Efficient Test Set Evaluation

Hans-Joachim Wunderlich
Maren Warnecke

Universität - GHS - Siegen
FB 12, Hölderlinstr. 3, 5900 Siegen, Germany

Abstract: The fault coverage obtained by a set of test patterns is usually determined by expensive fault simulation. Even using fault dropping techniques fault simulation provides more information than actually needed. For each fault the pattern is determined which detects this fault first. This is mainly redundant information if diagnosis is not required. We can dispense with this high resolution and restrict our interest on the set of faults which is detected by a set of patterns. It is shown theoretically and practically that this information is obtainable in an highly efficient way.

KEYWORDS: Test, fault simulation

1. Introduction

Fault simulation is among the most expensive tasks in circuit design. Its average computing time depends on the size of the circuit, the average size of the list of faults to be simulated and the number of patterns. In summary the average complexity of fault simulation turns out to be between quadratical and cubical [8]. Hence many techniques to improve the efficiency of fault simulation were presented during the last years. They are primarily based on structural analysis to avoid unnecessary computations (e.g. [10, 1, 13, 12]) and on Parallel Pattern Single Fault Propagation (PPSFP) for exploiting the parallelism of general purpose computers [4, 17].

Compared with classical methods as parallel, deductive or concurrent fault simulation [14, 3, 16], innovative techniques may be up to 1.000 or 10.000 times faster, but simulating a single pattern still has a quadratical worst case complexity [13]. Complexity analysis shows that there is no hope for linear time fault simulation [9]. Hence many approximate techniques, which are divided into optimistic and pessimistic methods, were proposed to accelerate the evaluation of test sets (e.g. [5, 2, 11]).

For fault f and input pattern t let the detectability $d_f(t)$ be computed by an approximate simulator. A method will be called optimistic, if $d_f(t) = 0$ implies that fault f is not detectable by pattern t. It will be called pessimistic, if $d_f(t) = 1$ implies that fault f is detected by pattern t. Exact fault simulation is optimistic as well as pessimistic, the "Fast Fault Grading" method proposed by [5] is optimistic, and "Critical Path Tracing" [2] is pessimistic. Unfortunately the latter method has also polynomial complexity with a degree larger than 1 [9].

In the following section we present a technique to compute a pessimistic measure $e_f(t)$ and an optimistic measure $\neg n_f(t)$ for all the faults by a single pass through the circuit with complexity O(N ln N). Hence $n_f(t) = 1$ implies that fault f is not detectable by pattern t. Another optimistic linear time algorithm for identifying a subset of the undetectable faults was also presented in [Kris90]. Figure 1 illustrates how the set F of faults is partitioned by the two measures $e_f(t)$ and $n_f(t)$. As an abbreviation we use the variable $u_f(t) := \neg e_f(t) \wedge \neg n_f(t)$ for indicating that no information is achieved.

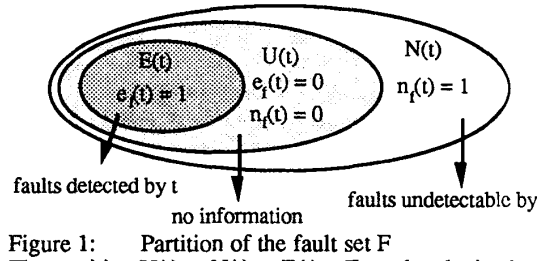

Figure 1: Partition of the fault set F

The partition $U(t) \cup N(t) \cup E(t) = F$ can be obtained very efficiently. The algorithm for computing these measures is called Test Set Evaluation (TEVA), and is presented in section 2. In section 3 we explain how to evaluate a set of test patterns by this algorithm. After applying a number of patterns to TEVA all faults of E(t) can be dropped and we switch to an exact simulator. For the patterns already applied to TEVA we only have to simulate the faults of U(t), for the following patterns we have to simulate all faults not yet definitely detected. This way we lose the information about the first pattern detecting a fault, but we still retain the exact information about the total fault coverage. Also in this section it is analyzed when to switch from TEVA to exact fault simulation in order to obtain the highest gain in efficiency. Results are presented in section 4.

2. Approximate Fault Simulation

The technique presented is based on the principle of single path sensitizing. For each node k and input pattern t two values are computed, which will be called $e_k(t)$ and $n_k(t)$. $e_k(t)$ is true if a single path from k to one of the primary outputs is sensitized. In this case k is definitely observable. If $n_k(t)$ is true, all pathes from k to the primary outputs are blocked, and k is definitely not observable. For all nodes whose values $e_k(t)$ and $n_k(t)$ are false no statement can be made.

In order to explain the algorithm in detail, some formal notations are required. Let C be a combinational circuit

0-8186-2645-3/92 $3.00 © 1992 IEEE

with primary inputs I and primary outputs O, let v be a node of C. The boolean function $v^* : \{0, 1\}^I \rightarrow \{0, 1\}$ is defined by $v^*(t) = 1 \Leftrightarrow$ node v becomes logical "1" if t is applied. The formulas $v^*(t, 0_i)$ and $v^*(t, 1_i)$ are defined in a similar way, but in addition node i is set to 0 or 1, respectively. If f is a stuck-at fault s0-k [or s1-k] we define $e_f(t) = k^*(t) \wedge e_k(t)$ [$e_f(t) = \neg k^*(t) \wedge e_k(t)$] and $n_f(t) = \neg k^*(t) \vee n_k(t)$ [$n_f(t) = k^*(t) \vee n_k(t)$]. Hence $e_f(t) = 1$ implies definite fault detection, and $n_f(t) = 1$ implies that fault f is undetectable.

Let k be a node with the immediate predecessor v. The term $\frac{\delta k}{\delta v}(t)$ describes the boolean difference, it is true if the value of v can be observed at k. In figure 2 we have $\frac{\delta g}{\delta k} = 1$, and $\frac{\delta g}{\delta b} = 0$.

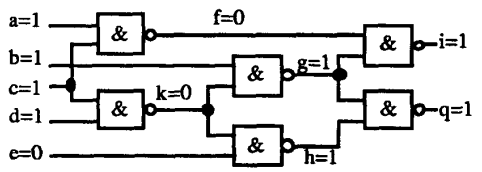

Figure 2: Example circuit C17

As combinational circuits can be considered as acyclic graphs, a path from node v to a node o is uniquely determined by its set of nodes $\omega(v, o)$. If $\omega(v, k)$ is a path from v to k and $\omega(k, o)$ is a path from k to o, the formula $\omega(v, k) + \omega(k, o)$ denotes the well-defined path from v to o formed by the concatenation of $\omega(v, k)$ and $\omega(k, o)$.

Lemma 1: Node v is not observable if for each output $o \in O$ and for each path $\omega(v, o)$ there is a node $z \in \omega(v, o)$ such that $z^*(t, 1_v) = z^*(t, 0_v)$.
Proof: Straightforward

A path $\omega(v, o)$ is called *blocked* by node $v_{i+1} \in \omega(v, o)$ if v_i is not observable at v_{i+1}, and all the other immediate predecessors $k \neq v_i$ of v_{i+1} are independent of v, i. e. $k^*(t, 0_v) = k^*(t, 1_v)$.

Without loss of generality we assume a maximum fanout of 2 in the circuit. With the help of the following definition we can decide whether a fanout stem is definitely not observable:

Definition 1: Let g, h be two immediate successors of v. The two node sets A_1, A_2 satisfy the *blocking condition* of v if:

1) For each output o and path $\omega_1(g, o)$ there is a blocking node in A_1.
2) For each output o and path $\omega_2(h, o)$ there is a blocking node in A_2.
3) For every $b \in A_1 \cap A_2$, the immediate predecessor of b in $\omega_1(g, o)$ is also predecessor in $\omega_2(h, o)$.

Obviously the blocking condition is satisfied for $A_1 \cap A = \emptyset$. In figure 3a) the path $\omega_1(g, o)$ is blocked by $A_1 := \{o\}$ and ω_2 (h, o) is blocked by $A_2 := \{o\}$, too, but the blocking condition is violated due to 3), so that we cannot make any statement about the observability of k.

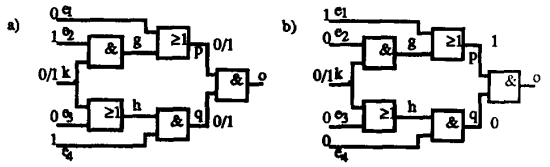

Figure 3: Blocking condition

In figure 3b) we have $A_1 := \{p\}$ and $A_2 := \{q\}$, i.e the blocking condition is satisfied and k is definitely not observable.

Lemma 2: If A_1, A_2 satisfy the blocking condition for v then the set $A_1 \cup A_2$ contains blocking nodes for all paths $\omega(v, o)$ and v is not observable.
Proof: see [18], p. 170.

Now we can derive a sufficient condition of definite observability. Under patter $t \in \{0, 1\}^I$, a single path $\omega(v, w) := \{v = v_0, \ldots, v_n = w\}$ is sensitized if

a) All nodes of $\omega(v, w)$ have different values for v = 1 and v = 0: $v_i^*(t, 0_v) \neq v_i^*(t, 1_v)$, i = 0, ..., n.
b) All immediate predecessors $k \neq v_{i-1}$ of v_i, i = 1, ..., n, outside the path have the same value for v = 1 and v = 0, i.e. $k^*(t, 0_v) = k^*(t, 1_v)$.

Figure 4 illustrates the concept of single path sensitizing:

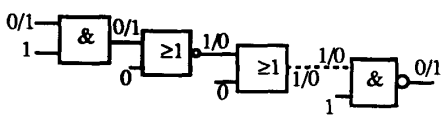

Figure 4: Single path sensitizing

If a single path from node v to a primary output is sensitized then v is observable, but this conclusion is not always true in the other direction. Finally we have:

Lemma 3: Let v be a fanout stem with the immediate successors g and h, i. e. there are edges $e_1 := (v, g)$ and $e_2 := (v, h)$. Let a path $\omega(g, o)$ be sensitized and let $\frac{\delta g}{\delta v} = 1$. Let $N[e_1]$ be a set of nodes such that for each path $\omega_1 := (v, g, \ldots, p) \propto (v, g) + \omega(g, o)$ with some $p \in \omega(g, o)$ there is a blocking node $b \in N[e_1]$, $b \neq p$. Let $N[e_2]$ be analogously defined for all paths (v, h, ..., o). If $N[e_1] \cap N[e_2] = \emptyset$ and $N[e_2] \cap \omega(g, o) = \emptyset$ then $(v, g) + \omega(g, o)$ is a sensitized single path. Additionally $N[v] := N[e_1] \cup N[e_2]$ contains blocking nodes for all paths $\tau(v, p) \propto (v, g) + \omega(g, o)$ with some $p \in (v, g) + \omega(g, o)$.
Proof: see [18], p. 170.

Lemma 3 gives us a means for deciding whether a single path starting from a fanout stem is sensitized. The complexity of this decision is in the order of the cardinalities of the sets N.

We clarify lemma 3 with the help of the example circuit of figure 2. If we want to compute the observability of the fanout stem c, the two sets N[e_1] (e_1 := c, f) and N[e_2] (e_2 := c, k) have to be determined. We set N[e_1] := Ø and N[e_2] := {h, i}, and the path ω(f, i) is sensitized. Because of N[e_2] ∩ ω(f, i) ≠ Ø, the condition of single path sensitizing is not satisfied and we cannot make any statement about the observability of k.

The approximate simulation algorithm TEVA consists of two steps. First the fanout-free regions are processed by finding out if the path from node k to the next fanout stem or primary output, which is called v(k), is sensitized. In this case e'_k is true; if k is a fanout stem or primary output by itself, i. e. k = v(k), formula e'_k is true, too.

Case 1) v(k) = k: Set $e'_k = 1$.

Case 2) v(k) ≠ k: Set $e'_k = \frac{\delta\, k2}{\delta\, k} \wedge e'_{k2}$, where k2 is the successor of k.

During the second step the whole circuit is processed by calculating e_k and n_k for each node. Whithin fanout-free regions and for primary outputs this is straightforward:

Case 1) k is a primary output: Set $e_k = 1$; $n_k = 0$.

Case 2) k is inside a fanout-free region: Set $e_k = e'_k \wedge e_{v(k)}$; $n_k = \neg e'_k \vee n_{v(k)}$.

Case 3)

In order to compute e_k and n_k for fanout stems we have to determine the already introduced set N[k] which contains the blocking nodes and a set called S[k]. If e_k is true, S[k] will contain all the nodes of the sensitized path from k to a primary output of the circuit, otherwise S[k] is empty.

The successors of fanout stems are divided into independent and dependent successors. If there are reconvergent paths to the same primary output, they are called dependent and we apply lemma 3. Otherwise there are no reconvergencies and all paths starting at the two successors lead to disjoint primary outputs.

Before processing fanout stem k, we first have to determine the two values e and n and the set N for the two edges leading from that fanout stem to his immediate successors g and h: We define $e_{g,k} = \frac{\delta g}{\delta k} \wedge e_g$ and $n_{g,k} = \neg \frac{\delta g}{\delta k} \vee n_g$. If $\left(e'_g \wedge \frac{\delta g}{\delta k}\right) = 0$ we set N[k, g] := {k'}, where k' is the node of path ω(g, v(g)) nearest to k, else we set N[k, g] := N[v(g)].

After processing the edge (k, h) the same way, we can determine the observability of k as shown in figure 5.

2.2 Complexity and accelerations

The algorithm consists of two passes through the circuit. Except for fanout stems the number of operations at a node is bounded by a constant. At fanout stems we have to intersect sets which increase linearly with the circuit size C. Overall this leads to a quadratical worst-case complexity, which can be reduced further.

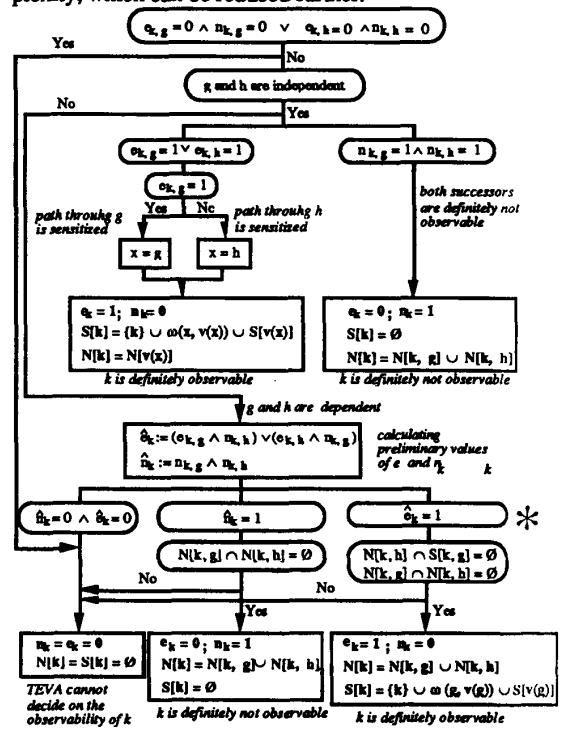

Figure 5: Determining the observability of fanout stems

* The path through g is sensitized. If the path through h is sensitized, e_k, n_k, N[k] and S[k] are computed the analogous way.

As the sets S represent paths of linearly ordered nodes they can be organized as balanced trees such that checking membership can be done in O(ln |S|). Moreover the tradeoff between the information obtained by this algorithm and its complexity can be controlled by an additional parameter β which should limit the cardinality of the sets N. Whenever |N[k, g] ∪ N[k, h]| > β we dispense with further computations for k, and set N[k] = S[k] = Ø and n_k = e_k = 0. With this restriction the entire algorithm has complexity O(C · ln(C)). Further improvements are obtainable by an analysis of the circuit structure as used in exact fault simulation [MaRa88, Schu88]. E. g. if node k has a dominator d the computation can be accelerated [15]. Suppose d is definitely not observable, then k is not observable either. We set n_k = 1, s_k = 0, N[k] = N[d], and S[k] = Ø.

Finally the principle of Parallel Pattern Single Fault Propagation [4, 17] can be applied to all steps of the algorithm with the exception of the set operations. The results

to be reported in section 4 are obtained by the approximate method and by an exact fault simulation both using the PPSFP priciple and a preprocessing task for determining dominators.

3. Evaluation of large test sets

The improvements of efficiency obtained by the approximate method increase with the number of faults to be processed. If the fault list is reduced by fault dropping the exact simulation may become even faster. This section addresses the problem of balancing the computational load between the approximate method and the exact fault simulation. A similar problem was dealt with in [7] where the best point for switching from random pattern simulation to deterministic test generation was discussed.
Using TEVA the fault coverage obtained by the test set T := $\{t_1, ..., t_\alpha\}$ is determined in the following way:

1) Set $F_0 := F$ and for all $f \in F$ set $\overline{T(f)} := \emptyset$
2) For $i = 1, ..., \alpha$: Set $\hat{F}_i :=$ the set of faults in F_{i-1} detected by t_i using TEVA. Set $F_i := F_{i-1} \setminus \hat{F}_i$ .If TEVA classifies fault $f \in F_i$ as not detectable by pattern t_i (i. e. $n_f(t_i) = 1$) then set $\overline{T(f)} := \overline{T(f)} \cup \{t_i\}$.
3) Do exact fault simulation for all faults $f \in F_\alpha$ with patterns from $T \setminus \overline{T(f)}$.

First we determine the expected number of patterns and the expected computational effort for detecting a fault by exact simulation, then the same values are determined for TEVA. Finally a switching criterion is established and the possible savings are quantified.
Let c_S: Characteristic constant of the complexity of the exact fault simulation method used.
C: Size of the circuit.
α: Number of test patterns to be evaluated.
Let d_f, e_f, n_f and u_f be the probability that $d_f(t)$, $e_f(t)$, $n_f(t)$ and $u_f(t)$, respectively, are true if t is a random pattern.
The expected number of random patterns until detecting and dropping fault f is described by the formula

$$(1) \quad E(f) = \sum_{i=1}^{\alpha} i(1-d_f)^{i-1} \cdot d_f.$$

Some simple transformations yield

$$(2) \quad E(f) = -d_f \frac{\delta \sum_{i=1}^{\alpha} (1-d_f)^i}{\delta d_f} = \frac{1-(1-d_f)^\alpha}{d_f} - \alpha(1-d_f)^\alpha.$$

The probability of not detecting a fault during simulation is $(1-d_f)^\alpha$, and $\alpha \cdot (1-d_f)^\alpha$ patterns are expected. Hence on the average

$$(3) \quad a(f) := E(f) + \alpha(1-d_f)^\alpha := \frac{1-(1-d_f)^\alpha}{d_f}$$

patterns have to be simulated for fault f, and the expected simulation costs are

$$(4) \quad c_S \cdot \sum_{f \in F} a(f) \cdot C.$$

For estimating the total effort for the combination of approximate and exact fault simulation the following parameters are used

$\hat{e}_f$: $\frac{e_f}{d_f}$ describes the quality of the pessimistic approximation of TEVA
$\hat{n}_f$: $\frac{n_f}{1-d_f}$ describes the quality of the optimistic approximation of TEVA
α_f : $(1-n_f)\alpha$ expected number of patterns $T \setminus \overline{T(f)}$ to be simulated
$\tilde{d}_f$: $\frac{d_f}{1-n_f}$ is the conditional detecting probability if a pattern from $T \setminus \overline{T(f)}$ is simulated.
$C_T(\beta)$: the characteristic constant of the complexity of TEVA using parameter β.

As the effect of fault dropping on the computational effort of TEVA is negligible step 2 leads to an worst case effort of order

$$(5) \quad C_T(\beta) \cdot \alpha \cdot \ln(C) \cdot C$$

The probability of a fault not being detected in Step 2) is $(1-e_f)^\alpha$. Hence in step 3 we expect a simulation time in the order of

$$c_S \cdot \sum_{f \in F} a'(f)C \text{ where}$$

$$a'(f) = (1 - e_f)^\alpha \frac{1 - (1-\tilde{d}_f)^{\alpha_f}}{\tilde{d}_f}.$$

Due to the power series of the exponential function we estimate $(1-e_f)^\alpha = (1-\hat{e}_f \cdot d_f)^\alpha \approx (1-d_f)^{\hat{e}_f \alpha}$, and

$$\frac{1 - (1 - \tilde{d}_f)^{\alpha_f}}{\tilde{d}_f} = \frac{1 - \left(1 - \frac{d_f}{1 - n_f}\right)^{\alpha_f}}{\frac{d_f}{1 - n_f}} \approx \frac{1 - (1 - d_f)^\alpha}{\frac{d_f}{1 - n_f}},$$

and hence we have

$$a'(f) = (1-d_f)^{\hat{e}_f \alpha} (1-\hat{n}_f(1 - d_f)) \frac{1-(1-d_f)^\alpha}{d_f}.$$

The costs of simulating fault f are $a(f) = \frac{1-(1-d_f)^\alpha}{d_f}$, the costs of simulating fault f after TEVA are a'(f) which is only a fraction:

$$(6) \quad \frac{a'(f)}{a(f)} = (1-d_f)^{\hat{e}_f \alpha} \cdot (1-\hat{n}_f(1-d_f)).$$

The results reported in section 4 show average values for $\hat{e}_f$ significantly larger than 0.5 and for $\hat{n}_f$ significantly larger than 0.9. Hence the fraction (6) is very small, its

exact size depends on the actual detectibility d_f, too.
In general it is not useful to process all the patterns by TEVA as after reducing the fault list by fault dropping TEVA loses and exact simulation gains efficiency. The optimal switching point is determined as described in [7].

Assume after applying i patterns the subset $\tilde{F}$ of faults is still undetected by TEVA. Then the expectation value for each $e_f(t)$, $f \in \tilde{F}$, is $E(e_f) = \frac{1}{i+2}$ [7].

An estimation of the portion of faults not detectable by TEVA but by simulation is $1-\hat{e}$, where $\hat{e} := \frac{1}{|F|} \sum_{f \in F} \hat{e}_f$ is the average quality of TEVA. Hence by applying another pattern to TEVA we expect an increase of fault coverage of at least $\Delta_T := (|\tilde{F}| - (1-\hat{e})|F|) \frac{1}{i+2}$, until $|\tilde{F}| > (1-\hat{e})|F|$.

By applying another pattern to exact simulation we expect an increase of fault coverage by

$$\Delta_S := |\tilde{F}| \cdot E(p_f) = |\tilde{F}| \cdot \frac{1}{\hat{e}} E(e_f) = \frac{|\tilde{F}|}{\hat{e}(i+2)}.$$

The costs of fault coverage in time units determine the switching point: Using formula (4) and (5), if

$$\Delta_S \cdot \frac{1}{C_S |\tilde{F}|} > \Delta_T \cdot \frac{1}{C_T(\beta) \ln(C)}$$

then exact simulation becomes more efficient than TEVA. As Δ_S and Δ_T are only estimations the switching point should not be precomputed statically but checked dynamically. If the expected gain Δ_T in fault coverage is not reached for several times, we should dispense with TEVA and switch to exact simulation. On the other hand, if the actual increase Δ_T in fault coverage exceeds we continue with TEVA.

4. Experimental results

The usefulness of the presented approach depends on the answers of two questions. First we have to examine if the estimated values are sufficiently precise, i. e. if the average value of u_f is small enough. Second we investigate whether the approximate technique is fast enough in order to achieve an improvement of performance for the combination of exact and approximate simulation. It will be shown that both questions can be positively answered for the large combinational and sequential ISCAS-benchmark circuits [Brgl85, 89]. Only the combinational parts of the sequential circuits are used.
In order to evaluate the average quality of the pessimistic and the optimistic approximation of TEVA, each circuit was simulated with a test set T of 6000 pattern by both techniques. For each pattern t we determined $d_f(t)$, $e_f(t)$, $n_f(t)$ and $u_f(t) := 1-e_f(t) - n_f(t)$. The average quality of the pessimistic and the optimistic estimation is

$$\hat{e} := \frac{1}{|F|} \sum_{f \in F} \hat{e}_f \text{ and } \hat{n} := \frac{1}{|F|} \sum_{f \in F} \hat{n}_f .$$

The average degree of uncertainty is $u := \frac{1}{|F|} \sum_{f \in F} \frac{\sum_{t \in T} u_f(t)}{|T|}$.

The values of d, e and n are computed the same way. Table 2 contains these experimentally determined values, which demonstrates the quality of TEVA.

circuit	u	$\hat{e}$	$\hat{n}$
C1908	0.156	0.514	0.913
C2670	0.185	0.294	0.908
C3540	0.156	0.086	0.929
C5315	0.189	0.229	0.886
C6288	0.254	0.096	1
C7552	0.171	0.324	0.977
S9234	0.116	0.612	0.933
S13207	0.099	0.649	0.962
S15850	0.111	0.584	0.949
S35932	0.151	0.545	0.897
S38584	0.078	0.744	0.967
S38417	0.113	0.588	0.945

Table 2: Quality of the estimations

The quality of the pessimistic approximation increases with the circuit size and is for the six larger circuits on the average 0.62. The average quality of the optimistic approximation is 0.942. The quality of the approximate method also depends on the circuit structure, e.g. the quality of the estimations for circuit C6288 is much worse than for the other circuits. This circuit has a large portion of fanout stems, each of these fanout stems has dependent successors. In contrast to this, the quality of the approximations for S38584 is very high. This circuit has less fanout stems than S35932 and S38417. A lot of these fanout stems have only independent successors.
Table 3 shows that the high quality estimations are obtained faster than exact simulation can be performed, and that TEVA depends linearly on the circuit size.

	TEVA		SIM		
circuit	time	CPU-ms / node	time	CPU-ms / node	$\frac{time_{SIM}}{time_{TEVA}}$
C1908	0.6	6.6	1.3	13.9	2.1
C2670	0.4	2.6	1.4	9.8	3.8
C3540	0.8	4.8	3.5	20.3	4.2
C5315	0.6	2.4	4.2	17.0	7.0
C6288	0.9	3.5	14.3	58.4	16.7
C7552	1.5	4.0	11.8	31.6	7.8
S9234	1.6	2.7	13.2	22.6	8.3
S13207	2.0	2.3	17.8	20.6	8.9
S15850	2.4	2.4	30.7	29.0	12.7
S35932	5.8	3.3	184.5	103.5	32.2
S38584	8.2	3.5	214.0	89.7	26.3
S38417	6.6	3.2	180.2	87.0	27.0

Table 3: Computing times in CPU-s for simulating 32 patterns

The last column is the ratio of $time_{SIM}$ and $time_{TEVA}$

which increases with the circuit size. The preprocessing task is the same for both techniques, has to be performed only once and is not taken into account. All results of TEVA are obtained on a SPARC 1+ workstation using the parameter $\beta = 5$.

The complexity constants C_S and $C_T(\beta)$ are determined experimentally using quadratical minimization techniques and we get $C_S = 9.73 \cdot 10^{-9}$ and $C_T(5) = 1.02 \cdot 10^{-6}$.

Table 4 contains the fault coverage F_S where it is useful to switch from TEVA to exact simulation. α_S is the number of patterns simulated by TEVA before switching to exact simulation.Moreover it shows the time for the exact simulation and the fault coverage obtained by SIM.

		TEVA		SIM	
circuit	α_S	time	F_S	time	F
S9234	928	63.8	0.36	199.1	0.69
S13207	1312	153.0	0.50	365.1	0.74
S15850	576	67.4	0.47	268.5	0.79
S35932	160	51.0	0.47	289.3	0.90
S38584	4046	1444.6	0.68	2628.3	0.84
S38417	576	210.8	0.48	1087.5	0.92

Table 4: Fault coverages and CPU-times in s for exact simulation and TEVA (simulating α_S patterns)

TEVA requires computing times 1.5 through 5 times less than exact simulation in order to determine the detection of 52% through 81% of the detectable faults. Computing time is still saved if TEVA and exact simulation are combined. The results of simulating α_S patterns with TEVA and than simulating the following patterns with the exact simulator are shown in table 5. The combination of TEVA and exact simulation described in section 3 could not be examined because at the current stage of the implementation of the exact fault simulator it is not possible to eliminate those patterns where $n_f(t) = 1$. In order to exploit the additional information of the optimistic approximation obtained by TEVA the exact simulator will be modified.

	simulation		TEVA+simulation	
circuit	F	time	F	time
S9234	0.7349	325.7	0.712	245.9
S13207	0.8133	571.7	0.8016	458.0
S15850	0.8395	413.4	0.8132	249.9
S35932	0.8959	426.87	0.8959	326.9
S38584	0.9311	4218.6	0.9256	4071.4
S38417	0.8616	1757.2	0.8525	1143.4

Table 5: Fault coverages and computation time in CPU-s for the exact simulation and combining TEVA with simulation for $2\alpha_S$ pattern

Experimental results have demonstrated that the choice of the parameter β, which limits the cardinality of the sets N, has only small influence on the accuracy of the results. For all of the examined circuits the fault coverage is nearly constant - independent of the value we assign to β. Since the parameter primarily influences the computation time and not the accuracy of the simulation results, the value 5 for β seems to be a good choice.

Conclusion

An efficient method for evaluating the fault coverage obtained by large test sets has been presented. With nearly linear effort it is determined whether a certain pattern definitely detects a fault or is unable to detect or whether no information is obtainable by this method. Only for the latter case exact simulation is required. Overall this leads to a significant reduction of computing time.

References

1 Antreich, K.J.; Schulz, M.H.: Accelerated Fault Simulation and Fault Grading in Combinational Circuits; IEEE Trans. CAD, Vol. CAD-6, No. 5, 1987, pp. 704-712

2 Abramovici, M.et al.: Critical Path Tracing - An Alternative to Fault Simulation; Proc. of 20th Design Automation Conference, 1983, pp. 214-220

3 Armstrong, D.B.: A Deductive Methode for Simulating Faults in Logic Circuits in: IEEE Trans. on Comp., Vol. C-21, No. 5, May 1972

4 Barzilai, Z. et al.: HSS- A High Speed Simulator; in: IEEE Transactions on Computer-Aided Design, Vol. 6, No. 4, July 1987, pp 601-617

5 Brglez, F.: A Fast Fault Grader: Analysis and Applications; Proc. International Test Conference, pp. 785-794, 1985

6 Brglez, F.et al.: Combinational Profiles of Sequential Benchmark Circuits in Proc. ISCAS'89, Portland, Oregon 1989, pp.1929-1934

7 Daehn, W.: A Switching Criterion for Hybrid ATPG; Proc. 1st European Test Conference, 1989, pp.26-32

8 Goel, P.: Test Generation Costs Analysis and Projections; Proc. 17th Design Automation Conference, June 1980, pp. 77-84

9 Harel, D.; Krishnamurthy, B.: Is there Hope for Linear Fault Simulation? Proc. 17th FTCS, 1987, pp. 28-33

10 Hong, S.J.: Fault Simulation Strategy for Combinational Logic Networks; Proc. 8th International Symposium of Fault Tolerant Comp. (FTCS-8), 1978, pp. 96-99

11 Akers, S.B.et al.:Why is less information from logic simulation more useful in fault simulation? Proc. IEEE International Test Conference 1990, pp.786-800

12 Maamari, F.; Rajski J.: A Reconvergent Fanout Analysis for Efficient Exact Fault Simulation of Combinational Circuits; Proc. FTCS-18, Tokyo 1988

13 Schulz, M.H.: Testmustergenerierung und Fehlersimulation in digitalen Schaltungen mit hoher Komplexität; Informatik-Fachberichte 173, Springer-Verlag 1988

14 Thompson, E.W.; Szygenda, S.A.: Parallel Fault Simulation; Computer, March 1975

15 Tarjan, R.E.: Finding Dominators in a directed graph; SIAM Journ. of Computing, Volume 3, 1974 pp. 62-89

16 Ulrich, E.G. and Baker, T.: The Concurrent Simulation of Nearly Identical Digital Networks; Proc. of 10th Design Automation Workshop,vol.6, pp.145-150, 1973

17 Waicukauski, J.A. et al.: Fault Simulation for Structured VLSI; VLSI Systems Design, Dec. 1985

18 Wunderlich, H.J.: Hochintegrierte Schaltungen: Prüfgerechter Entwurf und Test, Springer-Verlag, Berlin, Heidelberg, New York, 1991