Reducing the Complexity of Path Classification by Reconvergence Analysis

Paul Tafertshofer Andreas Ganz Manfred Henftling

Institute of Electronic Design Automation
Technical University of Munich
80290 Munich, Germany
e-mail: {Tafertshofer,Ganz}@regent.e-technik.tu-muenchen.de

Siemens AG, HL IT PE 5
P.O.-Box 801709
81617 Munich, Germany
e-mail: mah@earl.hl.siemens.de

Abstract — In this paper we present a new and efficient method for path classification, i.e. for determining the set of functional unsensitizable or robust dependent paths. In a pre-processing step, the new method computes a minimal set of reconvergence regions that need to be considered for path classification. Functional sensitization is only performed for path segments contained in these regions. Thus, the complexity for path classification can be reduced from the total number of paths in the circuit to the number of paths contained in the minimal set of reconvergence regions.

I. Introduction

The increasing complexity of digital logic systems and the continuing move towards higher levels of integration have steadily augmented the importance of dynamic testing. Two major fault models have been proposed: the gate delay fault model [1] and the path delay fault model [2]. In this paper we concentrate on the path delay fault model. A major disadvantage of the path delay fault model is the huge size of its fault dictionary. Therefore, in many practical cases it is extremely difficult or even impossible to enumerate all path delay faults [3] for test pattern generation. Recently, it has been shown that many paths in a circuit cannot influence its performance. Hence, such paths need not be considered for delay testing [4, 5, 6, 7].

In [4] *robust dependent (RD)* paths are introduced. It is shown that *RD* paths need not be considered for path delay testing since they cannot influence the circuit performance unless some testable path delay fault also occurs. The presented algorithm for computing a maximal set of *RD* paths, however, is computationally feasible only for small circuits. In [5] the concept of *functional unsensitizable (FU)* paths is proposed. This set of paths cannot cause a delay fault under all delay assignments. An approximative procedure for finding all *FU* paths based on the idea of mandatory (or necessary) assignments [8] is given. The approach tries to avoid enumeration of all paths by using functional unsensitizable prime segments. In [6] a common framework for *FU* and *RD* paths is established. It is shown that the class of *FU* paths forms a subset of the maximum robust dependent set as computed by the method of [4]. An algorithm for identifying near maximum robust dependent sets is proposed based on the algorithm suggested in [5].

In this paper we will address the problem of efficiently determining the sets *FU* and *RD*, respectively. We will refer to the task of partitioning the set of all paths into one of the above mentioned categories as *path classification*.

Despite of the various proposed methods for improving the efficiency of path classification, still huge numbers of paths have to be considered. In the worst case, the complexity still remains the number of paths in the circuit. So as to reduce this worst-case complexity, we propose the use of reconvergence analysis. As shown in section II B, almost all *RD* or *FU* paths pass through at least one reconvergence region in the circuit. Thus, the number of paths that need to be considered can be reduced without significant loss of accuracy by constraining our algorithm to reconvergence regions, i.e. the worst-case complexity is reduced to the number of paths found in a reconvergence region.

Reconvergence analysis is a well-known technique in CAD. It has been used to speed up fault simulation of stuck-at faults in [9] as well as for corolla based partitioning [10]. In [11] it is applied to detecting seven different types of reconvergent structures in order to find robust untestable paths in a circuit. We propose a method for reconvergence analysis that is based on the fundamental techniques of [9] but has been extended significantly to fit the specific needs of our problem. The presented procedure computes a minimal set of reconvergence regions, which are not fully covered by any other reconvergence region. Only these regions need to be considered for identifying *RD* or *FU* paths in a circuit.

The paper is organized as follows. Section II starts with a review of the algorithms presented in [5] and [6], respectively. Next, the benefits as well as the validity of our approach are shown. In section III we give an overview of the basic notation used throughout this paper. Section IV describes the applied methods of reconvergence analysis and its advantages. The new approach for computing *RD* as well as *FU* paths by considering only a minimal set of reconvergence regions is discussed in section V. Experimental results are given in section VI. Section VII concludes the paper.

II. Basic Idea

Before giving the details of the proposed approach, we first explain its basic idea and the resulting advantages with help of some simple examples.

A. Review of Path Classification

When determining the sets of *FU* and *RD* paths, we try to classify the set of all paths into two disjoint sets. One set that does not have to be considered for path delay testing and one that has to be further examined by an atpg tool.

The method of [5] for computing *FU* paths is based on the idea of mandatory (or necessary) assignments for path sensitization [8]. That is, all necessary signal assignments for propagating a desired signal transition on a given path from a primary input to a primary output are injected. This injection of necessary signal assignments is also referred to as *functional sensitization.* Next, all possible implications from these assignments are carried out. If a conflicting signal requirement is found by implication, the considered path is determined *FU*. The algorithm of [5] requires all off-path inputs of gates which have a non-controlling final value at its on-path signal to have a non-controlling final value as well.

The algorithm of [6] for identifying *RD* paths is an extension of the method of [5] which imposes additional conditions for propagating the transition. Additionally, non-controlling values are required at some subset of the off-path inputs at those gates which have a controlling final value at their on-path input signal. This subset is determined by a given order in the input signals. That is, all input signals that are smaller with respect to the given order than the on-path signal are assigned a non-controlling value whereas no requirements are issued for the remaining off-path signals. It is obvious that the method for identification of *RD* paths also finds all *FU* paths as it injects a superset of the necessary assignments for detection of *FU* paths. In [6] it is proven that the set of *RD* paths forms a superset of the set of *FU* paths.

Both methods can only compute a good lower bound on the complete sets of *FU* and *RD* paths since conflicts are only discovered by an implication phase following the injection of necessary assignments and no justification of the signal requirements is performed.

B. How Reconvergence Analysis Can Help

Even though the methods for determining *FU* and *RD* paths have been optimized for efficiency, they still have a worst case complexity of number of all paths in the circuit.

Functional sensitization according to [5, 6] can be started at the primary outputs (*PO*) and continues towards the primary inputs (*PI*). This is shown in figure 1a. Let's consider one of the dashed paths ending at a *PO*. During the traversal of the path, every gate on it is first functionally sensitized. Then, an implication phase is performed based on the required signal values for functional sensitization. If these necessary assignments cause a conflict in the signal assignment of the given circuit, all paths containing the currently considered path segment are marked *RD* or *FU*. This is indicated by the grey shaded cone in figure 1a. Let's assume that the conflicting signal assignment is caused by trying to sensitize a path through the gate marked with a flash. Obviously, this conflict originates in the fact that the considered path passes through the reconvergence region consisting of the white gates, i.e. we would already discover the conflict if only the path segments in the reconvergence region were sensitized. Since all paths leading from a *PO* to the reconvergence region (marked by dashed lines) are processed independently, the same reconvergence region is processed multiple times without need.

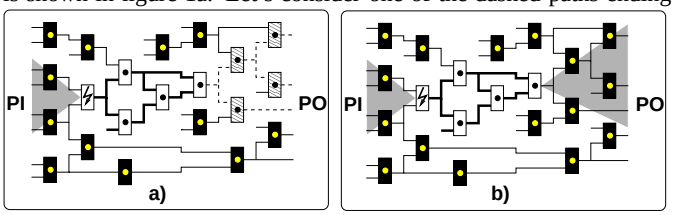

Figure 1: Benefits of confinement to reconvergence regions

ASP-DAC '97
0-89791-851-7$5.00

Therefore, we propose to limit functional sensitization to path segments contained in reconvergence regions. Thus, every path contained in the reconvergence region found in figure 1b is processed only once. If a conflict is found when trying to sensitize the gate marked with the flash, all paths containing the currently considered path segments are marked *RD* or *FU*. This is indicated by two grey shaded cones in figure 1b. So, the complexity of path classification is reduced to the number of paths in a reconvergence region. We will later show that in many practical cases this reduction is substantial.

Next, the validity of our approach is discussed. Intuitively, signal reconvergencies form a prerequisite for the existence of *RD* and *FU* paths, since circuits without reconvergencies (tree structures) are fully testable for robust (and nonrobust) delay faults. This dependency leads to the following theorem:

Theorem 1 *In an arbitrary combinational circuit a FU (RD) path, as determined by functional sensitization, is either caused by existence of a reconvergence region through which the path passes or by existence of a redundant stuck-at fault.*

Proof: Let's assume a given path does not pass through a reconvergence region but is *FU* or *RD*. Then justification of at least one of the required off-path signals causes a conflict. Since the considered path does not pass through a reconvergence region none of the other necessary assignments along the path can be the origin of this conflict. Thus, this conflict has to be caused by a redundant stuck-at fault at an off-path input. Due to this redundant stuck-at fault the required signal value cannot be justified.

As redundant stuck-at faults can easily be detected by stuck-at atpg, it is no deficiency of the proposed method that these paths remain undetected.

The proposed approach first computes a minimal set of reconvergence regions. Then, only path segments contained in this set are functionally sensitized according to the rules for identification of *FU* and *RD* paths, respectively.

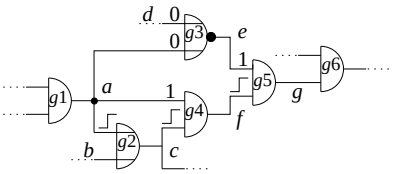

Figure 2: Example circuit

Let's assume that the reconvergence region determined by fanout stem a and reconvergent signal g in figure 2, which shows six gates that may be part of a larger combinational circuit[1], is contained in such a minimal set. We will now consider the path segment $(g1, g2, g4, g5)$ with rising transitions at signals a, c and f for functional sensitization according to the rules given in [5]. Consequently, we have to assign non-controlling values to the off-path signals of gates $g4$ and $g5$. We assign logical value 1 to signals e and a. Performing an implication step detects a conflicting assignment at signal a. Thus, we have found that every path containing gates $g1, g2, g4$ and $g5$ with rising transitions at signals a, c and f is *FU*. This may be a large number and it has been identified by local analysis of gates $g1, g2, g3, g4$ and $g5$ only.

The potential reduction in complexity is best illustrated with help of the circuit found in figure 3 which has been taken from [3]. It is the third member of a class of circuits which are generated by concatenating the circuit part marked by the shaded box n times. The number of structural paths in the circuit equals $3 \cdot 2^n - 2$ (for $n \geq 2$) whereas the number of reconvergence regions is only n. Since every reconvergence region contains two structural path segments, the proposed method only has to consider $2n$ path segments for functional sensitization whereas the approaches of [5, 6] have to cope with $3 \cdot 2^n - 2$ paths. That is, exponential complexity could be reduced to linear complexity for this extreme example.

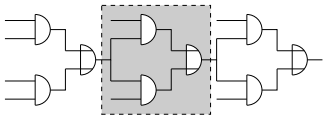
Figure 3: Circuit according to [3]

III. Basic Definitions

The combinational part of a given circuit is modelled by a directed, acyclic graph $G = (V, E)$. The edge set E denotes the set of all signals in the circuit and the node set V comprises all internal gates as well as the primary inputs and primary outputs. The set of all primary inputs is given by *PI* and the set of all primary outputs by *PO*. Fanout stems are modelled at the corresponding gate, i.e. the set of all fanout stems is given by $FO \subseteq V$. $RN \subseteq V$ denotes the set of all reconvergence nodes, i.e. the set of all gates that can be reached by at least two disjoint paths starting at the same fanout stem.

A path p in G is defined as a tuple $p = (x_i, \ldots, x_\alpha, x_\beta, \ldots, x_j)$ with nodes $x_i, \ldots, x_j \in V$ on the path being the components of the tuple and $(x_\alpha, x_\beta) \in E$. The set of all possible paths is denoted by *PM*. The set of structural paths *PS* comprises those paths $p \in PM$ that start at a primary input and end at a primary output, i.e. $first(p) \in PI$ and $last(p) \in PO$. The function $first(p)$ ($last(p)$) returns the first (last) gate on the path. A functional path $p_f \in PF$ is defined by choosing all transitions at all signals[2] on a structural path $p_s \in PS$.

We will further need the following definitions. A reconvergence region $rr(fo, rn)$, which is uniquely determined by its fanout stem $fo \in FO$ and its corresponding reconvergence node $rn \in RN$, is defined as the set of all disjoint paths starting at fo and ending at rn. The set of all nodes $n \in V$ contained in reconvergence region $rr(fo, rn)$ is given by $set(rr(fo, rn)) = \bigcup_{p \in rr(fo, rn)} set(p)$, where $set(p)$ returns the set of all nodes contained in the respective path p. The set of all reconvergence regions found in the circuit is given by *RR*.

The set $RN(fo)$ contains all reconvergence nodes belonging to a specific fanout stem fo and is defined as $RN(fo) = \{rn \in RN | \exists rr(fo, rn)\}$. $RR(fo) = \{rr \in RR | \exists rr(fo, rn)\}$ denotes all reconvergence regions having fo as their fanout stem.

The set $FO(rn)$ represents all fanout stems fo that have node rn among their corresponding reconvergence nodes, i.e. $FO(rn) = \{fo \in FO | \exists rr(fo, rn)\}$. $RR(rn) = \{rr \in RR | \exists rr(fo, rn)\}$ denotes all reconvergence regions having rn as their reconvergence node.

In the remaining sections of the paper, we only consider combinational circuits consisting of basic gates (NOT,AND,OR,NAND,NOR,XOR,XNOR,BUF). The presented methods for reconvergence analysis, however, remain applicable to circuits composed from arbitrary gates as well as the combinational part of sequential circuits.

IV. Reconvergence Analysis

As has been pointed out in section II B, functional sensitization is applied only within reconvergence regions. In order to minimize the required effort we propose a reconvergence analysis method that computes a minimal set of reconvergence regions. Since all reconvergence regions contained in this set are not fully covered by any other reconvergence region, we refer to these regions as *maximal reconvergence regions* rr_{max}.

Our method for determining a minimal set of maximal reconvergence regions consists of two subsequent steps. Both steps can be further divided into a forward as well as a backward phase.

A. Step 1

The forward phase of step 1 is similar to the method proposed in [9]. Starting at a fanout stem all immediate successors are given different markers. Then, in a levelized breadth first traversal of the corresponding output cone all markers are driven towards the primary outputs. Nodes that have been marked with at least two different markers are found to be reconvergence nodes. Thus, a set of reconvergence nodes $RN(fo)$ can be determined for every fanout stem $fo \in FO$. In a similar manner, a levelized breadth first traversal of the input cone belonging to a reconvergence node rn yields the set of all reconvergent fanout stems $FO(rn)$ for this node. This procedure is carried out during the backward phase.

During the forward phase, for every fanout stem $fo \in FO$ the set of corresponding maximal reconvergence nodes $rn_{max} \in RN_{max}(fo)$, which is contained in $RN(fo)$, is determined. A maximal reconvergence node rn_{max} with respect to a fanout stem fo is a node whose corresponding reconvergence region $rr(fo, rn_{max})$ is not fully covered by any other reconvergence region $rr \in RR(fo)$. We can easily compute these nodes if we interpret the graph $G = (V, E)$ as representing an order relation[3]. We start by determining all maximal elements with respect to the set M, which is given by all nodes contained in the union of all reconvergence regions belonging to fanout stem fo. It is determined by $M = \bigcup_{rn \in RN(fo)} set(rr(fo, rn))$. The set of maximal elements $\max(M)$ is obtained by evaluating $\max(M) = M \cap [\bigcap_{x \in M} \overline{pre(x)}]$ where $pre(x)$ denotes all predecessors of node x in G. The max-operation, as well as the min-operation used during the backward phase, can be carried out very efficiently on the graph G. In the following, superscript '1' indicates that the marked sets are computed in step 1.

[1] As a matter of fact, the circuit of figure 2, which has been selected for its simplicity, is redundant. More complicated examples without such redundancies could easily be given.

[2] For circuits containing only simple logic gates it is sufficient to define the signal transition at the *PO* and at the output of all XOR-gates (XNOR-gates) found on path p_s.

[3] As a matter of fact, the transitive closure of graph $G = (V, E)$ represents the corresponding order relation.

Thus, we get the set of maximal reconvergence nodes rn_{max} for fanout stem fo as $RN^1_{max}(fo) = \max(M)$ and the set of all rn_{max}, as computed by step 1, as $RN^1_{max} = \bigcup_{fo \in FO} RN^1_{max}(fo)$, respectively. The computed $rn_{max} \in RN^1_{max}$ correspond to closing reconvergence gates as defined in [9].

Next, the backward phase is performed. For every reconvergence node $rn_{max} \in RN^1_{max}$ the set of corresponding minimal fanout stems $fo_{min} \in FO^1_{min}(rn_{max})$, which is contained in $FO(rn_{max})$, is determined. A minimal fanout stem fo_{min} with respect to reconvergence node rn is a node whose corresponding reconvergence region $rr(fo_{min}, rn)$ is not fully covered by any other reconvergence region $rr \in RR(rn)$. The set of minimal fanout stems fo_{min} belonging to a given rn_{max} is computed by finding the minimal elements with respect to a subset $N = \bigcup_{fo \in FO(rn_{max})} set(rr(fo, rn_{max}))$. Set N contains all nodes which are contained in the union of all reconvergence regions $rr \in RR(rn_{max})$. The set of minimal elements $\min(N)$ is determined by evaluating $\min(N) = N \cap [\bigcap_{x \in N} \overline{suc(x)}]$ where $suc(x)$ denotes all successors of node x in G. Thus, we get the set of minimal fanout stems fo_{min} for reconvergence node rn_{max} as $FO^1_{min}(rn_{max}) = \min(N)$ and the set of all fo_{min} as computed by step 1 as $FO^1_{min} = \bigcup_{rn \in RN_{max}} FO^1_{min}(rn)$, respectively.

Let us now illustrate step 1 with help of the simple network found in figure 4 which can be seen as describing a part of the structure of a large circuit. The set of fanout stems is given by

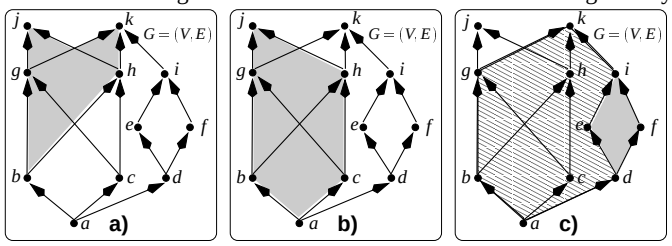

Figure 4: Step 1: a) forward phase, b) backward phase c) overlapped rr_{max}

$FO = \{a,b,c,d,g,h\}$ and the set of reconvergence nodes is found to be $RN = \{g,h,i,j,k\}$. Basic reconvergence analysis determines the set of all reconvergence regions RR in the network as $RR = \{(a,g),(a,h),(a,j),(a,k),(b,j),(b,k),(c,j),(c,k),(d,i)\}$[4]. During the forward phase of step 1 the set $RN^1_{max}(fo)$ is determined for every $fo \in FO$. This can be seen in figure 4a for fanout stem b. Computing the set of maximal elements with respect to $M = \{b,g,h,j,k\}$ (indicated by the shaded region) results in $\max(M) = \{j,k\}$, i.e. $RN^1_{max}(b) = \{j,k\}$. After all $fo \in FO$ have been processed, the set of maximal reconvergence nodes is given as $RN^1_{max} = \{i,j,k\}$. Next, during the backward phase all minimal fanout stems are computed beginning at every maximal reconvergence node $rn_{max} \in RN^1_{max}$. This procedure is illustrated in figure 4b for maximal reconvergence node j. Finding the set of minimal elements in $N = \{a,b,c,g,h,j\}$ (indicated by the shaded region) yields $\min(N) = \{a\}$, i.e. $FO^1_{min}(j) = \{a\}$. After all $rn_{max} \in RN^1_{max}$ have been processed, the set of minimal fanout stems is determined as $FO^1_{min} = \{a,d\}$. Thus, the complete set of maximal reconvergence regions RR^1_{max} as computed by step 1 is given by $RR^1_{max} = \{(a,j),(a,k),(d,i)\}$; i.e. the number of reconvergence regions that have to be considered could be reduced from 9 contained in the initial set RR to 3 in RR^1_{max}. In other words, the fully covered reconvergence regions $RR = \{(a,g),(a,h),(b,j),(b,k),(c,j),(c,k)\}$ were eliminated by step 1.

As can be seen in figure 4c, reconvergence region (d,i), which is fully covered by (a,k), is, however, still contained in RR^1_{max}. This is because neither $i \in RN(a)$ nor $(d \in FO(k)) \vee (d \in FO(j))$. Region (d,i) is eliminated by step 2 such that the final minimal set of maximal reconvergence regions is correctly computed as $RR_{max} = \{(a,j),(a,k)\}$.

[4]For reasons of brevity, we will denote the reconvergence region $rr(fo,rn)$ determined by fanout stem fo and reconvergence node rn by the ordered pair (fo,rn) throughout this example.

B. Step 2

Step 2 removes all reconvergence regions from RR^1_{max} that are fully covered by another reconvergence region but cannot be identified by step 1. Thus, the final set of reconvergence regions RR_{max} only contains maximal reconvergence regions rr_{max} which are not covered by any other $rr \in RR$. Therefore, RR_{max} is truly minimal and only these reconvergence regions have to be considered for identifying RD and FU paths.

Similarly to basic reconvergence analysis, at the beginning of step 2 for each minimal fanout stem $fo_i \in FO^1_{min}$ markers are driven towards the primary outputs . Next, for each reconvergence node $rn_j \in RN^1_{max}(fo_i)$ a levelized backward traversal of the corresponding reconvergence region $rr(fo_i, rn_j)$ is carried out. During the traversal, we check if other maximal reconvergence nodes rn_{max} are encountered and store such nodes in the unique-table `rn_hash`. If a minimal fanout stem fo'_{min} is encountered during the same traversal we check if any maximal reconvergence node $rn'_{max} \in RN^1_{max}(fo'_{min})$ has already been stored in `rn_hash`. If this holds true, region $rr(fo'_{min}, rn'_{max})$ is fully covered and can therefore be eliminated. For the example circuit of figure 4, after completion of step 2 reconvergence region $rr(d,i)$ is eliminated from RR_{max} such that the circuit is found to have only two maximal reconvergence regions $rr(a,j)$ and $rr(a,k)$. The set of minimal fanout stems is determined as $FO_{min} = \{a\}$ and the set of maximal reconvergence nodes as $RN_{max} = \{j,k\}$.

Experimental results for both steps are given in section VI.

V. A Reconvergence Analysis Based Approach to Identifying *FU* and *RD* Paths

In section II B we have shown that almost all FU and RD paths can be identified by executing functional sensitization only within reconvergence regions. If we restrict the set of reconvergence regions to the set of maximal reconvergence regions RR_{max}, however, a single phase of functional sensitization can only compute a lower bound on the number of FU and RD paths as determined by the methods of [5] and [6], respectively.

This observation is now explained with help of figure 5 and we will show how a two-phased approach for functional sensitization can help. In figure 5 you find a part of the graph G describing the structure of a given circuit with two reconvergence regions $rr_1(fo_1, rn_1)$ and $rr_2(fo_2, rn_2)$. Only reconvergence region rr_1 is contained in RR_{max} as rr_2 is fully covered by rr_1.

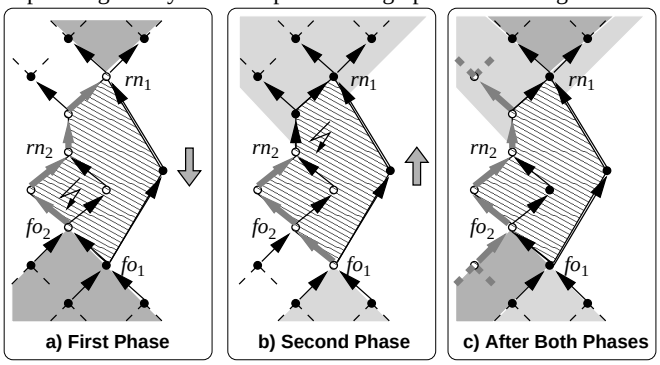

Figure 5: The two-phased approach

During the first phase of functional sensitization, maximal reconvergence regions are traversed starting from a maximal reconvergence node rn_{max} towards its corresponding minimal fanout stem fo_{min}. During this traversal of $rr_{max}(fo_{min}, rn_{max})$ necessary assignments for functional sensitization of path segments in rr_{max} are injected. If a conflict occurs at fanout stem fo_c during the implication step, all functional paths containing the path segment $(fo_c, \ldots, rn_{max})$ can be marked FU and RD, respectively.

Let's consider the maximal reconvergence region $rr_1(fo_1, rn_1)$ in figure 5a. If we try to sensitize the path segment $(fo_2, \ldots, rn_1)$ marked by grey arrows, a conflict may occur at fanout stem fo_2. Then, all functional paths containing the segment $(fo_2, \ldots, rn_1)$ are found FU and RD, respectively. This is indicated by shaded cones in figure 5a.

Let's assume that the conflict also occured if only path segment $(fo_2, \ldots, rn_2)$ were sensitized, i.e. $rr_2(fo_2, rn_2)$ causes the conflict.

Then, all functional paths containing the segment $(fo_2, \ldots, rn_2)$ need to be marked *FU* and *RD*, respectively. As $rr_2 \notin RR_{max}$, it is not processed explicitly. Thus, too few paths might be marked *FU* or *RD* if we only apply a single phase. This effect is avoided by means of a second phase.

In the second phase, maximal reconvergence regions are traversed starting from a minimal fanout stem fo_{min} towards its corresponding maximal reconvergence node rn_{max}. Again, necessary assignments for functional sensitization of the region's path segments are injected during the traversal. If a conflict arises at reconvergence node rn_c, all functional paths containing the path segment $(fo_{min}, \ldots, rn_c)$ are marked *FU* and *RD*, respectively. In the example of figure 5b, the conflict will now occur at rn_2 since we assumed it would happen if we only considered rr_2. Thus, all functional paths containing the path segment $(fo_1, \ldots, rn_2)$ are found *FU* and *RD*, respectively, which is indicated by shaded cones in figure 5b.

Figure 5c shows the situation after completion of both phases. It can be seen that even after completion of the second phase some *FU* or *RD* paths which pass through $rr_2(fo_2, rn_2)$ might remain undetected. Such a path has been marked by grey arrows in figure 5c. Even though this path passes through rr_2 it has not been considered by the two phases. This is because the path contains neither rn_1 nor fo_1.

A possible solution to this problem is storing information about the conflict found in phase one at node fo_2. Then, if a conflict also occurs at node rn_2 during the second phase, we can retrieve this information. After a final check if $rr_2(fo_2, rn_2)$ caused the conflict, all paths containing the segment $(fo_2, \ldots, rn_2)$ can be marked *FU* or *RD*. Thus, after completion of both phases all paths that contain the path segment $(fo_2, \ldots, rn_2)$ are marked correctly, and the non-maximal reconvergence region rr_2 has been considered implicitly. Of course, storing and retrieving information about conflicts induces some considerable overhead. Moreover, experimental results presented in section VI show that a faster approach without storing conflict information already discovers nearly all *FU* and *RD* paths. Therefore, we will from now on only examine the fast approach. This is particularly permissible, as the methods of [5] and [6] compute only a lower bound to the number of *FU* and *RD* paths in a given circuit, too.

VI. Experimental Results

The proposed methods have been implemented in C-language and all experiments have been carried out on a DECAlpha-Station 250-4/266. All ISCAS-85 and the combinational part of ISCAS-89 benchmark circuits have been investigated. For reasons of brevity, we only present results for computation of *FU* paths. Results for *RD* paths, which are very similar, may be found in [12].

Table 1 gives an overview on the proposed method for computing

Circuit	#*RR*	Step 1		Step 2	
		$\#RR^1_{max}$	t in s	$\#RR_{max}$	t in s
c2670	2422	655	0.1	88	0.3
c3540	23251	3413	0.5	456	6.7
c5315	3708	1536	0.4	434	0.9
c6288	224480	23449	3.0	392	16.7
c7552	8439	2947	1.1	382	1.5
s13207	9089	1413	1.9	719	2.5
s15850	12669	2891	4.0	1033	6.5
s35932	13015	6646	26.6	2934	37.3
s38417	20662	8802	21.2	2978	35.6
s38584	18990	7578	19.0	5020	30.9

Table 1: Computation of maximal reconvergence regions

the minimal set of maximal reconvergence regions RR_{max}. Column 2 contains the number of all reconvergence regions #*RR* found in the circuit. Column 3 lists the number of maximal reconvergence regions $\#RR^1_{max}$ as computed by step 1. The required time is given in column 4. The final number of maximal reconvergence regions $\#RR_{max}$ as generated by step 2, is found in column 5. Column 6 displays the required time. $\#RR_{max}$ clearly shows a significant reduction in the number of reconvergence regions that need to be considered for the subsequent reconvergence analysis based approach for identifying *FU*(*RD*) paths.

In table 2, we compare the results generated by the first phase of our approach for identifying *FU* paths with the results gained by the method of [5], which was re-implemented for experimental purposes[5]. Column 2 gives the number of *FU* paths as computed by the re-implemented version of [5] and column 3 lists the required time. The number of *FU* paths as computed by the first phase of the proposed approach and required run time are found in columns 4 and 5, respectively. The results found in table 2 indicate that often the first phase alone is sufficient for finding the great majority of *FU*(*RD*) paths in a circuit.

[5]Comparing the results of our method with a re-implemented version guarantees that the reported results are independent of the used data structures for executing implication.

Circuit	[5]		first phase	
	fu_c	t_c in s	fu_1	t_1 in s
c1355	6776160	57	6345888	15
c2670	1194077	20	1100383	8
c3540	42744454	7530	41851110	6557
c5315	2107569	74	2103051	39
c7552	1006505	250	1005051	240
s13207	1956761	1161	1554060	162
s15850	277312406	37032	222248884	7346
s35932	276137	348	273417	313
s38417	976628	423	975572	285
s38584	1463359	1639	1035087	514

Table 2: Comparing the approach of [5] and the first phase (*FU*)

Table 3 compares the method of [5] with the complete two-phased approach. These results clearly demonstrate that the presented two-

Circuit	[5]		two-phased	
	fu_c	t_c in s	fu_t	t_t in s
c1355	6776160	57	6776160	98
c2670	1194077	20	1159903	19
c3540	42744454	7530	42285726	8758
c5315	2107569	74	2105947	93
c7552	1006505	250	1006471	326
s13207	1956761	1161	1929259	299
s15850	277312406	37032	273605500	11753
s35932	276137	348	274457	493
s38417	976628	423	976148	452
s38584	1463359	1639	1418899	1295

Table 3: Comparing the method of [5] and the two-phased approach (*FU*)

phased approach is capable of identifying the vast majority of *FU*(*RD*) paths in a circuit. Run time for relatively small circuits is not reduced by the new approach. These circuits, however, could already be processed efficiently by the methods of [5] and [6]. Yet, speed-ups of up to a factor 10 are achieved for large circuits with many paths. For example, the time required for identifying *RD* paths in the circuit containing the most paths (s15850) could be reduced by 67.1%. Circuit c6288 could not be processed as our current implementation requires the generation of a path array for path counting purposes. Please note that this is no deficiency of the proposed method as both reconvergence analysis and functional sensitization can be carried out for this circuit. As mentioned before, good approximative results are already achieved in comparatively very short time by sole application of the first phase.

VII. Conclusion

We have presented a new and fast method for identifying both *FU* and *RD* paths in combinational circuits. It consists of a reconvergence analysis phase and a subsequent approach for identifying *FU* as well as *RD* paths by local analysis of reconvergence regions. First, the proposed reconvergence analysis phase is used to compute a minimal set of maximal reconvergence regions. Then, the new approach remains limited to path segments contained in this set. Thus, less paths have to be treated explicitly and *FU* as well as *RD* paths in a circuit can be determined very fast. For a special class of circuits the computational complexity could be reduced from exponential to linear complexity. The presented experimental results clearly show that the new approach is particularly suitable for large circuits with many paths. These circuits can only be processed with great expenses by other methods.

References

[1] Z. Barzilai and B. K. Rosen, "Comparison of ac self–testing procedures," in *IEEE International Test Conference (ITC)*, pp. 89–94, Oct. 1983.

[2] G. L. Smith, "Model for delay faults based upon paths," in *IEEE International Test Conference (ITC)*, pp. 342–349, Nov. 1985.

[3] I. Pomeranz and S. M. Reddy, "An efficient non–enumerative method to estimate path delay fault coverage," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pp. 560–567, Nov. 1992.

[4] W. K. Lam, A. Saldanha, R. K. Brayton, and A. L. Sangiovanni-Vincentelli, "Delay fault coverage, test set size, and performance trade offs," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems CAD*, vol. 14, pp. 32–44, Jan. 1995.

[5] K.-T. Cheng and H.-C. Chen, "Delay testing for non-robust untestable circuits," in *IEEE International Test Conference (ITC)*, pp. 954–961, Oct. 1993.

[6] U. Sparmann, D. Luxenburger, K.-T. Cheng, and S. M. Reddy, "Fast identification of robust dependent path delay faults," in *ACM/IEEE Design Automation Conference (DAC)*, vol. 32, pp. 119–125, June 1995.

[7] M. A. Gharaybeh, M. L. Bushnell, and V. D. Agrawal, "Classification and test generation for path-delay faults using single stuck-fault tests," in *IEEE International Test Conference (ITC)*, pp. 139–148, Oct. 1995.

[8] T. Kirkland and M. R. Mercer, "A topological search algorithm for atpg," in *ACM/IEEE Design Automation Conference (DAC)*, vol. 24, pp. 502–508, June 1987.

[9] F. Maamari and J. Rajski, "A method of fault simulation based on stem regions," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems CAD*, pp. 212–220, Feb. 1990.

[10] S. Dey, F. Brglez, and G. Kedem, "Corolla based circuit partitioning and resynthesis," in *ACM/IEEE Design Automation Conference (DAC)*, (Orlando), pp. 607–612, 1990.

[11] W. C. Wu, C. L. Lee, and J. E. Chen, "Identification of robust untestable path delay faults," in *IEEE Asian Test Symposium (ATS)*, pp. 229–235, 1995.

[12] P. Tafertshofer, A. Ganz, and M. Henftling, "Reducing the complexity of path classification by reconvergence analysis," Tech. Rep. TUM-LRE-96-3, Technical University of Munich, Oct. 1996.