On the Evaluation of Arbitrary Defect Coverage of Test Sets

Ankur Jain†, Vamsi Boppana‡, Michael S. Hsiao‡ and M. Fujita†
† Department of Electrical & Computer Engineering, Rutgers University, Piscataway, NJ
‡ Advanced CAD Research, Fujitsu Labs of America, Inc., Sunnyvale, CA

Abstract

Efficient methods to evaluate the quality of a test set in terms of its coverage of arbitrary defects in a circuit are presented. Our techniques rapidly estimate arbitrary defect coverage because they are independent of specific, physical, fault models. We overcome the potentially explosive computational requirements associated with considering all possible defects by implicitly evaluating multiple faults (of all types) simultaneously and by exploiting the local nature of defects. Our experiments show that a strong correlation exists between stuck-at fault coverage and defects whose behavior is independent of the input vectors. Our techniques are capable of identifying regions in the circuit where defects may escape the test set. We also demonstrate how the chances of detection of an arbitrary defect by a test set vary when a single stuck-at-fault within the vicinity of that defect is detected multiple times by the test set.

I Introduction

Our work draws motivation from the fact that the grading of a test set can only be as good as the underlying fault model. Hence, the quality of a given test set can only be established by evaluating its coverage of truly arbitrary defects in a circuit. It has been suggested previously that it is critical to evaluate not only the single stuck-at-fault coverage, but also the coverage of other types of possible defects (such as bridging faults, multiple stuck-at faults, transition faults, stuck-open faults etc.) that can occur in a design or fabricated chip [1, 2, 3, 4, 5, 11]. These requirements arise from the quality demands imposed by the increasingly complex design and manufacturing processes of present day.

One alternative for overcoming the potential quality problems associated with the use of a single specific fault model is to generate vectors based on models that are functional [6]. Examples of these approaches include universal tests [7, 8], c-testable arrays [6], functional test generation targeting multiprocessor faults [6] and tests based on unimplemented blocks [9]. These methods typically do not use any information about the structure of the circuit. The other approach to solve this problem is to evaluate test sets based on physical fault models different from the ones used in the generation of test vectors [3, 4, 10, 1]. For example, detections from a combination of bridging faults, stuck-open faults and stuck-at faults are used to evaluate the coverage of a given test set. Finally, analysis of pattern-dependent and timing-dependent failures in an experimental test chip was presented in [11]. Results on an experimental test chip for the detection of pattern-dependent failures were collected, and use of multiple detections for improving defect coverage was discussed.

In contrast, in this paper, we present efficient methods to evaluate a test set for its coverage of arbitrary defects in a circuit without assuming any particular physical fault model. Our technique is not a purely functional one because it uses critical information about the structure of the circuit in its computation. Our technique is based on the observation that defects (even when they are affecting multiple nodes in the circuit) are usually “local” in nature. For instance, a bridging fault usually occurs between nodes in nearby surroundings. This observation motivates us to model arbitrary defects as errors in a set of structurally close nodes or a vicinity.

A defect in a vicinity/region can cause the nodes of that region to assume some arbitrary values that may be different from the good circuit value of those nodes. We model such regional defects by first using the X-list-based error model (a logic-level model) [12], in which all the nodes in the X-lists (representing the vicinity/region of interest) are initially set to unknown values during simulation. In doing so, propagation behavior of arbitrary defects can be examined. Taking this approach a step further, we can compute the excitation and propagation behaviors of error regions by explicitly setting specific error values in the X-lists. Henceforth, the terms X-list, region, and vicinity are used interchangeably in this paper.

Our defect coverage computation techniques compute the defect coverage for each X-list and for the entire test set. The use of the logic-level X-list model reduces the need for layout or transistor-level information; only the logic-level abstraction of a circuit is needed to estimate the defect coverage of a given test set. In addition, the model can also capture design errors in addition to physical defects.

Since our techniques do not explicitly enumerate all types of faults that may cause a real defect and handle multiple faults (of possibly different types) simultaneously,
they are extremely fast. Our study of the effect of stuck-at coverage on the defect coverage demonstrates a correlation between stuck-at fault coverage and the coverage of those defects whose behavior is independent of the input test vector (vector-independent analysis). On the other hand, single stuck-at test sets could prove to be insufficient for defects that behave differently with differing input vectors (vector-dependent analysis). In addition, we observe that in some circuits, detectability of an arbitrary error $E$ by a test set is increased when a single stuck-at-fault within the vicinity of $E$ is detected multiple times by the test set.

The rest of the paper is organized as follows: Section II describes the modeling of arbitrary defects, including vector-independent and vector-dependent defects. Section III presents the algorithms for evaluating vector-independent defect coverage. Effects of multiple detections of single stuck-at-faults within a vicinity are analyzed in Section IV. Section V describes an algorithm for evaluating vector-dependent defect coverage for a given test set. Section VI describes the experimental setup and gives the results. Finally, Section VII concludes this paper.

II Modeling Arbitrary Defects

Two gate-level error models that capture the locality of errors have been proposed in the past to address the application of diagnosis: topologically-bounded error model and the region-based error model [13]. In this work, we use the region-based error model for capturing arbitrary defects in the vicinity of any node in the circuit. A group of nodes belonging to a “region” is allowed to carry a defect on any nodes within the group. Such groups are called region-based X-lists. These X-lists form the basic mechanisms to model, introduce, and simulate defects. These objects provide a logic-level alternative to capture “locality” of defects, much like inductive fault analysis [14] does at the layout level.

Each X-list is formed by including all the nodes within a fixed structural distance (called radius) from a single node of the circuit. That single node is called the center of the X-list. An X-list of radius one around gate $G$ is illustrated in Figure 1. This X-list consists of gates $\{A, B, C, D, G\}$; regions $R1$ and $R2$ are not a part of this X-list. X-lists are formed for every node in the circuit; thus, there can be as many X-lists as the number of nodes in the circuit.

During simulation, all the nodes of an X-list are first set to unknown values $X$ to cover any arbitrary error that may occur on the nodes of the X-list. If no $X$ from the X-list propagates to a primary output for a given vector $V$, we can conclude that for this vector $V$, any defect within the X-list region will not be detectable. On the other hand, if one or more $X$’s propagate to the primary outputs, some of the defects within the region may be detectable. To determine which specific defects can be detected by $V$, enumeration of all possible error values on the nodes in the X-list is required.

Consider an X-list, $L$, consisting of three nodes $[a, b, c]$. The set of values that these nodes can take is shown below:

$$000$$
$$001$$
$$010 \text{ good circuit value}$$
$$\ldots$$
$$111$$

At any given vector, only one of the $2^3$ possible values on the nodes of X-list $L$ can be a good-circuit value; any defect spanning the nodes of X-list $L$ can manifest as any one of the $2^3$ values (including the good-circuit value). By exhaustively simulating all possible error values for all vectors, arbitrary defect coverage can be obtained.

A typical X-list (of radius 2) consists of 8-10 nodes. However, the number of nodes in an X-list may range from as low as 2 (for instance, X-lists formed at PO’s) to as high as 50 nodes depending on several factors: structural radius of each region, fanin-fanout characteristics of the circuit, and the location of the region in the circuit. Consider, for example, an X-list with 10 nodes. The error space for this X-list is defined as the enumeration of all possible error values for these 10 nodes. Therefore, the cardinality of error space of this X-list with 10 nodes is $2^{10}$. When it becomes computationally expensive to simulate the entire error space for large X-lists, we use one of the two approaches given below to deal with this limitation:

- Sample a set of errors from the error space of a X-list. The sample size, $s$, is fixed for all X-lists of the circuit for a given run. Thus, for any X-list with $n$ nodes, the number of errors considered for evaluation is:

$$2^n \quad \text{ if } 2^n < s$$
$$s \quad \text{ if } s < 2^n$$

- Limit the nodes in a X-list to a reasonable number and consider all possible errors.

Figure 1: An X-list.
A Vector Dependence of Defects

An arbitrary defect in a circuit can be vector-dependent, i.e., the faulty values created by the defect/error are dependent on the input vector. For example, consider circuits shown in Figure 2. Figure 2(a) shows the original circuit,

\[ \begin{array}{c}
    x \\
    0 \\
    0(1) \\
    G1 \\
    G2 \\
    1 (0) \\
\end{array} \]

(a) Error-free circuit

while the erroneous/faulty version of the circuit is shown in part (b), where the NAND gate G2 behaves incorrectly as if it was an AND gate. Two vectors X00 and X01 are applied respectively to the circuit, and the corresponding good and the faulty circuit outputs are shown. When vector X00 is applied, the fault in the circuit manifests as a stuck-at-0 fault, whereas when vector X01 is applied, the fault manifests as stuck-at-1 fault. Clearly, this example shows that a vector-dependent defect may manifest as different erroneous/faulty values to different input patterns applied.

Hence, in order to correctly model the behavior of a vector-dependent defect, it is necessary to enumerate all possible error values for each unique input pattern. On the other hand, a vector-independent defect (for e.g., a stuck-at fault) exhibits the same erroneous behavior for all input vectors, and thus, the error values in the affected region need only to be enumerated once for all input vectors. Furthermore, in order to obtain a 100% vector-dependent defect coverage in a region, the test set must exercise all possible patterns at the fanin nodes of that region. Consider a region with 10 inputs. There are \(2^{10}\) possible input values to that region. To exercise all these values, a test set consisting of at least \(2^{10}\) vectors is required. We observed that a typical test set produces only a small percentage of these values at the inputs of a region. Thus, in order to evaluate vector-dependent defect coverage, we use the notions of activity level and observability of a region under a given test set.

Definition 1: Activity level of a region \(L\) under a given test set \(T\) is defined as the percentage of possible input patterns obtained at the inputs of \(L\) when \(T\) is applied.

For instance, in a 2-input region (4 possible input patterns), if a test set \(T\) produces only two different input patterns to this region, the activity level of the region under \(T\) is 50%.

Definition 2: Observability of a region \(L\) under a given test set \(T\) is defined as the percentage of defects in \(L\) that are observable at the primary outputs, out of the total number of defects excited by \(T\).

Let us consider a 3-node, 2-input region \(L\). Any vector-dependent defect in this region can manifest as one of eight possible values on its nodes under a given value at its two inputs. If the activity level of \(L\) under a given test set \(T\) is 50%, the defects can have a total of 16 possible manifestations (8 for each of the two patterns produced at inputs of \(L\) by \(T\)). If 12 out of these 16 error manifestations are observed at PO’s when test set \(T\) is applied, the observability of \(L\) under \(T\) is 75%. Given two test sets \(T_1\) and \(T_2\) with similar activity levels, the test set that produces higher observability is a better test set. A test set with both high activity level and observability is desired.

In the following sections we present algorithms for evaluating vector-independent and vector-dependent arbitrary defect coverages using previously generated test sets. Let \(N\) be the total number of nodes in the circuit; then, there are \(N\) X-lists in the circuit. Each X-list \(L\) is centered at a different node in the circuit. The test set being evaluated is represented by \(T\), and a vector in \(T\) is represented by \(V\).

III Vector-Independent Defect Coverage

A vector-independent defect manifests itself the same way for all vectors applied. For example, single stuck-at faults are vector-independent, since the node with a fault always exhibits the same error behavior. Likewise, multiple stuck-at faults also belong to the same vector-independent family.

We now present an algorithm to estimate the coverage of all vector-independent defects. X-lists are first extracted from the circuit based on a given structural radius. A vector-independent defect in a region can cause the nodes in the X-list to take on any logic value 0 or 1. Therefore, for an \(n\)-node X-list, there exists up to \(2^n\) vector-independent defects. Each of these \(2^n\) defects in the region can be modeled by fixing values on the nodes specific to that defect during simulation. In order to consider all possible arbitrary defects in a region, an exhaustive simulation using all \(2^n\) error values in the X-list region is required.

We simulate all vectors on all X-lists. The nodes of a X-list \(L\) are initially fixed to unknown value \(X\) in a process called \texttt{X-simulate}(\(V, L\)). If at the given vector \(V\), no unknown value \(X\) in the X-list propagates to a primary output, then we don’t need to enumerate and simulate the possible error values in the X-list region for that vector; otherwise, enumeration of the error values needs to be done. For large X-lists, because it is computationally expensive to simulate all error values, we have a choice of sampling the error values or limiting the number of nodes.
in the X-list, as described in the previous section. We use value sampling of large X-lists for vector-independent defect coverage evaluation; however, the node-limiting technique would work as well.

begin
  for each X-list \( L \) in circuit
    \( \{ \text{ErrorSpace}[L] \} = \{ \text{InitialErrorSpace}[L] \} \)
  for each vector \( V \in T \)
    for each X-list \( L \) in circuit
      if \( \{ \text{ErrorSpace}[L] \} \neq \emptyset \)
        \( \text{X-simulate}(V, L) \)
      if one or more \( X \) propagate to POs
        for each error \( E \) in \( \{ \text{ErrorSpace}[L] \} \)
          \( \text{ErrorSpaceSimulate}(V, L, E) \)
        if \( V \) detects \( E \)
          \( \{ \text{ErrorSpace}[L] \} = \{ \text{ErrorSpace}[L] \} - E \)
  for each X-list \( L \) in circuit
    \( \text{X-list-Coverage}(L) = \frac{\| \text{InitialErrorSpace}[L] \| - \| \text{ErrorSpace}[L] \|}{\| \text{InitialErrorSpace}[L] \|} \)
  \( \text{TestSet-Coverage}(T) = \sum_{L} \left( \frac{\| \text{InitialErrorSpace}[L] \| - \| \text{ErrorSpace}[L] \|}{\| \text{InitialErrorSpace}[L] \|} \right) \)
end

Figure 3: Vector-Independent Defect Coverage Estimation

The algorithm for defect coverage evaluation is described in Figure 3. The set of \( s \) sampled error values for each X-list \( L \) constitutes its error space denoted by \( \text{InitialErrorSpace}[L] \). All these errors are individually simulated by setting the specific error values on the nodes of X-list. When the algorithm is finished, the defect coverage computed at the last step gives us the defect coverage under the given test set \( T \).

IV Effect of Multiple Detections On Defect Coverage

It has been observed that multiple-detects of single stuck-at-faults increase the defect coverage \([2, 10]\). In this section, we study if our defect coverage analysis using X-lists also supports this observation. We analyze how the defect coverage of a given region depends on the detection of representative stuck-at-faults in that region, where the representative fault is the fault at the center of the region.

The arbitrary errors are again modeled with X-lists of a fixed radius around each node in the circuit. Because we are interested in all the possible defects in an immediate vicinity, exhaustive simulation of all possible error values in the region is preferred. However, in order to maintain computational feasibility, we use the node-limiting technique to limit each X-list to have at most 10 nodes.

The procedure for analyzing effects of a multiply-detected single stuck-at-fault on detection of the defects in its vicinity is described in Figure 4. There is a first phase to the procedure that fault simulates the given test set \( T \) using the single stuck-at faults of the circuit without fault dropping. Since no fault dropping is performed, each fault in the fault-list of the circuit can be detected multiple times by different vectors of the test set \( T \).

A \textit{Regional-Category}, \( R \), consists of vicinities in which faults at center nodes are detected same number of times. Four different \textit{Regional-Categories} where a fault at the center node of the region is detected once, twice, 3-10 times, and greater than 10 times are considered. We name these categories \( R_0 \), \( R_1 \), \( R_2 \), and \( R_3 \) respectively.

\begin{verbatim}
// F: stuck-at-0 or stuck-at-1 fault
// CL: Center node of X-list L

Fault Simulate T without fault dropping
for each X-list L
  for each fault \( F \in \{ CL \text{ stuck-at-0, } C_L \text{ stuck-at-1} \} \)
    \( \{ \text{ErrorSpace}[L, F] \} = \{ \text{InitialErrorSpace}[L, F] \} \)
for each vector \( V \in T \)
  for each fault \( F \) in the circuit
    Let \( L \) be the vicinity of fault \( F \)
    if \( V \) detects \( F \)
      if \( \{ \text{ErrorSpace}[L, F] \} \neq \emptyset \)
        for each error \( E \in \{ \text{ErrorSpace}[L, F] \} \)
          \( \text{ErrorSpaceSimulate}(V, L, E) \)
        if \( V \) detects \( E \)
          \( \{ \text{ErrorSpace}[L, F] \} = \{ \text{ErrorSpace}[L, F] \} - E \)
for each Regional-Category \( R \in \{ R_0, \ldots, R_3 \} \)
  Regional-Defects-Missed(\( R \)) = \sum_{L \in R} \| \text{ErrorSpace}[L, F] \|
end

Figure 4: Computing Correlation of Multiple Detections and Defect Coverage

\end{verbatim}

After the first phase is finished, we evaluate vector-independent defect coverage (a variation of algorithm given in Figure 3) and study its variation with multiple detections of stuck-at faults using the algorithm given in Figure 4. The set of errors (out of the entire error space of vicinity) that include the center node of \( L \) as a stuck-at fault \( F \) constitute the initial error space of the vicinity with respect to \( F \) and is denoted by \( \text{InitialErrorSpace}[L, F] \). Thus, initial error spaces for each center node stuck-at 0 and stuck-at 1 are compliments of each other and they together constitute the entire error space of the vicinity. Errors in the vicinity of a stuck-at fault \( F \) are simulated only for vectors that detect \( F \). In the end, fraction of errors that remain undetected in each of the four \textit{Regional-Categories} \( \{ R_0, \ldots, R_3 \} \) is evaluated and is used to the analyze the effect of multiple detects of single stuck-at-faults on defect coverage in its vicinity.

V Vector-Dependent Defect Coverage

The algorithms described so far are useful only for vector-independent arbitrary defects. In this section we discuss
the evaluation of vector-dependent defect coverage for a given test set, since the vector-independent defect assumption may not capture all real defects.

Under the vector-dependent defect assumption, an arbitrary defect may not have the same behavior for all vectors. Due to this reason, the set of possible erroneous values on the nodes of a region can change with different vectors, as explained earlier. If two vectors produce the same good circuit values on the immediate fan-in nodes of a region, the error behavior for these two vectors will be the same. In such a case, only the errors of that region which could not be detected by first vector are simulated with the second vector. For the vectors that produce different good circuit values on the fan-in nodes of a region, all possible error manifestations of a defect must be considered. For large regions, we again use node-limiting technique to obtain a fair estimate of vector-dependent defect coverage in a reasonable amount of time.

As discussed earlier, vector-dependent defect coverage of a test set can be interpreted by the activity level the test set produces in the circuit and the observability of different regions that is obtained by applying the test set. Both activity level and observability of a X-list L are computed as described in section II. Figure 5 shows the algorithm for computing activity level and observability of each X-list in the circuit. \( S_{L,V} \) represents the input pattern to X-list L from vector V. For each new \( S_{L,V} \) containing previously unseen error-free input values to X-list L, a new error-value set \( \text{ErrorSpace}[L, S_{L,V}] \) is generated for simulation. Since our algorithm computes activity level and observability of all X-lists in the circuit, it is capable of indicating specific regions in the circuit where the given test set has higher chances of missing an arbitrary defect.

**VI Experimental Results**

In this section we present the results of experiments performed using procedures based on algorithms discussed in previous sections. The procedures were implemented in C & C++; ISCAS85 combinational benchmark circuits were used. All experiments were performed on Sun SPARCstation 20 with 512MB of memory. Test vector sets generated by HITEC [15] for single stuck-at-faults were used.

Table 1 presents the results of vector-independent defect coverage on HITEC test sets. Sample size of 256 was used for large X-lists in Table 1. In this table, the number of vectors and single stuck-at-fault coverage are first given for each benchmark circuit. Next, the defect coverage and simulation times (in seconds) are reported for X-list radii of 1, 2, and 3. The number of X-lists in each circuit is the same as number of nodes in the circuit. However, the size of the X-list of the same radius will differ in the circuit. The defect coverages indicate the percentage of simulated defects modeled by all X-lists that could be detected by the single stuck-at test sets.

As shown in Table 1, the defect coverages are very high (near 100%) for most circuits. In the case of three circuits, c432, c2670, and c3540, where lower single stuck-at fault coverage had been obtained, the corresponding defect coverages are also affected. Note that the simulation times are very small for all circuits, indicating the efficiency of our algorithm. From this experiment, we observe that stuck-at fault coverage may indeed be a good measure of the vector-independent defect coverage.

Now we move on to discussing the effect of multiple detections of a stuck-at fault on the detection of defects within its vicinity. We again use HITEC test sets to fault simulate (without any fault dropping) the circuits initially, and statistics regarding the % of total faults detected once, twice, between three and ten times (3-10) and more than ten times (>10) are gathered. For each vector, all X-lists centered at nodes for which a stuck-at-fault \( F \) is detected by that vector are considered. For each such X-list, all errors in \( \text{ErrorSpace}[L, F] \) (refer to algorithm in Figure 4) are simulated. Thus, the errors in the vicinity of a fault detected five times by given test set are simulated at five vectors. Because we are still considering vector-independent defect model, all errors detected by a vector are dropped and are not considered when the X-list is simulated again at a subsequent vector.

Table 2 gives the results showing variation in the defect coverage with increasing number of detections of relevant stuck-at faults. In this table, the number of vectors and fault coverage are first given for each circuit, followed by the statistics of the detection multiplicity of

```c
// \( S_{L,V} \): set of fan-in node values of X-list L at vector V
// \{\text{ErrorSpace}[L, S_{L,V}]\} is the set of error values for X-list L
// corresponding to fan-in node value set \( S_{L,V} \).
for each X-list L
\{S_L\} = \emptyset
for each vector V \in T
for each X-list L
if \( S_{L,V} \notin \{S_L\} \)
    \{\text{InitialErrorSpace}[L, S_{L,V}]\} = \text{GetNewErrorSet}(L)
    \{\text{ErrorSpace}[L, S_{L,V}]\} = \{\text{InitialErrorSpace}[L, S_{L,V}]\}
\{S_L\} = \{S_L\} + S_{L,V}
if \{\text{ErrorSpace}[L, S_{L,V}]\} \neq \emptyset
\text{X-simulate}(V, L)
    if one or more X propagate to POs
for each error E \in \{\text{ErrorSpace}[L, S_{L,V}]\}
    \text{ErrorSpaceSimulate}(V, L, E)
    if V detects E
        \{\text{ErrorSpace}[L, S_{L,V}]\} = \{\text{ErrorSpace}[L, S_{L,V}]\] - E
for each X-list L
X-list-activity(\(L\)) = \sum_{S_L} \left\lceil \frac{|S_L|}{k} \right\rceil \text{ possible input patterns to } L
X-list-observability(\(L\)) = \frac{\sum_{S_L} \{\|\text{InitialErrorSpace}[L, S_{L,V}]\| - \|\text{ErrorSpace}[L, S_{L,V}]\|\}}{\sum_{S_L}\|\text{InitialErrorSpace}[L, S_{L,V}]\|}
```

Figure 5: Vector-Dependent Defect Coverage Estimation
Table 1: Vector-Independent Defect Coverage Analysis (sample size = 256)

<table>
<thead>
<tr>
<th>Circuit</th>
<th>No. of Vectors</th>
<th>Single s-at FC(%)</th>
<th>Vec-Ind. Defect Coverage</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Radius 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Cov(%)</td>
</tr>
<tr>
<td>c432</td>
<td>100</td>
<td>93.37</td>
<td>99.97</td>
</tr>
<tr>
<td>c499</td>
<td>184</td>
<td>99.38</td>
<td>100.00</td>
</tr>
<tr>
<td>c880</td>
<td>75</td>
<td>100.00</td>
<td>100.00</td>
</tr>
<tr>
<td>c1355</td>
<td>88</td>
<td>99.49</td>
<td>100.00</td>
</tr>
<tr>
<td>c1908</td>
<td>280</td>
<td>99.36</td>
<td>100.00</td>
</tr>
<tr>
<td>c2670</td>
<td>102</td>
<td>95.74</td>
<td>99.91</td>
</tr>
<tr>
<td>c3540</td>
<td>350</td>
<td>95.89</td>
<td>99.95</td>
</tr>
<tr>
<td>c5315</td>
<td>264</td>
<td>98.82</td>
<td>100.00</td>
</tr>
<tr>
<td>c6288</td>
<td>46</td>
<td>99.54</td>
<td>100.00</td>
</tr>
<tr>
<td>c7552</td>
<td>450</td>
<td>98.09</td>
<td>100.00</td>
</tr>
</tbody>
</table>

Table 2: Effect of Multiple-Detects on Regional Defect Coverage

<table>
<thead>
<tr>
<th>Ckt</th>
<th>Vec</th>
<th>Stuck-at FC(%)</th>
<th>Once</th>
<th>Twice</th>
<th>3-10</th>
<th>&gt; 10</th>
<th>% Defects undetected in vicinity of faults detected</th>
<th>Time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>c432</td>
<td>100</td>
<td>93.3</td>
<td>0.52</td>
<td>1.03</td>
<td>46.9</td>
<td>49.48</td>
<td>1.51</td>
<td>0.01(1.56)</td>
</tr>
<tr>
<td>c499</td>
<td>184</td>
<td>99.3</td>
<td>0.00</td>
<td>5.11</td>
<td>4.89</td>
<td>90.00</td>
<td>0.13</td>
<td>0.00(0.00)</td>
</tr>
<tr>
<td>c880</td>
<td>75</td>
<td>100.0</td>
<td>0.33</td>
<td>1.32</td>
<td>22.19</td>
<td>76.16</td>
<td>0.03</td>
<td>0.00(0.00)</td>
</tr>
<tr>
<td>c1355</td>
<td>198</td>
<td>99.4</td>
<td>4.98</td>
<td>4.98</td>
<td>10.14</td>
<td>79.90</td>
<td>1.71</td>
<td>0.15(11.72)</td>
</tr>
<tr>
<td>c1908</td>
<td>280</td>
<td>99.3</td>
<td>0.00</td>
<td>4.15</td>
<td>10.00</td>
<td>85.85</td>
<td>0.38</td>
<td>0.00(0.00)</td>
</tr>
<tr>
<td>c2670</td>
<td>102</td>
<td>95.7</td>
<td>0.59</td>
<td>2.04</td>
<td>17.25</td>
<td>80.12</td>
<td>2.05</td>
<td>0.01(2.08)</td>
</tr>
<tr>
<td>c3540</td>
<td>350</td>
<td>95.8</td>
<td>0.00</td>
<td>0.75</td>
<td>10.47</td>
<td>88.78</td>
<td>0.16</td>
<td>0.00(0.00)</td>
</tr>
<tr>
<td>c5315</td>
<td>264</td>
<td>98.8</td>
<td>0.05</td>
<td>0.32</td>
<td>4.42</td>
<td>95.21</td>
<td>0.29</td>
<td>0.00(0.00)</td>
</tr>
<tr>
<td>c6288</td>
<td>46</td>
<td>99.5</td>
<td>0.00</td>
<td>0.00</td>
<td>6.05</td>
<td>93.95</td>
<td>0.01</td>
<td>0.00(0.00)</td>
</tr>
<tr>
<td>c7552</td>
<td>450</td>
<td>98.0</td>
<td>0.85</td>
<td>0.53</td>
<td>4.33</td>
<td>94.28</td>
<td>1.10</td>
<td>0.02(6.36)</td>
</tr>
</tbody>
</table>

% Defects undetected in vicinity of faults detected: ≥ Once - Gives the total % of Defects left undetected. Once, Twice, 3-10 Times, >10 Times - Number outside parenthesis is the % undetected defects of the particular category over the entire error space. The number inside parenthesis is the % undetected defects in that regional category.

stuck-at-faults; specifically, the percentage of faults detected once, twice, etc. are reported. Next, the percentage of undetected defects within different region categories are reported for each circuit. For instance, the Twice column indicates the percentage of undetected defects within the regions centered around faults that are detected exactly twice by the test set. There are two numbers under columns Once, Twice, 3 – 10, and > 10. The first number indicates the % undetected defects of the particular category over the entire error space, and the second number (inside parenthesis) indicates the % undetected defects in X-lists belonging to that particular regional category. For instance, in circuit c1355, 1.71% of defects for all X-lists with a detected stuck-at-fault at its center node remain undetected. The contribution of 1.71% comes from different region categories: 0.15% from Once, 0.18% from Twice, 0.16% from 3 – 10, and 1.29% from > 10. Because 79.90% of the stuck-at faults are detected more than 10 times by the test set, most of the X-lists fall into the > 10 category, thus yielding 1.29% of 1.71% undetected faults. The numbers inside the parentheses (11.72%, 2.34%, etc.) for c1355 indicate that 11.72% of defects within regions centered around once-detect stuck-at faults are not detectable by the test set, 2.34% undetectable by the test set for regions centered around twice-detect faults, etc. Note that in column 3 – 10, the percentage of undetects is a summation of thrice-detect, 4-times-detect, ..., up to 10-times detect. The case of column > 10 is similarly computed.

Finally, the results for vector-dependent arbitrary defect coverages are reported in Table 3. The size of X-lists was limited to four nodes for this experiment. The table gives, for each circuit, number of Vectors simulated, Activity and Observability obtained by the simulated test set and the Time taken in seconds. For instance, the test set for c6288 (46 vectors) produces an activity level of 46.84% and observability of 92.98% in the circuit. Also, the time taken for this evaluation is 110.18 seconds.

Note that the simulation times in vector-dependent-defects case are significantly higher than the vector-independent counterpart. This is due to the fact that in the vector-dependent model, a different error-value set for an X-list must be considered for every vector that has a different fan-in values to the X-list region. In general, the number of error values needed is one order of magnitude higher than in the case of vector-independent defects.

We observe that activity level achieved by the test set for some circuits is very low. For example, c1908 has only 2.49% activity level under the given test set. This is due
to the fact that a given test set may not exercise many patterns that are obtainable at the inputs of a region. As pointed out earlier in our discussion of vector-dependent defects, if two test sets produce similar amount of activity in the circuit, the one that produces higher observability is a better test set.

We observe from this experiment that test sets with nearly 100% single stuck-at fault coverage may not be sufficient to detect all the vector-dependent defects. Nevertheless, the vector-dependent defect model is pessimistic, since we are considering all possible error values in a region and that each error may manifest differently and independently with different test vectors.

VII Conclusions

We have presented efficient algorithms for estimating arbitrary defect coverages of a given test set. No specific, physical, fault model was used as the underlying defect model. Instead, X-lists are used to capture all possible arbitrary defects. Both vector-independent and vector-dependent analyses are considered, thus we can capture both physical defects (such as stuck-at-faults) as well as design errors (such as an AND gate changed to an OR gate). Our experiments showed that while test sets with high single stuck-at fault coverage are sufficient to detect nearly 100% of vector-independent arbitrary defects, the same may not be true for vector-dependent defects. We have also demonstrated the effect of multiple detections of single stuck-at-faults on detectability of defects located around them.

Extensive experimentation with physical failure data from a manufacturing process environment (and error data from a design process) is required to validate the estimation techniques proposed. Furthermore, future extensions of this research include test generation targeting arbitrary errors based on the X-list-based error models, particularly targeting regions for which defect coverage is low, and more efficient techniques for vector-dependent defect simulation.

References

[13] Names, “Withheld to comply with blind review”, in Submitted for publication, 19XX.