On the Compaction of Independent Test Sequences for Sequential Circuits

Petros Drineas
Computer Science Department
Rensselaer Polytechnic Institute
Troy, NY 12180
drinep@cs.rpi.edu

Yiorgos Makris
Electrical Engineering Department
Yale University
New Haven, CT 06520
yiorgos.makris@yale.edu

Extended Abstract

Deterministic test generation methods typically target a primary fault and generate a test sequence for detecting it. Since the generated test sequence may also detect ancillary faults, fault simulation is subsequently employed and both the primary and the ancillary faults are eliminated from the fault list. The same fault dropping mechanism is also employed in simulation-based test generation methods, wherein random, pseudo-random, or algorithmically constructed test sequences are fault-simulated on the circuit. In either case, the primary objective is the derivation of a set of test sequences that detects all faults and fault dropping is an essential element in order to reduce test generation time. As a result, test generation methods typically produce a suboptimal set of test sequences, i.e. a set wherein some test sequences (or portions thereof) may be redundant. Elimination or pruning of redundant test sequences is the objective of test compaction, which may be performed either during test generation (dynamic compaction), or after test generation (static compaction). Efficient test compaction methods are very important in order to reduce test storage, test application time, and by extension, test cost.

In this paper, we study a specific instance of the problem, namely the compaction of independent test sequences for sequential circuits. Such test sequences do not rely on any assumptions regarding the initial state of the circuit and are, thus, independent of it. It is also assumed that each test sequence is fault simulated only once, yet without fault dropping so that all detectable faults are obtained. Based on this information, it is possible that some test sequences may be eliminated or pruned without any reduction in fault coverage. Since each test sequence consists of a number of test vectors, the optimization objective of test compaction in this scenario is the minimization of the total number of test vectors in the compacted set of test sequences.

This instance of test compaction was first formulated in [1], where it is shown to be NP-hard and is approximated through Genetic Algorithms. An fast and efficient *Branch-&-Bound* Algorithm for solving this problem has also been proposed recently [2]. While significant levels of compaction within reasonable time are experimentally observed, no indication of proximity to the optimal solution is provided through these method. This deficiency is addressed through the work presented herein; more specifically, we contribute a formulation of the problem as an Integer Program, which is subsequently approximated through Randomized Rounding [3] of its Linear Program relaxation. The major advantage of this approach is that it provides a lower bound for the size of the optimal set of compacted test vectors, namely the optimal solution of the Linear Program relaxation of the Integer Program. Such a lower bound not only establishes a mechanism for assessing the quality of test compaction, but may also provide an informed termination criterion for iterative approaches, such as the solution proposed in [1]. Moreover, experiments with alternative test sets for the ISCAS89 [4] benchmark circuits show that the proposed solution yields almost optimal solutions.

In order to evaluate the proposed methodology we repeat the experiment described in [1], wherein the authors generated sets of independent test sequences for the ISCAS89 [4] benchmark circuits using two different ATPG tools, GATTO [5] and HITEC [6]. Details and the resulting fault detection matrices are available at [7]. These matrices are the starting point for our experiments. Test sequences are extended into subsequences, the proposed method is applied and results are reported in Figures (1)-(2)[1].

The number of test sequences and total vectors in the original test set before compaction are reported in columns 2 and 3. The number of test sequences and total vectors in the compacted test set yielded by the proposed method are reported in columns 4 and 5. The difference between the number of vectors in the identified solution and the theoretical lower bound given by the Linear Program solution is reported in column 6. Column 7 indicates the size of the compacted test set as a percentage of the size of the original test set. Finally, column 8 indicates the test compaction efficiency of the Genetic Algorithms method proposed in [1].

The most important observation is that our approach almost always identifies the optimal solution. As shown in the tables, the distance from the theoretical lower bound is 0 for most circuits. The same observation applies for the results of the Genetic Algorithm described in [1]. One can also observe that, for some circuits, out method achieves better compaction ratio over [1] (i.e. GATTO test set for S3271, HITEC test sets for S1269 and S3271).

The actual running times of our approach are comparable to those reported in [1]. We caution the reader, however, that such a comparison is rather misleading: our algorithm is im-

---

[1] A “*” in the table of Figure (2) indicates a minor discrepancy between the numbers reported in [1] and the size of the tables available from [7].

Circuit	Original Test Set		Compacted Test Set		Distance From Lower Bound	Proposed Method	GA [1] Method
	# Seq	# Vec	# Seq	# Vec		% Red	% Red
S208	36	1096	6	347	0	31.66	31.66
S298	24	302	11	141	0	46.69	46.69
S344	19	141	10	66	0	46.81	46.81
S349	19	144	11	84	0	58.33	58.33
S382	17	840	7	485	0	57.74	57.74
S386	38	418	15	221	0	52.87	52.87
S400	16	916	7	502	0	54.08	54.08
S420	33	797	8	333	1	41.78	41.78
S444	22	1434	9	788	0	54.95	54.95
S499	29	465	9	192	0	41.29	41.29
S510	37	989	7	237	0	23.96	23.96
S526	18	1050	9	769	0	73.24	73.24
S526n	16	862	6	523	0	60.67	60.67
S641	48	395	24	221	0	55.95	55.95
S713	55	557	23	250	0	44.88	44.88
S820	38	669	14	347	0	51.87	51.87
S832	33	425	10	196	0	46.12	46.12
S838	37	1323	12	476	3	35.98	35.75
S938	37	1323	11	473	0	35.75	35.75
S953	75	1099	32	539	0	49.04	49.04
S967	72	1223	31	660	1	53.96	54.70
S991	20	448	9	365	0	81.47	81.47
S1196	133	1805	74	1124	0	62.27	62.66
S1238	123	1554	74	1004	0	64.61	64.80
S1269	52	450	29	245	0	54.44	54.44
S1423	107	2691	28	1279	0	47.53	47.71
S1488	65	1824	19	946	0	51.86	51.86
S1494	62	1244	19	652	0	52.41	52.41
S1512	52	772	14	289	0	37.44	37.44
S3271	132	2529	50	1178	0	46.58	60.58
S3384	58	888	22	410	0	46.17	46.17
S4863	112	1533	42	790	8	50.88	48.66
S5378	71	919	42	493	0	53.65	53.65
S6669	64	592	36	301	0	50.84	51.18
S13207	34	544	9	187	0	34.38	34.38
S15850	10	153	3	91	0	59.48	59.48

Figure 1. Results for GATTO Test Sets

plemented in MatLab – a slow, interpreted language – while the Genetic Algorithm of [1] is implemented in `C`. Moreover, the two approaches are examined on different platforms. On the other hand, we should emphasize that the theoretical running time of our approach is linear; such a statement cannot be made for the Genetic Algorithm of [1], where the rate of convergence relies heavily on the quality of the initial population. Additionally, the main contribution of the proposed method is the problem formulation which, unlike previous approaches, allows for a lower bound to be obtained.

References

[1] F. Corno, P. Prinetto, M. Rebaudegno, and M. Sonza Reorda, “New static compacion techniques of test sequences for sequential circuits,” in *European Design and Test Conference*, 1997, pp. 37–43.

Circuit	Original Test Set		Compacted Test Set		Distance From Lower Bound	Proposed Method	GA [1] Method
	# Seq	# Vec	# Seq	# Vec		% Red	% Red
S208	44	739	10	292	1	39.51	39.27
S298	20	188	7	116	0	61.70	54.38*
S344	11	61	6	45	0	73.77	75.41
S349	16	84	9	63	0	75.00	76.19
S382	16	358	2	155	0	43.30	43.45
S386	58	258	31	162	0	62.79	62.79
S400	16	354	2	155	0	43.75	43.70
S420	52	786	10	274	0	34.86	34.90
S444	18	305	2	204	0	66.89	66.56*
S510	38	845	27	623	0	73.73	73.67*
S526	18	260	2	172	0	66.15	66.54
S526n	17	257	2	169	0	65.76	65.23*
S713	74	270	35	169	1	62.59	63.33
S820	121	1170	62	671	0	57.35	57.44
S832	112	1058	60	617	0	58.32	58.51
S838	52	671	12	310	1	46.20	45.93
S938	52	671	12	310	1	46.20	45.93
S953	111	825	38	404	0	48.97	48.97
S967	120	831	38	407	0	48.98	48.98
S991	50	83	25	46	0	55.42	55.42
S1196	189	509	110	339	2	66.60	66.60
S1238	191	513	110	334	2	65.11	64.72
S1269	67	255	26	136	0	53.33	61.18
S1423	50	282	16	187	0	66.31	66.43
S1488	24	69	16	56	0	81.16	81.16
S1494	60	523	43	418	0	79.92	81.02
S1512	60	282	14	117	0	41.49	41.70
S3271	61	1158	19	489	0	42.23	49.70
S3384	18	212	8	164	0	77.36	77.83
S4863	106	373	57	256	0	68.63	68.35*
S5378	95	250	50	153	1	61.20	60.80
S6669	68	466	23	259	0	55.58	55.58
S9234	7	19	2	9	0	47.37	52.63
S13207	15	97	6	57	0	58.76	59.79
S15850	15	39	4	14	0	35.90	38.46
S38417	281	806	14	131	0	16.25	16.38
S38584	48	509	30	435	0	85.46	85.46

Figure 2. Results for HITEC Test Sets

[2] J. Raik, A. Jutman, and R. Ubar, “Fast static compaction of tests composed of independent sequences: Basic properties and comparison of methods,” in *International Conference on Electronics, Circuits and System*, 2002, pp. 445–448.

[3] P. Raghavan and C. Thompson, “Randomized rounding: A technique for provably good algorithms and algorithmic proofs,” *Combinatorica*, vol. 7, no. 4, pp. 365–374, 1987.

[4] “ISCAS’89 benchmark circuits information,” Available from `http://www.cbl.ncsu.edu`.

[5] F. Corno, P. Prinetto, M. Rebaudegno, and M. Sonza-Reorda, “GATTO: A genetic algorithm for automatic test pattern generation for large synchronous sequential circuits,” *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 15, no. 8, pp. 991–1000, 1996.

[6] T. Niermann and J. H. Patel, “HITEC: A test generation package for sequential circuits,” in *European Conference on Design Automation*, 1992, pp. 214–218.

[7] “Test sequence tables used in [1],” Available from `http://www.cad.polito.it/tools`.