論　文

部分回路除去に対する含意関係の不変性について

市原　英行[†]　　梶原　誠司[††]　　樹下　行三[†]

On Invaliant Implication Relataions for Removing Partial Circuits

Hideyuki ICHIHARA[†], Seiji KAJIHARA[††], and Kozo KINOSHITA[†]

あらまし　大規模論理回路の簡単化のためにはテスト生成手法を応用することが有効であるが，そこではテスト生成アルゴリズムのSOCRATESで用いられる静的学習で得られる回路内部の信号線の信号値の含意関係を利用することが重要な役割を占めている．しかしながら，信号値の含意関係は回路構造に依存するため，多段階に回路変換を繰り返す場合には，変換ごとに含意関係を求め直す必要があった．本論文では，回路内の一部のゲートや信号線を除去するような回路変換において，変換前の回路に対する静的学習で得られた含意関係のうち，どのようなものが変換後の回路で無効になるかを考察する．次に冗長除去において，静的学習をやり直さなくても回路変換前後で不変である含意関係を冗長判定に用いて，冗長除去を効率化する手法を提案する．ベンチマーク回路に対する実験では，従来手法と比較して，本手法により約60倍の高速化した例が存在した．

キーワード　テスト生成，静的学習，論理合成，含意関係，冗長除去

1. まえがき

これまでに論理回路のテスト生成手法に関する研究は数多く行われており，特に組合せ回路に対しては高速な手法が提案され大きな成果を上げている[1]～[3]．中でもSOCRATES[3]では，静的学習(static learning)と呼ばれる処理において，前もって個々のゲートの入出力関係だけからでは求まらないような信号線の信号値の含意関係を学習し，それらをテスト生成中の含意操作に用いることで，テスト生成の高速化に役立てている．静的学習の導入により，一意的に決まる信号値を多く求めることができ，従来とは比較にならないほど少ないバックトラック回数で高速に冗長故障を指摘することができるようになった．これらのテスト生成手法は大規模回路に対しても有効であり，近年では，こうしたテスト生成手法を大規模論理回路の論理合成に応用する研究がなされている[4]～[12]．

これまでに，縮退故障のテスト生成において，故障の検出可能性から回路の冗長性を判断し，その冗長性を除去することで回路を簡単化できること[4]～[7]，更に非冗長な回路であっても，冗長な回路を付加することによりそれまで非冗長であった箇所を冗長にし，それらを取り除くことにより更に回路を小さくできることが示されている[8]～[12]．この際の付加される冗長な回路は，SOCRATES[3]の学習操作に基づいて回路内部の信号の含意関係を調べることにより探し出すことができ[8],[10]，また，その後新たに冗長になった箇所を見つける処理にも静的学習の結果を利用することができる．このように，静的学習で回路内部の信号の含意関係を求め，それらを利用することは，テスト生成において重要な役割を果たしており，冗長除去や冗長の付加と除去の繰返しによる回路簡単化にも応用できる．

静的学習が一般のテスト生成に用いられる場合には，前処理として一度だけ行えばよいため，総処理時間のうち静的学習に費やされる時間は少ない．しかしながら含意関係は回路構造に依存するため回路の一部を変換すると，変換前の回路に対しては成立した含意関係が回路変換後には成立しなくなる可能性があり，これら成立しなくなった含意関係を用いるなら，テスト生成や論理合成で誤った結果が得られる．そのため従来の方法では，回路を変換するごとに，変換前の回路で得られた含意関係を一度すべて破棄し，新たに静的学習を行っていた．従って，何度も回路変換を行う場合には，静的学習が何度も繰り返され，特に規模が

† 大阪大学大学院工学研究科応用物理学専攻，吹田市
Dept. of Applied Physics, Osaka University, Suita-shi, 565 Japan
†† 九州工業大学情報工学部電子情報工学科，飯塚市
Dept. of Computer Sciences and Electronics, Kyushu Institute of Technology, 820 Japan

大きな回路になると，静的学習に多くの時間が費やされることになる．

本論文では，回路内の一部のゲートや信号線を除去するような回路変換において，変換前の回路に対する静的学習で得られた含意関係の不変性について考察し，部分回路除去で含意関係が無効となるための必要条件を示す．次に，その手法を冗長除去による回路簡単化に応用した手法を提案する．一度に除去される信号線やゲートの数は一般的に少ないため，無効となる含意関係も少ないと考えられる．このため，冗長除去では，最初に静的学習で得られた含意関係の中から無効になる可能性があるものを取り除き，確実に不変なものだけを用いても，冗長判定は十分高速に行えると考えられる．すなわち，利用できる含意関係が少なくなるが，一から静的学習を行わないことで計算時間を短縮できる．

以下，本論文は次のように構成される．**2.** では，本論文に用いる用語の定義を行い，静的学習を部分回路変換に用いるときの問題点について述べる．**3.** では，回路変換により変換前の含意関係が成り立たなくなるのは，どのような場合に起きうるかを考察し，その必要条件を示す．**4.** では，静的学習の回数を少なくすることで冗長除去の処理を効率化する手法とベンチマーク回路に対する実験結果を示し，**5.** で本論文のまとめを述べる．

2. 準　　備

2.1 諸 定 義

まず本論文で用いる語句を定義する．なお，本論文では，NOT，AND，NAND，OR，NOR，EXOR ゲートで構成された組合せ回路を扱うものとする．

[定義1　制御値]　ゲートの一つの入力値だけにより出力値が一意に決まるとき，その入力値を制御値と呼ぶ．AND, NAND ゲートの場合には 0，OR, NOR ゲートの場合には 1 である．EXOR ゲートには，制御値は存在しない．また，制御値でない信号値を非制御値と呼ぶ．

SOCRATES で行われる静的学習 (static learning) は，個々のゲートの入出力関係からは直接的に求まらないような信号値の含意関係を，テスト生成に前もって抽出する操作である [3]．以降，本論文では，含意関係を単に含意と記述する．静的学習により得られる含意は間接含意 (indirect implication) と呼ばれ，単にゲートの入出力関係とゲートのつながり方からわかる含意は直接含意 (direct implication) と呼ばれる[8]．例えば，図1に示した回路では，間接含意 "$d=1$ ならば $a=1$" が得られる．これは，$a=0$ からの含意操作 (implication procedure) によって得られた直接含意 "$a=0$ ならば $d=0$" の対偶である．なお本論文中では，含意操作はある信号値が決まったことから一意に決まる信号値を割り当てる操作のことである．また，ある含意 "$a=0$ ならば $d=0$" は，以後，"$a=0 \Rightarrow d=0$" と記述することにする．間接含意は，テスト生成の含意操作において信号線 d に 1 を割り当てるときに使用され，一意的に決定する信号値をより多く求めることに役立っている．

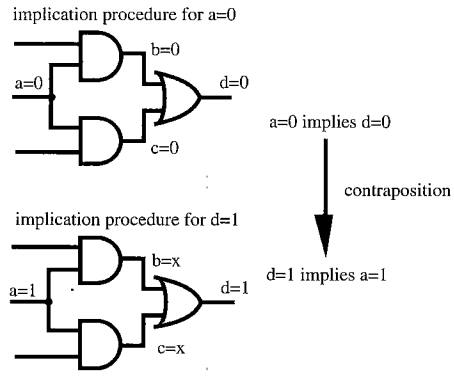


図1　静 的 学 習
Fig. 1　Static learning.

静的学習では，信号線 i の値を V_i（以下，$V_x \in \{0, 1\}$）として含意操作を行い，その結果，ゲート G の出力信号線 j の値が V_j となったとき，"$j=\overline{V}_j \Rightarrow i=\overline{V}_i$" という信号値の含意を記憶するものである．間接含意として記憶する含意は，以下に示す学習基準 (learning criterion) を同時に満たすときであり，それ以外の信号線の値は，$j=\overline{V}_j$ の含意操作から直接得られるので，記憶する必要はない[3].

[学習基準]

(1) ゲート G のすべての入力値が非制御値である．

(2) 外部出力方向への含意操作が信号値 V_j の決定に関与している．

2.2 部分回路変換と間接含意

静的学習により得られた間接含意は，回路の一部を除去することにより成立しなくなる場合がある．例えば与えられた回路が図2 (a) のような回路を含んでいるとする．この回路に対して，"$a=0 \Rightarrow h=0$"という直接含意から，"$h=1 \Rightarrow a=1$"という間接含意が得られる．今仮に，ゲート $G1$ とその入出力 $a1, b, e$ を除去し

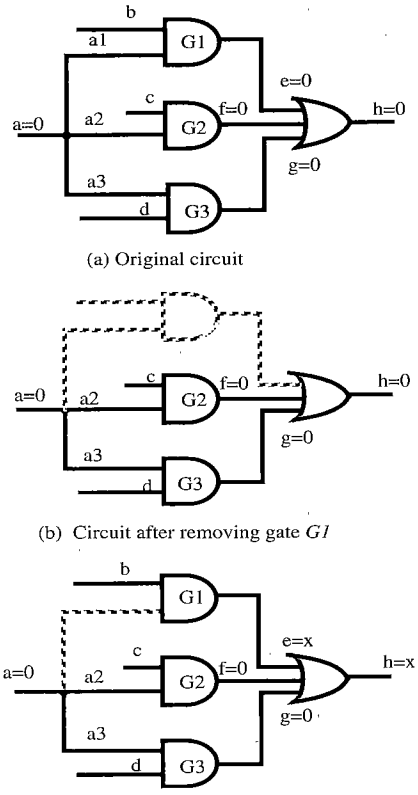
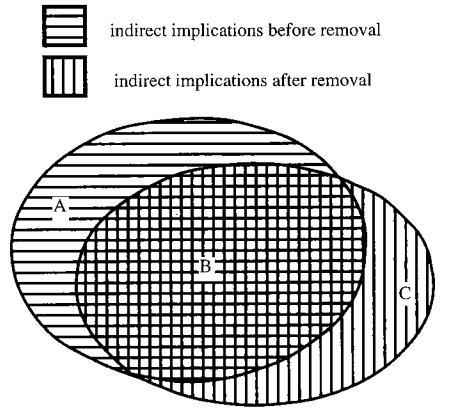


図2　部分回路除去に対する含意関係の不変性
Fig. 2　Invariability of implication under removing a partial circuit.

図2(b)のような回路が得られたとする．この回路に対しても，もとの回路で得られた間接含意 "$h=1 \Rightarrow a=1$" は成立する．しかしながら，信号線 *a1* のみを除去したとすると図2(c)のような回路が得られ，この回路に対しては，"$h=1 \Rightarrow a=1$" という間接含意は成立しなくなる．ここでは例を示さないが，部分回路除去により新たに間接含意が得られる場合があることを記しておく．

図3に，ある部分回路除去の前後で得られる間接含意の関係を示す．A は回路変換によって無効となる間接含意の集合を，B は回路変換後も成り立つ間接含意の集合を，C は回路変換後の回路に対し新たに得られた間接含意の集合を表す．すなわち，$A \cup B$ が回路変換前に有効な間接含意の集合で，$B \cup C$ が回路変換後に有効な間接含意の集合である．部分回路除去を繰り返すときは，回路変換後成り立たなくなった間接含意は変換後の回路において使用できないため，$A \cup B$ の間接含意を一度破棄し，改めて静的学習を行い $B \cup C$ の間接含意を学習し直すことが必要となる．



A: invalidated indirect implications for removing a partial circuit
B: invariant indirect implications for removing a partial circuit
C: indirect implications obtained newly after removing a partial circuit

図3　回路除去前後での間接含意
Fig. 3　Indirect implications before and after removing a partial circuit.

しかし1回の処理で除去される信号線やゲートの数が，数本または数個と少ない場合には，A に含まれる間接含意は少ないと考えられる．これは，多くの間接含意は変換後の回路においても有効であるにもかかわらず，それらを破棄して同じ間接含意を学習することになり，部分回路除去を繰り返せば計算時間が増加する大きな原因の一つになる．

本論文では，このような回路の除去において間接含意の再学習を行わず，A の間接含意を完全に取り除き，残った B の間接含意のみを以後の処理に用いることを試みる．C の間接含意もまた A の間接含意と同様に少ない可能性が高いため，B の間接含意だけでも引き続き行う処理に十分対応できる．静的学習のやり直しを避けることにより，計算時間の短縮が期待できる．B の間接含意だけでは不十分な場合には，そのとき初めて静的学習をやり直せばよい．

次章では，図3の A の部分に相当する無効となる間接含意がどのように生じるかを考察し，それらを取り除く手法について述べる．なお本手法は，無効となる間接含意が生じるための必要条件のみ（必要十分条件ではない）を考慮するため，図3の B の部分に相当する無効にならない間接含意も除去される可能性があるが，A の無効になる間接含意は必ず除去される．

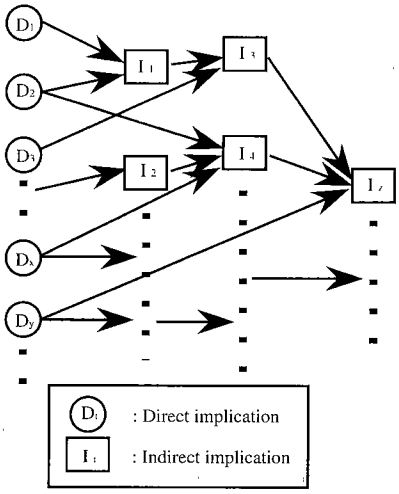
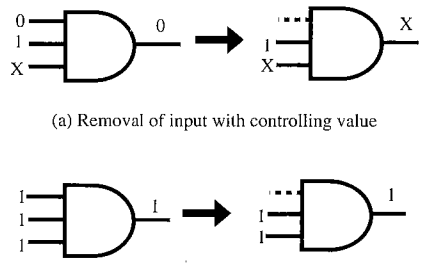


図4 静的学習に使用した含意関係
Fig. 4 Implications used in static learning.



(a) Removal of input with controlling value

(b) Removal of input with non-controlling value

図5 入力の除去と出力値の変化
Fig. 5 Removing input and change of assigned value of output.

3. 部分回路除去における間接含意の不変性

3.1 学習関与直接含意

間接含意は含意操作の結果として得られるので，回路の一部を除去したときの間接含意の不変性は，その間接含意を学習するときに用いた含意の不変性を考慮すればよい．学習するときに用いられる含意は直接含意だけでなく，既に学習された間接含意も含まれる．このため，間接含意は，直接含意だけ使って学習したものと，直接含意と間接含意の両方を使って学習したものとの2種類に分けることができる．

図4は静的学習において間接含意を得るために行った含意操作で用いた含意の関係を有向グラフで示したものである．節点 D_i と I_i は，それぞれ直接含意と間接含意を示す．有向辺の始点となる含意は有向辺の終点の間接含意を得るのに必要とした含意である．例えば，間接含意 I_3 は直接含意 D_3 と間接含意 I_1 を用いて学習したことを示している．なお直接含意は，一つの論理ゲートの入出力間での含意（以下，ゲートでの含意），と分岐点の幹と枝の間での含意（以下，分岐点での含意）に分解して表現できるため，D_i はそのどちらかを表す．また，ゲートの複数の入力が制御値をもつ場合のように，複数の含意が重複して信号値を決めることもあるが，本手法では含意操作において最初に適用した含意のみを扱う．

このように各間接含意を求めるときに用いた含意は，ループのない有向グラフで表現でき，どの間接含意に至るパスも必ず直接含意から始まることになる．ここで，間接含意 I_i に到達可能な直接含意を I_i の学習関与直接含意と呼ぶことにする．このとき，このグラフの性質から，部分回路除去の際の間接含意の不変性に関して次の補題が成り立つ．

[補題1] 部分回路除去により間接含意 $I_i: a = V_a \Rightarrow b = V_b$ が無効となるための必要条件は，以下の二つの条件のいずれかが成り立つことである．

(1) I_i の学習関与直接含意のうち少なくとも一つが無効となる．

(2) 信号線 a, b のいずれかが除去される．

（証明）(1)は，対偶を考察することより自明であり，(2)も，信号線 a または b がなくなれば，間接含意 I_i そのものが無意味になるので明らかである．

例えば，I_3 が無効になるとき，学習関与直接含意 D_1, D_2, D_3 の内少なくとも一つが無効であるか，または，I_3 を表現する信号線が存在しないことになる．

上記(2)の条件を満たす間接含意を見つけることは容易であるので，以下では，(1)の条件に関して，無効となる直接含意について考察する．

ある直接含意が回路の除去前と除去後で異なるとすれば，その原因は，除去された部分と残った部分の境界部分（入力の一部が除去されたゲート，または一部の分岐枝が除去された分岐点）における含意操作で，異なる信号値が生じることにある．これは，回路除去とは無関係な範囲内でしか含意操作が行われないなら，除去前後で決して異なる信号値が生じないので，明らかである．つまり，入力の一部が除去されたゲートか一部の分岐枝が除去された分岐点において，除去前の回路で用いた直接含意 $j = V_j \Rightarrow i = V_i$ のうち，信号線 j の除去により，除去されずに残った信号線 i の値を V_i にできなくなる含意が含まれていれば，V_i の

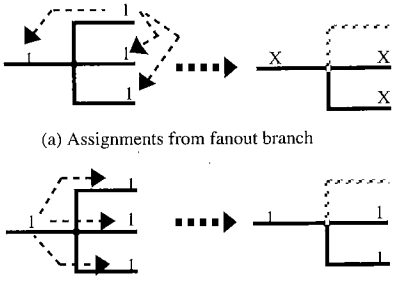
値の違いが他に波及して，含意操作の結果が異なってくる．このように，除去された回路とされなかった回路で境界部分の直接含意が異なる場合を判断することにより，無効となる直接含意はすべて見つけることができる．

3.2 論理ゲートの入力の除去と含意関係

直接含意は，ゲートでの含意と分岐点での含意の組合せで表現できる．本節では，あるゲートの入力を除去したとき，そのゲートでの除去前の含意が無効になる場合を考える．なお，ゲートの出力を除去する場合はゲートそのものがなくなるため考える必要がない．

ゲートの入力が決まることにより出力が決まる含意は，入力値が制御値を含む場合と，すべて非制御値である場合に分けることができる．除去によりあるゲートでの含意が無効になるか否かは，除去した信号線で制御値と非制御値のどちらの含意を考えているかによる．図5にANDゲートの入力を除去する例を示す．左図が除去前に成り立っていた含意を示していて，右図がその含意が入力除去後に成立するか否かを示している．含意が成立しなくなるのは図5(a)のような場合である．除去前の回路で考えている含意は，除去した信号線の値'0'（制御値）により出力値が決まる含意であり，残りの入力の信号値は出力値の決定にには関与していない．よって，変換後の回路ではこの含意は無効となる．図5(b)は，除去したゲートの入力に非制御値である'1'を与える含意を考えた場合である．このときは残りのゲートの入力値が出力値をそのままに保つため，入力の除去に対して含意は不変である．よって次の補題が明らかに成り立つ．

[補題2] 2入力以上もつゲートGの入力の集合を$I=\{I_1, \cdots, I_m\}$，ある部分回路除去により除去されたGの入力の集合を$RI \subset I$，残った入力の集合を$NRI = I - RI$とする．また，RIは空集合ではないとする．この部分回路除去により無効となるGでの直接含意は，GがEXORゲート以外のとき，Gの出力値が除去前の回路で入力$la\,(la \in RI)$での制御値により決まる含意である．またGがEXORゲートの時は，$la\,(la \in RI)$での信号値にかかわらず無効となる．



(a) Assignments from fanout branch

(b) Assignments from fanout stem

図6 分岐の枝の除去と信号値の変化
Fig. 6 Removal of fanout branch and change of assigned value.

複数のゲート入力に制御値が割り当てられ，そのうち一部の入力が除去される場合，ゲートの出力は除去後も同じ値をとることができる．しかし，本手法では，ゲートの出力を決めるため最初に適用した含意のみを扱うため，無効とした含意には含意操作に影響しないものを含む場合がある．また，EXORゲートは，定義からは制御値はもたないが，入力信号線の除去に出力値が影響をうけるため，変換後の回路ではそのゲートでの含意は無効となる．

3.3 分岐の枝の除去と含意関係

次に，分岐の枝の除去が分岐点での含意に与える影響を考える．なお分岐の幹を除去した場合は，その分岐点自体がなくなるため考える必要はない．

分岐点における含意にはそれぞれ図6(a)と図6(b)のように，前方から後方への含意と後方から前方への含意がある．図6(a)は最初に分岐の枝の信号値が決まり，それから幹や残りの分岐の枝の信号値が決まるような，前方から後方への含意を示している．この含意は，最初に信号値の決まった分岐の枝を除去した分岐点では成り立たないことになる．すなわち，このような含意は回路変換後は無効となる．図6(b)は分岐の幹の値が最初に決まり，それから分岐の枝が決まるような，後方から前方への含意を示している．この含意は，分岐の枝のうちの1本を除去した分岐点でも成り立つ．これより次の補題が明らかに成り立つ．

[補題3] ある分岐点FPにおいて，分岐の幹をst，枝の集合を$BR=\{br_1, ..., br_m\}$とし，BRの一部を除去する回路変換により除去された分岐の枝を$RBR \subset BR$，残った枝を$NRBR=BR-RBR$とする．$br_a\,(br_a \in RBR)$からst，$br_b\,(br_b \in NRBR)$への含意は，変換後の回路では無効となる．

3.4 無効となる可能性をもつ間接含意

以上の議論に基づいて，部分回路除去により無効となる可能性をもつ間接含意の条件を具体的に示す．これは除去した信号線に対し，ある間接含意を得るときに行った含意操作で，変換前の回路においてどのような信号値を与えていたかを調べることで判断できる．なお，**3.1**で述べたように，除去したすべての信号線に

ついて調べる必要はなく，除去した信号線のうち，回路に残っているゲートまたは信号線に接続していたものだけである．これらの信号線を境界信号線と呼ぶことにすると，境界信号線のうち出力側にあるものは，必ず，出力が残っているゲートの除去された入力信号線であり，また，入力側にある境界信号線は，必ず，幹が残っている分岐の除去された枝となる．

ある間接含意を学習するために行った含意操作において，以下の条件のうち一つでも満たせば，その間接含意は無効となる可能性をもつ．

条件(1)：出力側の境界信号線に制御信号値を割り当てたとき．

条件(2)：EXOR ゲートの入力である境界信号線に信号値が割り当てられたとき．

条件(3)：入力側の境界信号線に，同じ分岐点の幹や他の枝より先に信号値を割り当てたとき．

条件(4)：含意操作が無効であると判断された間接含意を用いたものであるとき．

条件(5)：間接含意に示す信号線を除去したとき．

条件 (1) と条件 (2) は補題 2 から，条件 (3) は補題 3 から，条件 (4) は**3.1** での考察から，そして，条件 (5) は補題 1 から得られる．

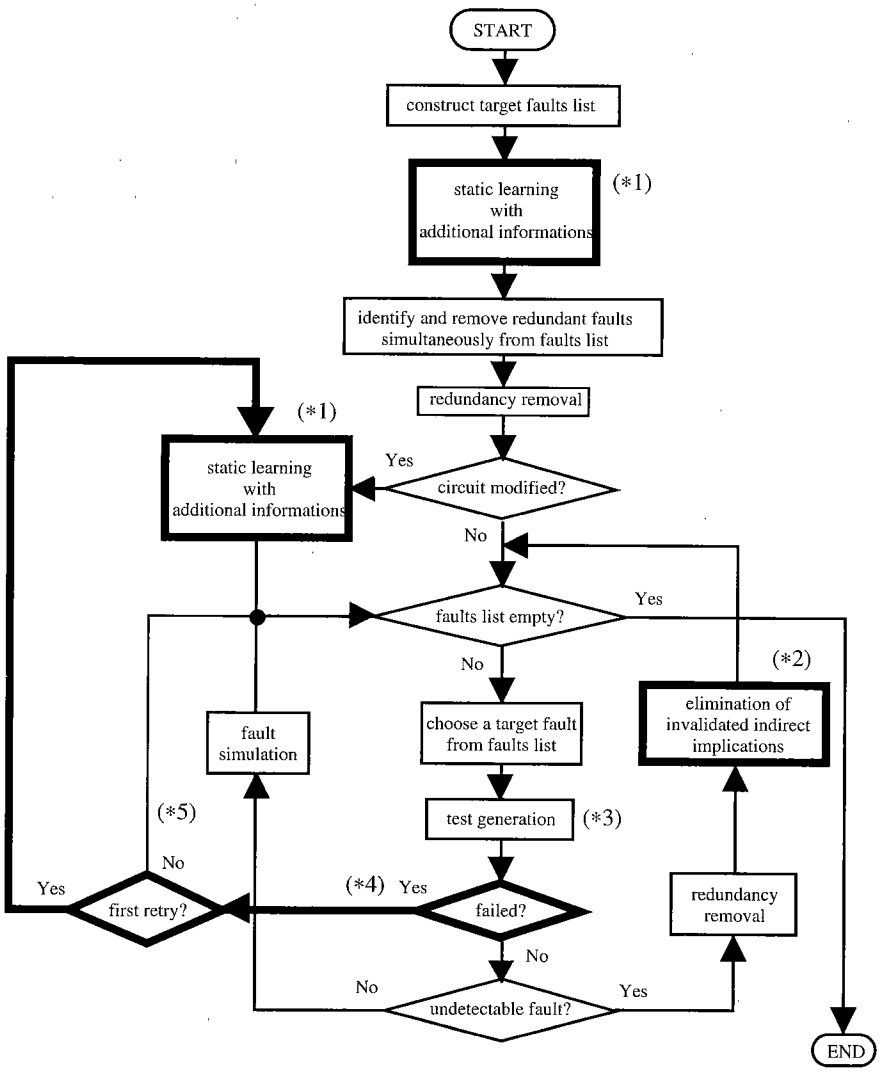


図7　冗長除去のアルゴリズム
Fig. 7　Proposed algorithm for redundancy removal.

4. 不変な間接含意を用いた冗長除去手法

4.1 アルゴリズム

ここでは，前章で得られた方法を回路の組合せ回路的冗長の除去に適用する場合について述べる．回路の冗長除去では，一つの冗長故障を取り除くと他の冗長故障が検出可能になる場合や，検出可能な故障が冗長故障になる場合があるため，テスト生成を繰り返すことが求められており [4]～[6]，そのため静的学習も繰り返し行われる．そこで，前節で得られた手法を用いて，無効となる間接含意を取り除くことで，静的学習を繰り返し行わない手法を提案する．

図7にアルゴリズムを示す．このアルゴリズムは，検出不能故障のクラス化による組合せ回路の冗長除去の方法 [5] を基本としたものであり，太い線で囲った部分が本手法で新たに付加した部分となる．静的学習時には無効となる可能性をもつ間接含意破棄手法の前処理として，図4のように学習時に必要とした含意間の関係を示すグラフに相当する情報の記憶を行っている(*1)．具体的には，回路内の全ての信号線に対してそれぞれの信号線が除去されたときに，**3.4** の条件(1)～(5)により無効となる可能性をもつと判断できるすべての間接含意を記憶している．

本手法では無効となる可能性をもつ間接含意の破棄(*2)だけを考えているため，変換前の回路では得ることができなかったが変換後の回路では得ることができるはずの間接含意（図3のCの部分に相当）を得ることができない．このため，冗長性判定の能力は，静的学習を回路変換の度に繰り返す手法と比べて落ちると考えられる．このアルゴリズムでは，一つの故障に対するテスト生成 (*3) において，3回以内のバックトラックで生成できなかった場合，その故障に対するテスト生成は失敗したとして(*4)，静的学習をやり直している．なお一度静的学習をやり直したにもかかわらずテスト生成に失敗する故障については，本手法で用いたテスト生成アルゴリズムでは判定できない故障なので，2度以上の静的学習のやり直しは行わない(*5)．

4.2 実験結果

上記のアルゴリズムを富士通 S-4/LC ワークステーション上にC言語を用いてプログラム化し，ISCAS'85のベンチマーク回路 [15] とISCAS'89のベンチマーク回路 [16] の組合せ回路部分を用いて実験を行った．本手法の有効性を示すため，静的学習を回路変換のたびにやり直す冗長除去の手法 [5] と比較し，その結果を表1に示す．表中の “previous method” の欄と“proposed method” の欄にはそれぞれ，[5] の手法による処理時間と提案手法による処理時間を示している．また，“static learning” の“repetition” と “time” の欄にはそれぞれの手

表1 ベンチマーク回路に対する実験結果
Table 1 Experimental results for benchmark circuits.

cirsuit	previous method (sec)	static learning repetition	static learning time (sec)	proposed method (sec)	static learning repetition	static learning time (sec)	acceleration (prev./prop.)
c432	3	5	0.1	3	1	0.2	1.0
c499	4	9	0.1	3	1	0.3	1.3
c880	3	1	0.1	3	1	0.5	1.0
c1355	20	9	0.5	17	1	1.3	1.2
c1908	17	9	0.5	16	2	1.4	1.1
c2670	31	41	0.6	25	2	2.2	1.2
c3540	95	37	1.9	51	2	5.1	1.9
c5315	124	58	1.1	81	2	5.8	1.5
c6288	68	3	0.9	91	1	8.0	0.7
c7552	786	139	2.8	437	2	12.0	1.8
s9234	3445	169	4.6	2887	2	17.2	1.2
s13207	4956	383	13.2	917	2	40.8	5.4
s15850	2806	294	8.3	725	2	39.3	3.9
s35932	73635	657	77.3	1221	1	278.0	60.3
s38417	2995	70	9.1	2898	2	159.4	1.0
s38584	27875	254	110.6	4670	2	354.5	6.1

法において静的学習が行われた回数と1回目の静的学習に費やした時間を示している．すなわち“repetition”と“time”に示す値の積が，それぞれの手法の処理時間のうち静的学習に費やした時間とほぼ等しいことになる．提案手法では，付加情報の記憶のため1回に必要な静的学習の時間が増えるものの，静的学習の回数を減らすことで，手法[5]の静的学習に必要な時間を短縮することが実現できているのがわかる．また“acceleration”の欄には，提案手法が手法[5]と比較して何倍高速になったかを示している．

規模が小さい回路では，1回の静的学習に必要な時間が短く，また，静的学習の繰返し回数が少ないため，本手法の有効性はあまり認められない．逆に，1回の静的学習に必要な時間が長く，また，静的学習の繰返し回数が多い回路(s13207, s15850, s35932, s38584)では，本手法の有効性が顕著に現れている．特に，s35932では約60倍もの高速化が見られた．唯一，c6288に対しては，本手法が従来手法より多くの時間を要しているが，これは間接含意の有効性を判断するときに必要な付加情報の計算と記憶のため，従来法と比べて，静的学習により多くの時間を費やしたことが理由として考えられる．

最後に，結果として示していないが，提案手法と従来手法では，除去される故障の順序が異なる場合があるため，結果として得られる回路は違うものになる可能性があるが，除去された信号線数やゲート数の差はどの回路においても非常に小さく，この観点から手法の優劣を論ずることはできなかった．また，提案手法において，新たに生じる間接含意をつけ加えていないためテスト生成に失敗して，再学習を必要とした回路は一つもなかった．

5. むすび

本論文では，部分回路除去に対する間接含意の不変性について考察し，静的学習のやり直しを行わない冗長除去手法を提案した．また，ベンチマーク回路に対する実験では，従来法と比較することで，本手法の有効性を確かめることができた．本論文では，回路の除去する場合のみを対象としたが，含意関係をもとに回路を付加しながら回路変換を行う手法に対しても，同様の考察を行い効率化することが考えられるが，それは今後の課題である．

文　献

[1] P. Goel, “An implicit enumeration algorithm to generate tests for combinational logic circuits,” IEEE Trans. Comput., vol.C-30, no.3, pp. 215-222, March 1981.

[2] H. Fujihara and T. Shimono, “On the acceleration of test generation algorithms,” IEEE Trans. Comput., vol.C-32, pp. 1137-1144, Dec. 1983.

[3] M. H. Schultz, E. Trischler, and T. Sarfert, “SOCRATES: A highly efficient automatic test pattern generation system,” IEEE Trans. CAD., vol.7, pp. 126-137, Jan. 1988.

[4] D. Bryan, F. Brglez, and R. Lisanke, “Redundancy identification and removal,” MCNC Workshop on Logic Synthesis, May 1989.

[5] S. Kajihara, H. Shiba, and K. Kinoshita, “Removal of redundancy in logic circuits under classification of undetectable faults,” 22th Int'l Sympo. on Fault-Tolerant Comput., pp. 263-270, July 1992.

[6] M. Abramovici and M. A. Iyer, “One-pass redundancy identification and removal,” Proc. Int'l Test Conf., pp. 807-815, Sept. 1992.

[7] R. Jacoby, P. Moceyunas, H. Cho, and G. Hachtel, “New ATPG techniques for logic optimization,” Int'l Conf. on CAD, pp. 548-551, Nov. 1989.

[8] W. Kurz, P. R. Menon, “Multi-level logic optimization by implication analysis,” Int'l Conf. on CAD, Nov. 1994.

[9] K. T. Cheng and L. A. Entrrena, “Multi-level logic optimization by addition and removal,” European Conf. on Design Automation, pp. 373-377, Feb. 1993.

[10] C. L. Berman and L. H. Trevillyan, “Global flow optimization in automatic logic design”, IEEE Trans. CAD 10, pp. 557-564, May 1991.

[11] M. Yaguchi, Y. Nakamura, K. Wakabayashi, and T. Fujita, “Multi-level logic minimization based on multi-signal implications”, Proc. DAC'95, pp. 658-662, June 1995.

[12] S. C. Chang amd M. M. Sadowska, “Perturb and simplyfy: Multi-level boolean network optimizer”, Int'l Conf. on CAD, pp. 2-5, Nov. 1994.

[13] F. Breglez and H. Fujiwara, “A neutral netlist of 10 combinational benchmark circuits and a target translator in fortran,” IEEE Int'l Sympo. on Circuits and Systems; Special Session on ATPG and Fault Simulation, pp. 663-698, June 1985.

[14] F. Breglez, D. Bryan, and K. Kozminski, “Combinational profiles of sequential benchmark circuits,” Proc. IEEE Int'l Sympo. on Circuits and Systems, pp. 1929-1934, May 1989.

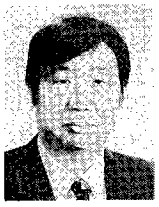
（平成8年3月11日受付，7月12日再受付）

市原　英行

平7阪大・工・応用物理卒．現在同大大学院工学研究科応用物理学専攻博士前期課程在学中．テスト生成に関する研究に従事．

梶原　誠司　（正員）

昭62広島大・総合科学・総合科学卒，平1同大大学院工学研究科情報工学専攻博士課程前期了，平4大阪大学大学院工学研究科応用物理学専攻博士後期課程了．博士（工学）．同年大阪大学工学部応用物理学科助手，平8九州工業大学情報工学部助教授．論理回路のテスト生成，テスト容易化設計などの研究に従事．情報処理学会，IEEE各会員．



樹下　行三　（正員）

昭34阪大・工・通信工卒．昭39同大大学院工学研究科通信工学専攻博士課程了．工博．同年同大工学部電子工学科助手，昭41同学科助教授，昭53広島大学教授，平1より大阪大学工学部教授．論理回路およびメモリのテスト容易化設計，故障診断，テスト生成などの研究に従事．昭62～平1本会FTS研究会専門委員長，平4IEEE第1回ATS実行委員長など．情報処理学会会員，IEEEフェロー．