Compaction of Pass/Fail-based Diagnostic Test Vectors for Combinational and Sequential Circuits [*]

Yoshinobu Higami, Kewal K. Saluja[*], Hiroshi Takahashi, Shin-ya Kobayashi and Yuzo Takamatsu

Department of Computer Science, Ehime University
[*]Department of Electrical and Computer Engineering, University of Wisconsin-Madison

Abstract— Substantial attention is being paid to the fault diagnosis problem in recent test literature. Yet, the compaction of test vectors for fault diagnosis is little explored. The compaction of diagnostic test vectors must take care of all fault pairs that need to be distinguished by a given test vector set. Clearly, the number of fault pairs is much larger than the number of faults thus making this problem very difficult and challenging. The key contributions of this paper are: 1) to use techniques for reducing the size of fault pairs to be considered at a time, 2) to use novel variants of the fault distinguishing table method for combinational circuits and reverse order restoration method for sequential circuits, and 3) to introduce heuristics to manage the space complexity of considering all fault pairs for large circuits. Finally, the experimental results for ISCAS benchmark circuits are presented to demonstrate the effectiveness of the proposed methods.

I. Introduction

Increase in testing cost is one of the most significant problems in testing LSI circuits. This is primarily due to the fact that testing large scale circuits requires a large number of test vectors which in turn increase the test application time, tester memory space, and hence the total test cost. Numerous research papers have proposed various compaction techniques to reduce the number of test vectors and the volume of test data [4]. In particular, [8] studied the effect of detection-oriented test compaction on fault diagnosis experimentally and showed the loss of diagnosis capability of such methods. Little or no attention has been paid to reduce the number of diagnostic vectors. We believe that with respect to fault diagnosis, reducing the number of test vectors is also important. Reduction of the number of diagnostic test vectors will shorten real execution time for fault diagnosis, reduce tester memory space, and thus reduce the cost of design debug. Further, it will also allow a reduction in the size of fault dictionary, which is an important method for fault diagnosis.

Below, we briefly explain the relation between the compaction for fault detection and fault diagnosis. This is included here for the sake of completeness of this paper and can be found in [5]. While compacting detection test vectors, it is important to find redundant vectors efficiently. An approach to find redundant vectors is to use a fault detecting table which contains information about detection of all faults by every test vector. If the fault detecting table is given, compaction of detection test vectors can be formulated as a set-cover problem.

Compaction of diagnostic test vectors can also be similarly formulated by using a fault distinguishing table, which contains information about distinguishability of all fault pairs by every test vector. Once the fault distinguishing table is obtained, the minimum number of diagnostic test vectors can be obtained by solving the set-cover problem as before. However, it is important to note that number of fault pairs is a square function of the number of faults in the circuit and hence it may be rather difficult, if not impossible, to store a complete fault distinguishing table to arrive at an optimal solution. How to reduce the size of fault pairs that need to be stored in memory is one of the very important components for compaction of diagnostic test vectors

In this paper, we propose algorithms to compact diagnostic test vectors for combinational and sequential circuits. The proposed algorithms introduce some heuristics for reducing the size of the fault distinguishing table to be considered at any time. The fault distinguishing table is constructed by considering a chosen subset of fault pairs. A nearly minimum set of test vectors that can distinguish the fault pairs is selected by solving the set-cover problem. After that, fault simulation is performed to find the fault pairs that are distinguished by the selected test set. If not all the fault pairs are distinguished, then further construction of the fault distinguishing table and selection of test vectors is performed. In the present work, we only consider the pass/fail information, that is, we do not take into account the locations of primary outputs where a fault effect is propagated. Thus our method is well suited for BIST-based fault diagnosis [2, 9]. None the less we must add that our method is general and is also applicable for diagnosis methods that contain the information on locations of primary outputs as well as extra observation points [7] to which a fault effect is propagated. Note also that this work is different from the method that deal with generation of diagnostic test sets as in [1].

Compaction of test vectors for sequential circuits is even more difficult than that for combinational circuits, just as di-

[*]The work was supported in part by JSPS under the Grant-in-Aid for Scientific Research

agnostic fault simulation and test generation for sequential circuits are more difficult than for combinational circuits. The proposed method uses a variant of reverse order restoration (ROR) technique proposed in [3] and found to be very efficient for detection-oriented test compaction for sequential circuits. When diagnostic test sequences are compacted using ROR, the list of fault pairs distinguished by the original test sequence is required. However, it is difficult to store information about all fault pairs since the number of fault pairs is much larger than the number of faults. Therefore, we also propose a method to reduce the number of target fault pairs for ROR.

The rest of the paper is organized as follows. In Section 2 we give the necessary definitions. In Section 3 a compaction algorithm for combinational circuits is explained and the experimental results for implementing our algorithm are given. In Section 4, compaction algorithms for sequential circuits are explained, and experimental results for sequential circuits are given. Finally in Section 5, conclusions are described.

II. Preliminaries

The problem we consider in this paper is to find a subset of test vectors or test subsequences from a given test set or test sequence such that the new set or sequence is as small as possible while providing the same level or better diagnosis as the original test set or test sequence. We need the following definitions for developing algorithms.

[**Definition 1**]: Two faults f_1 and f_2 are said to be distinguished by a test vector v if there exists at least one test vector v such that v detects f_1 but not f_2 or that v detects f_2 but not f_1.

[**Definition 2**]: A **fault detecting table (FDETT)** for a test set T contains information about faults detected by each test vector. Similarly, a **fault distinguishing table (FDIST)** for T contains information about fault pairs distinguished by each test vector.

Note that we do not take into account the faults that are not detected by the original test set or the original test sequence. This is because a circuit with such an undetected fault can not be identified to be a faulty circuit, and thus such a fault can not be the target of fault diagnosis.

III. Compaction for combinational circuits

A. Basic idea

If we can construct a complete FDIST, that is, if the information about all the pairs of detected faults is available, the minimum test vectors can be selected by solving the set-cover problem. However, since the number of fault pairs is generally very large, it is difficult to store a complete FDIST. Therefore we must work with a partial FDIST. The partial FDIST includes information only about a subset of all possible candidate fault pairs. Using such a partial FDIST, a small number of test vectors are selected, and then fault simulation is performed with the selected test vectors. If the selected test vectors do not achieve the original diagnostic resolution, another subset of fault pairs is selected and a partial FDIST is obtained to select additional test vectors. Such process is repeated until all distinguishable fault pairs are distinguished. The following theorem provides conditions for an initial choice of test vectors.

[**Theorem 1**]: Let V_0 be a set of test vectors, and Fd_1 be the set of those faults each of which is detected by only one test vector in V_0. Further, let FP be a set of fault pairs constructed from faults in Fd_1 and distinguished by $^{\forall}v \in V_0$. Now collect the test vectors that detect $^{\forall}f \in Fd_1$, and the set of the test vectors is denoted by V_1. If any two test vectors are removed from V_1 and the resulting vector set is V_1', then at least one fault pair in FP can not be distinguished by $^{\forall}v \in V_1'$.

(Proof:) Let v_1 and v_2 be test vectors in V_1, and f_1 and f_2 be faults detected only by v_1 and v_2, respectively. If v_1 and v_2 are removed from V_1, then fault pair $< f_1, f_2 >$ is not distinguished by the remaining vectors in V_1. Since V_1 contains only test vectors that detect $f \in Fd_1$, the above statements are true no matter which pair of test vectors are selected as v_1 and v_2.

B. Proposed algorithm

We now state an algorithm for diagnostic compaction for combinational circuits, called DCOMP-C. This algorithm makes use of the above result in selecting the initial vector set, thus keeping the FDIST to a manageable size. The steps of the algorithm are explained following the algorithm proper and an example is also given.

Algorithm: DCOMP-C

```
/* $V_0$: a given test set */
1) Set $V_s = \phi$.
2) Perform fault simulation with $V_0$ and obtain $Fd_1$, which is a set of faults detected by only one test vector in $V_0$.
3) Collect test vectors that detect faults in $Fd_1$, add the test vectors to $V_s$, and remove the test vectors from $V_0$.
4) Perform fault simulation with $V_s$ and select $n_p$ fault pairs among the fault pairs that are not distinguished by any $v \in V_s$. Let $P$ be a set of the selected fault pairs.
5) Perform fault simulation with $V_0$ and $P$ in order to construct an FDIST.
6) If no fault pairs in $P$ are distinguished by any $v \in V_0$, then go to 8).
7) Select test vectors among $V_0$ that distinguish fault pair $p \in P$ by solving the set-cover problem, add the selected test vectors to $V_s$, remove them from $V_0$, and go to 4).
8) Obtain all the fault pairs that are not distinguished by $v \in V_s$. Let $P$ be a set of selected fault pairs.
9) Make an FDIST with $V_0$ and $P$.
10) Select test vectors among $V_0$ that distinguish fault pair $p \in P$ and add them to $V_s$. ($V_s$ is a resultant compacted test
```

set.)

In steps 2) and 3), test vectors that detect the faults detected once are collected. Theorem 1 implies that all such test vectors are necessary even in the compacted test set for diagnosis, precisely speaking, all except for one test vector are necessary. In steps 4) and 5), n_p fault pairs are selected among the fault pairs that are not distinguished by the currently obtained test set, and an FDIST is constructed, where n_p is a predetermined number. After the FDIST is constructed, approximately minimum number of test vectors can be obtained by solving the set-cover problem so that they can distinguish the selected fault pairs in step 7). This process is repeated until no fault pair among the selected fault pairs is distinguished by the remaining vectors in V_0, in step 6). Usually, set P includes faults distinguished and not distinguished by V_0. After selecting sufficiently many test vectors, P includes no fault pairs distinguished by V_0. In such a case, the process goes to step 8), where fault pairs undistinguished by V_s are all collected. By selecting test vectors that can distinguish such fault pairs except for indistinguishable ones, the obtained test set V_s can distinguish all the fault pairs distinguished by the initial V_0.

Example1: Consider a test set $V_0 = \{v_1, v_2, v_3, v_4, v_5\}$ and faults $f_1, f_2, f_3, f_4, f_5, f_6$ and f_7 that are detected by $v \in V_0$. Table I shows an FDETT for this example. (Note that DCOMP-C does not use such a table explicitly.) Consider the case where DCOMP-C is applied to V_0. In step 2), faults that are detected only by one test vector are collected, and $Fd_1 = \{f_1\}$ is obtained. In step 3), test vectors that detect the fault in Fd_1 are collected, and $V_s = \{v_1\}$ is obtained. In step 4), fault simulation is performed with v_1, and n_p fault pairs are selected among the fault pairs that are not distinguished by v_1. Now suppose that n_p is set to 2 and $P = \{< f_2, f_3 >, < f_4, f_5 >\}$ is obtained. In step 5), fault simulation is performed with $V_0 = \{v_2, v_3, v_4, v_5\}$, and an FDIST is constructed. In step 7), the set-cover problem is solved. In this case, suppose that v_2 and v_3 are selected. The process goes to step 4) again. Next suppose that $P = \{< f_3, f_4 >, < f_5, f_6 >\}$ is obtained in step 4). In step 6), since no fault pair in P is distinguished by V_0, the process goes to step 8). Fault pairs that are not distinguished by current V_s are all collected, and $P = \{< f_3, f_7 >, < f_4, f_7 >\}$ is obtained. In step 9), the FDIST is constructed, and in step 10), v_5 is selected. Finally the compacted test set $V_s = \{v_1, v_2, v_3, v_5\}$ is obtained.

C. Experimental results

We implemented DCOMP-C in C programming language and ran it on a Pentium IV 2.6GHz platform targeting ISCAS'85 and ISCAS'89 (scan version) benchmark circuits. In these experiments, 1024 random vectors were used as a given test set. Table II shows results by DCOMP-C, where name of the circuit (circuit), fault coverage (cov), the number of fault pairs (pair), the number of undistinguished fault pairs (undis), the number of compacted test vectors (vect), the percentage

TABLE I
EXAMPLE OF AN FDETT

	v_1	v_2	v_3	v_4	v_5
f_1	d				
f_2		d	d	d	
f_3			d	d	
f_4			d	d	
f_5				d	d
f_6				d	d
f_7			d	d	d

d: detected

of removed test vectors for the number of original test vectors (%) and CPU time in seconds (cpu) are shown. The predetermined number n_p was set to 100,000 and 1,000. It is evident from these results that the compaction algorithm provides substantial test compaction without loss of diagnostic resolution. In comparison of results between $n_p = 100,000$ and $n_p = 1,000$, fewer test vectors and shorter CPU time are achieved with $n_p = 100,000$ than $n_p = 1,000$. For the circuit s38584, the test vectors were compacted to 509 test vectors, while handling over half a billion fault pairs. In general, as the table shows, the proposed algorithm can deal with circuits having several hundred million fault pairs in an efficient manner.

IV. Compaction for sequential circuits

A. Reverse order restoration

Test compaction for sequential circuits is substantially more difficult than that for combinational circuits. In case of sequential circuits when a test vector is deleted from a test sequence, faults detected by the original test sequence may become undetected by the compacted test sequence as well as new faults may be detected by the compacted sequence. Therefore in order to keep fault coverage and diagnostic resolution as high as the original one, fault simulation must be performed or state transition must be checked. Reverse order restoration (ROR) has been proposed [3] as an efficient detection oriented test compaction method. ROR first removes all the test vectors except for initialization vectors. Next it restores test vectors so that a subset of all faults are detected. The restoration is repeated until all the originally detected faults are detected. As this method forms the basis of our approach, we briefly explain how ROR works through the following example.

Example 2: Suppose that a test sequence T_0 consisting of seven vectors $v_1, v_2, ..., v_7$ is given. Sequence T_0 detects 3 faults f_1, f_2 and f_3, and f_1, f_2 and f_3 are detected by v_3, v_4 and v_7, respectively. Table III shows an FDETT for T_0. ROR first removes all the test vectors except for the initialization vectors. Now suppose that v_1 and v_2 are restored. Compacted test sequence T_c initially consists of v_1 v_2. Next, faults that are detected at the latest time are collected. In this example this is fault f_3 which is detected by the test vector v_7. Therefore, v_7 is restored. After that, fault simulation is performed to

TABLE II
RESULTS BY DCOMP-C

circuit	cov	pair	undis	$n_p = 100,000$ vect	%	cpu(s)	$n_p = 1,000$ vect	%	cpu(s)
c432	97.52	$1.30 * 10^5$	93	68	93.4	0.1	71	93.1	0.1
c880	97.52	$4.45 * 10^5$	104	63	93.8	0.2	70	93.2	0.3
c1355	98.57	$1.25 * 10^6$	878	88	91.4	0.8	89	91.3	1.3
c1908	94.12	$1.84 * 10^6$	1208	139	86.4	2.1	144	85.9	2.8
c2670	84.40	$3.08 * 10^6$	1838	79	92.3	2.8	79	92.3	3.9
c3540	94.49	$5.95 * 10^6$	1585	205	80.0	10.6	207	79.8	16.6
c5315	98.83	$1.57 * 10^7$	1579	188	81.6	15.4	199	80.6	25.1
c6288	99.56	$2.97 * 10^7$	4491	37	96.4	1659	38	96.3	3560
c7552	91.97	$2.76 * 10^7$	4438	198	80.7	33.8	208	79.7	110.5
s5378	94.55	$9.47 * 10^6$	1802	231	77.4	12.2	245	76.1	21.0
s9234	73.86	$1.31 * 10^7$	6798	246	76.0	36.1	261	74.5	100.6
s15850	85.05	$4.97 * 10^7$	9368	261	74.5	141.8	279	72.8	389.4
s35932	89.81	$6.16 * 10^8$	16655	91	91.1	962.2	96	90.6	2023
s38417	86.58	$3.64 * 10^8$	12480	436	57.4	1848	453	55.8	7021
s38584	90.60	$5.41 * 10^8$	31607	509	50.3	2069	537	47.6	10030

check if $T_c = v_1\ v_2\ v_7$ detects f_3. If it does, fault simulation is performed again in order to drop faults detected by T_c from the fault list. Otherwise, test vector v_6 is restored, and it is checked whether $T_c = v_1\ v_2\ v_6\ v_7$ detects f_3. After f_3 is detected by T_c, test vectors are restored while targeting fault f_2. These steps are repeated until all the target faults are detected.

TABLE III
EXAMPLE OF AN FDETT

	v_1	v_2	v_3	v_4	v_5	v_6	v_7
f_1			d				
f_2				d			
f_3							d

d: detected

B. Basic algorithm

We now present a diagnostic compaction algorithm for sequential circuits using ROR, called DCOMP-S. Similar to the detection-oriented ROR, DCOMP-S first removes all the test vectors, except for initialization vectors. After that, it restores test vectors such that a targeted subset of fault pairs are distinguished. The restoration is repeated until all fault pairs distinguished by the original test sequence are distinguished. The steps of the algorithm are explained following the algorithm and an example is given to further clarify the steps.

```
Algorithm: DCOMP-S
/* $T_0$: a given test sequence */
/* $v_i$: $i$-th test vector in $T_0$ */
/* $F_0$: a set of faults detected by $T_0$ */
/* $FP_0$: a set of fault pairs that are constructed from $f \in F_0$ and distinguished by $T_0$ */
1) Set $T_c = \phi$ and $FP_{trg} = \phi$.
2) Restore test vectors from $v_1$ to $v_{init}$ as initialization vectors.
3) Drop fault pairs distinguished by $T_c$ from $FP_0$.
4) Find the latest time among the times when fault pairs in $FP_0$ are distinguished. Let the time to be $t$.
5) Add fault pairs that are distinguished by $v_t$ to $FP_{trg}$.
6) Restore $v_t$ to $T_c$.
7) Check whether fault pairs in $FP_{trg}$ are distinguished by $T_c$.
8) If at least one fault pair is not distinguished, then $t = t - 1$ and go to 5).
9) Drop fault pairs distinguished by $T_c$ from $FP_0$, and set $FP_{trg} = \phi$.
10) If $FP_0 \neq \phi$, then go to 5).
```

First, the algorithm removes all the test vectors and restores $init$ test vectors as initialization vectors, where $init$ is a predetermined number and it is usually a small number. At this time, compacted test sequence $T_c = v_1\ v_2 \ldots\ v_{init}$. Fault pairs that are distinguished by T_c are dropped from FP_0 in step 3). Next the algorithm investigates the time when each fault pair is distinguished in T_0, and finds the latest time t among them in step 4). Note that when a fault pair is distinguished at more than one time, only the latest time is used and all other times are ignored. Fault pairs that are distinguished by v_t in T_0 are collected in step 5), and they are targeted for vector restoration. In step 6), test vector v_t is restored to T_c. If FP_{trg} includes only fault pairs distinguished by v_t, then v_t is concatenated at the end of T_c. Otherwise, v_t is inserted at the one time earlier position than v_{t+1} in T_c. If the current T_c does not distinguish at least one fault pair in FP_{trg}, time t is decreased in step 8) and v_t is further restored in step 6). If T_c distinguishes all the pairs in FP_{trg}, other fault pairs that are distinguished by T_c are dropped from FP_0 in step 9).

Example 3: Suppose that a test sequence T_0 consisting of 7 vectors is given, and that T_0 distinguishes 5 fault pairs fp_1, fp_2, fp_3, fp_4 and fp_5. Table IV shows an FDIST of T_0.

TABLE IV
AN FDIST OF T_0

	v_1	v_2	v_3	v_4	v_5	v_6	v_7
fp_1							D
fp_2						D	
fp_3				D			
fp_4					D		
fp_5				D			

D: distinguished

Now suppose that v_1 and v_2 are first restored as initialization vectors. Fault simulation is performed to check if each fault pair is distinguished by $T_c = v_1\ v_2$. In this case no fault pair is distinguished. Next the time when each fault pair is distinguished is investigated, and it is found that the latest one t is set to 7 in step 4). In step 5) fault pairs distinguished by v_7 are collected, and $FP_{trg} = \{fp_1\}$ is obtained. Vector v_7 is restored in step 6), and it is checked whether fp_1 is distinguished by $T_c = v_1\ v_2\ v_7$. If it is not distinguished, then t is decreased and the process goes to step 5). In step 5), fault pair fp_2 is added to FP_{trg}, and in step 6), v_6 is restored. It is inserted between v_2 and v_7, thus $T_c = v_1\ v_2\ v_6\ v_7$ is obtained. In step 7), it is checked whether fp_1 and fp_2 are both distinguished by T_c. If both of them are distinguished, then it is checked whether other fault pairs fp_3, fp_4 and fp_5 are distinguished by T_c. In this case suppose that fp_4 and fp_5 are distinguished. Next the process goes to step 4), and $t = 4$ is obtained, because the remaining fault pair fp_3 is distinguished by v_4 in T_0. In step 6), v_4 is restored, and it is concatenated at the end of T_c. It is checked whether fp_3 is distinguished by T_c in step 7). If it is distinguished, then the process is terminated and finally $T_c = v_1\ v_2\ v_6\ v_7\ v_4$ is obtained.

C. Results by DCOMP-S

We implemented DCOMP-S algorithm using C programming language, and experimented with ISCAS'89 benchmark circuits on a Pentium IV 2.6GHz platform. We used test sequences generated by HITEC[6] as initially given test sequences. Table V shows the results by DCOMP-S. In the table, circuit name (circuit), the original test length (len), compacted test length (comp), the percentage of removed test vectors for the original test length (%), the total number of target fault pairs (pair), the number of fault pairs undistinguished by the original test sequences (undis) and CPU time in seconds (cpu) are given. Variable *init* was set to 2% of the original test length. It is found that for every circuit, more than 10% test vectors could be removed from the original test sequences. In the best case, for s526, about 72% test vectors could be removed. These are all relatively small circuits. DCOMP-C can not be used for large circuits as the size of the FDIST will be too large to store. This aspect is dealt in the next subsection.

TABLE V
RESULTS BY DCOMP-S

circuit	len	comp	%	pair	undis	cpu(s)
s344	127	82	35.4	50721	367	0.3
s349	134	85	36.6	52650	402	0.3
s382	2074	846	59.2	48516	1806	16.4
s386	286	164	42.7	49141	106	0.7
s400	2214	845	61.8	61075	2343	24.9
s444	2240	956	57.3	75466	2994	21.3
s526	2258	631	72.1	64980	6617	19.4
s641	209	123	41.1	80601	129	0.7
s713	173	113	34.7	113050	383	0.8
s820	1115	745	33.2	330078	284	33.9
s832	1137	761	33.1	333336	270	42.6
s1196	435	295	32.2	766941	167	18.2
s1238	475	316	33.5	822403	191	22.9
s1423	150	134	10.7	261003	3823	4.9
s1488	1170	787	32.7	1041846	355	75.7
s1494	1245	772	38.0	1054878	411	106.2

D. Algorithm applicable for large circuits

DCOMP-S algorithm needs the information of all the fault pairs that are distinguished by the original test sequence, and hence can compact test sequences only for small circuits. For large circuits, FP_{trg}, a set of target fault pairs, can not be stored. We now propose another algorithm applicable for large circuits, called DCOMP-LS. DCOMP-LS has similar steps as DCOMP-S. It first constructs a compacted test sequence T_c by selecting S test vectors v_1 to v_S among the original test sequence T_0, where S is predetermined. Fault simulation is performed in order to collect fault pairs that are distinguished by T_0 but not distinguished by the initial T_c. The set of the collected fault pairs is referred to by FP_{trg}. By making S large, FP_{trg} can be small enough to store in a computer memory. Although DCOMP-S requires the list of all the target fault pairs, denoted by FP_0, DCOMP-LS requires only a subset of fault pairs FP_{trg}, which can be made much smaller than FP_0. Only the fault pairs in FP_{trg} are targeted during vector restoration. Next test vectors are restored from T_0 such that fault pairs in FP_{trg} are all distinguished. DCOMP-LS algorithm is described below.

```
Algorithm: DCOMP-LS
/* $T_0$: a given test sequence */
1) Set $T_c = \phi$.
2) Restore test vectors from $v_1$ to $v_S$ in $T_c$.
3) Perform fault simulation, and collect fault pairs distinguished by $T_0$ but not distinguished by $T_c$. Let $FP_{trg}$ be a set of the collected fault pairs.
4) Same as step 4) to 10) in DCOMP-S algorithm.
```

E. Results by DCOMP-LS

DCOMP-LS algorithm was implemented and experiments were performed on ISCAS' 89 benchmark circuits. First we compare the results by DCOMP-LS with those by DCOMP-

S. In this experiment, test sequences generated by HITEC[6] were also used as original test sequence, and variable S in DCOMP-LS was set to 50% of the original test length. Table VI shows the results, where column "S" and "LS" show the results by DCOMP-S and DCOMP-LS, respectively. The results by DCOMP-S are the same as in Table V. Columns "compacted vectors" show the number of compacted test vectors, column "%" shows the percentage of the number of deleted test vectors for the original test length, and columns "cpu(s)" show CPU times in seconds. Although DCOMP-LS could not achieve as short test sequences as DCOMP-S, it still deleted about 3% to 35% test vectors. CPU times by DCOMP-LS were shorter than DCOMP-S for most of circuits. For s382, s400 and s444, DCOMP-LS restored much more test vectors than DCOMP-S, and this increased the fault simulation time, and thus increased CPU time.

TABLE VI
COMPARISON BETWEEN DCOMP-S AND DCOMP-LS

	compacted vectors				cpu(s)	
circuit	S	%	LS	%	S	LS
s344	82	35.4	98	22.8	0.3	0.1
s349	85	36.6	111	17.2	0.3	0.1
s382	846	59.2	1580	23.8	16.4	18.5
s386	164	42.7	233	18.5	0.7	0.6
s400	845	61.8	1865	15.8	24.9	42.4
s444	956	57.3	1648	26.4	21.3	34.4
s526	631	72.1	1468	35.0	19.4	19.6
s641	123	41.1	177	15.3	0.7	0.4
s713	113	34.7	154	11.0	0.8	0.4
s820	745	33.2	940	15.7	33.9	19.1
s832	761	33.1	998	12.2	42.6	31.5
s1196	295	32.2	366	15.9	18.2	12.5
s1238	316	33.5	398	16.2	22.9	17.7
s1423	134	10.7	145	3.3	4.9	2.6
s1488	787	32.7	992	15.2	75.7	35.2
s1494	772	38.0	1074	13.7	106.2	47.1

TABLE VII
RESULTS BY DCOMP-LS FOR LARGE CIRCUITS

circuit	len	comp	%	pair	undis	cpu(s)
s5378	912	839	8.0	$5.064 * 10^6$	$5.912 * 10^3$	20.0
s35932	496	496	0.0	$6.090 * 10^8$	$6.641 * 10^6$	3409
s35932	546	536	1.8	$6.098 * 10^8$	$6.641 * 10^6$	2502
s35932	1024	921	10.1	$3.753 * 10^8$	$1.121 * 10^8$	3522

Table VII shows the results for s5378 and s35932. In the second and the third row, the results for HITEC test sequences are shown, in the fourth row, the results for HITEC sequence plus 50 random test vectors are shown, and in the fifth row, the results for 5 initialization vectors plus 1019 random test vectors are shown. Variable S in DCOMP-LS was set to 90% of the original test length. Each column in Table VII has the same meaning as that in Table V. For s5378 with the HITEC sequence and s35932 with initialization vectors plus random vectors, 8.0% and 10.1% test vectors could be deleted by DCOMP-LS, respectively, but for s35932 with only HITEC sequence, no compaction was achieved. It should be noted, however, that the proposed algorithm can deal with large circuits that have more than 600 million fault pairs.

V. CONCLUSION

In this paper, we proposed compaction algorithms of diagnostic test vectors for combinational and sequential circuits. The algorithm for combinational circuits uses an FDIST table and finds a subset of test vectors that achieve the same diagnostic resolution as that for the original test set. Since for large circuits a complete FDIST can not be stored, the algorithm uses a partial FDIST and repeats selection of test vectors.

The algorithm for sequential circuits uses a ROR technique. Since the original method needs a complete list of target fault pairs, it is difficult to apply it to large sequential circuits. Therefore, we developed heuristics for use in the improved algorithm for reducing the target fault pairs, and it is applicable to large circuits.

REFERENCES

[1] D. H. Baik, Y. C. Kim, K. K. Saluja, and V. D. Agrawal, "Exclusive test and its applications to fault diagnosis," *Proc. Int. Conf. on VLSI Design*, pp. 143–148, January 2003.

[2] J. G.-Dastidar, d. Das and N. A. Touba, "Fault Diagnosis in Scan-based BIST using both Time and Space Information," in *Proc. Int. Test Conf.*, pp. 95–102, 1999.

[3] R. Guo, I. Pomeranz, and S. M. Reddy, "On speed-up vector restoration based static compaction of test sequences for sequential circuits," in *Proc. Asian Test Sympo.*, pp. 467–471, Dec. 1998.

[4] Y. Higami, S. Kajihara, H. Ichihara and Y. Takamatsu, "Test Cost Reduction for Large Circuits: Reduction of Test Data Volume and Test Application Time," in *Wiley Interscience Journal of Systems and Computers in Japan*, vol. 36, No. 6, pp. 69-83, 2005.

[5] W. H. Kautz, "Fault Testing and Diagnosis in Combinational Digital Circuits," in IEEE Trans. on Computers, vol. EC-15, pp 352-366, April 1968.

[6] T. M. Niermann and J. H.Patel, "HITEC: A test generation package for sequential circuits," in *Proc. European Conf. on Design Automation*, pp. 214–218, Feb. 1991.

[7] I. Pomeranz, S. Venkataraman, and S. M. Reddy, "Z-DFD: Design-for-diagnosability based on the concept of Z-detection," in *Proc. Int. Test Conf.*, pp. 489–497, 2004.

[8] Y. Shao, R. Guo, I. Pomeranz, and S. M. Reddy, "The effects of test compaction on fault diagnosis," in *Proc. Int. Test Conf.*, pp. 1083–1089, 1999.

[9] H. Takahashi, Y. Yamamoto, Y. Higami, and Y. Takamatsu, "Enhancing BIST Based Single/Multiple Stuck-at Fault Diagnosis by Ambiguous Test Set," in *Proc. Asian Test Symp.*, pp. 216–212, 2004.