On the Derivation of a Minimum Test Set in High Quality Transition Testing

Tsuyoshi Iwagaki and Mineo Kaneko

Japan Advanced Institute of Science and Technology (JAIST), Ishikawa 923-1292, Japan

E-mail: {*iwagaki, mkaneko*}*@jaist.ac.jp*

Abstract

This paper discusses a test generation method to derive high quality transition tests for combinational circuits. It is known that, for a transition fault, a test set which propagates the errors (late transitions) to all the primary outputs reachable from the fault site can enhance the detectability of unmodeled defects. In this paper, to generate a minimum test set that meets the above property, the test generation problem is formulated as a problem of integer linear programming. The proposed formulation guarantees that minimum two-pattern tests for a transition fault are generated so that the errors will be observed at all the primary outputs reachable from the fault site. A case study using a benchmark circuit is presented to show the feasibility of the proposed method.

1 Introduction

The purpose of manufacturing test is to separate defective circuits from good ones. The behavior of a defect can be expressed by a fault model. To cope with various types of defects, several fault models such as the stuck-at fault model and the transition fault model are usually considered during test generation phases. When a target fault model is specified, test engineers try to generate tests with 100% fault coverage under the fault model. Obtained tests are then applied to actual circuits for defect screening. However, some defective circuits can pass the screening due to the presence of unmodeled defects even though the fault coverage of the applied tests is 100%. One way to avoid this undesirable situation is to develop a dedicated fault model for such defects. However, since it is costly to do so in general, several alternatives which assume conventional fault models have been discussed to enhance the detectability of unmodeled defects [1, 2, 3, 4].

Multiple-detection tests [1] have been shown to have an ability of detecting unmodeled defects. In order to clarify how effective multiple-detection tests are, some metrics were discussed in [2, 3, 4]. This paper focuses on the metric in [2]. In [2], the authors considered a test set for transition faults that propagates the errors (late transitions) of each transition fault to all the primary outputs reachable from the fault site, and showed it is effective in screening defective circuits compared to a conventional test set. To derive such a test set, some test generation procedures have been proposed in [5, 6]. The procedures in [5, 6] used a Boolean satisfiability technique with some heuristics and an existing test generation tool, respectively. Given a combinational circuit and a transition fault in the circuit, the following simple question can arise:

- What is the minimum number of two-pattern tests that detect the fault at all the primary outputs reachable from the fault site?

To the best of our knowledge, there has been no answer to this question yet. One goal of this paper is to give an answer to it. In this paper, we try to tackle this problem by using a technique of integer linear programming (ILP).

The rest of this paper is organized as follows. Section 2 gives the concept of test generation using ILP, then, in Section 3, an ILP formulation is presented to derive a minimum test set for a transition fault that meets the above property. Section 4 presents a case study to show the feasibility of our proposed method, and finally, Section 5 concludes the paper and describes our future work.

2 Preliminaries

Our test generation method is based on integer linear programming (ILP). In this section, we describe how to translate the test generation problem for a transition fault in a combinational circuit into an ILP problem.

2.1 Concept of ILP-based test generation

ILP-based test generation has first been presented for the stuck-at fault model [8]. Figure 1 represents the concept of ILP-based test generation. In this framework, given a combinational circuit and a fault, the circuit and the detection condition of the fault are first translated into the corresponding constraints that consist of inequalities and equalities with integer variables (especially 0-1 variables). Then, a feasible assignment to the variables that meets the constraints is obtained by an ILP solver. The assigned values of the variables that correspond to the circuit inputs form a test for the fault. If one wants to optimize some property during

978-1-4244-4206-5/09/$25.00 ©2009 IEEE

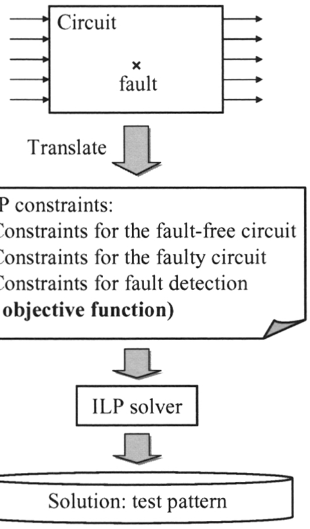

Figure 1: Concept of ILP-based test generation

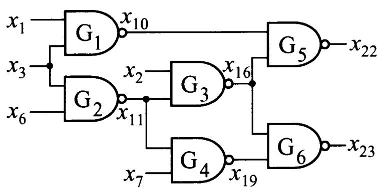

Figure 2: ISCAS '85 benchmark circuit c17

test generation, one can add it as an objective function to the ILP problem.

In the following, we explain how to translate the test generation problem for a transition fault in a combinational circuit into an ILP problem by using an example. More formal descriptions of ILP-based test generation can be found in [8, 9].

2.2 Transition Test Generation Using ILP

A two-pattern test for a transition fault, which is of the slow-to-rise type or slow-to-fall type, satisfies the following two conditions:

1. The first vector sets an appropriate value to the fault site.
2. The second vector detects the corresponding stuck-at fault.

Since there is no correlation between the first vector and the second vector, they can be considered separately during test generation. Before describing how to generate a two-pattern test for a transition fault, we first explain how to express the circuit behavior by using ILP constraints.

Table 1 shows inequalities in ILP constraints to express the behaviors of primitive gates with one or two inputs. In the first column of the table, y represents a gate output and each of x, x_1 and x_2 represents a gate input, where they can take '0' or '1.' A feasible assignment to the variables of the inequalities for a gate corresponds to the behavior of the gate. For example, a 2-input AND gate produces '0' if at least one input has '0.' This behavior corresponds to the first and second inequalities in Table 1. Indeed, if x_1 or x_2 takes '0,' y has to be '0' in those inequalities. Furthermore, if both inputs take '1,' the AND gate produces '1.' This behavior is expressed by the last inequality in the table. In this way, each gate in a combination circuit can be interpreted as inequalities in ILP constraints. Given a combinational circuit, we can obtain ILP constraints for the whole circuit by replacing each gate with its corresponding inequalities repeatedly. Now, let us consider the circuit shown in Figure 2. For example, we can obtain the following constraints for c17:

G_1: $x_1 + x_{10} \geq 1, x_3 + x_{10} \geq 1, -x_1 - x_3 - x_{10} \geq -2$,
G_2: $x_3 + x_{11} \geq 1, x_6 + x_{11} \geq 1, -x_3 - x_6 - x_{11} \geq -2$,
G_3: $x_2 + x_{16} \geq 1, x_{11} + x_{16} \geq 1, -x_2 - x_{11} - x_{16} \geq -2$,
G_4: $x_{11} + x_{19} \geq 1, x_7 + x_{19} \geq 1, -x_{11} - x_7 - x_{19} \geq -2$,
G_5: $x_{10} + x_{22} \geq 1, x_{16} + x_{22} \geq 1, -x_{10} - x_{16} - x_{22} \geq -2$,
G_6: $x_{16} + x_{23} \geq 1, x_{19} + x_{23} \geq 1, -x_{16} - x_{19} - x_{23} \geq -2$.

Any feasible assignment for these constraints simulates the behavior of c17. In Figure 2, when we have $x_1 = 1$, $x_2 = 1$, $x_3 = 0$, $x_6 = 1$ and $x_7 = 1$, the circuit behaves as follows: $x_{10} = 1$, $x_{11} = 1$, $x_{16} = 0$, $x_{19} = 0$, $x_{22} = 1$ and $x_{23} = 1$. These values satisfy the above constraints, and *vice versa*.

Given a combinational circuit C and a transition fault f in C, the following procedure is performed to generate a two-pattern test in this paper.

1. Extract the fanin cone C^{g1} reachable to f and the fanout cone C^{f} reachable from f, from C.
2. Copy C as C^{g2}.
3. Translate C^{g1}, C^{g2} and C^{f} into the corresponding ILP constraints, and create additional constraints to express the connection between C^{g2} and C^{f}.
4. Create the constraints for detecting f.
5. Apply an ILP solver to the above constraints.

Here, we consider Figure 2 and the slow-to-rise transition fault on x_{11}. To generate a two-pattern test for the fault, we first perform steps 1 and 2 of the above procedure. Figure 3 shows the obtained three circuits. Figure 3(a) represents the fault-free version of the original circuit associated with x_{11}. This fault-free circuit is used to generate the first vector of a two-pattern test, and the behavior of it is expressed by the following constraint:

G_1^{g1}: $x_3^{g1} + x_{11}^{g1} \geq 1, x_6^{g1} + x_{11}^{g1} \geq 1, -x_3^{g1} - x_6^{g1} - x_{11}^{g1} \geq -2$.

Table 1: Inequalities in ILP constraints expressing the behaviors of primitive gates

Gate types	Inequalities
$y = \mathrm{AND}(x_1,x_2)$	$x_1 - y \geq 0, x_2 - y \geq 0, -x_1 - x_2 + y \geq -1$
$y = \mathrm{NAND}(x_1,x_2)$	$x_1 + y \geq 1, x_2 + y \geq 1, -x_1 - x_2 - y \geq -2$
$y = \mathrm{OR}(x_1,x_2)$	$-x_1 + y \geq 0, -x_2 + y \geq 0, x_1 + x_2 - y \geq 0$
$y = \mathrm{NOR}(x_1,x_2)$	$-x_1 - y \geq -1, -x_2 - y \geq -1, x_1 + x_2 + y \geq 1$
$y = \mathrm{XOR}(x_1,x_2)$	$x_1 - x_2 + y \geq 0, -x_1 + x_2 + y \geq 0, x_1 + x_2 - y \geq 0, -x_1 - x_2 - y \geq -2$
$y = \mathrm{XNOR}(x_1,x_2)$	$x_1 - x_2 - y \geq -1, -x_1 + x_2 - y \geq -1, x_1 + x_2 + y \geq 1, -x_1 - x_2 + y \geq -1$
$y = \mathrm{NOT}(x)$	$x + y \geq 1, -x - y \geq -1$
$y = \mathrm{BUFFER}(x)$	$x - y \geq 0, -x + y \geq 0$

Figure 3(b) represents the fault-free version of the original circuit. This fault-free circuit is used together with the circuit of Figure 3(c) in order to generate the second vector of a two-pattern test, and the behavior of it is expressed by the following constraints.

G_1^{g2}: $x_1^{g2} + x_{10}^{g2} \geq 1, x_3^{g2} + x_{10}^{g2} \geq 1, -x_1^{g2} - x_3^{g2} - x_{10}^{g2} \geq -2,$
G_2^{g2}: $x_3^{g2} + x_{11}^{g2} \geq 1, x_6^{g2} + x_{11}^{g2} \geq 1, -x_3^{g2} - x_6^{g2} - x_{11}^{g2} \geq -2,$
G_3^{g2}: $x_2^{g2} + x_{16}^{g2} \geq 1, x_{11}^{g2} + x_{16}^{g2} \geq 1, -x_2^{g2} - x_{11}^{g2} - x_{16}^{g2} \geq -2,$
G_4^{g2}: $x_{11}^{g2} + x_{19}^{g2} \geq 1, x_7^{g2} + x_{19}^{g2} \geq 1, -x_{11}^{g2} - x_7^{g2} - x_{19}^{g2} \geq -2,$
G_5^{g2}: $x_{10}^{g2} + x_{22}^{g2} \geq 1, x_{16}^{g2} + x_{22}^{g2} \geq 1, -x_{10}^{g2} - x_{16}^{g2} - x_{22}^{g2} \geq -2,$
G_6^{g2}: $x_{16}^{g2} + x_{23}^{g2} \geq 1, x_{19}^{g2} + x_{23}^{g2} \geq 1, -x_{16}^{g2} - x_{19}^{g2} - x_{23}^{g2} \geq -2.$

Figure 3(c) represents the faulty version of the original circuit associated with x_{11}. This faulty circuit is used to generate the second vector of a two-pattern test, and the behavior of it is expressed by the following constraints.

G_3^{f}: $x_2^{f} + x_{16}^{f} \geq 1, x_{11}^{f} + x_{16}^{f} \geq 1, -x_2^{f} - x_{11}^{f} - x_{16}^{f} \geq -2,$
G_4^{f}: $x_{11}^{f} + x_{19}^{f} \geq 1, x_7^{f} + x_{19}^{f} \geq 1, -x_{11}^{f} - x_7^{f} - x_{19}^{f} \geq -2,$
G_5^{f}: $x_{10}^{f} + x_{22}^{f} \geq 1, x_{16}^{f} + x_{22}^{f} \geq 1, -x_{10}^{f} - x_{16}^{f} - x_{22}^{f} \geq -2,$
G_6^{f}: $x_{16}^{f} + x_{23}^{f} \geq 1, x_{19}^{f} + x_{23}^{f} \geq 1, -x_{16}^{f} - x_{19}^{f} - x_{23}^{f} \geq -2.$

In Figure 3(c), since we can assume that x_{11}^{f} has a stuck-at 0 fault, and that x_2^{f}, x_7^{f} and x_{10}^{f} have the same values of the corresponding signals of Figure 3(b), we must have the following constraints:

$x_{11}^{f} = 0,$
$x_2^{g2} - x_2^{f} = 0,$
$x_7^{g2} - x_7^{f} = 0,$
$x_{10}^{g2} - x_{10}^{f} = 0.$

Now, we consider the detection conditions for the slow-to-rise transition fault on x_{11}. According to the first detection condition mentioned before, x_{11} must be set to '0' under the first vector of a two-pattern test. Hence, the following constraint is required:

$x_{11}^{g1} = 0.$

Moreover, according to the seconds detection condition, in order to detect the corresponding stuck-at fault, we need to differentiate the fault-free circuit from the faulty one. To translate this condition into ILP constraints, we introduce variables e_{22}, e_{23} with the following constraints:

$x_{22}^{g2} - x_{22}^{f} + e_{22} \geq 0, -x_{22}^{g2} + x_{22}^{f} + e_{22} \geq 0,$

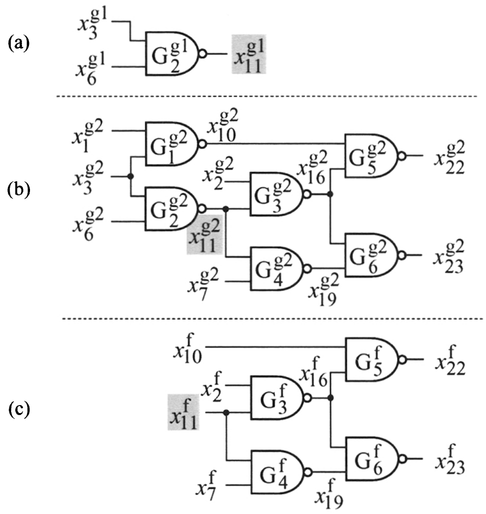

Figure 3: Three circuits for fault detection: (a) Fault-free circuit for generating the first vector of a two-pattern test; (b) Fault-free circuit for generating the second vector of a two-pattern test; and (c) Faulty circuit for generating the second vector of a two-pattern test

$x_{22}^{g2} + x_{22}^{f} - e_{22} \geq 0, -x_{22}^{g2} - x_{22}^{f} - e_{22} \geq -2;$
$x_{23}^{g2} - x_{23}^{f} + e_{23} \geq 0, -x_{23}^{g2} + x_{23}^{f} + e_{23} \geq 0,$
$x_{23}^{g2} + x_{23}^{f} - e_{23} \geq 0, -x_{23}^{g2} - x_{23}^{f} - e_{23} \geq -2.$

Each of e_{22} and e_{23} takes '1' if and only if the corresponding primary outputs of the fault-free circuit and faulty circuit take different values.

Finally, since the error must be propagated to at least one primary output, we have the following constraint:

$e_{22} + e_{23} \geq 1.$

In this way, a two-pattern test can be generated by applying any ILP solver to all the above constraints.

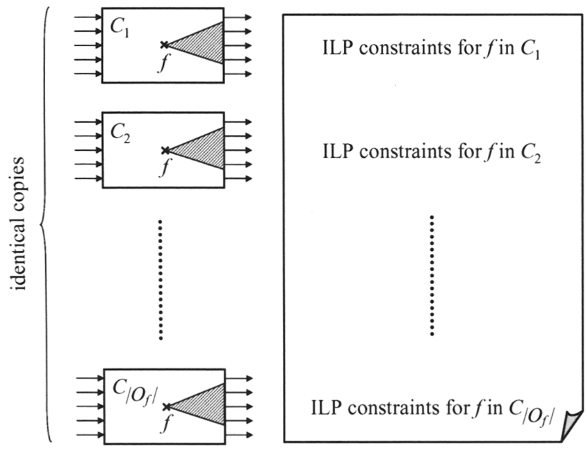

Figure 4: ILP model for generating a minimum test set

3 Proposed Method

3.1 Our Test Generation Problem

We formally state our test generation problem as follows.

Input: A combinational circuit C and a transition fault f in C

Output: A two-pattern test set T_f that propagates the errors caused by f to its all reachable primary outputs

Objective: Minimizing $|T_f|$

To solve this problem, we derive the following formulation.

3.2 ILP Formulation

The upper bound of $|T_f|$ is $|O_f|$, where O_f represents all the primary outputs reachable from f, because one test is enough to propagate the error of f to each reachable primary output. We make use of this upper bound to formulate an ILP problem. Here, we prepare $|O_f|$ copies of the given circuit, and associate ILP constraints to detect f with each copy (Figure 4). This implies that, for f, $|O_f|$ vector pairs can be generated simultaneously. Notice that the constraint for the first vector of a two-pattern test can commonly be used by all the copies. If we identify useless copies of them as much as possible, we will finally obtain a minimum test set for f. To achieve this, we consider additional constraints in the following. Note that the following additional constraints are used instead of the constraint for fault detection, i.e., the last constraint of the previous section.

For each i ($1 \leq i \leq |O_f|$) and j ($1 \leq j \leq |O_f|$), we introduce a 0-1 variable $e_{i,j}$. Variable $e_{i,j}$ takes '1' if the error of f is propagated to the j-th primary output in C_i, otherwise it takes '0.' In general, there is a redundant primary output at which the error of f never reaches for any vector pair. For such a primary output, we prepare a 0-1 variables r_j for each j. Equation $r_j = 1$ indicates the error of f does not reach at the j-th primary output of any copy of the circuit, and $r_j = 0$ indicates the error of f reaches at the j-th primary output of at least one copy. By using this variable, we have the following constraints for each j.

$$\sum_{i=1}^{|O_f|} e_{i,j} + r_j \geq 1 \tag{1}$$

This means that the error of f must be propagated to the j-th primary output of at least one copy, or the j-th primary output of every copy must be redundant. Since $e_{i,j} = r_j = 1$ never happen for all i, j, we also have the following constraints.

$$e_{i,j} + r_j \leq 1 \tag{2}$$

Now, we introduce a variable u_i for each i to identify copies of the circuit that are mandatory. Variable u_i takes '1' if the error of f is propagated to at least one primary input of the i-th copy, otherwise it takes '0.' This state can be expressed by the following constraint.

$$-e_{i,j} + u_i \geq 0 \tag{3}$$

Since at least one u_i has to take '1' if f is testable, i.e., a test is generated in at least one copy, we also have the following constraint.

$$\sum_{i=1}^{|O_f|} u_i \geq 1 \tag{4}$$

Finally, we have to minimize the following equation for test minimization.

$$\sum_{i=1}^{|O_f|} u_i + |O_f| \cdot \sum_{j=1}^{|O_f|} r_j \tag{5}$$

The first term counts the number of copies that are used for propagating the errors to all the reachable primary outputs. From inequality (1), it can be seen that r_j can be set to '1' freely. To prevent r_j from being '1' freely, the term is multiplied by $|O_f|$ in the second term of the above equation. Therefore, after running an ILP solver, r_j will take '1' if and only if the error of f never reaches at the j-th primary output of any copy, i.e., the j-th primary output of the circuit is redundant.

By using the values assigned to the primary inputs of copies whose u_i take '1,' we can form a minimum test set for f.

Table 2: Values of $e_{i,j}$

	$j=1$	$j=2$	$j=3$	$j=4$	$j=5$
$i=1$	0	1	0	0	0
$i=2$	0	1	0	0	1
$i=3$	0	0	0	0	0
$i=4$	1	0	0	0	1
$i=5$	0	1	1	0	0

Table 3: Values of u_i

$i=1$	1
$i=2$	1
$i=3$	0
$i=4$	1
$i=5$	1

3.3 Example

To clarify our ILP formulation, we give an example here. We use a combinational circuit C with five primary outputs as an example circuit. To generate a minimum test set for a fault f in C, five copies $C_1, C_2, \ldots, C_5$ of C need to be prepared. Now, let us consider a situation where ILP constraints for the test generation were provided for an ILP solver, and, during solving the ILP problem, the temporary feasible assignment shown in Tables 2–4 was obtained.

Table 2 represents that the errors of f reach at the 2nd primary output of C_1, at the 2nd and 5th primary outputs of C_2, at the 1st and 5th primary outputs of C_4, and at the 2nd and 3rd primary outputs of C_5, respectively. Note that, in C_3, no test is generated Since, in any of C_1, C_2, C_4 and C_5, the error appears at least one primary output, each u_i except u_3 takes '1' as shown in Table 3. Notice that it is possible for u_3 to take '1' because it also satisfies inequality (3). However, in the final solution after solving the ILP problem, such an assignment will be rejected. Table 4 shows that the 4th primary output of any copies has no error.

3.4 Sizes of variables and constraints

Here, we estimate the sizes of variables and constraints in our test generation problem. Let n be the number of signal lines in a combinational circuit. It is enough to prepare $2n$ variables for fault detection (Figure 3). As mentioned in Section 3.2, since $|O_f|$ copies of the original circuit are produced, totally $2n \cdot |O_f|$ variables are required for fault detection. Since the additional variables of r_j and u_i, where $1 \leq i \leq |O_f|$ and $1 \leq j \leq |O_f|$, are used to derive a minimum test set, totally $2|O_f|$ variables are also needed. Thus, we need to prepare at most $2n \cdot |O_f| + 2|O_f|$ variables. The number of constraints for fault detection and for test set minimization can roughly be estimated as $O(n \cdot |O_f|)$ and $O(|O_f|^2)$, respectively.

Table 4: Values of r_j

$j=1$	$j=2$	$j=3$	$j=4$	$j=5$
0	0	0	1	0

4 Case study

To show the feasibility of our proposed method, we performed a case study using one ISCAS '85 benchmark circuit (c2670). Our case study was done on a Linux workstation (CPU: AMD Opteron 250 2.4 GHz ×2, Memory: 8 GB), and CPLEX (version 11.01) from ILOG and Galena from [10] were used as ILP solvers. In the case study, several slow-to-rise faults in the circuit were chosen as target faults, and, for each fault, its ILP model was obtained by using a Perl program.

Table 5 shows the test generation results for the faults. Columns "Signal name" and "#reachable" represent the signal name of each fault site and the number of primary outputs reachable from the fault site, respectively. Columns "#variables" and "#constraints" list the number of variables and constraints in the ILP model for each fault, respectively. Columns "#tests," "#unobservable" and "CPU time" give the number of two-pattern tests generated by CPLEX or Galena, the number of redundant primary outputs reachable from the fault site and computation time including model construction time, respectively. From the results, the following remarks can be made:

- If the error of a fault can be propagated to all the reachable primary outputs with one test, its computation time can be short, otherwise its computation time can increase.
- The presence of redundant primary outputs for a fault can make the computation time large.

The results also show that CPLEX did not work well for almost all instances. From this point of view, our ILP problems seem to be hard. However, Galena solved them successfully. This is because Galena is tuned specifically for 0-1 ILP problems where all variables take '0' or '1.' It is conceivable that our method is applicable for larger instances if we use a tuned 0-1 ILP solver.

In the future, we should verify the above remarks for various benchmark circuits. If the remarks are true, we can use those facts to improve our ILP model. For example, if we identify redundant primary outputs by using a preprocessing technique, we can remove the variables and constraints for them in our ILP model. Furthermore, this can also reduce the number of duplicated circuit copies used in our ILP model.

5 Conclusions and Future Work

In this paper, we presented an integer programming formulation to generate high quality transition tests for com-

Table 5: Test generation results for slow-to-rise faults in c2670

Signal name	#reachable	#variables	#constraints	#tests		#unobservable		CPU time [s]	
				CPLEX	Galena	CPLEX	Galena	CPLEX	Galena
"139"	5	7,171	17,292	1	1	0	0	8.00	1.00
"100"	6	8,551	20,624	1	1	0	0	9.78	0.58
"104"	7	10,606	25,545	1†	1	1†	1	> 3,600	2.49
"82"	8	11,185	27,098	1	1	0	0	4.89	0.69
"246"	10	14,111	34,232	1	1	2	2	63.18	1.55
"1068"	11	16,748	40,413	1†	1	1†	1	> 3,600	7.03
"78"	12	18,613	44,822	2†	2	1†	1	> 3,600	7.32
"1065"	13	20,273	48,760	2†	2	1†	1	> 3,600	5.91
"92"	15	23,851	57,482	3†	3	1†	1	> 3,600	32.57
"1075"	18	32,457	78,576	2†	2	2†	1	> 3,600	74.02
"1042"	21	36,690	89,088	2†	1	2†	1	> 3,600	74.23
"227"	28	58,017	142,074	—‡	2	—‡	1	> 3,600	195.73

† Temporary solution within 3,600 seconds
‡ No feasible solution within 3,600 seconds

binational circuits. When a combinational circuit and a transition fault in the circuit are given, our method always generates a minimum test set that propagates the errors of the fault to all the primary outputs reachable from the fault site. In addition to theoretical interests, we believe that our discussion can be useful if one investigates a new heuristic technique for test minimization or evaluates existing heuristic techniques such as [5, 6].

In the future, we should evaluate the proposed method for various benchmark circuits, and should consider improving our ILP model and adopting heuristic techniques. Moreover, from a practical point of view, it should be important to discuss minimizing tests for not one but all faults in a circuit in our future work.

Acknowledgments

The authors would like to thank the reviewers of this paper for their helpful comments. This work was supported in part by the research promoting expenses for assistant professors of JAIST.

References

[1] S. C. Ma, P. Franco and E. J. McCluskey, "An experimental chip to evaluate test techniques: experimental results," *Proc. International Test Conference*, pp. 663–672, 1995.

[2] C.-W. Tseng and E. J. McCluskey, "Multiple-output propagation transition fault test," *Proc. International Test Conference*, pp. 358–366, 2001.

[3] B. Benware, C. Schuermyer, N. Tamarapalli, K.-H. Tsai, S. Ranganathan, R. Madge, J. Rajski and P. Krishnamurthy, "Impact of multiple-detect test patterns on product quality," *Proc. International Test Conference*, pp. 1031–1040, 2003.

[4] H. Tang, G. Chen, S. M. Reddy, C. Wang, J. Rajski and I. Pomeranz, "Defect aware test patterns," *Proc. Design, Automation and Test in Europe*, pp. 450–455, 2005.

[5] B. Vaidya and M. B. Tahoori, "Delay testing based on transition faults propagated to all reachable outputs," *Proc. International Workshop on Defect Based Testing*, pp. 67–75 2004.

[6] I. Park, A. Al-Yamani and E. J. McCluskey, "Effective TARO pattern generation," *Proc. VLSI Test Symposium*, pp. 161–166 2005.

[7] N. K. Jha and S. Gupta, *Testing of digital systems*, Cambridge University Press, 2003.

[8] J. P. M. Silva, "Integer programming models for optimization problems in test generation," *Proc. Asia and South Pacific Design Automation Conference*, pp. 481–487, 1998.

[9] P. F. Flores, H. C. Neto and J. P. M. Silva, "An exact solution to the minimum size test pattern problem," *ACM Transactions on Design Automation of Electronic Systems*, Vol. 6, Issue 4, pp. 629–644, Oct. 2001.

[10] D. Chai and A. Kuehlmann, "A fast pseudo-boolean constraint solver," *Design Automation Conference*, pp. 830–835, 2003.