等価検証の効率化に関する一考察

松永, 裕介
九州大学大学院システム情報科学研究院

出版情報：電子情報通信学会技術研究報告, CPSY2002-63(2002-11), pp.73-78, 2002-11. 電子情報通信学会CPSY研究会
バージョン：
権利関係：

等価検証の効率化手法に関する一考察

松永　裕介†

† 九州大学大学院システム情報科学研究院
情報工学部門
〒 816–8580 福岡県春日市春日公園 6–1

E-mail: †matsunaga@slrc.kyushu-u.ac.jp

あらまし　組合わせ回路の等価検証問題は $\mathcal{NP}$ 完全問題であることが知られているが，実際の設計においては構造の似通った 2 つの回路が等価かどうかを比較する場合が多く見られる．本稿ではそのような構造の類似性を利用して高速に等価性判定を行なうアルゴリズムについて述べ，前向き走査と後ろ向き走査という効率化手法に関する理論的な考察および実験結果の比較解析を行なう．

キーワード　形式的検証，等価性検証，二分決定グラフ

On acceleration methods of equivalence checking

Yusuke MATSUNAGA†

† Department of Computer Science and Communication Engineering
Graduate School of Information Science and Electrical Engineering
Kyushu University
6–1 Kasuga Koen, Kasuga, Fukuoka, 816-8580, Japan

E-mail: †matsunaga@slrc.kyushu-u.ac.jp

Abstract　While equivalence checking for combinational circuits is known as an $\mathcal{NP}$ complete problem, there are many cases in the real-world that two circuits to be verified have structural similarity. This paper describes equivalence checking algorithms utilizing such structural similarity. Theoretical study and empirical analysis for two acceleration methods — forward scan and backward scan — are shown.

Key words　formal verification, equivalence checking, BDDs

1. はじめに

今日の一般の LSI 設計では RTL もしくはそれよりも高位のレベルのハードウェア記述から合成を行なう設計手法が主流となっているため，ゲートレベル (論理レベル) の回路を直接，設計する機会は少なくなっている．とはいえ，細部のタイミング調整や負荷の調整などの理由で RTL 合成後のネットリストを変更する場合も残っており，そのような場合には，回路変更が変更前の回路の機能と同一の機能を持っていることを保証する等価検証が重要な技術となっている．また，汎用の EDA ツールを用いないカスタム設計の場合には，合成の間に人手が介在することもあるので，RTL 記述に対して合成後 (さらに修正後) のゲートレベルのネットリストの等価検証を行なうといったような使用方法もある．本稿では，この等価性検証問題を効率よく行なうための手法を紹介し，その得失について実験および考察を行なう．

2. 等価性検証問題

等価検証問題は，対象の回路が順序回路 (すなわち有限状態機械) か組合わせ回路 (すなわち論理関数) かによって大きく 2 種類に分けることができる．実際の回路はほとんどの場合は順序回路であるが，順序回路の等価性検証問題は組合わせ回路の検証問題に比べて格段に難しい．そこで，何らかの手段で 2 つの回路間のフリップフロップ (レジスタ) の対応関係を求めることによって，問題を組み合わせ回路の等価性検証問題として扱う手法が広く用いられている．つまり，もともとのフリップフロップへの入力を外部出力と見なし，もともとのフリップフロップからの出力を外部入力と見なすことによって，回路を組合わせ回路のように扱うわけである．このようにフリップフロップを取り除いた組合わせ回路が等価であればもとの順序回路も等価であることが容易に証明できる．ただし，当然のことながら，このフリップフロップの対応づけを行ない，組合わせ回路化した回路が等価である，という条件はもとの順序回路が等価であることの十分条件ではあっても必用条件ではない．組み合わせ回路としては異なっていても順序回路としての振る舞いは等価である例も存在するので注意が必要である．本稿では等価性検証問題のうち，組合わせ回路の等価性検証問題を扱うものとする．以降，特にことわりの無いときには組合わせ回路の等価性検証問題をただ単に等価性検証問題と表記するものとする．

順序回路の等価性検証問題よりも簡単だとはいっても組合わせ回路の等価性検証問題は $\mathcal{NP}$ 完全問題であり(注1)，回路の入力数 n に対する多項式時間で解くことのできるアルゴリズムを開発できるのぞみは少ない．従来，このような検証は論理シミュレーションを利用して行なわれてきた．具体的には，変更前の回路と変更後の回路に対して同一のテストパタンを入力し，出力が同一かどうかを比較することで検証が行なわれる．2 つの回路の出力結果が異なっていた場合，その 2 つの回路は等価ではないことが容易にわかるが，2 つの出力結果が同一だった場合には，考えられるすべての入力パタンを入力しない限り等価であることの証明はできない．回路の入力数を n とすると考えられる入力パタンの総数は 2^n となるので，全パタンを網羅的に試すことができる回路の入力数は数十入力が限界である．

これに対し，2 分決定グラフ (Binary Decision Diagrams: BDDs) [1] と呼ばれるデータ構造を用いた手法が提案されている [2], [3]．これは 2 分決定グラフが論理関数のカノニカル (canonical) な表現形式である— 同一の変数順序のもとではグラフの形が唯一に定まる性質をもっている— ことと，多くの場合コンパクトに論理関数を表すことができる，という性質を用いたもので，検証対象回路の外部出力の論理関数を表す 2 分決定グラフを作って同一であるか比較することで等価かどうかを判断することができる．2 分決定グラフを用いた手法はシミュレーションを用いた手法よりは適用可能な回路規模が大きいが，場合によっては 30 入力程度の回路の論理関数をコンパクトに表せない場合もあり，実用的にはロバストであるとはいいがたい．

等価性検証問題は $\mathcal{NP}$ 完全問題であるので，一般的な問題に対する解法を求めることは難しい．そこで，検証対象の回路間にある程度の構造の類似性がある場合を対象にしたヒューリスティックアルゴリズムの開発が行なわれている [4], [6]〜[10]．これらの手法は基本的には [4] で述べられているような分割統治法を基本としている．つまり，比較対象の 2 つの回路 A と B がそれぞれ k 個の部分回路 $\{A_1, A_2, \ldots, A_k\}$ と $\{B_1, B_2, \ldots, B_3\}$ に同じ構造で分割されているものとする．もしも $\forall i A_i \equiv B_i$ が成り立っていたら $A \equiv B$ であることがわかるというものである．個々の等価検証問題に分割することでひとつの問題あた

(注1)：正確には等価でないことを確かめる問題が $\mathcal{NP}$ 完全問題である．

りの規模を小さく抑えることができる．ただし，常に，このような適切な分割が求められるとは限らない．不適切な分割を用いると対応する部分回路が等価でなくなる場合がある (c.f. $A_i \neq B_i$)．ある分割を用いて部分回路の等価検証を行なった結果，等価でないと分かった場合にありうる状況は次の 2 つである．

（1） 回路全体も等価ではない．

（2） 回路全体は等価だが，分割が不適切だった．

後者はフォールスネガティブ (false negative) と呼ばれており，分割統治法を用いた場合に取り扱わなければならないやっかいな問題である．つまり，部分回路の検証結果が等しくなかった場合に，それがフォールスネガティブなのかどうかを判定しないと回路全体が等価かどうかの判断ができない．

このフォールスネガティブを取り扱うための手法は以下のように分類することができる．

- テストパタン生成手法 (ATPG) の技術を用いたもの [6]
- 二分決定グラフによる論理関数処理を用いたもの [9]
- 両者の折衷手法 [7], [8], [10]

テストパタン生成手法の技術の利点は時間さえかければ回路規模に比例したメモリ使用量で処理を行なえるということである．逆に，最悪の場合には多大な計算時間を必要とする．テストパタン生成問題の場合には，仮定している故障を検出するテストパタンを生成できた時点で処理を打ち切ることができる．等価検証の場合も同様に 2 つの回路が異なっていたらその時点で処理を打ち切ることができるが，多くの場合，異なっていないことを証明するために検証を行なっているので途中での打ち切りはほとんど発生しないと考えてよい．そのため，この手法は他の手法に比べてより多くの計算時間を必要とする傾向がある．

二分決定グラフを用いた手法の利点は一旦，二分決定グラフを用いてコンパクトな論理関数表現を得ることができれば，あとはその二分決定グラフのサイズの多項式時間で処理を行なえるということである．欠点は場合によっては二分決定グラフのサイズが入力数の指数乗に比例した大きさになってしまうということである．もっとも，グラフのサイズが指数爆発するときには瞬時にそのような状態になる場合が多いので，指数爆発したらあきらめるという方針で使う限り無駄な時間を浪費することはない．

両者の折衷案では，テストパタン生成手法を適用する際の含意情報 (A という信号線が 1 になったら B という他の信号線も必ず 1 になるというような情報) を求める際に二分決定グラフを用いるものがあるが [7], [8]，実験結果を見る限り処理が複雑になるわりにはあまり効果的ではないようである．テストパタン生成手法では充足される条件を求めるのに適しており，二分決定グラフを用いた手法では充足されない条件を証明するのに適しているので，問題によって向き不向きが分かれる．このような特徴を活かして二分決定グラフを用いた手法でサイズが爆発してしまったらテストパタンを用いた手法に切り替える，といった実用的なヒューリスティックが考えられる [10].

3. 効率化手法

分割統治法をベースにした等価検証を効率的に行なう手法としては，後ろ向き走査 [9]，前向き走査 [10] が提案されている．まず，これらの手法の理論的な基礎となっている Cerny と Mauras の手法 [5] について述べたあとで 2 つの手法について紹介する．

3.1 Cerny と Mauras の手法

Cerny と Mauras の手法 [5] 自体はあまり実用的ではないが，理論的な基礎として重要なのでここで取り上げる．ただし，概念は文献 [5] と同一のものであるが，説明の都合上，一部の記号の意味が異なっている．今，等価検証対象の 2 つの回路を C_1 および C_2 とする．C_1 の入力変数を $X = \{X_1, X_2, \ldots, X_n\}$，出力変数を $Y = \{Y_1, Y_2, \ldots, Y_m\}$ とする．同様に C_2 の入力変数を $X = \{X_1, X_2, \ldots, X_n\}$($C_1$ と同じ)，出力変数を $Z = \{Z_1, Z_2, \ldots, Z_m\}$ とする．回路 C_1 の入力と出力の振る舞いを表す特徴関数 $\mathcal{R}^{C_1}(X, Y)$ を以下のように定義する．

$$\mathcal{R}^{C_1}(x, y) = \begin{cases} 1 & \text{回路 } C_1 \text{ に } x \text{ を入力すると} \\ & \text{出力 } y \text{ が得られる．} \\ 0 & \text{上記以外} \end{cases}$$

同様に回路 C_2 の入力と出力の振る舞いを表す特徴関数 $\mathcal{R}^{C_2}(X, Z)$ も以下のように定義する．

$$\mathcal{R}^{C_2}(x, z) = \begin{cases} 1 & \text{回路 } C_2 \text{ に } x \text{ を入力すると} \\ & \text{出力 } z \text{ が得られる．} \\ 0 & \text{上記以外} \end{cases}$$

ここで，回路 C_1 と C_2 が等価であるとは次式 (1) が成り立たないことと同値である．

$$\exists x, y, z \mathcal{R}^{C_1}(x, y) \wedge \mathcal{R}^{C_2}(x, z) \wedge (y \neq z) \qquad (1)$$

つまり，この式はある入力 x を回路 C_1 および C_2 に

加えると相異なる出力 $y \not\equiv z$ が得られることを示している．

ここで，回路 C_1 と C_2 を図 1 のようにカットラインに沿って分割するものとし，C_1 のカットライン上の信号変数を $U = \{U_1, U_2, \ldots, U_k\}$，$C_2$ のカットライン上の信号変数を $V = \{V_1, V_2, \ldots, V_l\}$ とする．また，回路 C_1 の分割された部分回路をそれぞれ C_{1L}，C_{1R} とし，回路 C_2 の分割された部分回路をそれぞれ C_{2L}，C_{2R} とする．回路 C_1 の特徴関数 $\mathcal{R}^{C_1}$ は次

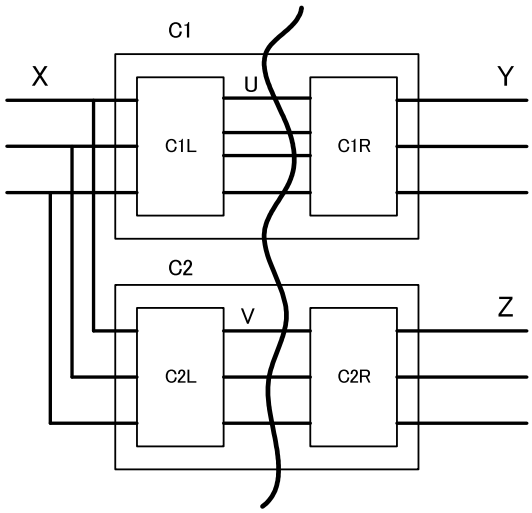

図 1　回路の分割

のように書くことができる．

$$\mathcal{R}^{C_1}(x,y) = \exists u \mathcal{R}^{C_{1L}}(x,u) \wedge \mathcal{R}^{C_{1R}}(u,y) \qquad (2)$$

ここで $\mathcal{R}^{C_{1L}}$ および $\mathcal{R}^{C_{1R}}$ はそれぞれ部分回路 C_{1L}，C_{1R} の特徴関数を表す．同様に回路 C_2 の特徴関数 $\mathcal{R}^{C_2}$ は次のように書くことができる．

$$\mathcal{R}^{C_2}(x,z) = \exists v \mathcal{R}^{C_{2L}}(x,v) \wedge \mathcal{R}^{C_{2R}}(v,z) \qquad (3)$$

$\mathcal{R}^{C_{2L}}$ および $\mathcal{R}^{C_{2R}}$ はそれぞれ部分回路 C_{2L}，C_{2R} の特徴関数を表す．

変数 U および V の値の集合 $\Gamma(u,v)$ および $\Omega(u,v)$ を次式のように定義する．

$$\Gamma(u,v) = \exists y, z \mathcal{R}^{C_{1R}}(u,y) \wedge \mathcal{R}^{C_{2R}}(v,z) \wedge (y \not\equiv z) \qquad (4)$$

$$\Omega(u,v) = \exists x \mathcal{R}^{C_{1L}}(x,u) \wedge \mathcal{R}^{C_{2L}}(x,v) \qquad (5)$$

つまり，$\Gamma(u,v)$ は外部出力 y と z に対して異なる値を出力させるような信号線 u, v の値を組み合わせを表しており，cross-observability と呼ばれる．一方，$\Omega(u,v)$ は任意の外部入力 x の値に対して u, v の取りうる値の組合わせを表しており，cross-controllability (cross-satisfiability) と呼ばれる．

ここで，以下の定理が成り立つ．

［定理 1］　回路 C_1 と C_2 が等価であるための必用十分条件は式 (6) が成り立たないことである．

$$\exists u, v \Gamma(u,v) \wedge \Omega(u,v) \qquad (6)$$

証明:

$$\begin{aligned}
&\exists u, v \Gamma(u,v) \wedge \Omega(u,v) \\
=\ &\exists u, v \exists y, z \mathcal{R}^{C_{1R}}(u,y) \wedge \mathcal{R}^{C_{2R}}(v,z) \wedge (y \not\equiv z) \\
&\wedge \exists x \mathcal{R}^{C_{1L}}(x,u) \wedge \mathcal{R}^{C_{2L}}(x,v) \\
=\ &\exists x, y, z (\exists u, v \mathcal{R}^{C_{1L}}(x,u) \wedge \mathcal{R}^{C_{1R}}(u,y)) \wedge \\
&(\exists u, v \mathcal{R}^{C_{2L}}(x,v) \wedge \mathcal{R}^{C_{2R}}(v,z)) \wedge (y \not\equiv z) \\
=\ &\exists x, y, z \mathcal{R}^{C_1}(x,y) \wedge \mathcal{R}^{C_2}(x,z) \wedge (y \not\equiv z)
\end{aligned}$$

最後の右辺の式は式 (1) そのものである．　□

この定理 1 は回路 C_1 および C_2 の外部入力と外部出力に対する関係 ($\mathcal{R}^{C_1}$ および $\mathcal{R}^{C_2}$) を知らなくても分割された回路の特徴関数から回路全体の等価検証を行なえることを示している．この条件は必用十分条件であり，完全であるのでここにはフォールスネガティブが起こることはない．もっとも，そのような完全な情報を持つために，一般にはこの Γ や Ω をコンパクトな形で表現することは難しい．

3.2　前向き走査

前節で述べた Cerny と Mauras の定理は等価条件を完全に表すものであるが，その反面，この条件を実際に計算することは難しい．そこで，実用的なヒューリスティックとしては式 (6) の条件を緩めて十分条件にすることで検証を行なう手法が提案されている．結果的には文献 [4], [6]〜[10] の手法はすべてこの範疇に入るといえる．具体的には $\Omega(u,v)$ の代りに以下のような関数 $\Psi(u,v)$ を用いるものである．

$$\Psi(u,v) = \bigwedge_{u_i \text{ と } v_j \text{ は等価信号線対}} (u_i \equiv v_j) \qquad (7)$$

つまり，cross-controllability のうち，等価信号対の情報 (u_i と v_j が異なる値になることがないということ) のみを集めてそれを $\Omega(u,v)$ の代りにするというものである．明らかに $\Psi(u,v) \supseteq \Omega(u,v)$ である ($\Psi(u,v)$ は実際に起こり得ない u, v 間の値の組み合わせを含む可能性がある) ので，式 (6) の $\Omega(u,v)$ を $\Psi(u,v)$ に置き換えた次式

$$\Gamma(u,v) \wedge \Psi(u,v) \qquad (8)$$

が成り立たなければ式 (6) も成り立つことはない．つまり，式 (8) は等価のための十分条件であるという

ことができる．実際には $\Psi(u,v)$ を表す二分決定グラフを保持する必要はなく，もしも u_i と v_j が等価であると判定されたら v_j のファンアウト先の結線をそのまま u_i に移すことで内部等価点の情報を反映することができるため，この近似は非常に効率がよい．一方，式 (8) が必用条件ではなくなったため，フォールスネガティブが起こる可能性が生じたというわけである．等価検証のための効率のよいヒューリスティックを開発するためには，式 (8) のような近似を行ないつつ，なおかつフォールスネガティブを起こしにくい (必要以上に情報を落とさない) ような条件を求めることが鍵となる．

文献 [10] ではこの内部等価点の情報を計算するために 2 つの手法を提案している．一つめは structure hash と呼ばれるもので，回路構造のみを見て等価と判定できる内部等価点を探すものである．回路構造を表す内部のデータ構造を 2 入力 AND ゲートとインバータ(注2)に制限することによって，構造的な等価点を容易に判定できるようになっている．もう一つの手法は BDD hash と呼ばれるもので，回路中に適当なカットラインを設け，そのカットラインを外部入力と見なしたときの回路各部の論理関数を二分決定グラフ (BDD) で表し，同一の二分決定グラフを持つゲートを等価と判定する，というものである．回路中に適当なカットラインが引いてあり，そのカットライン上の変数 (信号線) の集合を T としたとき，T を外部入力と見なした時の信号線 A の論理関数 $F_A^T(T)$ が信号線 B の論理関数 $F_B^T(T)$ と等しかった場合，真の外部入力 X に基づく信号線 A の論理関数 $F_A^X(X)$ と信号線 B の論理関数 $F_B^X(X)$ も等しいことが容易にわかるので，上記の方法によって適切な内部等価点 (の部分集合) を求めることができることがわかる．文献 [10] では，まず外部入力をカットラインとして内部等価点をもとめ，次には今求められた内部等価点を新たなカットラインとして次の内部等価点を求めるヒューリスティックを提案している．その際に，計算された二分決定グラフをそのサイズの小さい順にヒープ木に登録することによって，常に小さな二分決定グラフだけを選んで新しい論理関数の計算を行なうようにしている．計算の結果，そのサイズが制限値を越えた場合にはそれ以上の処理は行なわない．このようなヒューリスティックを用いることにより，定められたサイズ以下で計算可能な (判定可能な) 内部等価点の情報を求めることができる．ただし，この情報だけでは等価であるための必用十分条件とはならないため，簡単な後ろ向き走査やテストパタン生成手法を併用している．この手法の短所は二分決定グラフの計算を行なう時に，ただ単にそのサイズが制限値以下という条件で可能な限りすべての信号線の論理関数を計算してしまうことにある．そこで計算し判定した情報が後で用いられるかどうかを前もって知ることはできない．

3.3 後ろ向き走査

前向き走査が $\Omega(u,v)$ を近似した $\Psi(u,v)$ を用いたものであるのに対して，後ろ向き走査 [9](注3)は $\Gamma(u,v)$ そのものを計算するものである．今，内部等価点対の候補 (A,B) が実際に等価であるか判定するものとする．後ろ向き走査では A,B よりも入力側に適当なカットライン U,V を求め，そのカットライン上の $\Gamma(u,v)$ および $\Psi(u,v)$ を計算することで等価判定を行なうものである．ただし，この $\Psi(u,v)$ は前向き走査で求めるのではなく，以前の後ろ向き走査で求めた情報を用いる．この後ろ向き走査も前向き走査と同じく $\Psi(u,v)$ を用いているために検証が失敗する場合がありうる．その場合にはもとのカットラインよりも入力よりに新たなカットライン U',V' を設けて，そのカットライン上の $\Gamma(u',v')$ を計算することで更なる判定を行なう．理論的にはこのカットラインが外部入力まで到達すれば外部入力上では $\Omega(u,v)=\Psi(u,v)$ であるので完全な検証が行なえることになる．また，カットライン上の他の信号線とは独立に 0/1 の値を取ることができる信号線に関する情報は存在作用素 ($\exists$) 演算を用いて $\Omega(u,v)$ から消去することが可能である．このような独立な信号線の削除が行えるので，後ろ向き走査は前向き走査と比べて同じ内部等価対を検証するのに必要な計算量を低く抑えることができる．また，現在のカットラインから次のカットラインを求めるときに，回路の構造の情報を利用して適切なカットラインを選択するヒューリスティックが提案されている [9]．以上の点は後ろ向き走査が前向き走査に対して有利と思われる点である．

後ろ向き走査の問題点は，検証を行なうべき内部等価対の候補の選定にある．この候補を求める処理には通常は乱数を用いたシミュレーションが用いられる．適当なパタン数を入力した結果，同一の値を

(注2)：実際にはインバータを表すゲートは用いず，ゲートとゲートを結ぶ枝 (配線) に反転属性を負荷することで極性反転を表している．

(注3)：文献 [9] 中には後ろ向き走査 (に対応した英語) という用語は使用されていない．ここでは文献 [10] の手法との比較のためにこのような用語を用いている．

とった信号線の対を検証すべき候補とする．シミュレーションを行なうパタン数が少ないと本来は等価でない信号線の対が候補に含まれることになり無駄な検証を行なわなければならない．一方，候補の情報の確度を高めるためにシミュレーションのパタン数を増やすことは計算時間の増大を招くため，適切なパタン数を求めることが重要である．

4. 実験結果

ここまで述べたような効率化手法の効果をはかるため，計算機実験を行なった．現在の所オープンになっている等価性検証のベンチマーク回路はどれも規模が小さいため，あまり適切ではないが，他に選択が余地がないので ISCAS'85 の回路のオリジナルと冗長故障を削除したものの間の等価性検証を行なった (答えはすべて等価となる)．手法としては以下の 4 通りを比較した．

（1） 後ろ向き走査のみ ([9] 相当)
（2） 後ろ向き走査＋ structure hash
（3） 前向き走査 (structure hash + BDD hash)
（4） 前向き走査＋後ろ向き走査

すべての手法において二分決定グラフのサイズの制限は 1,000 ノードとしている．表 1 に実験結果を示す．使用計算機は Pentium-III 1GHz (FreeBSD-4.7-RELEASE) である．

表 1 実験結果

	計算時間 (秒)			
回路名	手法 1	手法 2	手法 3	手法 4
c432	0.13	0.10	0.06	0.06
c499	0.19	0.15	0.12	0.18
c1355	0.32	0.23	0.14	0.26
c1908	0.45	0.28	0.60	0.68
c2670	0.51	0.35	0.96	1.10
c3540	0.73	0.55	1.11*	1.25
c5315	1.12	0.77	2.17	2.45
c6288	2.02	1.50	1.74	1.73
c7552	2.05	1.56	1.68	2.26
合計	7.52	5.49	8.58	9.97

* は検証が失敗したことを示す.

手法 2(後ろ向き走査＋ struct hash) がもっとも短い時間で検証を行なっている．手法 3(前向き走査) は小さな回路では手法 2 よりも高速であるが，回路規模が大きくなると遅くなっている．無駄な論理関数の計算を行なっているものと思われる．また，c3540 の場合には前向き走査だけでは完全な検証を行なうことはできなかった．具体的には一つの外部出力の等価性判定が二分決定グラフのサイズの制限のために成功していない．例えば，サイズの制限を 10,000 に緩めると 4.31 秒かかって検証に成功する．ある意味二分決定グラフの大きさと性能をトレードオフにかけているということもできるが，後ろ向き走査では 1,000 ノードも必要とせずに検証を行なっているので不必要な計算を行なっていることがわかる．手法 4(前向き走査＋後ろ向き走査) は総じて手法 1 よりも悪い結果となっている．これは前向き走査で得られる情報は後ろ向き走査でも得られるので前向き走査で可能な限り多くの信号線に対して論理関数を計算している処理の部分が冗長になっているものと考えられる．

5. おわりに

本稿では組合わせ回路の等価検証を行なう実用的な手法のついて述べ，その効率化手法として知られているいくつかの手法についての考察を行なった．考察および実験結果から筆者の提案している後ろ向き走査に structure hash 手法を加えたものががもっとも効果的であるといえる．今後の課題として更なる効率化のためのヒューリスティック開発や，順序回路の等価検証問題を組合わせ回路の等価検証問題に落とすためのフリップフロップのマッチング問題のためのアルゴリズム開発などが考えられる．

文献

[1] R.E. Bryant, "Graph-based algorithms for boolean function manipulation", *IEEE Transactions on Computer*, C-35(12), pp. 677–691, Aug. 1986.

[2] M. Fujita, H. Fujisawa, and N. Kawato, "Evaluation and improvements of Boolean comparison method based on Binary Decision Diagrams", In *Proc. of ICCAD*, pp. 2–5, Nov. 1988.

[3] S. Malik, A. Wang, R. Brayton, and A. Sangiovanni-Vincentelli, "Logic verification using Binary Decision Diagrams in a logic synthesis environment", In *Proc. of ICCAD*, pp. 6–9, Nov. 1988.

[4] C.L. Berman and L.H. Trevillyan, "Functional comparison of logic designs for VLSI circuits", In *Proc. of ICCAD*, pp. 456–459, Nov. 1989.

[5] E. Cerny and C. Mauras, "Tautology checking using cross-controllability and cross observability", In *Proc. of ICCAD*, pp. 34–37, Nov. 1990.

[6] W. Kunz, "Hannibal: An efficient tool for logic verification based on recursive learning", In *Proc. of ICCAD*, pp. 538–543, Nov. 1993.

[7] S.M. Reddy, W. Kunz, and D.K. Pradhan, "Novel verification framework combining structural and OBDD methods in a synthesis environment", In *Proc. of 32nd DAC*, pp. 414–419, Jun. 1995.

[8] J. Jain, R. Mukherjee, and M. Fujita, "Advanced verification techniques based on learning", In *Proc. of 32nd DAC*, pp. 420–426, Jun. 1995.

[9] Y. Matsunaga, "An efficient equivalence checker for combinational circuits", in *Proc. of 33th DAC*, pp. 629–634, Jun. 1996.

[10] A. Kuehlmann, F. Krohm, "Equivalence checking using cuts and heaps", In *Proc. of 34th DAC*, pp. 263–268, Jun. 1997.