are shown to be particularly effective for solving circuit delay computation problems. Moreover, more realistic delay models, which take into account extracted interconnect delays and fanout data, were incorporated into the proposed modeling and algorithmic framework. Preliminary results suggest that the approach is still feasible, though necessarily more inefficient.

Additional work involves concluding the experiments described in this paper, as well as experimenting with a larger number of benchmarks. Another experiment that will be useful in identifying which SAT algorithms can actually be used in practice for circuit delay computation is to study families of circuits, for which we can increase the problem complexity by selecting larger members of that family. A well-known example is the family of carry-skip adders [15], where measures of size/complexity include the number of bits in the adder as well as the number of bits per block. Furthermore, additional experiments ought to include more detailed gate delay models with the goal of evaluating how large the CNF formulas can become, and how the different SAT algorithms handle larger CNF formulas, with more path sensitization options.

References

[1] P. Ashar, S. Malik and S. Rothweiler, “Functional Timing Analysis using ATPG,” in *Proceedings of the European Design Automation Conference*, 1993.

[2] R. I. Bahar, E. A. Frohm, C. M. Gaona, G. D. Hachtel, E. Macii, A. Pardo and F. Somenzi, “Algebraic Decision Diagrams and Their Applications,” in *Proceedings of the International Conference on Computer-Aided Design*, November 1993.

[3] P. Barth, “A Davis-Putnam Based Enumeration Algorithm for Linear Pseudo-Boolean Optimization,” Technical Report MPI-I-95-2-003, Max-Planck-Institut für Informatik, January 1995.

[4] R. Bayardo Jr. and R. Schrag, “Using CSP Look-Back Techniques to Solve Real-World SAT Instances,” in P*roceedings of the National Conference on Artificial Intelligence (AAAI-97)*, 1997.

[5] J. Benkoski, E. Vanden Meersch, L. Claesen and H. De Man, “Efficient Algorithms for Solving the False Path Problem in Timing Verification,” in *Proceedings of International Conference on Computer-Aided Design*, pp. 44-47, 1987.

[6] H.-C. Chen and D. H. Du, “Path Sensitization in Critical Path Problem,” *IEEE Transactions on Computer-Aided Design*, vol. 12, no. 2, pp. 196-207, February 1993.

[7] J. Crawford and L. Auton, “Experimental Results on the Cross-Over Point in Satisfiability Problems,” in *Proceedings of the 11th National Conference on Artificial Intelligence (AAAI-93)*, pp. 22-28, 1993.

[8] M. Davis and H. Putnam, “A Computing Procedure for Quantification Theory,” *Journal of the Association for Computing Machinery*, vol. 7, pp. 201-215, 1960.

[9] S. Devadas, K. Keutzer and S. Malik, “Computation of Floating-Mode Delay in Combinational Circuits: Practice and Implementation,” *IEEE Transactions on Computer-Aided Design*, vol. 12, no. 12, pp. 1924-1936, December 1993.

[10] O. Dubois, P. Andre, Y. Boufkhad and J. Carlier, “SAT versus UNSAT,” *Second DIMACS Implementation Challenge*, David S. Johnson and Michael A. Trick (eds.), DIMACS Series in Discrete Mathematics and Theoretical Computer Science, 1993.

[11] ES2 ECPD07 Library Databook, July 1995.

[12] J. W. Freeman, *Improvements to Propositional Satisfiability Search Algorithms*, Ph.D. Dissertation, Department of Computer and Information Science, University of Pennsylvania, May 1995.

[13] D. S. Johnson and M. A. Trick (eds.), *Second DIMACS Implementation Challenge*, DIMACS Series in Discrete Mathematics and Theoretical Computer Science, 1993. DIMACS benchmarks available in ftp://Dimacs.Rutgers.EDU/pub/challenge/sat/benchmarks/cnf.

[14] T. Larrabee, *Efficient Generation of Test Patterns Using Boolean Satisfiability*, Ph.D. Dissertation, Department of Computer Science, Stanford University, STAN-CS-90-1302, February 1990.

[15] P. C. McGeer and R. K. Brayton, *Integrating Functional and Temporal Domains in Logic Design: The False Path Problem and its Implications*, Kluwer Academic Publishers, 1991.

[16] P. McGeer, A. Saldanha, P. R. Stephan, R. K. Brayton and A. L. Sangiovanni-Vincentelli, “Timing Analysis and Delay-Test Generation Using Path Recursive Functions,” in *Proceedings of the International Conference on Computer-Aided Design*, November 1991.

[17] P. McGeer, A. Saldanha, R. K. Brayton and A. L. Sangiovanni-Vincentelli, “Delay Models and Exact Timing Analysis,” in *Logic Synthesis and Optimization*, T. Sasao (Ed.), 1993.

[18] J. P. M. Silva and K. A. Sakallah, “Efficient and Robust Test-Generation Based Timing Analysis,” in *Proceedings of the International Symposium on Circuits and Systems*, pp. 303-306, 1994.

[19] J. P. M. Silva and K. A. Sakallah, “GRASP—A New Search Algorithm for Satisfiability,” in *Proceedings of the International Conference on Computer-Aided Design*, November 1996.

[20] L. G. Silva, J. P. M. Silva, L. M. Silveira and K. A. Sakallah, “Satisfiability Models and Algorithms for Circuit Delay Computation,” in the ACM Workshop on Timing Issues in the Specification and Synthesis of Digital Systems (TAU), December 1997.

[21] P. R. Stephan, R. K. Brayton and A. L. Sangiovanni-Vincentelli, “Combinational Test Generation Using Satisfiability,” Memorandum no. UCB/ERL M92/112, Department of Electrical Engineering and Computer Sciences, University of California at Berkeley, October 1992.

[22] H. Yalcin and J. P. Hayes, “Hierarchical Timing Analysis Using Conditional Delays,” in *Proceedings of the International Conference on Computer-Aided Design*, November 1995.