社団法人　情報処理学会　研究報告
IPSJ SIG Technical Reports

2006－ＳＬＤＭ－125（９）
2006／5／12

テスト生成における間接含意の効率的な生成方法

吉村　正義†　　梶原　誠司††　　松永　裕介†††

† 福岡知的クラスター研究所　〒814-0001 福岡市早良区百道浜 3-8-33
†† 九州工業大学　〒820-8502 福岡県飯塚市大字川津 680-4
††† 九州大学　〒816-8580 福岡県春日市春日公園 6-1
E-mail: †yosimura@fleets.jp, ††kajihara@cse.kyutech.ac.jp, †††matsunaga@c.csce.kyushu-u.ac.jp

あらまし　テストパタン自動生成システムにおいて，間接含意は，含意操作の対象を拡大し，テストパタン生成の探索範囲を狭める重要な技術である．一方，回路規模の増大に伴い，間接含意の生成量が増大し，問題となっている．そこで本論文では，間接含意を，非循環有向グラフを用いてモデル化し，より小さな間接含意集合の生成方法を提案する．またベンチマーク回路において，従来手法との比較評価を行う．

キーワード　テストパタン自動生成システム，含意操作，間接含意，非循環有向グラフ

Efficient generation method of indirect implication on ATPG

Masayoshi YOSHIMURA†, Seiji KAJIHARA††, and Yusuke MATSUNAGA†††

† Fukuoka Laboratry for Emerging & Enabling Technology of SoC(FLEETS)　3-8-33 Momochihama, Sawara-ku, Fukuoka, 814-0001, Japan
†† Kyushu Institute of Technology　680-4 Kawazu, Iizuka-shi, Fukuoka, 820-8502, Japan
††† Kyushu University　6-1 Kasuga-koen, Kasuga, Fukuoka 816-8580 Japan
E-mail: †yosimura@fleets.jp, ††kajihara@cse.kyutech.ac.jp, †††matsunaga@c.csce.kyushu-u.ac.jp

Abstract　On ATPG, indirect implication is an important technich for reduced area of searching on generating test pattern. This is a big problem that increasing indirect list increases with increasing circuit. We proposed that an efficient genration method of indirect implication list for small size with DAG. Finally, Our method is evaluated on bench mark circuit.

Key words　ATPG, implication, indirect implication, DAG

1. はじめに

近年の半導体集積技術の進展に伴い，LSI の回路規模は増大し，LSI の検査に関するコストが増大している．それに対して，LSI の検査を容易にするテスト設計が重要となってきている．一般の順序回路に対する自動テストパターン生成（ATPG）は困難な問題であり，高い故障検出効率を得るテストパターンを生成するには，フルスキャン設計 [1], [2] が必要である．

フルスキャン設計は回路中のすべてのフリップフロップ（FF）をスキャン FF で構成し，そのスキャン FF をシフトレジスタ状に構成することによって，テスト時にスキャン FF を外部入出力と等価とみなすことができる．フルスキャン設計方法では，スキャン FF の入出力を外部入出力とみなした回路（核回路）が組み合わせ回路であるため，核回路に対して組合せ ATPG の適用が可能となり，高い故障検出効率を得ることができる．

一方，LSI の微細化に伴い，外部入力から外部出力までのゲート段数が増える傾向にある．ゲート段数の増加に伴い，組合せ ATPG を用いたテストパターン生成の適用範囲となる核回路の規模が増大することによって，組合せ ATPG がより困難となる．さらに LSI の微細化に伴い，従来の縮退故障を想定して生成されたテストパターンでは検出できない故障が増加してきている．この問題に対して，縮退故障ではなく，遷移故障やパス遅延故障などの故障を想定して生成したテストパターンを用いることによって，LSI の検査時により多くの故障を検出できるようにしている．しかしこの遷移故障やパス遅延故障では，状態を遷移させる必要があるため，テストパターン生成時において，状態を設定する時刻と，状態を遷移させる時刻の 2 時刻分考慮する必要がある．つまりテストパターン生成においては，核回路を 2 時刻分，時間展開する必要がある．つまり従来の 2 倍のゲート段数となり，テストパターン生成がますます困難になる．

それに対して，ATPG をより効率的にする手法の一つとし

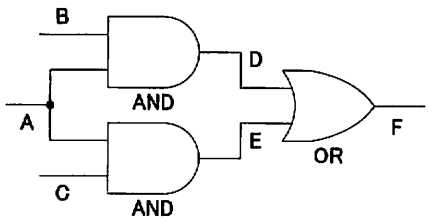

図 1　間接含意の例
Fig. 1　Example of indirect implicaiton

て，間接含意 [3], [4] がある．間接含意は，ATPG の前処理の一つとして行われ，ATPG 処理におけるテストパターン生成で用いられる．間接含意は，信号線間の含意の関係の対偶の関係によって求められる．しかし当該の間接含意を用いることなく，直接含意とその他の間接含意を用いて，含意できる間接含意が存在する．また回路規模が増大すると，回路規模の増大以上に間接含意の集合の大きさが増大し，間接含意を用いる上で大きな課題となっている．

そこで本論文では，より小さな間接含意の集合を得ることを目的として，回路の含意関係を，非循環有向グラフ（Directed Acyclic Graph : DAG）によって示し，非循環有向グラフの到達関係を用いることによって，間接含意が他の直接含意および間接含意によって被覆されているかを判断基準とすることができる．この基準に基づいて，より小さい間接含意の集合を得る方法を提案する．

2 章では，間接含意について説明する．3 章では，提案する非循環有向グラフを用いた間接含意の集合を得る手法について述べる．4 章では提案した手法と従来手法との比較実験を行う．5 章では本論文のまとめと今後の課題を示す．

2. 間接含意

間接含意とは，信号線間の含意の関係の対偶の関係に基づいて求められる．例を用いて説明を行う．図 1 において，信号線 A に値 0 を割り当てたとき，信号線 F は値 0 となる．すなわち，

$$(A = 0) \Rightarrow (F = 0) \tag{1}$$

となる．この含意の関係の対偶は，

$$(F = 1) \Rightarrow (A = 1) \tag{2}$$

である．式 (1) が真であるなら，式 (2) も真である．つまり，図 1 において，信号線 F が値 1 を割り当てたとき，必ず信号線 A は値 1 となる．

この信号線間の含意の関係の対偶を用いた含意操作を間接含意と呼ぶ．また間接含意は単純にすべての信号線と値 0, 1 から求めることができる．この間接含意を用いることによって，ATPG ににおけるテストパターン生成の際に，同じ信号線に同じ値を設定した場合においても，より多くの他の信号線の値を

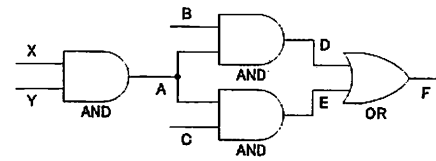

図 2　重複した間接含意の例
Fig. 2　Example of overlapping indirect implicaiton

一意に決定することができる．そのため，すでに設定した信号線の値によって矛盾が引き起こされる場合，早期に発見することができる．このことによってテストパターン生成時の課題の一つである冗長故障の判定が高速にできるという利点が生じる．

しかし，間接含意が得られる箇所が，テストパターン生成において，すべて有用であるわけではない．例えば，図 2 では，$F = 1 \Rightarrow A = 1$ に加えて，$F = 1 \Rightarrow X = 1$ と $F = 1 \Rightarrow Y = 1$ も間接含意として得ることができる．しかし，信号線 A は，信号線 X と信号線 Y を入力とする AND ゲートで接続されており，直接含意を用いることによって，$A = 1 \Rightarrow X = 1$ と $A = 1 \Rightarrow Y = 1$ は導き出される．つまり $F = 1 \Rightarrow X = 1$ と $F = 1 \Rightarrow Y = 1$ は間接含意としては成立するが，テストパターン生成において，有用ではない．

ここでテストパターン生成において有用である間接含意とは，当該間接含意の基点 (左辺) となる信号線に値を設定し，直接含意と当該の間接含意以外の間接含意を用いて，含意操作を行ったが，当該間接含意の終点の信号線に値が設定できない間接含意である．しかしこのテストパターン生成において有用とされる間接含意を求めることは，すべての間接含意の候補から，テストパターン生成に有用である間接含意の集合を抽出することが必要とされ，大変困難である．また回路規模が増大すると，回路規模の増大以上に得ることのできる間接含意量が増大し，テストパターン生成に間接含意を用いる上で大きな課題となっている．

3. 非循環有向グラフを用いた間接含意

本章では非循環有向含意グラフを用いた小さな間接含意の集合を求める手法について提案する．

3.1 非循環有向含意グラフ

まず含意グラフの説明を行う．信号線の含意関係は含意グラフ（Implication Graph : IG) [5] として表現できる．含意グラフは，頂点の集合 $V = \{x_1, \ldots, x_n\}$ と辺の集合 E を用いて，$G = (V, E)$ で表現される．ただし，x_i は対象となる回路の信号線と値 $v \in \{0, 1\}$ のすべての組合せから構成される．また辺 e は x_i から x_j へのパスであり，x_i の値を設定したとき，x_j の値が含意可能である場合，x_i から x_j への辺が存在する．

図 3 に示す 2 入力の NAND ゲートに対する含意グラフの表現を図 4 に示す．

図 3 では，NAND ゲートであるため，信号線 C が値 0 のときに，信号線 A と信号線 B がそれぞれ値 1 に含意される．また信号線 A または信号線 B が値 0 のとき，信号線 C は値 1 に

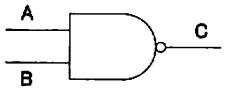

図3 NANDゲート
Fig. 3 A NAND gate

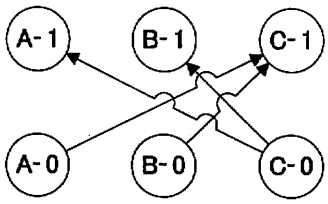

図4 含意グラフ
Fig. 4 Example of implication graph

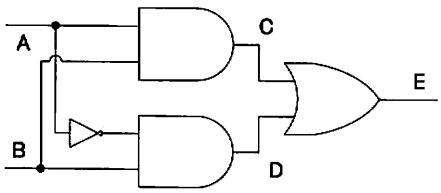

図5 等価な頂点が存在する回路
Fig. 5 Example circuit of equivalence vertex

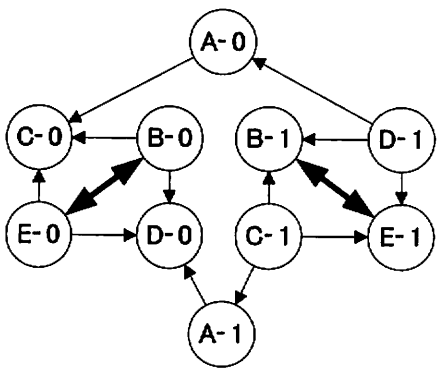

図6 等価な頂点が存在する含意グラフ
Fig. 6 Example implication graph of equivalence vertex

含意される．よって図3のゲートに対する含意グラフは図4で表現される．図4の頂点 C-0 は信号線 C が値0を割り当てられたことを示しており，同様に頂点 A-1 は，信号線 A が値1を割り当てられたことを示している．また頂点 C-0 から，頂点 A-1 への辺は，信号線 C に値0が割り当てられたとき，信号線 A に値1が含意されることを示している．

含意グラフには等価な関係を持つ頂点が存在する場合がある．例を挙げて説明を行う．図5の回路の含意グラフを図6に示す．図6では，頂点 B-1 と頂点 E-1，頂点 B-0 と頂点 E-0 との間に相互に辺を有している．これはお互いの信号線の値が等価であることを示している．

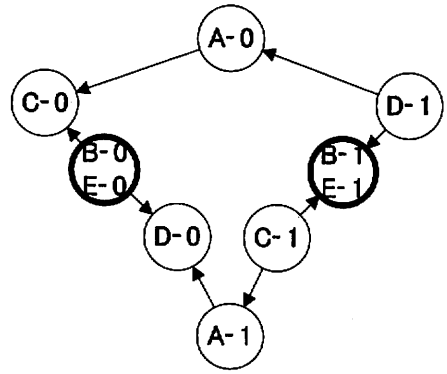

図7 一つの頂点とした含意グラフ
Fig. 7 Example implication graph of marged node

この同値の頂点を2つの要素をもつ一つの頂点として表現すると，図6は，図7に示すグラフで表される．図6での頂点 B-1 と E-1 が，図7では，頂点 B-1, E-1 として表されている．また B-0, E-0 も同様に表されている．よって含意グラフは，等価な頂点を一つの頂点とすることによって，非循環有向グラフとして表現することができる．この含意グラフを非循環有向グラフとして表現したグラフを非循環有向含意グラフとする．

3.2 非循環有向含意グラフと対偶の関係

この非循環有向含意グラフで定められた頂点と辺の関係は，含意できる範囲の大きさを示す特性を持つ．図7のグラフでは，頂点 D-1 から頂点 A-0 への辺が存在する．これは信号線 D に値1を割り当てたとき，信号線 A は値0に割り当てられる．逆に信号線 A に値0を割り当てても，信号線 D は値1に割り当てられない．よって，信号線 D に値1を割り当てたときにできる含意操作の範囲は，信号線 A に値0を割り当てたときにできる含意操作の範囲を包含する．また含意操作に基づいて得られる対偶の関係についても同様の包含の関係が成り立つ．つまり，信号線 A に値0を割り当て得られる含意操作の対偶の関係は，信号線 D に値1を割り当て得られる含意操作の対偶の関係に包含される．

逆に，先に信号線 A に値0を割りあてて得られる対偶を得ておけば，得た対偶を用いることによって，信号線 D に値1を割り当てて得られる新たな対偶を用いなくても必要な含意を行える場合がある．

つまり，非循環有向含意グラフにおいて示される頂点と頂点の関係に基づいて，含意操作の対偶を得る順序を決定することによって，より少ない対偶の関係の集合によって，必要な対偶の関係を表すことができると考えられる．

3.3 非循環有向含意グラフのレベル

より少ない対偶の関係の集合を求めるために，非循環有向含意グラフにおいて示される頂点と頂点の関係に基づいて，含意操作の対偶を得る順序を求めることを考える．本論文では，非循環有向含意グラフのレベル付けを行うことによって，順序を求める．非循環有向含意グラフに対して，以下のようにレベル付けを行う．

(1) 非循環有向含意グラフのいずれの頂点への辺を有していない頂点を求め，求めた頂点のレベルを0とする．

(2) レベルを1加算し，既にレベルが定められた頂点をグラフから除去する．いずれの頂点への辺を有していない頂点を求め，求めた頂点にレベルを設定する．

(3) すべての頂点のレベルが設定されるまで，2を繰り返す．

この手順を例を用いて説明を行う．例として図7のレベルを求める．まず手順1において，いずれの頂点への辺を有していない頂点を探す．図7では，頂点 $(C\text{-}0),(D\text{-}0),(B\text{-}1,E\text{-}1)$ が該当し，それぞれレベル0とする．次に手順2において，レベル0とした頂点を除いて，いずれの頂点への辺を有していない頂点を探す．図7では，頂点 $(A\text{-}0),(A\text{-}1),(B\text{-}0,E\text{-}0),(D\text{-}1)$ が該当し，レベル1とする．手順3では，すべての頂点のレベルが定まっていないので，手順2に戻る．手順2では，次に，レベルが定まっていない頂点を除いて，いずれの頂点への辺を有していない頂点を探す．図7では，頂点 $(C\text{-}1)$ が該当し，レベル2とする．手順3では，すべての頂点のレベルが定められたため，終了する．

3.4 小さい対偶の関係の集合を求める手順

ここで非循環有向含意グラフを用いた小さい対偶の関係の集合を求めるアルゴリズムを提案する．

(1) 回路の隣接の接続関係に基づいた非循環有向含意グラフを作成する

(2) 手順1で求めた非循環有向含意グラフに対するレベルを求める

(3) 手順2で求めたレベル0を除く，レベルの低い未処理の頂点を選択し，手順4へ進み，すべての頂点で終了した場合は終了する

(4) 手順3で選択した頂点の信号線に値を設定し，含意操作を行う

(5) 手順3で選択した頂点から到達可能な頂点の集合を求める

(6) 手順4の含意操作で得られた信号線と値の集合と手順5で求めた頂点の集合とを比較し，含意操作では求まるが，非循環有向含意グラフでは到達できない頂点の集合を求める

(7) 手順6で求めた集合が空集合であれば，手順3へ戻り，空集合でなければ，手順8へ進む

(8) 手順6で求めた集合の中で，最もレベルの高い頂点を一つ求める

(9) 手順3で選択した頂点から，手順6で求めた頂点へ含意関係を示す辺を加える．またこの辺に対する対偶の関係を間接含意の集合に加え，手順5へ戻る

ただし手順9で加えた辺によって，非循環有向含意グラフのレベルが変化した場合は，再度レベルを再度求める．また手順9で加えることによって，新たに等価な頂点が判明した場合はその場で，等価な頂点となるようグラフを変更し，レベルも再度求める．

提案したアルゴリズムを図2を用いて説明する．まず図2に対する隣接関係に基づいた非循環有向含意グラフを求める．図8に示す．

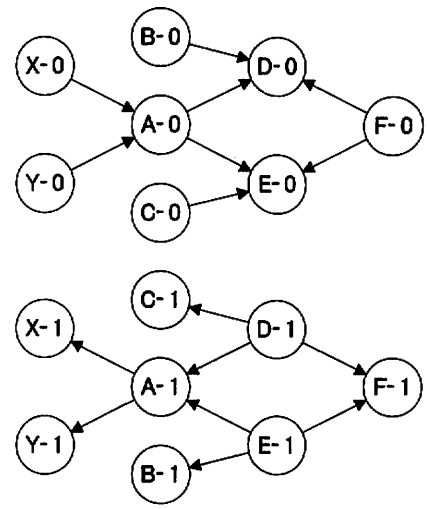

図8 非循環有向含意グラフ
Fig. 8 Directed Acyclic Implication Graph

手順2により，非循環有向含意グラフから，レベルを求める．求めた結果は次の通りとなる．

レベル0 $D\text{-}0, E\text{-}0, B\text{-}1, C\text{-}1, F\text{-}1, X\text{-}1, Y\text{-}1$

レベル1 $A\text{-}0, B\text{-}0, C\text{-}0, F\text{-}0, A\text{-}1$

レベル2 $X\text{-}0, Y\text{-}0, D\text{-}1, E\text{-}1$

手順3により，レベル1から順に頂点を選択する．まず頂点 $A\text{-}0$ を選択する．手順4により，信号線 A に値0を設定して，含意操作を行う．信号線 D, E, F が値0に含意される．手順5により，頂点 $A\text{-}0$ から到達可能な頂点を求める．頂点 $D\text{-}0, E\text{-}0$ を得る．手順6により，含意操作では求まるが，非循環有向含意グラフでは到達できない頂点の集合を求めると，頂点 $F\text{-}0$ が求められる．手順8により，頂点 $F\text{-}0$ を選択し，手順9によって，頂点 $A\text{-}0$ から頂点 $F\text{-}0$ へ新たな辺を追加する．対偶の関係として，信号線 F に値1を割り当てると，信号線 A に値1が割り当てられる関係を得ることができ，この対偶の関係を間接含意の集合に加える．またあらたに辺が追加されたため，レベルを再度計算し，頂点 $A\text{-}0$ をレベル2に，頂点 $X\text{-}0, Y\text{-}0$ をレベル3とする．手順5により，非循環有向含意グラフで再度到達可能な頂点の集合を求めると，頂点 $D\text{-}0, E\text{-}0, F\text{-}0$ を得る．これは含意操作で得られる集合と同じであるため，手順7により，頂点 $A\text{-}0$ に対する処理を終了する．次に，頂点 $B\text{-}0, C\text{-}0, F\text{-}0, A\text{-}1, D\text{-}1, E\text{-}1$ に対して処理を行う．ここでは新たな間接含意を得ることはできない．次に頂点 $X\text{-}0$ に対して，処理を行う．信号線 X に値0を設定して，含意操作を行う．信号線 A, D, E, F が値0に含意される．また頂点 $X\text{-}0$ から到達可能な頂点を求める．頂点 $A\text{-}0, D\text{-}0, E\text{-}0, F\text{-}0$ を得る．これは含意操作で得られる集合と同じであるため，頂点 $Y\text{-}0$ に対する処理を終了する．最後に頂点 $Y\text{-}0$ に対して，処理を行う．信号線 Y に値0を設定して，含意操作を行う．信号線 A, D, E, F

表 1 実験結果
Table 1 experimantal results

Cir	[3]		proposed-1		proposed-2	
	NII	CPU(s)	NII	CPU(s)	NII	CPU(s)
c432	121	0.02	79	0.01	86	0.04
c499	136	0.05	52	0.03	136	0.05
c880	78	0.02	61	0.01	74	0.01
c1355	304	0.08	368	0.04	304	0.07
c1908	948	0.09	305	0.05	835	0.08
c2670	1,239	0.09	404	0.06	1,035	0.09
c3540	3,831	0.40	846	0.23	3,191	0.38
c5315	2,454	0.18	1,262	0.11	1,531	0.19
c6288	1,051	0.11	1,741	0.06	974	0.10
c7552	8,070	0.55	2,771	0.36	4,508	0.50
b14_C	183,994	23.44	11,383	9.85	150,334	20.27
b15_C	463,769	293.76	44,462	76.15	429,669	275.80
b17_C	1,393,303	990.93	114,225	211.34	1,300,420	936.43
b20_C	251,154	34.40	28,831	14.50	215,250	31.62
b21_C	324,445	49.17	31,545	16.30	273,323	43.72
b22_C	414,191	58.06	46,972	24.91	343,646	50.78

が値 0 に含意される．また頂点 Y-0 から到達可能な頂点を求める．頂点 A-0, D-0, E-0, F-0 を得る．これは含意操作で得られる集合と同じであるため，頂点 Y-0 に対する処理を終了する．

これらの手順に示されるとおり，頂点 A-0 からを先に間接含意を求め，非循環有向含意グラフと比較することによって，頂点 X-0, Y-0 からの間接含意をテストパターン生成において有用でないと判断することができ，より小さい間接含意の集合を得ることができる．

4. 実験結果

3 章で述べた提案方法 (proposed-1) を用いて ISCAS'85 ベンチマーク回路と ITC'99 ベンチマーク回路 [6] に対して，実験を行った．比較対象として，文献 [3] で提案されている手法との比較を行った．また文献 [3] の手法に間接含意のみに着目して，包含される不要な間接含意を除去する手法 (proposed-2) に対しても比較を行った．実験には CPU が Pentium IV 3.05GHz, メモリが 1GB, OS が Linux である計算機を用いて行った．

提案手法に対するの実験結果である．Cir は回路名，NII 間接含意の集合の数，Time は CPU 時間 (秒) を，[3], proposed-1, proposed-2 はそれぞれの手法に対するを示す．

表 1 に示すとおり，非循環有向含意グラフを用いることによって，平均 43%の間接含意の集合を削減することができた．また比較的規模の大きな ITC99 ベンチマーク回路においても，処理時間が 211 秒と十分に短い時間で探索することができた．また proposed-2 では，処理時間も従来手法より短縮した上で，平均 16%の間接含意の集合を削減することができた．

5. まとめ

本論文では，含意グラフを拡張した非循環有向含意グラフを用いた間接含意の学習方法を提案した．提案したアルゴリズムによって，従来の手法や，間接含意同士の包含関係を調べる手法と比較して，より小さな間接含意の集合を得ることができた．

今後の課題としては，一部のベンチマーク回路 (c1355, c6288) で発生した提案手法の結果が悪い結果に対する解析と，さらに大きな回路に対する検証があげられる．

謝辞 本研究は，福岡地域の文部科学省知的クラスター創成事業の支援による．

文献

[1] H.Fujiwara. *Logic Testing and Design for Testability.* The MIT Press, 1985.

[2] M. Abramovici, M. A. Breuer, and A. D. Friedman. *Digital Systems Testing and Testable Design.* Computer Science Press, 1990.

[3] E. Trischler M. H. Shulz and T. M. Sarfert. Socrates: A highly effecient automatic test pattern generation system. *IEEE Trans. on Computer-Aided Design*, Vol. 7, No. 1, pp. 126–137, 1988.

[4] S. Kajihara H. Ichihara and K. Kinoshita. On processing order for obtaining implication relations in static learning. *IEICE Transaction on Information and Systems*, Vol. E83-D, No. 10, pp. 1908–1911, October 2000.

[5] V. D. Agrawal S. T. Chakradhar and S. G. Rothweiler. A transitive closure algorithm for test generation. *IEEE Trans. on Computer-Aided Design*, Vol. 12, No. 7, pp. 1015–1028, 1993.

[6] S.Davidson. ITC'99 Benchmark Circuits - Preliminary results. In *Proceedings of International Test Conference*, p. 1125, 1999.