単一経路活性化法に基づく検査系列生成プログラム（Eアルゴリズム）

高松雄三（佐賀大・理工）　藤原秀雄（大阪大・工）　樹下行三（広島大・総合科学）

1. まえがき

近年，論理回路の高集積化に伴って，検査系列を生成するのに要する時間が増大する傾向にある。また，検査対象となる回路は，一般には順序回路であり，その検査系列の生成は組合せ回路のそれと比べて更に複雑になっている。最近では，これらの問題を解決するための一つの方法として，スキャンパスを含む回路設計が行われている[1],[2]。このように設計された回路では，検査系列の生成の問題は組合せ回路のそれに帰着される。従って，効率のよい組合せ回路の検査系列生成アルゴリズムを考察することは尚重要な問題である。

これまで，組合せ回路の検査系列を生成する多くの手法が発表されており[3]～[6]，一つの経路だけで故障信号を伝搬させる一次元経路活性化法[3]，又，それを多重経路の活性化に拡張したDアルゴリズム[4]，などが知られている。

ここで報告するEアルゴリズムは，単一経路活性化に基づく手法であり，アルゴリズムそれ自体が特に新しいということではないが，簡単な検査系列生成アルゴリズム（Easy test generation Algorithm）として手軽に利用できること，および他の手法との比較を行うときの一つの基準となる手法を提案したい，ということが主旨である。

手法として，単一経路活性化法を用いたが，よく知られているように，この方法では検出可能なすべての故障に対して検査入力が生成されるとは限らない。このために，Dアルゴリズムでは多重経路活性化の手法を用いているが，これを省略したEアルゴリズムとDアルゴリズムとの差異が実用上どのように現れるか，という点についての実現でもある。

Eアルゴリズムは，故障点から出力端子に至る一つの経路だけで故障信号を伝搬させようとする手法であるから，どの経路を優先的に選ぶかということが検査系列生成時間の上で重要である。本手法では，回路テーブル生成時に決められる順序で行われるようになっており，前処理によってこの順序を変更することにより，検査系列生成時における適当な尺度[7],[8]を導入することができる。

さて，検査系列生成時間の比較という意味では，具体的なデータについて，プログラム実行結果の比較を行わなければならない。このとき，単にアルゴリズムだけでなく，それに伴うデータ構造も問題となる。例えば，Dアルゴリズムにおける tC ベクトル[4]の作り方とその変更の方法によって実行時間が影響される。

この意味で，本稿では，単にアルゴリズムだけでなく，使用しているデータ構造も併せて報告する。なお，プログラムは，比較のための一つの基準という意味で互換性の高い FORTRAN で記述している。

2. Eアルゴリズム

以下，本稿で扱う回路は，AND，OR，NOR，NAND，NOT，および2入力 EXCLUSIVE－OR（XOR

と略記する）から構成されている組合せ回路である。対象とする故障は，回路の信号線が0又は1に縮退した故障である。

Eアルゴリズムでは，Dアルゴリズムと同様に5つの信号値，0，1，D，$\bar{D}$，Xを用いる。ここで，D（$\bar{D}$）は正常のとき1（0）であり，故障のとき，0（1）であることを表し，又，Xは未定義であることを示す。

Eアルゴリズムは，（1）前方操作，（2）一致操作，および（3）含意操作，の3つの基本操作から構成されている。

2.1 前方操作

故障点の故障信号D（$\bar{D}$）を被検査回路の一つの出力端子まで伝搬するただ一つの経路を活性化する操作である。このただ一つの経路で故障信号を伝搬させることがEアルゴリズムの特徴である。

単一経路によるD（$\bar{D}$）の伝搬を基本としているので，前方操作における複数個の自由度のある選択は以下の3つの場合である。

（1）故障の基本Dキューブの選択
（2）分岐点における経路の選択
（3）XORの伝搬Dキューブの選択

2.2 一致操作

含意操作で定義された信号値のうち，未正当化の信号値（2.3参照）から後方操作して被検査回路の入力端子に信号値を矛盾なく割り付ける操作である。Dアルゴリズムにおける一致操作と基本的には同一である。この一致操作における選択は，AND，NAND（OR，NOR）ゲートのいずれの入力を0（1）に設定するかの選択，XORゲートの入力の2つの組合せのうち，どちらを設定するかの選択がある。

2.3 含意操作

2.1の前方操作および2.2の一致操作で逐次設定される信号値に対して一意に定まる回路の信号値をすべて設定する操作である。この操作において，ゲートの出力信号値が定まり，そのゲートの入力信号値が一意に定まらない（入力信号値の設定に選択が許される）場合は，そのゲートの出力信号値は未正当化であるので，その信号線に未正当化フラッグ（U-フラッグ）をセットする。

この操作において，同一信号線に異なる信号値を設定しようとした場合，又は，要求される信号値に設定できない場合，などが生じたとき，その時点での信号設定は不可能である。

DALG-II[4]と同様に，本稿のEアルゴリズムもこの含意操作を優先している。

3. プログラムの構成

ここでは，Eアルゴリズムに基づく検査系列生成法のプログラム化の手法について述べる。

3.1 全体の構成および概要

作成したプログラムは，図1に示すように，（I）回路記述から回路テーブルを作成する入力ルーチン，および（II）Eアルゴリズム実行ルーチンの2つから構成されている。

（I）の入力ルーチンにおける回路記述フォーマット，仕様については，文献（9）のそれとほぼ同様であるが，回路テーブルについては，入力側から出力側へ，出力側から入力側への両方向のポインターを有する回路テーブル（図2の回路テーブル参照）に改良されている。

（II）のEアルゴリズム実行ルーチンは，基本的には，以下示す3つのサ

ブルーチンを有している。

EDRIVE

前方操作を行うルーチンであり，指定された故障に対する故障の基本Dキューブの作成，および故障信号D（$\overline{D}$）を一つの出力端子まで一つの経路を伝搬させる操作を行い，成功すれば，故障信号の伝搬する出力端子，正常／故障時の出力信号値を求める。この操作中，複数個の自由度のある選択が生じた場合，すなわち，2.1の(1)，(2)，または(3)に対して図3で示すスタック（STACK(3,…)）にそれらは確保される。このとき，確保される順序は回路テーブル作成時に決められる順序に従っている。

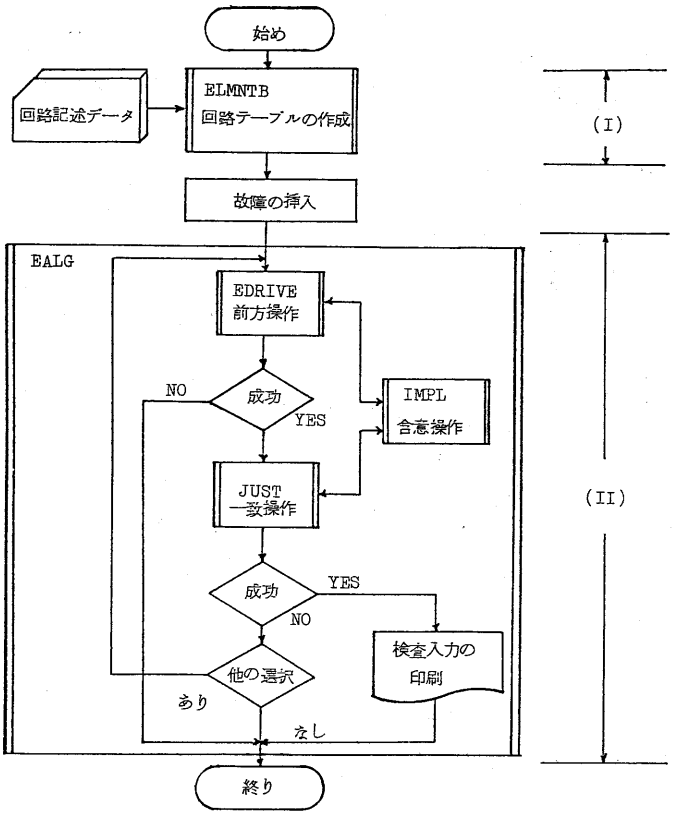


図 1　　全体の構成

JUST

IMPLでセットされたU-フラッグを有する信号線を探し，そのゲートについて一致操作を行う。このとき，複数個の自由度のある選択が生じた場合，すなわち，AND，NAND（OR，NOR）ゲートのいずれの入力を0(1)に設定するかの選択，XORゲートの入力の2つの組合せのうち，どちらを設定するかの選択，これらすべてをスタック（STACK(3,…)）に格納する。この操作をU-フラッグを有するすべての信号線に対して行い，一致操作がすべて成功すれば，検査入力が生成される。そうでなければ，スタックおよびTC(3,…)（図3参照）をJUSTが呼ばれた状態にに戻る。

IMPL

EDRIVE，JUSTで逐次設定された信号値により，一意に定まるべての信号値を定める。このとき，出力信号値が定まり，入力が一意に定まらないゲートの出力信号線にU-フラッグをセットする。この操作が矛盾なく終了すれば含意操作は成功となり，又，そうでなければ，失敗として，それぞれ呼ばれたサブルーチンに戻る。

含意操作で一意に定まるすべての信号線はTC内のリンクポインタ（図3参照）でリンクされている。

3.2 データ構造

ここでは，作成したプログラムで使用している データ構造について示す。

3.2.1 回路テーブル

図2は回路記述データから作成した回路テーブルの例である。

ここで，

ELMTBL(7,...)の各要素は，

TYPE：信号線番号(INDEX)Jを出力とするゲートの種類又はJの種類．

NFI：Jのファンイン数．

FIL：Jのファンインリスト．

NFI=1のとき，FILはELMTBL(7,...)へのポインタ．

NFI≧2のとき，FILはFLIST(...)へのポインタ．

NAME(1), NAME(2)：Jの信号線名．

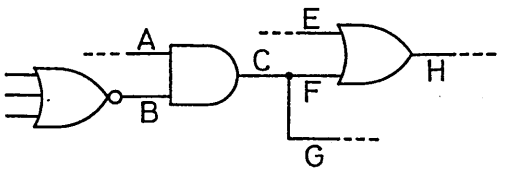
NFO：Jのファンアウト数．

FOL：Jのファンアウトリスト．

NFO=1のとき，FOLはELMTBL(7,...)へのポインタ．

NFO≧2のとき，FOLはFLIST(...)へのポインタ。であり，また，FLIST(...)の要素は，ファンイン，ファンアウトリストを示す。

その他，被検査回路の入力，出力端子をそれぞれ示すテーブルPITBL(3,...)，POTBL(3,...)などがある。



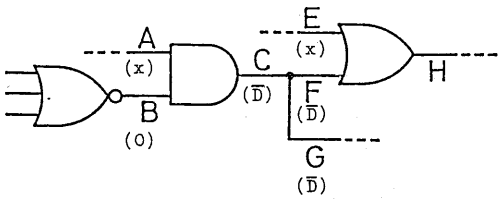
ELMTBL(7,...)

INDEX	TYPE	NFI	FIL	NAME (4) (5)	NFO	FOL
1				A		
2				B		
3	2(AND)	2	1	C	2	3
4				E		
5	3(FOUT)	1	3	F	1	7
6				G		
7	1(OR)	2	5	H		

FLIST(...)

NO.
1
2
5
6
4
5

図 2　回路テーブル



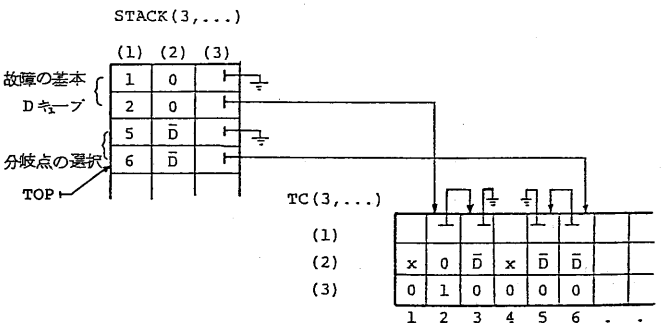


図 3　データ構造

3.2.2 スタックおよびリニアリスト

Eアルゴリズムを実行するため，図3で示すようなデータ構造を有するスタック(STACK(3,...))およびリニアリスト(TC(3,...))を用いる。両者は，図に示すようにリンクされている。ここで，STACK(3,...)の各要素は，

STACK(1,...)：信号線番号J.
STACK(2,...)：Jの信号値.
STACK(3,...)：TC(3,...)へのポインタ.
Top：スタックのトップを示すポインタ.
TC(3,...)の各要素は，
TC(1,J)：TC内のリンクポインタ.
TC(2,J)：信号線番号Jの信号値(0, 1, D, $\overline{D}$ またはX).
TC(3,J)：U-フラッグ(1のときU-フラッグセット).
を表す。

上記のデータ構造を用いることにより，逐次設定される信号値の設定，および，それらをリンクすることができる。すなわち，前方操作中の複数個の自由度のある選択の確保，一致操作中の複数個の自由度のある選択の確保が行われる。

こうして，Topで示されている時点での選択が失敗したとき，STACK(3,Top)でリンクされているTCの信号値，リンクポインタ，U-フラッグがリセットされ，次の選択に対して操作が行われる。このスタックおよびリニアリストを用いることにより，Eアルゴリズムの処理を効率よく実行することができる。

3.3 プログラムの仕様

作成したプログラムの仕様は次のとおりである。

(1) 使用言語：FORTRAN IV.

(2) プログラムの規模：

(I) 入力ルーチン-----約400ステップ

(II) Eアルゴリズム実行ルーチン-----約1000ステップ

(3) データ領域：

被検査回路のゲート数---------G

被検査回路の信号線数---------L

被検査回路のファンアウト点，の数--F

ゲートの平均ファンイン数-------m

ファンアウト点，の平均ファンアウト数--m'

とすると，

(I) 回路テーブル---$7L+mxG+m'xF$

(II) TC(3,...)-----3xL

である。

(4) 作業領域：

各操作における選択を確保するスタックSTACK(3,...)およびSET(2,...)が必要である。ここで，SET(2,...)は含意操作を行うときに使用する作業領域である。これらの大きさは被検査回路に依存するため，評価はむずかしい。

(5) 実行例：

付録に示す。

表 1　　実行結果の例

回路番号	ゲート数	信号線数	入力数	出力数	代表故障数 (N_F)	検出率 (%)	検査系列生成時間 (T) 秒	T/N_F 秒
#1	31	120	11	3	137	100	0.93	0.007
#2	46	129	9	2	132	100	0.97	0.007
#3	70	177	9	5	182	97.8	1.28	0.007
#4	63	200	14	8	237	100	1.63	0.007
#5	107	320	14	8	380	95.8	4.84	0.013
#6	118	335	10	4	384	93.8	4.75	0.012
#7	464	1243	8	7	1338	98.1	51.1	0.038
#8	152	421	12	6	473	99.2	7.84	0.016

4. 実行結果

現在，作成したプログラムを数10～数100ゲート程度からなる回路に適用して検査系列の生成を行っている。

表1にその実行結果の一部を示す。使用計算機は，九州大学大型計算機センター FACOM M-200である。

表1の結果から，検査系列生成時間(T)をゲート数(G)に対して図示すると，図4のようになる。

図4から，

$$T \fallingdotseq 0.46 \times G^{1.9} \quad (秒)$$

が得られる。これまで，$T \propto k \times G^{n}$ ($n \fallingdotseq 2.2 \sim 3$) であることが知られている[10],[11]。回路規模が小さいため明確なことは言えないが，Eアルゴリズムは高速に検査系列の生成が行えると思われる。

また，検査系列生成時間(T)／故障数(N_F)をゲート数(G)に対して図示すると，図5のようになる。

図5から，

$$T/N_F \fallingdotseq 0.18 \times G^{0.87} \times 10^{-3} \quad (秒)$$

が得られる。このことから，小規模の回路に対するT/N_Fは，ほぼゲート数(G)に比例している。

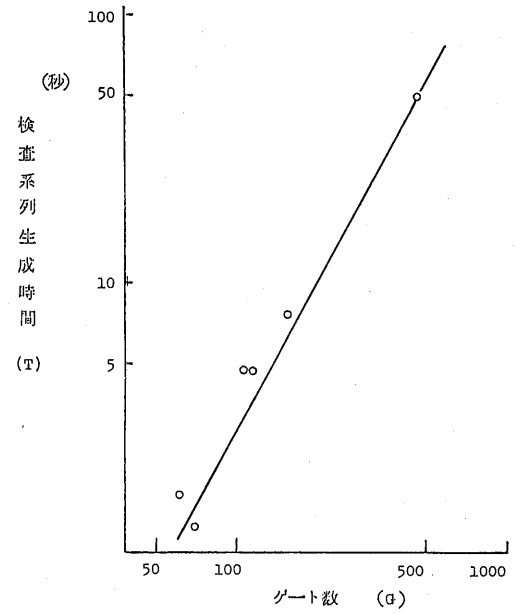


図 4 検査系列生成時間

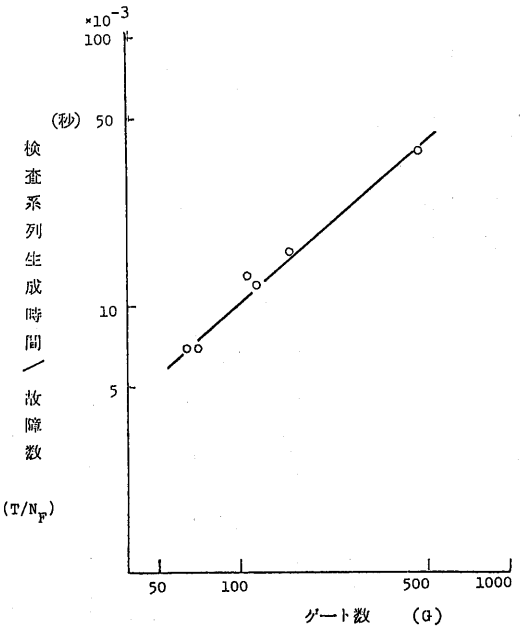


図 5 検査系列生成時間／故障数

5. むすび

単一経路活性化法に基づく検査系列生成法であるEアルゴリズムを提案し，そのプログラム化の手法および実行結果を示した。Eアルゴリズムは，アルゴリズム自身が簡単であり，3.2で示したデータ構造を用いることによりプログラム化も容易である。今後，(1)検査系列生成時間の高速化，(2)検出率の向上，に関してさらに検討する必要があるが，これらに対しては，検査系列生成のための尺度[7],[8]などの導入により改善が計れるものと考えてい

る。

アルゴリズムの評価については大規模な回路の実行結果が必要である。

現在，九州大学大型計算機センター（FACOM M-200），佐賀大学計算機センター（FACOM M-150F），大阪大学大型計算機センター（NEAC ACOS 900），広島大学総合情報処理センター（HITAC M-200H，M-180）上で実行可能であるので，その効果的な活用を期待している。

参考文献

(1) Yamada, A., et al: "Automatic test generation for large digital circuits", Proc., 14th Design Automation Conf., June, p.78, 1977.

(2) Eichelberger, E. B. and Williams, T. W. : "A logic design structure for LSI testability", ibid., p.462.

(3) Armstrong, D. B.: "On finding a nearly minimal set of fault detection tests for combinational logic nets", IEEE Trans. Electron. Comput., Vol. EC-15, 1, p.66, Feb. 1966.

(4) Roth, J. P., et al.: "Programmed algorithms to compute tests to detect and distinguish between failures in logic circuits", IEEE Trans. Electron. Comput., Vol.EC-16, 5, p.567, Oct. 1967.

(5) Goel, P.: "An implicit enumeration algorithm to generate tests for combinational circuits", The 10th Int. Symp. on Fault-Tolerant Computing, p.145, Oct. 1980.

(6) Kinoshita, K, Takamatsu, Y. and Shibata, M.: "Test generation for combinational circuits by structure description functions", ibid., p.152.

(7) Goldstein, L. H.: "Controllability/observability analysis of digital circuits", IEEE Trans. Circuits and Systems, Vol. CAS-26, 9, p.685, Sept. 1979.

(8) 藤原，尾崎："常見的手法によるテスト生成のための新しい尺度"，電子通信学会，電子計算機研究会資料 EC80-38(昭55年10月).

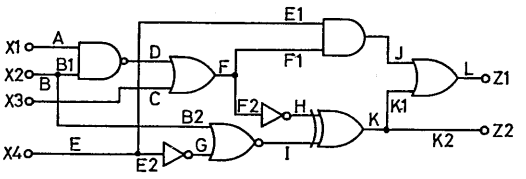
(9) 樹下，高松，笹田："構造記述関数による組合せ回路の検査系列の生成について"，情報処理学会，電子装置設計技術研究会資料 3-1，(昭54年12月).

(10) Bottorff, P. S., et al.: "Test generation for large logic networks", Proc., 14th Design Automation Conf., p.479, June 1977.

(11) Williams, T. W. and Parker, K. P.: "Testing logic networks and designing for testability", Computer, Vol.12, 10, p.9, Oct. 1979.

付録

〔実行例〕



(a) 論理回路の例

```
 1      ***** SAMPLE CIRCUIT *****
 2       IN/X1,X2,X3,X4/A,B,C,E/
 3       FOUT/B/B1,B2/
 4       NAND/A,B1/D/
 5       FOUT/E/E1,E2/
 6       OR/C,D/F/
 7       NOT/E2/G/
 8       NOT/F2/H/
 9       NOR/G,B2/I/
10       AND/E1,F1/J/
11       XOR/H,I/K/
12       FOUT/F/F1,F2/
13       FOUT/K/K1,K2/
14       OR/J,K1/L/
15       OUT/L,K2/Z1,Z2/
16       END
```

(b) 回路記述

ELMTBL(7,...)

INDEX	TYPE	NFI	FIL	LABEL NAME	NFO	FOL
1	0	1	1	A	1	7
2	0	1	2	B	2	1
3	0	1	3	C	1	10
4	0	1	4	E	2	5
5	3	1	2	B1	1	7
6	3	1	2	B2	1	14
7	-1	2	3	D	1	10
8	3	1	4	E1	1	16
9	3	1	4	E2	1	11
10	1	2	7	F	2	15
11	-3	1	9	G	1	14
12	3	1	10	F2	1	13
13	-3	1	12	H	1	17
14	-2	2	9	I	1	17
15	3	1	10	F1	1	16
16	2	2	11	J	1	20
17	5	2	13	K	2	17
18	3	1	17	K1	1	20
19	3	1	17	K2	1	22
20	1	2	19	L	1	21
21	4	1	20	Z1	0	0
22	4	1	19	Z2	0	0

FLIST(...)

1	5
2	6
3	1
4	5
5	8
6	9
7	3
8	7
9	11
10	6
11	8
12	15
13	13
14	14
15	15
16	12
17	18
18	19
19	16
20	18

(c) 回路テーブル

STACK(3,...)

3	0	3
15	3	0
12	3	15
18	2	0
19	2	18

TC(3,...)

7	5	14	0	1	2	10	0	0	0	0	13	17	6	12	0	0	19	22	0	0	0
1	1	0	4	1	1	0	4	4	3	4	3	2	0	3	4	2	2	2	4	4	2
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

(d) 信号線下の 1 縮退故障に対する検査入力生成時の最終 STACK(3,...) および TC(3,...).
