二分決定グラフを用いた誤り追跡入力の生成手法

1K−6

蛇蝮和寛[1]　丸田繁孝[2]　菅沼直昭[1]　冨田昌宏[1]　平野浩太郎[1]
[1]神戸大学　[2]神戸市役所

1．はじめに

二分決定グラフ（Binary Decision Diagram[1]:以下BDDと略す）には，論理関数をコンパクトに表現し，入力変数順の固定により関数を一意に表すことができるという特徴がある。

本稿では，論理診断手法の一つである拡張X－伝搬法[2]において重要な役割を果たす誤り追跡入力[2]（PLE：input Pattern for Locating design Errors）の生成に，BDD を適用する手法を提案する。拡張X－伝搬法では，少ないパターン数で限定効果の高い，有効な誤り追跡入力の組み合わせを適用することが重要である。BDD を用いることにより，don't carcを含む形で全ての誤り追跡入力を短時間で生成する。

2．ＢＤＤによる誤り追跡入力生成

n入力単一出力回路の，理想回路Fs，実回路Fgに，1つのブール変数Xまたはその補$\overline{X}$を導入した入力パターン

$v = (a_1, \cdots, a_{i-1}, X/\overline{X}, a_{i+1}, \cdots, a_n)$

が，

$$Fs(v) = X \quad (1)$$

$$Fg(v) = a \quad (2)$$

を満たす場合，これを誤り追跡入力と定義する。ただし，$X/\overline{X}$はXまたは$\overline{X}$の一方を表し，a_j（$j=1,\cdots,n;\ j\neq i$）及びaは定数（0または1）を表す。

関数FsとFgに対する不一致入力は，両者の不一致を示す関数：H＝1とする入力パターンである。Hは，

$$H = Fs \oplus Fg \quad (3)$$

で表される。両者が等しいとき常にH＝0が成り立つので，関数Hを求めることによって論理検証が行われる。Hの入力変数x_iによるブール微分が，

$$\frac{dH}{dx_i} = H_i(0) \oplus H_i(1)$$
$$= H(\cdots, x_{i-1}, 0, x_{i+1}, \cdots) \oplus H(\cdots, x_{i-1}, 1, x_{i+1}, \cdots) = 1 \quad (4)$$

を満たす入力パターンは，x_iについて0と1のうち，一方は一致入力でもう一方は不一致入力となる。Fs＝Xに限定し，入力パターンにおけるXと$\overline{X}$を区別するため，x_i＝0，1に対応した FsをFs_{i0}，Fs_{i1}とすると，

$$Fpx_i = \frac{dH}{dx_i} \cdot \overline{Fs_{i0}} \cdot Fs_{i1} = 1 \quad (5)$$

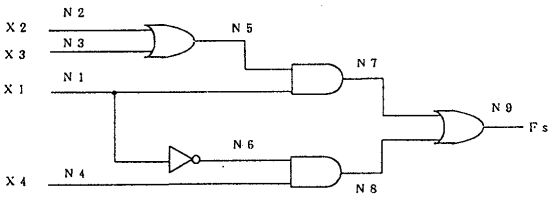
$$Fp\overline{x}_i = \frac{dH}{dx_i} \cdot Fs_{i0} \cdot \overline{Fs_{i1}} = 1 \quad (6)$$

の2式のうち，(5)式を満たすパターンのx_iにはXを与え，(6)式を満たすパターンには$\overline{X}$を与えたパターンが誤り追跡入力となる。

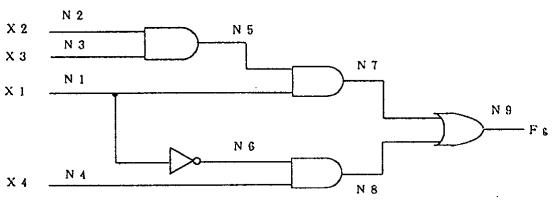
共有二分決定グラフ[3]の考えを取り入れたBDD演算パッケージ[4]を用いて，本手法を実装した。図1に示す理想回路と実回路の例について，入力x_2にX／$\overline{X}$を導入して誤り追跡入力を生成する過程を図2に示す。まずFs, Fg, Hそれぞれに対応するグラフを作成した後で，各入力変数について(4)-(6)式の演算を行う。変数に0または1の値を代入するrestrict演算[1]については，low, high間のポインタの代入操作による方法が提案されているが，Fs, Fg, Hに対応するグラフを保存するため，BDDの複写による方法を採用した。不一致関数Hからx_2=0, 1の代入操作によって得られた$H_2(0)$, $H_2(1)$のグラフを図2(a)に示す。代入操作の対象となる変数x_2よりも高位のノードについては，再帰的に複写を行う。複写元のノードに複写先ノードへのポインタを保持することにより複写先ノードとの対応付けを明示する。代入操作の対象となる変数のノードについては，代入値(0, 1) に応じて lowまたはhighのエッジを返す。Fpx_i, $Fp\overline{x}_i$を求めた後，satisfy-all[1]演算によって(5), (6)式を満足する誤り追跡入力を求める。図2(c)に示すFpx_2のグラフを1とする条件からは，誤り追跡入力

$(x_1, \cdots, x_4)$＝（1 X 0 D）　（D：don't care）

が求まる。



(a)　理想回路



(b)　実回路

図1　誤り追跡入力生成の対象回路例

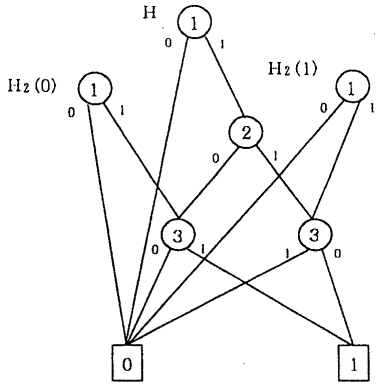
PLE Generation Algorithm Based on BDD
Kazuhiro JAHAMI[1], Shigetaka MARUTA[2], Naoaki SUGANUMA[1], Masahiro TOMITA[1] and Kotaro HIRANO[1]
[1]Kobe University　[2]Kobe City Government

3. 実験と評価

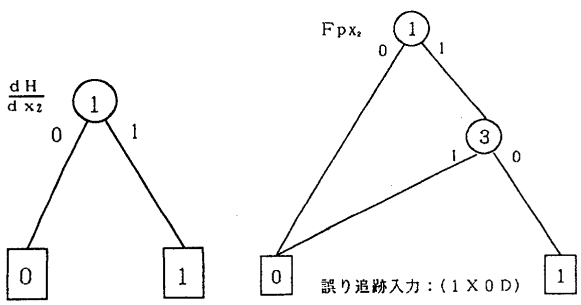
本手法を，パーソナル・コンピュータ MITAC4060G(CPU:80486DX 33MHz, 13MIPS)上にGCC ver1.39を用いて実現した。ISCASベンチマーク回路[5](Y1, Y4, C432)，シミュレーション・エンジンの回路[6](Y2, Y3)に誤りを付加した複数の回路を対象として実験を行った。ビット並列演算を導入したコンパイル法に基づく従来の網羅的手法と，今回提案する手法によって，全ての誤り追跡入力を生成した。ただし，C432については本手法による実験のみ，7出力全てについて行った。

実験対象回路数と，0，1，X，$\overline{X}$ のみで表された全誤り追跡入力数の平均値Na，従来手法による全誤り追跡入力生成の平均処理時間を表1に示す。また，BDD を用いて生成したdon't careを含む追跡入力数Ndc，圧縮率Ndc/Na，処理時間を表2に示す。don't careを用いることで，誤り追跡入力の個数は多くの例で 1/100以下に圧縮された。これにより，コンパクトに表現された全ての追跡入力の中から論理診断に有効な組を抽出することができる。従来手法と比較すると，BDD を用いて全誤り追跡入力を求める時間は，Y1のような小規模の回路を除いて 1/400程度となった。このため，全ての誤り追跡入力を求めてから有効な組を選択する時間の余裕を持つことが可能となった。

C432 では誤り追跡入力数が膨大となるため，satisfy-all演算を含む処理時間は増大するが，Fpx_i, $Fp\overline{x}_i$の計算については比較的短時間で完了する。従って，特に大規模回路については，BDD の形式で誤り追跡入力を記憶することが望ましいと考えられる。



(a) 不一致関数Hと$H_2(0)$, $H_2(1)$



(b) x2によるHのブール微分　　(c) Fpx_2のグラフ

図2　誤り追跡入力生成過程

表1　実験対象回路と全誤り追跡入力数

回路	ゲート数	入力数	対象回路数	全追跡入力数平均：Na	従来手法時間(秒)
Y1	40	8	41	241	0.021
Y2	60	22	18	826.453	218
Y3	75	19	18	26.806	12
Y4	116	25	16	9.5×10^{6}	1655
C432	203	36	5	8.7×10^{10}	—

表2　BDDを用いた誤り追跡入力生成の結果

回路	ノード数	追跡入力数平均：Ndc	圧縮率 Ndc/Na	処理時間(秒)
Y1	59	40	0.18	0.16 (0.16)
Y2	639	460	5.6×10^{-4}	0.39 (0.36)
Y3	207	78	2.9×10^{-3}	0.29 (0.28)
Y4	10,619	46,435	4.9×10^{-3}	4.3 (2.4)
C432	22,351	1.46×10^{7}	1.7×10^{-4}	630 (14)

処理時間：括弧内はFpx_i, $Fp\overline{x}_i$の計算時間
ノード数：全てのFpx_i, $Fp\overline{x}_i$を含む

4. まとめ

BDDを用いて，拡張X-伝搬法に適用する誤り追跡入力を生成する手法を提案した。従来は，ランダムに生成された誤り追跡入力を適用していたが，本手法により，don't careを含む全誤り追跡入力の高速生成を実現した。

今後の課題として，BDD による生成の段階で診断に有効な組を選択することが挙げられる。また，これまでの論理診断手法とは別に，BDD の特徴を積極的に利用した診断手法の検討も考えられる。

謝辞

BDD 演算パッケージに関して御協力を頂きました(株)富士通研究所ソフトウェア研究部の川戸信明部長，同プロセッサ研究部の藤田昌宏氏に深謝いたします。

参考文献

[1] R. E. Bryant, "Graph-based algorithms for boolean function manipulation," IEEE Trans. Computer, vol.C-35 (8), pp.667-691 (1986).

[2] 上田伸人他，"多重論理設計誤りを対象とする自動追跡手法，" 情処研報，91-DA-60，pp.185-192 (1991).

[3] 湊真一他，"論理関数の共有二分決定グラフによる表現とその効率的処理手法，" 情処研報，89-DA-50，pp.39-45 (1989).

[4] 松永祐介他，"順序付き2分決定グラフと許容関数を用いた段論理回路簡単化手法，" 電子情報通信学会論文誌，vol.J74-A，No.2，pp.196-205 (1991).

[5] F. Brglez and H. Fujiwara, : A Neutral Netlist of 10 Combinational Benchmark Circuits and a Target Translation in FORTRAN, ISCAS'85 (1985).

[6] 富田昌宏他，"1チップ・シミュレーションエンジン：TASS II，" 情報処理学会第42回全国大会講演論文集，vol.6，pp.178-179 (1991).