静的学習を用いた組合せテスト生成の高速化法

日大生産工(学部)　○大森　悠翔　　日大生産工　　細川　利典

1　はじめに

回路の高集積化が進むにつれて，テスト生成の対象となる回路の規模がますます大きくなる傾向にあり，それとともにテスト生成に要する計算費用が膨大なものとなってきている．これを解決するためには，スキャン設計1)のようなテスト容易な設計技法で論理設計を行うとともに，さらに効率の良いテスト生成法を考案することが必要である．テスト容易化設計としてフルスキャン設計が適用された論理回路はテスト生成の複雑度を組合せ回路のレベルに容易される．したがって，組合せ回路のテスト生成において効率の良い方法を考えれば十分である．

テスト生成において，多くの故障は無作為に生成したテストパターンで検出される確率が高いので，高い故障検出率を臨まない場合は，乱数パターンを用いて故障シミュレーションを行う方式が計算時間も速く有効である．しかし，高い故障検出率を望む場合には，無作為に生成した乱数パターンだけでは能率が悪く，次のような方法が必要となる．すなわち与えられた故障が検出可能であるか否かを判定でき，検出可能な場合には常にテストパターンを生成することができる方法である．このようなテスト生成法を”完全である”という．完全なテスト生成法としては，ブール微分法 2) 3), Dアルゴリズム 4), PODEMアルゴリズム 5)，FANアルゴリズム 6), SOCRATES 7)等が報告されている．

しかしながら，回路が大規模化すると，これらのアルゴリズムでは，現実的な時間で100%の故障検出効率(全故障数に対する検出故障数と冗長故障数の和の割合)を得るテストパターンを生成するのは困難である．検出困難な故障に対しては，バックトラックを多発し，テストパターンの生成(冗長故障判定も含む)を途中で打ち切っている．

本稿では，過去に提案された様々な手法 4) 5) 6) 7)を採用し，それらを組み合わせてバックトラック頻度の小さいアルゴリズムを提案する．

2　問題提示

テスト生成では，故障を検出するために外部入力に論理値(0 or 1)を割り当て，故障を仮定してシミュレーションを行い，少なくとも1つの外部出力で故障を検出できることを確認する．外部入力数をNとすると解空間は となる．

図1に示すように解空間は，次のような空間によって構成される．

- 解が存在する空間
- 特定できない解無し空間
- 解無し空間

ここで次のような事が重要となる．

1. 特定できない解無し空間を小さくする
2. 探索プロセスにおいて解無し空間を避ける

次に挙げるような手段によって2つの目的を達成することができる．

- 効率的な探索空間の枝刈り
- バックトラック数の減少
- できるだけ早期の矛盾の発見
- 特定できない解無し空間での無駄なバックトラックの回避

図1　解空間

Acceleration of A Test Generation Method Using Static Learning
Yuushou OOMORI, Toshinori HOSOKAWA

3　テスト生成のための効果的な手法

3.1.　諸定義

本稿では対象とする回路は組合せ回路であり，故障モデルは単一縮退故障を想定する．信号線値は，正常値，故障値ともそれぞれ0, 1，Xの値をもつ9値(1, 0, D, $\overline{D}$ X, 0/X, 1/X, X/1, X/0 ここでDは1/0, $\overline{D}$ は0/1を表す)として表現される．

3.2.　故障信号挿入

図2に示すように，故障信号線に故障信号を割り当て，テスト対象ゲートの入出力の値が一意的に決まる場合，その値を割り当てる．

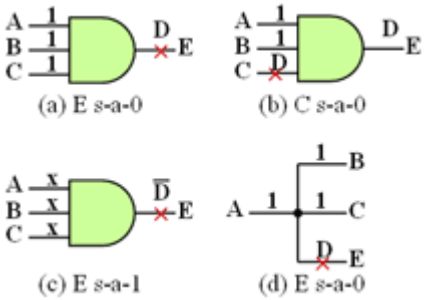

図2　故障信号挿入例

3.3.　含意操作

一意的に決定される信号線の値は全て決めるという処理を含意操作という．入力側から出力側への前方操作と，出力側から入力側への後方操作の両方を行う．一意的に決まる値が多いほど早期の矛盾の発見に繋がり解無し空間への探索を避ける効果がある．

3.4.　一意活性化操作

Dフロンティアが唯一の時，Dフロンティアから外部出力へ向かう故障の影響が唯一通過する各素子に対して値を設定する．含意操作同様，早期の矛盾の発見に繋がる．

3.5.　多重後方追跡

正当化，D伝搬の際に各素子に設定する値を外部入力まで後方追跡する．初期目標群(目標の集合)を設定し後方追跡を多重化している．これにより最も適した外部入力への値割当てが可能となる．

3.6.　バックトラック

含意操作で矛盾が生じるか，Dフロンティアがなくなりかつどの外部出力にも故障が伝搬していない時に決定木をバックトラックする．

4　静的学習

4.1.　直接含意と間接含意

含意には，ゲートの入出力の接続関係から求めることのできる直接含意と，ゲートの入出力の接続関係だけでは求めることのできない間接含意がある．例えば，図3の回路において，”a=0ならばb=0,c=0,d=0”となるのは直接含意である．直接含意は，論理ゲートの入出力関係と推移律を利用して，容易に得ることができる含意である．

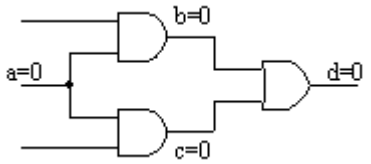

図3　直接含意の例

一方で，図4のように，”d=1ならばa=1”となるのは間接含意である．d=1のとき，直接含意では他の信号値は決定できないが，b=1またはc=1のいずれかが成り立たなければならないので，いずれの場合もa=1が必要となる．

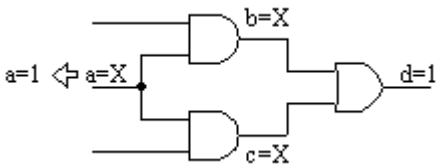

図4　間接含意の例

4.2.　学習規則A

静的学習は，直接含意の結果から間接含意を求める処理である．ある命題が真ならば，その対偶も真であるという関係を用いる．例えば，図3でa=0から含意操作を実行すると”a=0ならばd=0”という直接含意を得るが，この命題の対偶から，図4の”d=1ならばa=1”という間接含意が得られる．

静的学習の処理は回路内の各信号線について，その信号線の値を0に設定しての含意操作と1に設定しての含意操作をそれぞれ実行する．次の学習規則を満たす直接含意が存在するとき間接含意を保存する．

[学習規則A] 信号線sの値をv(v∈{0,1})とする含意操作により，ある2入力以上のゲートの出力線tが値w(w∈{0,1})をとり，かつ，wがそのゲートの非制御値であるとき，間接含意として”t = ¬wならば s = ¬v”を保存する．

4.3. 学習規則B

学習規則Aに当てはまらない直接含意の対偶は，間接含意にはならず，含意操作で容易に計算できると考えられてきた．ところが，図5の回路において，”c=0ならばd=1”の直接含意が求まるが，その対偶は”d=0ならばc=1”である．これは，図6のように，d=0からの含意操作により直接含意として求められる．

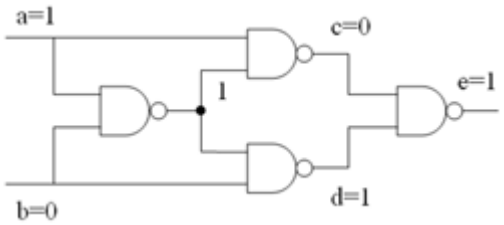

図5　c=0からの含意操作

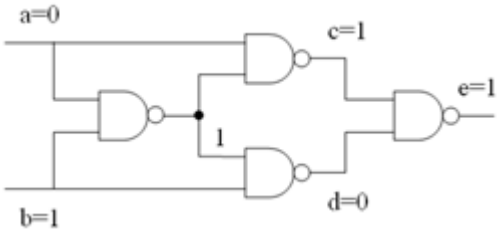

図6　d=0からの含意操作

図5の含意操作の結果には，学習規則Aにあてはまるものはないため，これまでのやり方では間接含意は得られていなかった．しかしながら，図5から，”c=0ならばb=0”が成り立つことがわかるが，この対偶の”b=1ならばc=1”は間接含意である．これは学習規則Aでは見逃されている．b=1からの含意操作を考えたとき，bはNANDゲートへの入力であり，b=1の含意操作が終了する理由の逆を考えて，後方への含意操作によりbが制御値をとることが判断できれば良い．一般化すれば，以下のような学習規則が得られる．

[学習規則B] 信号線sの値をv(v∈{0,1})とする含意操作において，後方への含意操作により，ある2入力以上のゲート入力線tが値w(w∈{0,1})となり，かつ，wがそのゲートの制御値であるとき，間接含意として”t = ¬wならば s = ¬v”を保存する．

4.4. 学習順序

静的学習は，学習処理の順序によって得られる学習の数に差が生じる 8)．図7の回路において2つの処理手順を適用し学習処理を行う．

処理手順Aは信号線fの学習を行った後，信号線cの学習を行う．処理手順Bは信号線fの学習を行う前に，信号線cの学習を行う．

2つの処理手順による学習の結果を表1に示す．

処理手順Aでは，f=1からの含意操作では含意関係をなにも得られず，c=0からの含意操作によってc=0 → f=0となり，f=1 → c=1の含意関係が得られるだけである．

処理手順Bでは，c=0からの含意操作によってc=0 → f=0となり，f=1 → c=1の含意関係が得られる．さらにf=1からの含意操作によってf=1 → g=0となる．結果としてg=1 → f=0の含意関係がf=1 → c=1という含意関係を用いたことによって得られる．このように，処理手順Bは，処理手順Aよりも多くの含意関係を得ることができる．

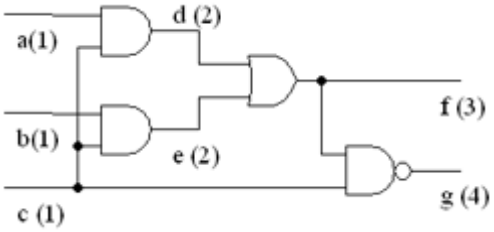

図7　回路例

表1　学習手順例

	初期割当	得られる含意関係
処理手順A	1. f = 1	無し
	2. c = 0	f = 1 → c = 1
処理手順B	1. c = 0	f = 1 → c = 1
	2. f = 1	g = 1 → f = 0

本稿では文献 8) で学習の順序として効果が高いとされている，回路の論理段数を基にした前方への幅優先探索による学習順序を用いている．図7の回路ではその学習順序はa-b-c-d-e-f-gとなる．

5 実験結果

FANアルゴリズムと静的学習を導入したアルゴリズム 7)を実装し，ISCAS85ベンチマーク回路に対して実験を行った．表2にFANアルゴリズムの実験結果を示し，表3に静的学習を導入したアルゴリズムの実験結果を示す．共にバックトラックリミットは100回とした．

表2　FAN アルゴリズム

回路名	故障検出率	故障検出効率	テストパターン数	打切り故障数	冗長故障数	バックトラック数	テスト生成時間[秒]
c880	100.00	100.00	87	0	0	16	0.33
c1355	88.06	88.06	62	188	0	23066	21.34
c1908	99.52	99.79	149	4	5	516	4.59
c2670	95.74	98.33	152	46	71	4893	15.81
c6288	99.56	100.00	31	0	34	138	7.22

表3　静的学習を導入したアルゴリズム

回路名	故障検出率	故障検出効率	テストパターン数	打切り故障数	冗長故障数	バックトラック数	テスト生成時間[秒]
c880	100.00	100.00	88	0	0	96	0.48
c1355	97.59	97.59	104	38	0	5568	9.53
c1908	99.52	99.89	155	2	7	314	6.19
c2670	95.74	98.36	148	45	72	4979	19.89
c6288	99.56	100.00	31	0	34	123	20.98

結果について考察してみるとc1355においてFANアルゴリズムでは打切り故障数188，故障検出率88.06%に対して，静的学習導入後は打ち切り故障数38，故障検出率97.59%となり故障検出率が向上していることがわかる．さらに，バックトラック数がFANアルゴリズムでは23066回なのに対して，静的学習導入後には5568回と1/4程度に削減されている．そのためテスト生成時間も削減されていることがわかる．

c1908，c2670のFANアルゴリズムと静的学習を導入したアルゴリズムの冗長故障数をみてみると静的学習を導入したアルゴリズムの冗長故障数の方が多いことがわかる．学習を導入したことによって冗長故障判定の性能が上がったと考えられる．

c6288ではFANアルゴリズムに比べて静的学習導入後の方はバックトラック数が少ないのに対してテスト生成時間が増加しているが，これは，前処理として学習処理を行っているためのオーバーヘッドが影響している．

6 おわりに

今回FANアルゴリズム，静的学習を導入したアルゴリズムを実装しテスト生成を行い，その結果を示した．表3の結果より打切り故障数やバックトラック回数を見ると，改良の余地があることがわかる．今後さらなるバックトラックの減少を目指すために，様々な手法を取り入れていく．

テスト生成時間削減のために，学習処理時の無駄な学習の保存を回避することを考えている．

「参考文献」

1) H． Fujiwara， “Logic Testing and Design for Testability，” The MIT Press，1985.

2) Sellers，F. F. . ：Analyzing Errors with the Boolean Difference，IEEE Trans．1968

3) Kinosita，K． ：Test Generation for Combinational Circuits by Structure Description Functions，Proc．1980

4) Roth，J．P．:Programmed Algorithms to Compute Tests to Detect and Destinguish between Failures in Logic Circuits，IEEE Trans．1967.

5) Goel，P．:An Implicit Enumeration Algorithm to Generate Tests for Combinational Logic Circuits， IEEE Trans．1981.

6) Fujiwara:on the Acceleration of Test Generation Algorithms, IEEE Trans. 1983.

7) MICHAEL H. SCHULZ, ERWIN TRISCHLER, and THOMAS M. SARFERT，A Highly Efficient Automatic Test Pattern Generation System， IEEE Trans．1988．

8) Hideyuki ICHIHARA， Seiji KAJIHARA， Kozo KINOSITA：On Processing Order for Obtaining Implication Relations in Static Learning，IEICE Trans． 2000．

9) Keitaro Saruwatari， Seiji Kajihara：On Efficient Identification and Preservation of Indirect Implications in Static Learning，TECHNICAL REPORT OF IEICE． 2003．