Minimal Delay Test Sets for Unate Gate Networks*

U. Sparmann†, H. Müller†, S.M. Reddy‡
† Computer Science Department, University of Saarland,
D 66041 Saarbrücken, Germany
‡ Department of ECE, University of Iowa,
Iowa City, Iowa 52242, USA

Abstract

We consider delay testing of a specific class of logic circuits, the so called 'unate gate networks (UGNs)', which are of importance for the realization of dynamic CMOS logic and in the field of on-line error detection. It has been shown earlier, that UGNs can be tested completely for delay faults with 'universal' test sets. This result even holds for designs which are not completely path delay testable, since the above test sets check the temporal correctness of a circuit by testing 'path systems' instead of single paths. A universal test set only depends on the computed function and thus, is valid for any unate gate network implementation of this function. This universal test property has to be paid by an increase in test set size, since a design independent test set will in general be larger than a design dependent one. In this paper, we show how to tailor a universal test set to a specific design in order to reduce its size maximally without loosing test quality. Experimental results demonstrate that the resulting delay test sets are very compact, and large savings in test set size of up to 96.71% can be achieved compared to the universal test set.

1 Introduction

Correct operation of logic circuits requires proper logic and timing behavior. Manufacturing defects and random variations in process parameters may cause propagation delays in the circuit to exceed their specification. Such defects are modeled as delay faults. Three approaches for testing delay faults have been suggested in the literature: gate delay test [BR83], (single) path delay test [Smi85], and path system delay test [KM94]. In this paper we will focus on the third approach, which subsumes path delay testing as a special case.

Delay testing of general combinational circuits is a hard problem. One of the main drawbacks here is that robust tests often only exist for a small fraction of the delay faults that need to be tested in order to verify the temporal correctness of a given design [RLP87, FFS91, SLCR95]. In [SMR94] it has been shown how to solve this problem for a special class of circuits, the so called 'unate gate networks (UGNs)'. By applying the methodology of path system testing it has been proven that any UGN can be completely checked for delay faults. Experimental results have shown that single path testing is in most cases not sufficient for verifying the temporal correctness of UGNs, i.e. path system testing is really necessary for a complete delay test of these circuits.

The result of [SMR94] has several important applications. The first one is for the testing of dynamic CMOS logic [KLL82, GD83, HGDT84] which is often applied in high speed circuits [PDY92] and must be structured as a unate gate network. Also self-checking circuits based on unordered codes are usually built as UGNs [DWB92, JW93]. Finally, the result can be applied in synthesis for testability. Any combinational circuit can be transformed into a UGN of the same depth with at most doubling its size[1] [Red73]. This implies a design for delay-testability method which does not compromise the speed of a given circuit implementation.

The delay test sets for UGNs derived in [SMR94] are *universal*, i.e. they only depend on the desired function but not its specific UGN implementation. As a consequence, the test set need not be changed when switching between different UGN realizations of the same function. This universal test property has to be paid by an increase in test application time, since a design independent test set will in general be larger than a design specific one. Experimental results for ISCAS89 [BBK89] and two-level MCNC [Yan91] benchmarks have shown that while for many functions the universal delay test sets are of reasonable size, there

*This work has been supported by DFG, SFB124 'VLSI design methods and parallelism'. The work of S.M. Reddy was done while visiting the University of Saarland, supported by Von Humboldt Foundation through a Senior Research Fellowship.

[1] As will be seen in the experimental results section, the overhead for transforming a general circuit into a UGN is usually much lower than this worst case bound.

are also some functions for which universal test set size is too large to be acceptable [SMR94].

In this paper we will address the question of how to handle UGNs for such problematic functions. A methodology will be developed which allows to tailor the universal test set to a specific design. Based on this methodology the size of the test set can be reduced maximally without loosing test quality, i.e. the reduced test set still checks the design under consideration completely for delay faults. The problem of reducing the size of the universal test set is solved in two steps. First, we characterize all necessary-to-test path systems for the specific design under consideration. Second, the task of reducing the universal test set maximally is transformed into a 'covering problem', where we select a minimum subset of the universal test set which is sufficient to check all these necessary-to-test path systems. Heuristics for solving this covering problem have been developed which allow to trade quality of the result against computation time. Experimental results on ISCAS89 and two-level MCNC benchmarks show that the resulting test sets are very compact, and reductions of up to 96.71% in test set size can be achieved.

The paper is organized as follows: In Section 2 we briefly review the results of [SMR94] and introduce some basic notions. The methodology for reducing the size of the universal test set maximally while still retaining complete delay testability for a given UGN is described in Section 3. Experimental results illustrating the practical applicability of our approach are given in Section 4.

2 Basic definitions and previous results

In this section we introduce some basic definitions and briefly review the results of [SMR94].[2] We restrict to single output circuits in the following, for multiple output circuits the theory can be applied to each output cone separately. As gate types we will consider only the elementary gates AND, NAND, OR, NOR, and NOT. The following definition introduces the class of circuits considered in this paper. As already mentioned in the introduction these circuits have several important applications, e.g. in the construction of dynamic CMOS logic and self-checking circuits.

Definition 1 *Let C be a combinational circuit.*

1. *For a path p in C the* inversion parity *of p is defined as the number of inverting gates* (NAND, NOR, NOT) *on p modulo 2.*
2. *C is called a* unate gate network (UGN) *iff for any pair (l_1, l_2) of lines in the circuit it holds that all paths from l_1 to l_2 have the same inversion parity.*

A function $f : \mathbf{B}^n \to \mathbf{B}$ is called *monotone* [Koh78] if for any two input vectors $v, w \in \mathbf{B}^n$ it holds that:

$$v \leq w \quad \Rightarrow \quad f(v) \leq f(w)$$

To explain the basic ideas of our approach clearly without being distracted by technical details we will restrict to UGNs computing monotone functions in the following. But note that this restriction is only for simplicity of presentation and can be easily removed[3]. The experimental results in Section 4 will be given for arbitrary UGNs.

Example 1 *Figures 1 and 2 show examples of unate gate networks implementing the monotone function $f = x_3 \vee x_2x_4 \vee x_1x_4 \vee x_2x_5$.*

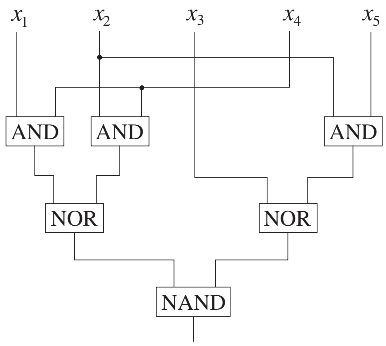

Figure 1: Unate circuit C'

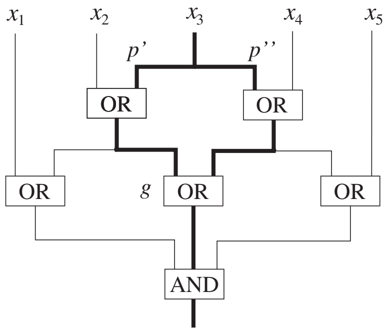

Figure 2: Unate circuit C''

Let C be an n-input circuit. The (worst case) delay of C for a given input vector v is denoted by $d(v)$. $d(C) := \max\{d(v) | v \in \mathbf{B}^n\}$ is the (worst case) delay of C. The purpose of delay testing is to verify for a given circuit C and a desired clock period τ whether $d(C) \leq \tau$.

[2]Universal test sets have also been studied in the context of stuck-at and stuck-open fault testing [Ake73, Red73, GJ88, CL94], for brevity we do not describe these results here.

[3]For functions which are not monotone the defintion of the universal test set is more technical [SMR94], the procedure for test set reduction remains the same [Mül96].

For a formal description of the universal delay test set corresponding to a monotone function f which can be used to verify $d(C) \leq \tau$ for any UGN C implementing f, we need the following definition.

Definition 2 *Let* $f : \mathbf{B}^n \rightarrow \mathbf{B}$ $(\mathbf{B} = \{0, 1\})$.

$v \in \mathbf{B}^n$ *is called a* true (false) vertex *of* f *iff* $f(v) = 1$ $(f(v) = 0)$.

A true (false) vertex v *of* f *is called a* minimal true (maximal false) vertex *iff there exists no true (false) vertex* w *such that* $w < v$ $(w > v)$.

The set of all minimal true (maximal false) vertices of function f *will be denoted by* $MINT_f$ $(MAXF_f)$.

The *universal delay test set* for a monotone function f is denoted by UTS_f and consists of the following set of two pattern sequences:

$$\begin{aligned} UTS_f \quad := \quad & \{< 00\ldots0, v > | v \in MINT_f\} \\ & \cup \{< 11\ldots1, v > | v \in MAXF_f\} \end{aligned}$$

In [SMR94] it has been shown that UTS_f is a complete delay test set for any UGN implementation of f, i.e.:

Theorem 1 *Let* f *be a monotone function and* C *an arbitrary UGN implementation*[4] *of* f. *Assume that* C *operates correctly*[5] *with respect to clock period* τ *for all tests from* UTS_f. *Then,* $d(C) \leq \tau$.

Example 2 *Consider the function*

$$f = x_3 \vee x_2x_4 \vee x_1x_4 \vee x_2x_5$$

of Example 1. For this function we have

$$\begin{aligned} MINT_f &= \{00100, 01010, 10010, 01001\}, \text{ and} \\ MAXF_f &= \{11000, 10001, 00011\}. \end{aligned}$$

The universal delay test set UTS_f *which checks any UGN implementation of* f*, like e.g. the circuits shown in Figures 1 and 2, completely for delay faults thus consists of seven two pattern sequences.*

As already mentioned in the introduction, the universal delay test set relies on path system testing instead of (single) path testing. Thus, it is even valid for implementations which are not path delay testable like e.g. the circuit shown in Figure 2. (The two paths p' and p'' drawn in bold in Figure 2 are neither robustly [RLP87] nor non-robustly [FFS91] testable for the $0 \rightarrow 1$ transition at their input x_3.)

[4] We assume that C has already been tested for static faults and thus, statically implements function f.

[5] We assume the usual test application procedure for delay testing with two pattern tests $< t_1, t_2 >$, i.e.: (1) The circuit is allowed to stabilize on the initialization vector t_1 by applying t_1 'long enough', and (2) t_2 is applied, and the response of the circuit is checked after time τ.

An interesting point to note is that for most circuit realizations, techniques for completely checking their temporal correctness do not exist. Testability for delay faults can only be guaranteed by design for testability modifications. From the above results it follows that such modifications are not necessary for UGN circuits.

The proof of Theorem 1 is based on the following lemma [SMR94] which will be of importance later:

Lemma 1 *Let* f *be a monotone function and* C *an arbitrary UGN implementation of* f.

(a) $d(v) \leq \tau$ $\forall v \in MINT_f \cup MAXF_f$ *implies that* $d(C) \leq \tau$.
(i.e. it suffices to only check the delays for input vectors from $MINT_f \cup MAXF_f$*.)*

(b) Let $init(v) := 00\ldots0$ $(11\ldots1)$ *iff* $v \in MINT_f$ $(v \in MAXF_f)$. *From the fact that* C *works correctly for test sequence* $< init(v), v >$ *in clock period* τ *it follows that* $d(v) \leq \tau$.
(i.e. the initialization vector $init(v)$ *creates the worst case initial condition for switching* C *under input vector* v.*)*

3 Reducing the size of the universal test set

In this section f denotes a monotone function with n inputs, and C is an arbitrary UGN implementation of f. We will show how to tailor the universal test set for f to the given design C, in order to minimize its size without loosing test quality. To achieve this goal we will proceed in two steps. First, we will determine the essential path systems which need to be checked in order to verify the temporal correctness of C (Section 3.1). After that subsets of the universal test set are characterized which are sufficient to check all these essential path systems (Section 3.2).

Let p be a physical path in circuit C leading from a primary input to a primary output. As usual in delay fault testing we will associate two logical paths with each physical path. A logical path is given as a tuple (p, t) with $t \in \{\uparrow, \downarrow\}$ being a transition $(\uparrow := 0 \rightarrow 1$, $\downarrow := 1 \rightarrow 0)$ at the primary input of p. $d(p, t)$ denotes the delay for propagating transition t over path p. The set of all logical paths in C is denoted by $\mathcal{LP}(C)$.

In path delay testing $d(C) \leq \tau$ is checked by measuring whether the delays of the logical paths of C are $\leq \tau$. This is not always possible, as illustrated by the following example.

Example 3 *Consider the two paths* p' *and* p'' *(drawn in bold in Figure 2) with respect to the* $0 \rightarrow 1$ *transition* *at their input x_3. At least one[6] of these paths has to be considered for path delay testing. But for none of them it can be checked (neither robustly nor non-robustly) whether its delay for the $0 \to 1$ transition is $\leq \tau$. This is due to the fact that both paths are always active at the same time for this transition. (Setting input $x_3 = 1$ implies value 1 at the output of gate g simultaneously over p' and p''.)*

The above problem can be solved based on the following observation: Since p' and p'' are always active simultaneously, it sufficies that only one of these two paths switches in time $\leq \tau$ for the circuit to function correctly. Thus, we will check that the system $[(p', 0 \to 1), (p'', 0 \to 1)]$ consisting of both logical paths switches fast enough, i.e. $\min\{d(p', 0 \to 1), d(p'', 0 \to 1)\} \leq \tau$.

3.1 Determination of the essential path systems

Definition 3 *A subset of logical paths $L \subset \mathcal{LP}(C)$ is called a* path system. *(To distinguish path systems from other sets, they will be written in brackets in the following, i.e. $L = [l_1, \ldots, l_\alpha]$, $l_i \in \mathcal{LP}(C)$.)*

The minimum of all path delays in system L is denoted by $d_{min}(L)$, i.e.:

$$d_{min}(L) := \min\{d(p,t) | (p,t) \in L\}$$

The following theorem relates the delay of the circuit for a given input vector to the minimum delays of path systems.

Theorem 2 *Let v be an arbitrary input vector to circuit C. There exists a set of path systems $PS(v) = \{L_i | L_i \subset \mathcal{LP}(C),\ i = 1, \ldots, k\}$, such that:*

$$d(v) = \max\{d_{min}(L) | L \in PS(v)\}$$

Since the formal proof of this theorem is quite technical it is deferred to the appendix.

Example 4 *Consider the unate circuit C'' of Figure 2 implementing the function $f = x_3 \vee x_2 x_4 \vee x_1 x_4 \vee x_2 x_5$ of Example 2. The assignment of values to the signal lines $(s_1, \ldots, s_{11})$ of this circuit for input vector $v_1 =$ (00100) is shown in Figure 3. The physical paths in this circuit are:*

$p_1 = s_1\text{-}s_8\text{-}s_{11}$, $p_2 = s_2\text{-}s_6\text{-}s_8\text{-}s_{11}$,
$p_3 = s_2\text{-}s_6\text{-}s_9\text{-}s_{11}$, $p_4 = s_3\text{-}s_6\text{-}s_8\text{-}s_{11}$,
$p_5 = s_3\text{-}s_6\text{-}s_9\text{-}s_{11}$, $p_6 = s_3\text{-}s_7\text{-}s_9\text{-}s_{11}$,
$p_7 = s_3\text{-}s_7\text{-}s_{10}\text{-}s_{11}$, $p_8 = s_4\text{-}s_7\text{-}s_9\text{-}s_{11}$,
$p_9 = s_4\text{-}s_7\text{-}s_{10}\text{-}s_{11}$, $p_{10} = s_5\text{-}s_{10}\text{-}s_{11}$.

[6] One of the two paths can be classified as 'robust dependent' [LSBS93, SLCR95].

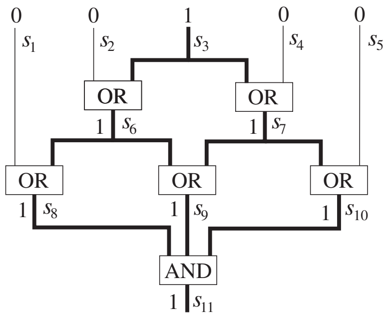

Figure 3: Example circuit C'' for input vector (00100)

For input vector $v_1 =$ (00100) the output value of the circuit is 1. In order to compute this output value, lines s_8, s_9, and s_{10} have to be switched to 1. s_8 (s_{10}) can only be switched over $s_3\text{-}s_6\text{-}s_8$ ($s_3\text{-}s_7\text{-}s_{10}$). s_9 can be switched either over $s_3\text{-}s_6\text{-}s_9$, or $s_3\text{-}s_7\text{-}s_9$. As a result, we obtain (path systems are written in brackets):

$$d(v_1) = \max\{d(s_3\text{-}s_6\text{-}s_8\text{-}s_{11}, \uparrow), d(s_3\text{-}s_7\text{-}s_{10}\text{-}s_{11}, \uparrow), d_{min}([(s_3\text{-}s_6\text{-}s_9\text{-}s_{11}, \uparrow), (s_3\text{-}s_7\text{-}s_9\text{-}s_{11}, \uparrow)])\}$$

i.e.:

$$PS(v_1) = \{\ [(p_4, \uparrow)];\ [(p_7, \uparrow)];\ [(p_5, \uparrow), (p_6, \uparrow)]\ \}$$

Similarly, we obtain for the remaining input vectors $v_2 = 01010$, $v_3 = 10010$, $v_4 = 01001$, $v_5 = 11000$, $v_6 = 10001$, and $v_7 = 00011$ from $MINT_f \cup MAXF_f$:

$PS(v_2) = \{\ [(p_2, \uparrow)];\ [(p_3, \uparrow), (p_8, \uparrow)];\ [(p_9, \uparrow)]\ \}$
$PS(v_3) = \{\ [(p_1, \uparrow)];\ [(p_8, \uparrow)];\ [(p_9, \uparrow)]\ \}$
$PS(v_4) = \{\ [(p_2, \uparrow)];\ [(p_3, \uparrow)];\ [(p_{10}, \uparrow)]\ \}$
$PS(v_5) = \{\ [(p_7, \downarrow)];\ [(p_9, \downarrow)];\ [p_{10}, \downarrow)]\ \}$
$PS(v_6) = \{\ [(p_3, \downarrow)];\ [(p_5, \downarrow)];\ [(p_6, \downarrow)];\ [(p_8, \downarrow)]\ \}$
$PS(v_7) = \{\ [(p_1, \downarrow)];\ [(p_2, \downarrow)];\ [(p_4, \downarrow)]\ \}$

The following lemma, which follows immediately from Theorem 2, shows how testing the delay for a given input vector can be done by path system testing.

Lemma 2 *Let τ be the desired clock period for circuit C. To verify that $d(v) \leq \tau$, it suffices to check that $d_{min}(L) \leq \tau\ \forall L \in PS(v)$.*

Based on Lemma 2 we can characterize a set $\mathcal{PS}$ of path systems such that it suffices to only test the path systems from $\mathcal{PS}$ in order to check C completely for delay faults.

Lemma 3 *Let:*

$$\mathcal{PS} := \bigcup_{v \in MINT_f \cup MAXF_f} PS(v)$$

Assume that $d_{min}(L) \leq \tau\ \forall L \in \mathcal{PS}$, then it follows that $d(C) \leq \tau$.

Proof:
Since $d_{min}(L) \leq \tau \; \forall L \in \mathcal{PS}$, it follows from the definition of $\mathcal{PS}$ and Lemma 2 that:

$$d(v) \leq \tau \;\; \forall v \in MINT_f \cup MAXF_f$$

Combining this with Lemma 1(a) finishes the proof. ■

The following simple observation which follows directly from the definition of d_{min} helps to further reduce the set of essential path systems. Let $L, L' \subset \mathcal{LP}(C)$, then:

$$L \supset L' \Rightarrow d_{min}(L) \leq d_{min}(L') \tag{1}$$

Theorem 3 *Let*

$$\mathcal{PS}_{min} := \{L' \in \mathcal{PS} | \nexists L \in \mathcal{PS} : \; (L \subset L') \wedge (L \neq L')\}$$

be the set of minimal path systems from $\mathcal{PS}$*.*

From the fact that $d_{min}(L') \leq \tau \; \forall L' \in \mathcal{PS}_{min}$ *it follows that* $d(C) \leq \tau$*.*

Proof:
We know that for every $L \in \mathcal{PS}$ there exists an $L' \in \mathcal{PS}_{min}$ such that $L \supset L'$. From Equation (1) and the fact that $d_{min}(L') \leq \tau \; \forall L' \in \mathcal{PS}_{min}$ it thus follows that $d_{min}(L) \leq \tau \; \forall L \in \mathcal{PS}$. Applying Lemma 3 finishes the proof. ■

3.2 Test set reduction

In order to reduce the size of the universal test set, we first need to characterize which essential path systems of circuit C are tested by a given input sequence of the universal test set.

Lemma 4 *Let* $v \in MINT_f \cup MAXF_f$*. If circuit* C *works correctly for input sequence* $< init(v), v >$ *in clock period* τ*, then* $d_{min}(L) \leq \tau \; \forall L \in PS(v)$*.*

Proof:
From Lemma 1(b) we know that if circuit C works correctly for input sequence $< init(v), v >$ in clock period τ, then $d(v) \leq \tau$. Since by Theorem 2 $d(v) = \max\{d_{min}(L) | L \in PS(v)\}$, we conclude that $d_{min}(L) \leq \tau \; \forall L \in PS(v)$. ■

The fact that a given path system $L \in \mathcal{PS}_{min}$ is usually checked by many test sequences of the universal test set, can be used to reduce the size of the universal test set as given next.

Theorem 4 *Consider a set of input vectors* $V \subset MINT_f \cup MAXF_f$ *such that:*

$$\bigcup_{v \in V} PS(v) = \mathcal{PS}_{min}$$

The test sequences from $\{< init(v), v > | v \in V\}$ *form a complete delay test set for* C*.*

Proof:
If C works correctly in clock period τ for all input sequences from $\{< init(v), v > | v \in V\}$ it follows from Lemma 4 and the fact that $\bigcup_{v \in V} PS(v) = \mathcal{PS}_{min}$ that:

$$d_{min}(L) \leq \tau \; \forall L \in \mathcal{PS}_{min}$$

Combining this with Theorem 3 yields the proof. ■

From Theorem 4 it follows that we can reduce the size of the universal test set maximally by searching for a minimum subset V of $MINT_f \cup MAXF_f$ such that $\bigcup_{v \in V} PS(v)$ still 'covers' all path systems from $\mathcal{PS}_{min}$.

Example 5 *Consider again the circuit of Figure 3. For this circuit we have (see Example 4):*

$$\begin{aligned}\mathcal{PS} = \{ \; & [(p_1,\uparrow)];\; [(p_2,\uparrow)];\; [(p_3,\uparrow)];\; [(p_4,\uparrow)];\\ & [(p_7,\uparrow)];\; [(p_8,\uparrow)];\; [(p_9,\uparrow)];\; [(p_{10},\uparrow)];\\ & [(p_3,\uparrow),(p_8,\uparrow)];\; [(p_5,\uparrow),(p_6,\uparrow)];\\ & [(p_1,\downarrow)];\; [(p_2,\downarrow)];\; [(p_3,\downarrow)];\; [(p_4,\downarrow)];\\ & [(p_5,\downarrow)];\; [(p_6,\downarrow)];\; [(p_7,\downarrow)];\; [(p_8,\downarrow)];\\ & [(p_9,\downarrow)];\; [(p_{10},\downarrow)] \;\}\end{aligned}$$

Since the path system $[(p_3,\uparrow),(p_8,\uparrow)]$ *is not minimal in* $\mathcal{PS}$*, it can be removed and we obtain:*

$$\mathcal{PS}_{min} = \mathcal{PS} \setminus \{[(p_3,\uparrow),(p_8,\uparrow)]\}$$

A minimum set $V \subset MINT_f \cup MAXF_f$*, which covers all systems from* $\mathcal{PS}_{min}$*, i.e.* $\bigcup_{v \in V} PS(v) = \mathcal{PS}_{min}$*, is:*

$$V = \{v_1, v_3, v_4, v_5, v_6, v_7\}$$

Thus, the universal test set can be reduced by the test sequence to input vector $v_2 = 01010$*. This is due to the fact that the path systems from* $\mathcal{PS}_{min}$ *corresponding to this vector* ($[(p_2,\uparrow)]$ *and* $[(p_9,\uparrow)]$) *are covered by the vectors* v_3 *and* v_4 *of* V *(see Example 4).*

4 Experimental Results

Searching for a minimum subset $V \subset MINT_f \cup MAXF_f$ which fulfills Theorem 4 reduces the universal test set maximally such that it still remains a complete delay test set for the given circuit. We have developed two different heuristics for computing such a subset V. The first heuristic produces near minimum solutions but can be expensive in terms of running time for circuits with very large path system sets $PS(v)$. The second heuristic allows to trade running time against quality of the result and thus, can be applied in cases were the first approach is too time consuming. Deciding which heuristic is appropriate for a given design can be done automatically by a structural analysis of the circuit. A detailed description of the algorithm is omitted here for space reasons and can be found in [Mül96].

circuit	$\vert\mathcal{LP}(C)\vert$	% rt LP	% gate increase
cs298	462	74.2 %	12.0 %
cs344	710	86.1 %	24.5 %
cs349	730	83.7 %	24.8 %
cs382	800	83.4 %	19.9 %
cs400	896	74.0 %	23.0 %
cs444	1070	54.8 %	21.6 %
cs510	738	98.8 %	6.2 %
cs526	820	84.6 %	9.5 %
cs641	3488	56.7 %	38.5 %
cs713	43624	2.7 %	55.3 %
cs820	984	99.6 %	1.5 %
cs832	1012	97.2 %	1.4 %
cs953	2312	99.6 %	4.8 %
cs1196	6196	57.8 %	33.6 %
cs1238	7118	50.4 %	33.3 %
cs1423	89452	31.6 %	37.0 %
cs1488	1924	97.5 %	4.7 %
cs1494	1952	96.4 %	4.7 %

Table 1: Statistics for ISCAS89 benchmarks

circuit	$\vert\mathcal{LP}(C)\vert$	% rt LP	% gate increase
apex1	13760	86.0 %	6.8 %
apex2	2860	73.4 %	13.3 %
apex3	35270	24.3 %	19.3 %
apex5	23848	35.0 %	24.3 %
bw	24380	2.6 %	65.0 %
clip	2694	22.0 %	56.2 %
duke2	5972	47.5 %	20.3 %
ex5	8318	25.0 %	14.1 %
inc	2884	13.6 %	69.8 %
misex1	1286	17.1 %	79.3 %
misex2	494	74.3 %	36.0 %
misex3	40578	22.7 %	50.3 %
misex3c	1856452	na[8]	77.1 %
pdc	3544	72.8 %	25.1 %
rd84	3994	22.3 %	54.2 %
spla	6274	69.8 %	13.5 %
squar5	930	15.4 %	73.5 %
table5	12694	70.2 %	15.0 %

Table 2: Statistics for MCNC benchmarks

To evaluate the efficiency of our test set reduction technique we conducted experiments on two sets of circuits. The first group of examples includes the combinational parts of the ISCAS89 benchmarks [BBK89] up to circuit cs1494. The second group consists of synthesized multi-level implementations[7] for the two-level MCNC benchmarks [Yan91]. The procedure of [Red73] was applied to transform these circuits into equivalent unate gate networks. As already mentioned in the introduction, this transformation does at most duplicate the number of gates in the circuit. The number of logical paths and the delay of the circuit remain the same. (For a more detailed description of the transformation process see [SMR94].)

Table 1 (Table 2) shows some statistics for the ISCAS89 benchmarks (a representative sample of the MCNC benchmarks). The circuit name and the number of logical paths in the circuit are given in the first two columns. Column three illustrates the path delay testability of the original non-unate circuits by giving the percentage of robustly testable logical paths (% rt LP). Test computation was done by the test generator of [RLP87] with the backtrack limit set to 1000. Column four gives the gate overheads for transforming the circuits into unate gate networks, which (by Theorem 1) are completely delay testable.

For most of the example designs the gate overhead for transforming the circuit is much less than duplication. Thus, the average increase in gate count for all example circuits considered is less than 33%, and for more than one third of the circuits the gate count increased by less than 15%. In addition, it should be noted that designing an optimal non-unate circuit and then transforming this circuit into an UGN, will usually not give the best possible UGN realization. Thus, the above gate overheads are upper bounds on the actual gate overheads implied by restricting to unate gate network realizations. Also, for circuits which are naturally designed as UGNs (like dynamic CMOS logic or self-checking circuits based on unordered codes) the test methodology given in this paper can be applied without inducing any gate overhead.

Tables 3 and 4 show the results of test set reduction for the transformed circuits. The circuit name and the number of logical paths are given in the first two columns. The third column lists the number of test sequences in the universal test set 'UTS'[9]. The fourth (fifth) column gives the number of test sequences in the reduced test set 'RTS' (the time needed to do the test set reduction on a SUN SPARC 10 workstation). Results which were obtained with the second heuristic are marked * in column five. All remaining circuits could be handled with the first heuristic. For both, the UTS as well as the RTS, the test sets were first obtained for each output cone separately and then combined using the technique of [CL94] in order to minimize the size

[7] Each circuit was synthesized by applying script.rugged in the SIS system [SSM+92].

[8] This circuit could not be handled by the test generator of [RLP87] because of its large number of paths.

[9] For the computation of the universal test set we assume that any input sequence $< init(v), v >$, $v \in MINT_f \cup MAXF_f$, can be applied to the circuit. This can be guaranteed by scanning the input register of the UGN and adding a set and reset capability to the output latches of the scan chain to generate the initialization vector $init(v)$ [PK93, SMR94].

circuit	$\vert\mathcal{LP}(C)\vert$	|UTS|	|RTS|	CPU-time
cs298	462	126	68	0:02
cs344	710	2798	112	0:23
cs349	730	2798	111	0:23
cs382	800	590	151	0:05
cs400	896	587	153	0:06
cs444	1070	587	191	0:06
cs510	738	628	161	0:06
cs526	820	310	159	0:03
cs641	3488	5098	318	1:39
cs713	43624	5463	1497	4:29*
cs820	984	895	162	0:09
cs832	1012	885	168	0:09
cs953	2312	2065	252	0:30
cs1196	6196	18324	740	6:40
cs1238	7118	15939	692	5:18
cs1423	89452	na	na	na
cs1488	1924	1572	378	0:17
cs1494	1952	1531	377	0:17

Table 3: Test set reduction for ISCAS89 benchmarks

circuit	$\vert\mathcal{LP}(C)\vert$	|UTS|	|RTS|	CPU-time
apex1	13760	20485	991	15:58
apex2	2860	4800	261	1:28
apex3	35270	8274	1331	7:10
apex5	23848	4995	895	2:08
bw	24380	351	318	0:10*
clip	2694	1674	315	0:22
duke2	5972	4723	406	2:12
ex5	8318	1647	676	0:45
inc	2884	393	224	0:06
misex1	1286	93	78	0:02
misex2	494	119	71	0:02
misex3	40578	15972	3482	16:30*
misex3c	1856452	36271	13117	17:00:00*
pdc	3544	4072	418	1:02
rd84	3994	5815	523	1:21
spla	6274	3328	653	1:01
squar5	930	108	72	0:01
table5	12694	40573	1333	37:59

Table 4: Test set reduction for MCNC benchmarks

of the overall test set.

As can be seen from these tables, the test set reduction depends extremely on the circuit under consideration. For most of the designs very large savings in test set size were obtained. (This is even true for those circuits which could only be handled with the second heuristic.) For 23 out of the 36 circuits the reduction is over 70%. The largest savings are for circuit table5 with a reduction of 96.71%. But there are also some circuits (like bw, misex1, or misex2 in Table 4) where the universal test set is already very compact and thus, only minor test set reductions could be achieved.

Comparing the reduced test set to the overall number of logical paths in the circuit, it can be seen that its size is always much less. As an example, circuit apex5 which has over 20,000 logical paths can be completely tested for delay faults with only 895 two pattern sequences. Thus, the reduced test set is very compact, i.e. its test sequences check a lot of logical paths (path systems) in parallel.

The time for test set reduction is less than 40 minutes for all the benchmarks. The only exception is circuit misex3c which took 17 hours because of its large number of paths. There is also one design (cs1423 in Table 3) for which the UTS could not be computed in acceptable running time because of its enormous size. The existence of such circuits is not surprising because of the following well known fact [Koh78]: the minimal true vertices (maximal false vertices) of a unate function f correspond to the prime implicants of f ($\overline{f}$). Thus, for functions where the set of prime implicants of f or $\overline{f}$ is extremely large and can not be computed in acceptable time our method is not applicable.

Even for a circuit like misex3c, where the size of the reduced test set (13,117) is much less than the overall number of logical paths (1,856,452), the RTS might still be considered too large for practical purposes. In path delay testing one common approach to handle such circuits with very large path numbers is to only check those paths with expected delay greater than a given threshold t. A similar methodology can also be applied for testing path systems, i.e. only those test sequences of the RTS are selected for testing purposes which check the necessary-to-test path systems $L \in \mathcal{PS}_{min}$ with expected delay $d_{min}(L)$ greater than t in the given design. Thus, the RTS can be further reduced to those test sequences which check the most critical parts of the given design.

5 Conclusion

We have studied delay testing for an important class of circuits the so called unate gate networks (UGNs). In [SMR94] it has been shown that any UGN can be checked completely for delay faults with universal test sets. These test sets only depend on the computed function and thus, are valid for any UGN implementation of this function. The above universal test property has to be paid by an increase in test set size, since a design independent test set will in general be larger than a design dependent one. In this paper it has been shown how to tailor a universal test set to a specific design in order to reduce its size maximally. This has been achieved without loosing test quality, i.e. the reduced test set still forms a complete delay test for the circuit under consideration. Experimental results have shown that the resulting test sets are very compact, and that the savings compared to the universal test set are usually very large (up to 96.71% for the examples considered).

Our results are of special interest in application areas where circuits are naturally built as UGNs, i.e. in the design of dynamic CMOS logic or self-checking circuits based on unordered codes. An additional application is in design for testability. Any combinational circuit can be transformed into an UGN of the same delay with at most doubling its size. This implies a design for delay-testability method which does not compromise the speed of a given circuit implementation.

Appendix: Proof of Theorem 2

For the proof of Theorem 2 it has to be shown that:

For every input vector v there exists a set $PS(v)$ of path systems such that:

$$d(v) = \max\{d_{min}(L)|L \in PS(v)\}$$

In order to proof the above equation we first need a simple lemma which allows to rewrite $\min\{\ldots\}$ formulas into $\max\{\ldots\}$ formulas. This lemma is essential because it allows us to transform delay formulas into the 'normal form' we need for Theorem 2.

Lemma 5 *Let $M_1, \ldots, M_k$ be subsets of a totally ordered set M, then:*

$$\min\{\max M_1, \ldots, \max M_k\} \\ = \max\{\min\{m_1, \ldots, m_k\}|m_i \in M_i\}$$

Proof:
(i) $\min\{\max M_1, \ldots, \max M_k\}$
$\leq \max\{\min\{m_1, \ldots, m_k\}|m_i \in M_i\}$:
This inequality follows directly from the fact that the set $\{\max M_1, \ldots, \max M_k\}$ is one of the sets $\{m_1, \ldots, m_k\}$, $m_i \in M_i$ $\forall i = 1, \ldots, k$, considered on the right hand of the inequality.

(ii) $\min\{\max M_1, \ldots, \max M_k\}$
$\geq \max\{\min\{m_1, \ldots, m_k\}|m_i \in M_i\}$:
Let $j \in \{1, \ldots, k\}$ be such that $\max M_j = \min\{\max M_1, \ldots, \max M_k\}$. Clearly, $\max M_j \geq \min\{m_1, \ldots, m_k\}$, $\forall \{m_1, \ldots, m_k\}, m_i \in M_i$. ■

For ease of presentation we will assume that the delays in C are only associated with the gates of the circuit. Let g be a gate in C with input signal s. The delay for propagating the transition $t \in \{\uparrow, \downarrow\}$ from input s to the output of g will be denoted by $d_{t,s}(g)$.

We are now in the position for proving Theorem 2.
Proof: (of Theorem 2)
The proof is done by induction on the number of gates in C.

INDUCTION BASE: C has no gates, i.e. C only consists of a single line s

$PS(v)$ contains only one path system which consists of exactly one logical path $(s, \overline{v} \to v)$, i.e. $PS(v) = \{[(s, \overline{v} \to v)]\}$.

INDUCTION STEP: C has $\alpha > 0$ gates

Let s be the output signal of C and g be the gate feeding s. W.l.o.g. we assume that g is an AND gate with two inputs s_1 and s_2, the other cases are handled analogously. The subcircuits of C feeding s_1 and s_2 are denoted by C_1 and C_2. The corresponding situation is depicted in Figure 4.

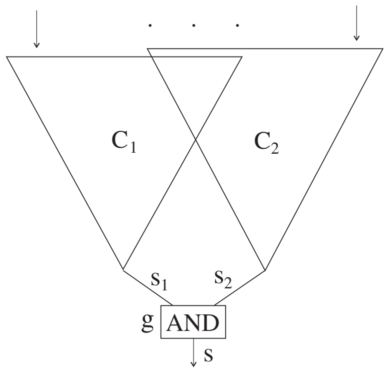

Figure 4: Structure of circuit C

Let $d_{C_1}(v)$ ($d_{C_2}(v)$) denote the (worst case) delay of circuit C_1 (C_2) under input vector v. Since the gate counts of C_1 and C_2 are less than α, we can inductively assume that Theorem 2 is fulfilled for these circuits, i.e. there exist sets of path systems $PS_{C_1}(v)$ and $PS_{C_2}(v)$ such that:

$$d_{C_1}(v) = \max\{d_{min}(L_1)|L_1 \in PS_{C_1}(v)\} \text{ and} \\ d_{C_2}(v) = \max\{d_{min}(L_2)|L_2 \in PS_{C_2}(v)\} \qquad (2)$$

For a set L of logical paths ending at an input signal of gate g, let

$$L \circ g := \{(l\text{-}g\text{-}s, t)|(l, t) \in L\}$$

denote the set of logical paths obtained by extending each path from L by gate g and its output signal s. Define:

$$PS_1(v) := \{L_1 \circ g|L_1 \in PS_{C_1}(v)\} \text{ and} \\ PS_2(v) := \{L_2 \circ g|L_2 \in PS_{C_2}(v)\} \qquad (3)$$

Depending on the input values of gate g for input vector v, we have to distinguish three cases:

CASE 1: One input value of g is controlling[10] (0) and the other input value is non-controlling (1)

W.l.o.g. assume that the value of s_1 (s_2) under input vector v is 0 (1). In this case:

$$d(v) = d_{C_1}(v) + d_{\downarrow,s_1}(g) \quad (4)$$

From equations (4), (2), and (3) it follows that:

$$\begin{aligned} d(v) &= \max\{d_{min}(L_1)|L_1 \in PS_{C_1}(v)\} + d_{\downarrow,s_1}(g) \\ &= \max\{d_{min}(L_1) + d_{\downarrow,s_1}(g)|L_1 \in PS_{C_1}(v)\} \\ &= \max\{d_{min}(L'_1)|L'_1 \in PS_1(v)\} \end{aligned}$$

Thus, we can set $PS(v) := PS_1(v)$.

CASE 2: Both inputs of g are set to the non-controlling value (1) by input vector v

In this case:

$$d(v) = \max\{d_{C_1}(v) + d_{\uparrow,s_1}(g), d_{C_2}(v) + d_{\uparrow,s_2}(g)\} \quad (5)$$

It follows from equations (5), (2), and (3) that:

$$\begin{aligned} d(v) &= \max\{\max\{d_{min}(L_1)|L_1 \in PS_{C_1}(v)\} + d_{\uparrow,s_1}(g), \\ &\qquad \max\{d_{min}(L_2)|L_2 \in PS_{C_2}(v)\} + d_{\uparrow,s_2}(g)\} \\ &= \max\{\max\{d_{min}(L'_1)|L'_1 \in PS_1(v)\}, \\ &\qquad \max\{d_{min}(L'_2)|L'_2 \in PS_2(v)\}\} \\ &= \max\{d_{min}(L')|L' \in PS_1(v) \cup PS_2(v)\} \end{aligned}$$

As a result, we can set $PS(v) := PS_1(v) \cup PS_2(v)$.

CASE 3: Both inputs of g are set to the controlling value (0) by input vector v

It follows that:

$$d(v) = \min\{d_{C_1}(v) + d_{\downarrow,s_1}(g), d_{C_2}(v) + d_{\downarrow,s_2}(g)\} \quad (6)$$

From equations (6), (2), and (3) we conclude that:

$$\begin{aligned} d(v) &= \min\{\max\{d_{min}(L_1)|L_1 \in PS_{C_1}(v)\} + d_{\downarrow,s_1}(g), \\ &\qquad \max\{d_{min}(L_2)|L_2 \in PS_{C_2}(v)\} + d_{\downarrow,s_2}(g)\} \\ &= \min\{\max\{d_{min}(L'_1)|L'_1 \in PS_1(v)\}, \\ &\qquad \max\{d_{min}(L'_2)|L'_2 \in PS_2(v)\}\} \end{aligned}$$

From Lemma 5 we know that for arbitrary subsets M_1 and M_2 of an ordered set M:

$$\begin{aligned} &\min\{\max M_1, \max M_2\} \\ &= \max\{\min\{m_1, m_2\}|m_i \in M_i\} \end{aligned}$$

Thus, we can rewrite the formula for $d(v)$ as:

$$d(v) = \max\{\min\{d_{min}(L'_1), d_{min}(L'_2)\}|L'_i \in PS_i(v)\}$$

Setting

$$PS(v) := \{L'_1 \cup L'_2 | L'_1 \in PS_1(v), L'_2 \in PS_2(v)\}$$

we obtain:

$$d(v) = \max\{d_{min}(L')|L' \in PS(v)\}$$

■

The above proof gives a constructive method for computing $PS(v)$ for a given input vector v. This is illustrated by the following:

Example 6 *Consider again our example circuit C'' for input vector $v = (00100)$ as shown in Figure 5.*

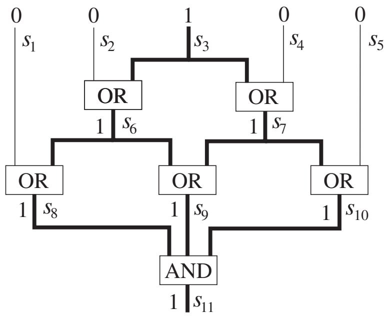

Figure 5: Example circuit C'' for input $v = (00100)$

Let C''_i denote the subcircuit of C'' with output line s_i, $i \in \{1, \ldots, 11\}$. $d_{C''_i}(v)$ denotes the (worst case) delay of C''_i for input vector v, and $PS_{C''_i}(v)$ is the set of path systems such that:

$$d_{C''_i}(v) = \max\{d_{min}(L)|L \in PS_{C''_i}(v)\}$$

By the proof of Theorem 2 (induction base) we obtain (path systems are written in brackets):

$$PS_{C''_3}(v) = \{\, [(s_3, \uparrow)] \,\}$$

Applying the induction step (Case 1) to C''_6 and C''_7 yields:

$$PS_{C''_6}(v) = \{\, [(s_3\text{-}s_6, \uparrow)] \,\},\ PS_{C''_7}(v) = \{\, [(s_3\text{-}s_7, \uparrow)] \,\}$$

For C''_8 and C''_{10} (C''_9) Case 1 (Case 3) of the induction step has to be applied, resulting in:

$$\begin{aligned} &PS_{C''_8}(v) = \{\, [(s_3\text{-}s_6\text{-}s_8, \uparrow)] \,\}, \\ &PS_{C''_{10}}(v) = \{\, [(s_3\text{-}s_7\text{-}s_{10}, \uparrow)] \,\},\ \textit{and} \\ &PS_{C''_9}(v) = \{\, [(s_3\text{-}s_6\text{-}s_9, \uparrow), (s_3\text{-}s_7\text{-}s_9, \uparrow)] \,\} \end{aligned}$$

Finally, we obtain by Case 2 of the induction step:

$$\begin{aligned} PS_{C''_{11}}(v) = \{\, &[(s_3\text{-}s_6\text{-}s_8\text{-}s_{11}, \uparrow)];\ [(s_3\text{-}s_7\text{-}s_{10}\text{-}s_{11}, \uparrow)]; \\ &[(s_3\text{-}s_6\text{-}s_9\text{-}s_{11}, \uparrow), (s_3\text{-}s_7\text{-}s_9\text{-}s_{11}, \uparrow)] \,\} \end{aligned}$$

[10] The controlling value for an AND, NAND (OR, NOR) gate is 0 (1). The non-controlling value is the complement of the controlling one.

References

[Ake73] S.B. Akers. Universal test sets for logic networks. *IEEE Transactions on Computers*, C-22:835–839, June 1973.

[BBK89] F. Brglez, D. Bryan, and K. Kozminski. Combinational profiles for sequential benchmark circuits. In *International Symposium on Circuits and Systems*, pages 1929–1934, 1989.

[BR83] Z. Barzilai and B.K. Rosen. Comparision of AC self-testing procedures. In *International Test Conference*, pages 89–91, 1983.

[CL94] B. Chen and C.L. Lee. A complement-based fast algorithm to generate universal test sets for multi-output functions. *IEEE Transactions on CAD*, pages 370–377, 1994.

[DWB92] K. De, C. Wu, and P. Banerjee. Reliability driven logic synthesis of multilevel circuits. In *Proceedings of the 1992 International Symposium on Circuits and Systems*, pages 1105–1108, 1992.

[FFS91] K. Fuchs, F. Fink, and M.H. Schulz. DYNAMITE: an efficient automatic test pattern generation system for path delay faults. *IEEE Transactions on CAD*, pages 1323–1335, 1991.

[GD83] N. Goncalves and H. DeMan. NORA: A race-free dynamic CMOS technique for pipelined logic structures. *IEEE Journal of Solid State Circuits*, 1983.

[GJ88] G. Gupta and N.K. Jha. A universal test set for CMOS circuits. *IEEE Transactions on CAD*, 7:590–597, 1988.

[HGDT84] L.G. Heller, W.R. Griffin, J.W. Davis, and N.G. Thoma. Cascode voltage switch logic: A differential CMOS logic family. In *IEEE International Solid State Circuits Conference*, 1984.

[JW93] N.K. Jha and S.-J. Wang. Design and synthesis of self-checking VLSI circuits. *IEEE Transactions on CAD*, pages 878–887, June 1993.

[KLL82] R.H. Krambeck, C.M. Lee, and H.-F.S. Law. High-speed compact circuits with CMOS. *IEEE Journal of Solid State Circuits*, 1982.

[KM94] W. Ke and P.R. Menon. Synthesis of delay-verifiable two-level circuits. In *European Design and Test Conference*, pages 297–301, 1994.

[Koh78] Z. Kohavi. *Switching and Finite Automata Theory*. McGraw-Hill, 1978.

[LSBS93] W.K. Lam, A. Saldanha, R.K. Brayton, and A.L. Sangiovanni-Vincentelli. Delay fault coverage and performance tradeoffs. In *30th Design Automation Conference*, pages 446–451, 1993.

[Mül96] H. Müller. Delay test of unate circuits. Master's thesis, Computer Science Department, University of Saarland, 66123 Saarbrücken, to appear 1996.

[PDY92] V. Peng, D.R. Donchin, and Y.-T. Yen. Design methodology and CAD tools for the NVAX microprocessor. In *IEEE Int. Conf. on Computer Design*, 1992.

[PK93] A.K. Pramanick and S. Kundu. Design of scan-based path delay testable sequential circuits. In *International Test Conference*, pages 962–971, 1993.

[Red73] S.M. Reddy. Complete test sets for logic functions. *IEEE Transactions on Computers*, C-22:1016–1020, November 1973.

[RLP87] S.M. Reddy, C.J. Lin, and S. Patil. An automatic test pattern generator for the detection of path delay faults. In *International Conference on CAD*, pages 284–287, 1987.

[SLCR95] U. Sparmann, D. Luxenburger, K.T. Cheng, and S.M. Reddy. Fast identification of robust dependent path delay faults. In *Proceedings of the 32nd Design Automation Conference*, pages 119–125, 1995.

[Smi85] G.L. Smith. Model for delay faults based upon paths. In *International Test Conference*, pages 342–349, 1985.

[SMR94] U. Sparmann, H. Müller, and S.M. Reddy. Universal delay test sets for logic networks. *To appear in: IEEE Transactions on VLSI Systems.* (Preliminary version available as technical report 5-30-94, Department of ECE, University of Iowa, Iowa City, IA 52242, USA).

[SSM+92] E. Sentovich, K. Singh, C. Moon, H. Savoj, R. Brayton, and A. Sangiovanni-Vincentelli. Sequential circuit design using synthesis and optimization. In *International Conference on Computer Design*, pages 328–333, 1992.

[Yan91] S. Yang. Logic synthesis and optimization benchmarks user guide (version 3.0). Technical report, Microelectronics Center of North Carolina, P.O. Box 12889, Research Triangle Park, NC 27709, January 1991.