確率的手法による組合せ回路のテスト・パターンの生成†

加納 弘††

組合せ回路の機能的な検査を行うために必要なテスト・パターンの系列を生成するための手法を紹介する．テスト・パターンの生成に関しては，乱数による方法，パス活性化法およびDアルゴリズムなどが提案されているが，これらと競合する確率的手法を提案し，実験を試みた．

論理シミュレーションを 0~1 の実数を利用して実行する．入力端子には初期値として0と1との間の中間値を与えてシミュレーションを行い，仮定した縮退故障回路の出力端子における出力値と正常な回路の出力端子における出力値との差を大きくするように，入力端子における値を逐次決めて行く．

これは，故障を検出する可能性（確率に近いと期待される）を高くするように，入力端子における値を選んで行くことに相当する．

参考文献の例題についての実験の結果は良好で，Dアルゴリズムに近い水準の効果を，簡単なプログラムと少ない計算機時間で実現することが期待される．

順序回路への適用も可能と考えられるが，これは今後の課題である．

1. はじめに

多数の IC を搭載した IC カードや論理 LSI の良否を判別するための論理機能の検査は論理装置を製造する上で必要不可欠な工程である．本論文では組合せ回路の直流的な論理機能を検査する場合に使用するテスト・パターンの系列を生成するための一つの手法について述べ，次いでその手法の適用結果について述べる．

故障は論理ゲートの入出力端子の単一の0または1縮退故障に限るものとする．

2. 擬似確率の概念

一般に論理回路は論理ゲートから構成されるが，これらの論理ゲートの入力は0または1のいずれかである．このように考えて論理回路をシミュレートすれば，2値論理シミュレーションである．

複数の入力端子と出力端子を持つ論理回路から，この論理回路に対応する一つのモデルを構成する．ここでは，入力端子，出力端子および各ゲートに入力端子，出力端子および各ゲートモデルが対応する．

このモデルにおいては，各ゲートモデルの入力値はすべて0と1との間の実数（0と1も含む）であり，出力値も実数であるような関数関係が定義される．

3. 擬似確率の計算方法

図1に代表的なゲートモデルの例を示し，これらについて入力端子に擬似確率 A および B ($0 \leqq A, B \leqq 1$) を与えた場合に出力端子に得られる擬似確率 Z の計算方法を示した．容易に分るように，$0 \leqq Z \leqq 1$ である．A と B に0または1を与えると Z も0または1であり，これは通常の論理演算と同じである．

図2は論理回路モデルの例を示し，これを構成する各ゲートモデルの入出力端子における擬似確率の関係式を示した．この例では，3つの関係式がすべて成り立つ．

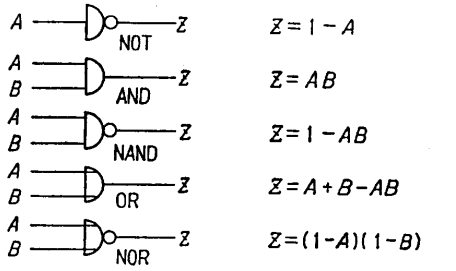

図1 代表的ゲートモデルと擬似確率の計算方法
Fig. 1 Typical gate models and calculation of pseudo probability.

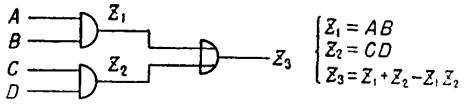

図2 論理回路モデルと擬似確率の計算方法
Fig. 2 Logic circuit and calculation [of pseudo probability.

† Test Pattern Generation for Combinatorial Logic Circuits by Probabilistic Method by HIROSHI KANO (Computer Development Laboratories).

†† (株)コンピュータ総合研究所

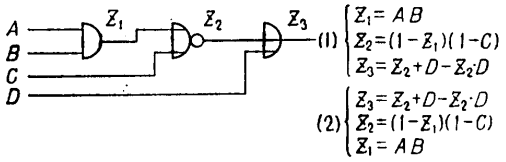

図 3 論理回路と計算順序
Fig. 3 Logic circuit and order of calculations.

一つの論理回路においては，すべての論理素子をANDやORに分解し，これらから構成されるものとする．一つの論理回路に対して擬似確率を計算する等式は論理ゲートの数だけ存在する．

組み合せ回路で，かつ計算式が擬似確率の伝播に従って順序づけられて並べられている場合には一度の計算で擬似確率が求められるが，もし丁度逆順に並べられている場合には論理の段数の数だけ全体の計算を繰り返す必要がある．擬似確率の一般的な計算方法は収束するまで繰り返し計算することであるが，入力端子からのレベル順に計算式を並べておく（レベルソート）のが計算時間の短縮のために有効である．

図3は一つの論理回路についての擬似確率の計算順序の異なる方法を与えた．(1)の順序では1回の計算で擬似確率が得られるが，(2)の方法では3回繰り返して計算して始めて擬似確率が求められる．

一つの論理回路に対して入力端子に擬似確率が与えられている場合に，各ゲートの出力端子における擬似確率を計算するために各内部ゲートに初期値0.5を与えてから計算を開始する．

図3の例で A, B, C, D がすべて“0”とし，(2)の順序で繰り返し計算をしてみる．

$$Z_3=0.5+0-0.5\cdot 0=0.5 \quad (1)$$
$$Z_2=(1-0.5)\cdot(1-0)=0.5 \quad (2)$$
$$Z_1=0\cdot 0=0 \quad (3)$$
$$Z_3=0.5+0-0.5\cdot 0=0.5 \quad (4)$$
$$Z_2=(1-0)(1-0)=1 \quad (5)$$
$$Z_1=0\cdot 0=0 \quad (6)$$
$$Z_3=1+0-1\cdot 0=1 \quad (7)$$
$$Z_2=(1-0)(1-0)=1 \quad (8)$$
$$Z_1=0\cdot 0=0 \quad (9)$$

全体の計算を3回繰り返すと擬似確率がすべて収束する．

論理ゲートに対応してゲートモデルといってきたが以下では単に論理ゲートまたはゲートと呼ぶことにする．

4. 組合せ回路におけるテスト・パターンの生成

擬似確率の概念と一つの計算方法について説明したが，次にこの方法の応用として組合せ回路におけるテスト・パターンの生成の方法を説明する．

以下の方法は J. P. Roth の考案した D-アルゴリズムと同じ目的を全く別の手法で達成しようとするものである．この手法を K-手法と名づける．

問題を次の形で把えてみる．

一つの組合せ回路が与えられている．入力は $I_1, I_2, \cdots, I_n$ であり，出力は $Z_1, Z_2, \cdots, Z_m$ であり，内部ゲートは $G_1, G_2, \cdots, G_l$ とする．ゲート G_k の出力端子の“0”縮退故障を検出するようなテスト・パターンを求める．1縮退の場合は(12)式の代りに

$$S=(1-G_k)\sum_{j=1}^{m}|Z_{kj}(1, I_1, \cdots, I_n) - Z_{kj}(0, I_1, \cdots, I_n)|$$

を用いる．

$Z_1, Z_2, \cdots, Z_m$ については

$$Z_j=Z_j(I_1, I_2, \cdots, I_n) \quad 1\leqq j\leqq m \qquad (10)$$

であるが，特に G_k を強制的に0または1と置いた場合の故障回路における $Z_1, Z_2, \cdots, Z_m$ を次のように表現する．

$$Z_{kj}=Z_{kj}(G_k, I_1, I_2, \cdots, I_n) \qquad (11)$$

ここに G_k は0または1である．

以上の関数を利用して

$$S=G_k(I_1, I_2, \cdots, I_n)\cdot\sum_{j=1}^{m}|Z_{kj}(1, I_1, \cdots, I_n) - Z_{kj}(0, I_1, \cdots, I_n)| \qquad (12)$$

とおく．ここに S は $S(k, I_1, I_2, \cdots, I_n)$ である．

式(12)の意味を考えてみる．ゲート G_k の“0”縮退故障を検出するためには検査パターンは次の条件を満たすものでなければならない．

(1) ゲート G_k の“0”縮退故障を検出するのであるから，$G_k=1$ となるようなパターンでなければならない．

(2) しかもそのパターンにおいて，G_k を強制的に“0”とした時（故障時）の出力と G_k が“1”の時(正常時)の出力が異なっていること，すなわち，故障の状態が出力に伝搬される必要がある．

このためには $|Z_{kj}(1, I_1, \cdots, I_n)-Z_{kj}(0, I_1, \cdots, I_n)|$ が少なくも一つの j について1以上になる入力パターンでなければならない．従って S が1以上となるパター

ンを求める．このSを利用してテスト・パターンを生成しよう．

入力パターンの候補を考える上で，Sが大きい程，求めるテスト・パターンに近づいていると考えて，以下の手法を考案した．

その求め方は以下の通りである．

（1） $I_1, I_2, \cdots, I_n$ に初期値をセットする

まず $I_1, I_2, \cdots, I_n$ をすべて0.5にする．0と1との中間値である．

（2） I_1 を求める

$S(k, 0, 0.5, \cdots, 0.5)$ と $S(k, 1, 0.5, \cdots, 0.5)$ を比較してより大きなSを与える I_1 を選ぶ．すなわち前者が大きければ $I_1=0$ とおき，後者が大きければ $I_1=1$ とおく．Sが同じであれば I_1 は0または1のいずれでもよいが，ここでは0とおく．

（3） I_i を求める

$I_1, \cdots, I_{i-1}$ まで求まった場合の I_i の求め方を説明する．これらのIを $\check{I}_1, \check{I}_2, \cdots, \check{I}_{i-1}$ とする．

$S(k, \check{I}_1, \cdots, \check{I}_{i-1}, 0, 0.5, \cdots, 0.5)$ と $S(k, \check{I}_1, \cdots, \check{I}_{i-1}, 1, 0.5, \cdots, 0.5)$ とを比較してより大きなSを与えるような I_i を選ぶ．前者が大きければ $\check{I}_i=0$ であり，後者が大きければ $\check{I}_i=1$ である．

（4） $\check{I}_1, \check{I}_2, \cdots, \check{I}_n$ を検定する

$I_1, I_2, \cdots, I_n$ がすべて求まったところで $S(k, \check{I}_1, \cdots, \check{I}_n)$ を調べる．このSが0でなければ $\check{I}_1, \cdots, \check{I}_n$ は所要の条件を満足するが，もし0であればこれは所要の条件を満たさないことになる．この場合の入力パターンの求め方については後述する．

以上で述べた手法（K-手法）を簡単な例に従って説明する．

図4に2ゲートから成る組合せ回路の例を示した．G_1 はANDゲートであり，G_2 はORゲートである．入力は I_1, I_2, I_3 の3入力であり，出力は Z_1（これは G_2 の出力である）のみである．仮定した故障は G_1 の出力の0縮退故障である．この G_1 の0縮退故障を検出するような入力パターンを前述の方法に従って求めてみる．

（1） I_1, I_2, I_3 に初期値をセットする

すなわち $I_1=I_2=I_3=0.5$ とおく．

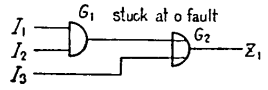

図4 簡単な回路の適用例
Fig. 4 Example of simple circuit.

（2） I_1 を求める

$G_k(I_1, \cdots, I_n)$ について求める．ここに $k=1$ である．（故障ゲートは G_1 である）

$$G_1(I_1, I_2, I_3)=I_1 \cdot I_2 \tag{13}$$

次に Z_{kj} を求める．出力は Z_1 のみであるので，Z_{11} を求めることになる．

$$Z_{11}=Z_1(G_1, I_1, I_2, I_3)$$
$$=G_1+I_3-G_1 \cdot I_3 \tag{14}$$

この Z_{11} は入力には I_1, I_2, I_3 を与えるのであるが，G_1 の出力だけは正常回路の出力とせず，強制的に G_1 という値（ここでは0または1を与えるのであるが）を与えた場合の出力 Z_1 を求めたのである．

次にSを求める．

$$S(1, I_1, I_2, I_3)$$
$$=Z_1(I_1, I_2, I_3)$$
$$\cdot \sum_{j=1}^{1} |Z_{1j}(1, I_1, I_2, I_3)-Z_{1j}(0, I_1, I_2, I_3)|$$
$$=I_1 \cdot I_2 \cdot |1-I_3| \tag{15}$$

$S(1, 0, 0.5, 0.5)$ と $S(1, 1, 0.5, 0.5)$ を比較する．

$$S(1, 0, 0.5, 0.5)=0 \tag{16}$$
$$S(1, 1, 0.5, 0.5)=0.25 \tag{17}$$
$$S(1, 0, 0.5, 0.5)<S(1, 1, 0.5, 0.5) \tag{18}$$

従って $\check{I}_1=1$ が求まる．

（3） I_2 を求める

$\check{I}_1=1$ が求まったとして I_2 を求めるためにSを計算する．

$$S(1, 1, 0, 0.5)=0 \tag{19}$$
$$S(1, 1, 1, 0.5)=0.5 \tag{20}$$
$$S(1, 1, 0, 0.5)<S(1, 1, 1, 0.5) \tag{21}$$

従って $\check{I}_2=1$ が求まる．

（4） I_3 を求める

$\check{I}_1=1$, $\check{I}_2=1$ が求まったとして I_3 を求める．

$$S(1, 1, 1, 0)=1 \tag{22}$$
$$S(1, 1, 1, 1)=0 \tag{23}$$
$$S(1, 1, 1, 0)>S(1, 1, 1, 1) \tag{24}$$

従って $\check{I}_3=0$ が求まる．

以上で $\check{I}_1=1$, $\check{I}_2=1$, $\check{I}_3=0$ が求まった．

（5） $\check{I}_1, \check{I}_2, \check{I}_3$ の検定

$S(1, 1, 1, 0)=1$ であるので，この $\check{I}_1, \check{I}_2, \check{I}_3$ すなわち 1, 1, 0 は求めるテスト・パターンである．

5. 組合せ回路についての適用結果

5.1 例 1

図5は参考文献1）の34頁の図3.4と同じ回路であ

表 1 例 1 の適用結果
Table 1 Applications for example 1.

故障ゲート番号	0,1 縮退の区分	I_1	I_2	I_3	I_4	I_5
1	0	1	1	1	1	0
	1	0	1	1	1	0
2	0	0	1	0	1	0
	1	0	1	1	1	0
3	0	1	1	0	1	1
	1	0	0	0	1	1
4	0	0	1	1	1	0
	1	1	1	1	1	0
5	0	1	1	0	1	0
	1	1	1	1	1	0
6	0	1	1	0	1	1
	1	1	1	1	1	0
7	0	0	1	1	1	1
	1	1	1	1	1	0
8	0	1	1	0	1	1
	1	0	0	1	1	1

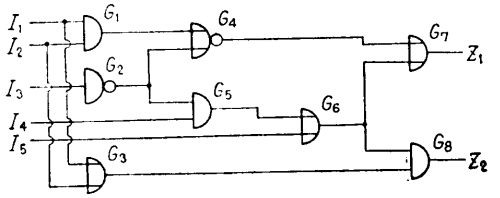

図 5 例 1 の回路
Fig. 5 Logic circuit of example 1.

る．この回路のゲート $G_1, G_2, \cdots, G_8$ の各出力について 0 縮退故障と 1 縮退故障について適用して得られた結果を**表 1** に示す．

この例では得られたテスト・パターンの検定結果はすべて必要な条件を満たしている．

5.2 例 2 Schneider の反例

図 6 は例 1 と同じ文献 1) の 36 頁の図 3.5 と同じ回路である．この回路に適用した結果を**表 2** に示す．

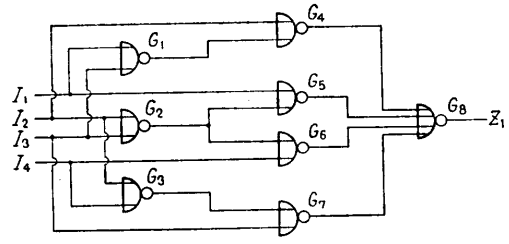

図 6 Schneider の反例
Fig. 6 Schneider's example.

表 2 Schneider の反例の適用結果
Table 2 Applications for Schneider's example.

故障ゲート番号	0,1 縮退の区分	I_1	I_2	I_3	I_4	Remark
1	0	0	0	0	0	
	1	1	0	1	1	
2	0	0	0	0	0	
	1	0	1	1	1	
3	0	1	0	0	0	
	1	1	1	0	1	
4	0	1	0	1	1	
	1	1	1	1	1	
5	0	0	1	1	1	
	1	1	1	1	1	
6	0	1	1	1	0	
	1	1	1	1	1	
7	0	1	1	0	1	
	1	1	1	1	1	
8	0	1	1	1	1	
	1	0	1	1	1	

Remark: Verification of this test pattern was not satisfactory, Test pattern (0,0,0,0) was generated by Re-Try.

この例では前述の例 1 と異なって，ゲート番号 3 の G_3 の 0 縮退故障を検出すべきテスト・パターンを検定した結果 $S(k, \bar{I}_1, \bar{I}_2, \bar{I}_3, I_4)=S(3,1,0,0,0)=0$ であるので検定条件を満たさない．

得られたテスト・パターンを検定した結果，検定条件を満たさない場合は，次章に述べる方法で再度テスト・パターンを求める．この時の結果についても後述する．

5.3 例 3 多重出力の論理回路

図 7 は参考文献 2) の 357 頁の図 7.7 と同じである．この回路に適用した結果を**表 3** に示す．

この例ではゲート番号 10(G_{10}) の 1 縮退故障と G_{13} の 0 縮退故障を検出すべきテスト・パターンを検定した結果は検定の条件を満たさない．この回路では論理

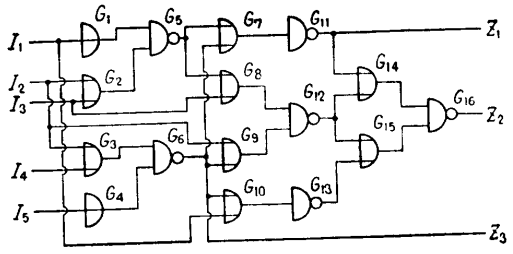

図 7 多重出力の論理回路
Fig. 7 Logic circuit with multiple outputs.

表 3 多重出力の論理回路の適用結果
Table 3 Applications for logic circuit with multiple outputs.

故障ゲート番号	0,1 縮退の区分	I_1	I_2	I_3	I_4	I_5	
1	0	1	1	0	1	1	
	1	0	1	0	1	1	
2	0	1	1	0	1	1	
	1	1	0	0	1	1	
3	0	1	1	1	1	1	
	1	1	0	1	0	1	
4	0	1	1	1	1	1	
	1	1	1	1	1	0	
5	0	0	1	0	1	1	
	1	1	1	0	1	1	
6	0	1	0	1	0	1	
	1	1	1	1	1	1	
7	0	0	1	1	1	1	
	1	1	1	1	1	1	
8	0	0	1	1	1	1	
	1	1	1	0	1	1	
9	0	0	1	1	1	1	
	1	0	0	1	1	1	
10	0	1	1	1	1	1	
	1	0	0	1	1	1	Remark 1
11	0	1	1	1	1	1	
	1	0	1	1	1	1	
12	0	1	1	0	1	1	
	1	0	1	1	1	1	
13	0	0	0	1	1	1	Remark 1
	1	1	1	1	1	1	
14	0	1	1	0	1	1	
	1	0	1	1	1	1	
15	0	1	1	0	1	1	
	1	1	1	1	1	1	
16	0	0	1	1	1	1	
	1	1	1	0	1	1	

Remark 1: Verification of these test patterns were not satisfactory. In this case suitable test patterns do not exist.

Remark 2: CPU time to generate these 16 test patterns was 1.04 sec by HITAC M-180.

の冗長の故に G_{10} の1縮退故障を検出するテスト・パターンがもともと存在しない.

6. 再試行の方法

以上で故障回路を検出すべきテスト・パターンを求める方法およびその適用結果について述べた. 得られたテスト・パターンは必ずしも必要な条件を満たすとは限らない. 解があるのに見つかっていない場合もあれば, 解が存在しないこともある.

本章では検定結果が不合格となった場合の再試行の方法について説明する. ゲート G_k が0縮退故障とする.

(1) $I_1, \cdots, I_n$ に再試行のための初期値を設定

検定結果は不合格であったとはいえ, 解に近いテスト・パターンが得られていると考える. 4章で述べた方法で得られたテスト・パターンを $I_{01}, I_{02}, \cdots, I_{0n}$ とする. この $I_{01}, \cdots, I_{0n}$ はすべて0か1である. この $I_{01}, \cdots, I_{0n}$ に対し, $I_{11}, I_{12}, \cdots, I_{1n}$ を次のように与える.

$I_{0j}=0$ ならば I_{1j} は0に近い正の値

$I_{0j}=1$ ならば I_{1j} は1より少し小さい値

(例えば, 0.01 や 0.99 を与える. 実験例ではこれらの値を与えた)

(2) I_{21} を求める

このために前述の S を利用する.

$$S(k, I_1, \cdots, I_n)=G_k(I_1, \cdots, I_n) \cdot \sum_{j=1}^{m} | Z_{kj}(1, I_1, \cdots, I_n) - Z_{kj}(0, I_1, \cdots, I_n)| \quad (25)$$

$S(k, 0, I_{12}, \cdots, I_{1n})$ と $S(k, 1, I_{12}, \cdots, I_{1n})$ を比較する.

$$S(k, 0, I_{12}, \cdots, I_{1n}) \geqq S(k, 1, I_{12}, \cdots, I_{1n}) \quad (26)$$

であれば, I_{21} は0に近い正の値 (例えば 0.01) と置き, 式(26) の不等号が逆であれば, I_{21} は1より少し小さい値 (例えば 0.99) を与える.

(3) I_{2j} を求める

$I_{21}, I_{22}, \cdots, I_{2(j-1)}$ が求まったとして, I_{2j} を求める方法を説明する.

$S(k, I_{21}, \cdots, I_{2(j-1)}, 0, I_{1(j+1)}, \cdots, I_{1n})$ と $S(k, I_{21}, \cdots, I_{2(j-1)}, 1, I_{1(j+1)}, \cdots, I_{1n})$ とを比較する. 前者が大きいか両者が等しければ, I_{2j} は0に近い正の値を与え, 後者が大きければ, I_{2j} は1より少し小さい値を与える.

この方法により $I_{22}, I_{23}, \cdots, I_{2n}$ をすべて求める.

(4) 新しいテスト・パターンの検定

$I_{21}, \cdots, I_{2n}$ が求まったとして新しくテスト・パターンの候補を決める.

$I_{2j} \leqq 0.5$ なら $\hat{I}_{2j}=0$

$I_{2j} > 0.5$ なら $\hat{I}_{2j}=1$

とおく．この $\bar{I}_{21}, \bar{I}_{22}, \cdots, \bar{I}_{2n}$ について S を計算して検定を行う．検定の結果

（i） 検定条件を満たす場合

この場合は，$\bar{I}_{21}, \cdots, \bar{I}_{2n}$ が求めるテスト・パターンである．5章の例2 Schneiderの反例のゲート番号3 (G_3) の0縮退故障の場合には，この $\bar{I}_{21}, \bar{I}_{22}, \bar{I}_{23}, \bar{I}_{24}$ が 0,0,0,0 となり，検定の条件を満たすテスト・パターンが求まった．

（ii） 検定条件を満たさない場合

この場合は次のステップで新しいテスト・パターンを求める．

（5） $I_{31}, \cdots, I_{3n}$, $\bar{I}_{31}, \cdots, \bar{I}_{3n}$ を求める

$I_{21}, \cdots, I_{2n}$ および $\bar{I}_{21}, \cdots, \bar{I}_{2n}$ が求まったとして次に $I_{31}, \cdots, I_{3n}$, $\bar{I}_{31}, \cdots, \bar{I}_{3n}$ を求める．

$S(k, I_1, \cdots, I_n)$ について $I_{31}, \cdots, I_{3(j-1)}$ が求まった場合に I_{3j} を求める方法を説明する．

$S(k, I_{31}, \cdots, I_{3(j-1)}, 0, I_{2(j+1)}, \cdots, I_{2n})$ と $S(k, I_{31}, \cdots, I_{3(j-1)}, 1, I_{2(j+1)}, \cdots, I_{2n})$ とを比較し，前者が大きいか両者が等しければ，$I_{3j}=0.01$ 後者が大きければ，$I_{3j}=0.99$ とおく．このようにして $I_{31}, \cdots, I_{3n}$ を求める．これから新しくテスト・パターンの候補を前述と同様に決める．

$I_{3j} \leqq 0.5$ なら $\bar{I}_{3j}=0$

$I_{3j} > 0.5$ なら $\bar{I}_{3j}=1$

とおく．この $\bar{I}_{31}, \cdots, \bar{I}_{3n}$ について次の検定を行う．

（i） $\bar{I}_{21}=\bar{I}_{31}, \cdots, \bar{I}_{2n}=\bar{I}_{3n}$ の場合

前述の例3のゲート番号 10(G_{10}) の1縮退故障と G_{13} の0縮退故障の場合は，この条件を満たす．すなわち故障を検出するテスト・パターンではなく，かつ新しいパターンを生成することができない．

（ii） $\bar{I}_{2j} \neq \bar{I}_{3j}$ となるような j が存在する場合

この場合は $\bar{I}_{31}, \cdots, \bar{I}_{3n}$ について改めて S を計算し，検定を行う．検定条件を満たせば求めるテスト・パターンである．もし検定条件を満たさなければ，更に次のステップへ進み同様の処理を行う．

（6） $I_{l1}, \cdots, I_{ln}$, $\bar{I}_{l1}, \cdots, \bar{I}_{ln}$ を求める

前述と同様に $I_{l1}, \cdots, I_{ln}$, $\bar{I}_{l1}, \cdots, \bar{I}_{ln}$ を求め，これらの値が検定条件を満たすか否かを調べる．検定条件を満たせば，それは求めるテスト・パターンである．もし検定条件を満たさなければ，一度得られたテスト・パターンの候補と比較する．

同じものがあれば，テスト・パターンは求められなかったとして処理を打ち切る．

同じものがなければ，$I_{(l+1)1}, \cdots, I_{(l+1)n}$, $\bar{I}_{(l+1)1}, \cdots, \bar{I}_{(l+1)n}$ を改めて求める．

このような処理を繰り返す．

前述の例では，2度繰り返すことによってテスト・パターンが見つかっているか，もともと適当なテスト・パターンが存在しないような縮退故障であるかの，いずれかであった．

7. 確率的手法についての考察

7.1 妥当性の検討

組合せ回路における仮定した縮退故障を検出するようなテスト・パターンを生成するためには，故障状態と異なる論理値を与え，かつ故障状態と正しい論理値が異なる値として出力端子に伝播せしめるような入力パターンを見つけ出す必要がある．この2条件を確率的に把え，この2条件を満たすには，入力端子に乱数をどのような重み（常に1ならば 1，通常の乱数ならば 0.5，常に0であれば 0，1が 1/3 で0が 2/3 ならば 0.33… と考える）で与えると故障を検出する可能性を高めるであろうかと考える．この可能性を高める方向に入力の値を定めて行くのである．

与えられたテスト・パターンが仮定した縮退故障を検出する可能性を意味する指標として，故障値と異なる値を与えることおよびその値が出力端子へ伝播することの二つに絞った．そこで，この二つの値の積を考えて，これらをすべての出力端子についての和を求めた．この和を最大にするような入力パターンが所定の条件を満たすのであるが，このため入力端子における値を逐次定めて行ったのである．

7.2 従来の方式との比較

乱数発生法では数多くの縮退故障を検出するテスト・パターンを短時間に発生することができるが，故障検出率がある程度に達すると，もはや新しい故障を検出するテスト・パターンを発生することは実際上不可能となってくる．

またDアルゴリズムについては一つの縮退故障を検出するテスト・パターンを生成するための計算機の処理時間は故障の検出に関連するゲート数に比例する以上に増大する。入力端子数を一定にとれば，本報告の手法ではゲート数に比例する処理時間で所要のテスト・パターンが生成でき，処理時間の点でDアルゴリズムよりも有利である．

従って，入力端子が少なくて論理の深い回路に適しているといえよう．

7.3 順序回路についての展望

順序回路の故障を検出するテスト・パターンは，記憶素子の情報保持の機能を含めてテストするものでなければならない．従ってある種の故障を検出するには複数個の一連のテスト・パターンの系列が生成される必要がある．この条件を満たすテスト・パターンを逐次本手法に従って生成することにより，順序回路のテスト・パターンの生成にも適用できる見通しである．

順序回路について換言すれば，本手法そのものが解答ではなく，本手法を応用することによって順序回路に適用できるであろうということであるが，応用する方法については現在検討中であり，次回に報告したい．

8. 結　言

以上で確率的手法によるテスト・パターンの生成について，手法を考案し，その 2〜3 の適用結果について述べた．これから得られる結論は以下の通りである．

確率的手法は組合せ回路の 0 または 1 縮退故障の検出に有効な手法である．入力端子についても出力端子についても同様である．本論文の説明では出力端子に限定したが，入力端子の縮退故障の場合には，その入力端子に 1 入力 1 出力のゲートが付加されていると考えることによって出力端子の故障に還元できる．

組合せ回路の 1 つの縮退故障を検出するテスト・パターンを生成するための計算機処理時間は，ゲート数と入力端子数の積に比例する．

ラッチを含む順序回路については本手法は有力な道具としての役割りを果たすことが見込まれる．

最後に本論文の作成に種々のご助言を頂いた広島大学樹下行三教授，(株)日立製作所村田健郎技師長に感謝致します．

参 考 文 献

1) Chang, H. Y. 他著 (鵜飼他訳)：ディジタル・システムの故障診断，産業図書，201 p. (1971).
2) Breuer, M. A. 著 (林訳)：ディジタル計算機の自動設計，産業図書，431 p. (1973).
3) Parker, K. P. and McCluskey, E. J.: Analysis of Logic Circuits with Faults Using Input Signal Probabilities, 4th Annual International Symp. on Fault-Tolerant Computing 74 IEEE, Digest of Papers, pp. 1-8〜1-12.

(昭和 52 年 8 月 26 日受付)
(昭和 53 年 8 月 25 日採録)