Satisfiability Models and Algorithms for Circuit Delay Computation

Luís Guerra e Silva, João P. Marques Silva, Luís Miguel Silveira and Karem A. Sakallah
Cadence European Laboratories
IST/INESC
1000 Lisboa, Portugal
{lgs,jpms,lms}@algos.inesc.pt,karem@eecs.umich.edu

Abstract

The existence of false paths represents a significant and computationally complex problem in computing the delay of combinational and sequential circuits. Existing research work on circuit delay computation by taking false paths into account has focused on three main areas: gate delay models, path sensitization models, and associated algorithms. In this paper we conduct a comprehensive study on modeling circuit delay computation as a sequence of instances of propositional satisfiability. Several path sensitization models and gate delay models are studied. In addition we evaluate several algorithms for propositional satisfiability seeking to identify which are best suited for solving the circuit delay computation problem.

1 Introduction

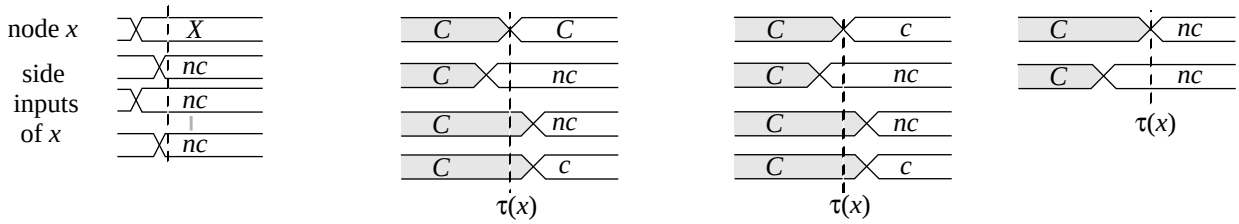
Recent years have seen an ever increasing need for more accurate delay estimation methodologies in digital circuits, in particular due to the decisive role delay estimation plays in determining limiting operating clock frequencies. A key problem associated with circuit delay estimation is the existence 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, 5, 8, 15-17, 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, within a unified framework for solving the false path problem which is based on propositional satisfiability models and algorithms. This study is necessarily incomplete, since several significant models and algorithms are not covered. However, it 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 each sensitization criterion. Section 4 briefly reviews some of the propositional satisfiability (SAT) algorithms used in this work. A comprehensive set of SAT algorithms is used, which cover a significant number of different algorithmic organizations proposed in recent years. Afterwards, in Section 5, the experimental procedure is described and experimental results are analyzed. Conclusions resulting from the proposed models, algorithms and experimental analysis are given in Section 6.

2 Definitions

In the following we shall assume a combinational circuit C, 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* delay value of a path in a circuit along which a signal transition is able to propagate, under a chosen propagation model, from the primary input to the primary output, under some primary input vector.



(a) Static (b) Viability (c) Floating-mode

Figure 1: A characterization of path sensitization criteria

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 [5]), 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. [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 (which is adapted from [19]), 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 inputs are required to assume non-controlling value 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, the primary input stimuli assumes 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 for at least one path with delay Δ to be 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,

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

which is basically equivalent to the simplified condition proposed in [16]. Furthermore, observe that each function $\chi^{f,\,t}(c)$ can be viewed as a node in a combinational circuit. Given the 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 be propagate from the fanin nodes to a given node *f*, we get the following definition of $\chi^{f,\,t}(c)$:

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

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 [5], 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 $\chi^{f,\,t}(c)$:

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

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.

3.5 Incorporating Pruning Information

In the previous sections we showed how different sensitization criteria can easily be captured as instances of propositional satisfiability. Nevertheless, and because path sensitization is a computationally hard problem, it is often useful to complete each generated instance with additional information which can be used for simplifying the search for a solution. One paradigmatic example is the structural information intrinsic to the problem being solved. For example, unique sensitization points (USP) (a well-known concept from testing [14, 18, 21]) can also be identified in circuit delay computation and used for effectively pruning the amount of search. Given the existence of a USP, the gate inputs not candidate to be in a sensitizable path must assume a non-controlling value.

4 Propositional Satisfiability Algorithms

Many search-based SAT algorithms have been proposed in recent years [3, 4, 6, 9-11, 20, 21], most of which consist of improvements to the classical Davis-Putnam backtrack search algorithm [7]. Despite their similarities, these algorithms present key distinguishing features in terms of search pruning ability.

1. The backtracking strategies may differ. For example, GRASP [20] and rel_sat [4] implement distinct non-chronological backtracking strategies.
2. Search may lead to conflicts (also known as dead ends). Necessary conditions for avoiding those conflicts can be captured as valid implicates of the boolean function associated with the given instance of SAT. Some algorithms can identify subsets of those implicates and include them in the CNF formula. For example, GRASP [20], rel_sat [4] and to some extent TEGUS [21] and NEMESIS [14], provide this ability. In TEGUS and NEMESIS, clauses are recorded during a preprocessing stage, often referred to as (static) learning.
3. The actual structure of how conflicts are identified also

provides useful insights on how to prune the search. For example, in GRASP the structure of conflicts is used for identifying a larger number of necessary assignments.

4. Restricted algebraic manipulation of the CNF formula may lead to significant simplifications. For example, CSAT [9] and SATO [12] use different preprocessing techniques for algebraic simplification.

5 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 delay 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 on these instances of SAT. For the results shown a SUN Sparc 5/85 machine, with 64 MByte of physical memory, was used. A description of the different SAT algorithms can be found in the bibliography.

The results for viability, static sensitization and floating mode sensitization are shown in Table 1**,** Table 2 and Table 3, respectively. For floating-mode, only a selected subset of the SAT algorithms was tested, since the remaining are known not to be competitive, given the results for viability and for static sensitization. In all tables, entries with a '*' indicate that the respective algorithm did not finish in less than 3,000 CPU seconds. As can be seen, static sensitization underestimates the circuit delay for C3540(N) and CLA.16, which agrees with the results of [18]. 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, sato [12] the basic Davis-Putnam procedure, which are unable to solve a significantly 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 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 shown are obtained with the version that is available in SIS [21].

6 Conclusions

In this paper we propose a unified propositional satisfiability modeling and algorithmic framework for studying circuit delay computation methodologies. Different models were considered and for all 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. Our practical experience indicates the choice of which path sensitization criterion to use does not significantly change the efficiency of the proposed approach.

Future research work mainly consists of incorporating more realistic gate delay models in the proposed framework, which will necessarily increase the accuracy of computed circuit delay estimates.

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] 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.

[6] 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.

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

[8] 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.

[9] O. Dubois, P. Andre, Y. Boufkhad and J. Carlier, "SAT versus UNSAT," *Second DIMACS Implementation Challenge*,

Circuit [18]	LTP/Δ	grasp[20]	rel_sat[4]	posit[10]	tegus[21]	ntab[6]	h2r[11]	sato[12]	c-sat[9]	dpl[3]
C432	17/17	0.03	0.03	0.02	0.13	0.10	0.18	0.02	0.10	0.45
C499	11/11	0.02	0.22	0.01	0.11	2.60	518.08	0.01	68.60	0.17
C880	24/24	0.04	0.02	0.01	0.11	0.10	0.08	0.02	0.10	0.15
C1355	24/24	0.12	0.09	0.19	0.32	0.70	2.27	0.20	0.50	*
C1908	40/37	0.26	1.43	0.48	3.29	5.10	4.37	19.99	5.40	107.37
C2670	32/30	2.83	3.21	*	*	*	*	*	*	*
C3540	47/46	0.54	0.29	0.69	4.72	3.10	2.81	9.38	3.10	519.63
C5315	49/47	1.27	0.54	1.03	*	4.90	6.44	*	5.90	*
C6288	124/123	11.19	42.79	*	86.73	1715.90	206.50	*	203.90	*
C7552	43/42	0.17	0.44	0.05	1.02	0.80	0.63	0.11	0.80	3.27
CBP.12.2	40/23	1.53	5.63	0.73	23.95	29.90	17.67	*	15.40	1228.67
CBP.16.4	44/27	1.03	6.66	0.56	16.40	45.90	16.53	*	17.70	310.65
CLA.16	34/34	0.04	0.02	0.00	0.07	0.00	0.05	0.01	0.00	0.03
TAU92EX1	27/24	0.63	2.73	0.06	5.73	0.11	2.65	36.44	1.90	13.27
MULT-CSA	78/78	5.90	13.89	4.43	518.06	5.50	88.83	*	21.80	*

Table 1: CPU times for viability

Circuit	LTP/Δ	grasp[20]	rel_sat[4]	posit[10]	tegus[21]	ntab[6]	h2r[11]	sato[12]	csat[9]	dpl[3]
C432	17/17	0.04	0.03	0.02	0.17	0.20	0.22	0.02	0.20	3.18
C499	11/11	0.01	0.08	0.02	0.10	0.10	627.47	0.01	74.6	0.15
C880	24/24	0.03	0.01	0.01	0.08	0.10	0.05	0.02	0.10	0.17
C1355	24/24	0.05	0.07	0.06	0.24	0.40	0.45	2.08	0.50	*
C1908	40/37	0.24	1.39	0.15	3.07	2.60	2.66	*	3.10	24.33
C2670	32/30	0.42	3.79	1291.82	*	*	*	*	*	*
C3540	47/46	0.61	0.35	0.45	2.08	2.80	2.59	*	2.90	246.43
C5315	49/47	0.99	0.30	0.61	90.06	5.40	4.18	*	4.00	*
C6288	124/123	17.94	197.5	2352.17	3.60	*	*	*	570.30	*
C7552	43/42	0.12	0.44	0.03	0.78	0.60	0.05	0.76	0.50	1.45
CBP.12.2	40/23	1.03	4.24	0.13	8.81	4.00	7.45	12.36	88.1	64.08
CBP.16.4	44/27	0.88	6.08	0.13	8.50	8.80	*	262.38	7.20	29.39
CLA.16	34/33	0.04	0.02	0.00	0.12	0.00	0.10	0.03	0.80	0.33
TAU92EX1	27/24	0.51	2.81	0.05	9.58	0.80	*	*	1.40	20.72
MULT-CSA	78/78	1.04	4.20	4.54	*	9.70	22.45	*	28.40	*

Table 2: CPU times for static sensitization

David S. Johnson and Michael A. Trick (eds.), DIMACS Series in Discrete Mathematics and Theoretical Computer Science, 1993.

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

[11] 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.

[12] S. Kim and H. Zhang, "ModGen: Theorem proving by model generation," in *Proceedings of the 12th National Conference of*

Circuit	grasp	rel_sat	posit	tegus	dpl
C432	0.07	0.03	0.04	0.20	3.47
C499	0.03	0.02	0.02	0.13	0.25
C880	0.12	0.02	0.02	0.13	0.58
C1355	0.11	0.09	0.34	0.46	*
C1908	0.73	0.62	0.84	65.21	96.62
C2670	5.78	5.31	*	*	*
C3540	1.36	0.42	0.97	645.52	537.87
C5315	2.63	1.18	6.01	*	*
C6288	55.49	29.17	*	*	*
C7552	0.25	0.11	0.11	4.24	7.78
CBP.12.2	14.11	6.11	89.68	208.92	*
CBP.16.4	12.57	4.50	3.80	109.30	*
CLA.16	0.08	0.02	0.01	0.11	0.65
TAU92EX1	1.24	0.60	0.22	206.84	80.53
MULT-CSA	61.38	13.49	27.64	468.52	*

Table 3: CPU times for floating mode sensitization

American Association on Artificial Intelligence (AAAI-94), pp. 162-167, 1994.

[13] Y. Kukimoto and R. Brayton, "Exact Required Timing Analysis via False Path Detection," in *Proceedings of the Design Automation Conference*, 1997.

[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, *Search Algorithms for Satisfiability Problems in Combinational Switching Circuits*, Ph.D. Dissertation, Department of Electrical Engineering and Computer Science, University of Michigan, May 1995.

[20] 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.

[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.