8M−1

トランスダクション法のための二分決定グラフを用いた束探索に基づく3段初期回路の生成

澤田 直* 石垣 博康+ 上林 弥彦+
*九州大学工学部 +京都大学工学部

1 まえがき

近年のVLSI技術の進歩や計算機能力の向上に伴い，計算機による論理回路設計が不可欠となってきている．

トランスダクション法[3]は何らかの方法で設計された初期回路を元に，そこに含まれる潜在的な冗長性を利用することにより回路の変形削減を行なう手法である．この手法は面積最適化だけでなく遅延最適化や設計の変更などにも強力であり，近年さまざまな研究がなされている．これまで著者らはその初期回路に注目し，できるだけ段数の少ない初期回路を生成することにより遅延を最適化する研究を行なってきた[5]．これは束(lattice)状に表された真理値表をもとに3段の回路を生成する初期回路生成部とそれにファンイン制限を施すファンイン制限部とからなっている．しかしこの初期回路生成部において束を直接扱うため，入力変数の数nに対して$O(2^n)$の記憶領域が必要であり大規模な回路を扱うことができなかった．

共有二分決定グラフ[2](Shared Binary Decision Diagram, 以下簡単のためSBDDと表す)は論理関数を計算機内部で表現するのに大変効率の良いデータ構造である。本稿では，論理関数の表現にSBDDを用い，束を順次計算することによって従来の手法を大規模な回路に適用できるように改良したことについて述べる．本手法を用いるのは不完全指定の多出力関数を実現する3段の論理回路を高速に作成するのに適しているためである．

2 束とNOR回路との対応

ある半順序集合において任意の2つの要素がただ1つの上限とただ1つの下限を持つとき，その半順序集合を束(lattice)と呼ぶ．本稿では外部入力の組合せ(これを入力ベクトルと呼ぶ)を要素とし，その任意の2つの要素a,bに対して$a \vee b$を上限に，$a \wedge b$を下限にするように半順序関係を定義したものを束として用いる．ある論理関数に対して代入したときその論理関数の値を0にする入力ベクトルを0ベクトル，1にする入力ベクトルを1ベクトルと呼ぶ．図1に束による真理値表の例を示す．

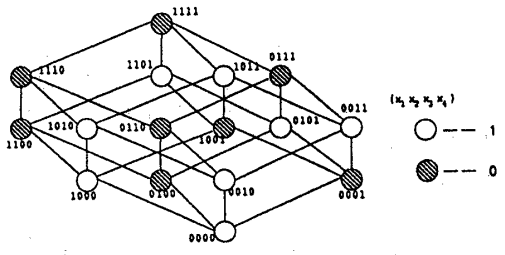

図 1: 束による真理値表の例

以下説明のために入力変数だけからなるゲートに対して代表ベクトルを定義する．

[定義 1] あるゲートvへ結線されている入力変数が全て0になり，その他の入力変数が全て1になっている入力ベクトルをゲートvに対する代表ベクトルと呼ぶ．

入力変数だけからなるNORゲートの出力を束上で見ると，その代表ベクトルの下界だけが1になり，残りは0になることが分かる．この性質を用いて3段NOR回路における各レベルのゲートに対する入力と最終的な出力との関係を示す．なお，ゲートの段数は出力側から1段目，2段目，3段目とする．

出力ゲートに対する入力: 出力ゲートv_oに入力変数x_iと複数のゲート$v_1,\cdots,v_j$が結線されているとすると，その出力関数$f(v_o)$は$\overline{x_i}\cdot\overline{f(v_1)}\cdots\overline{f(v_j)}$となる．従って$x_i=1$のときに$f(v_o)=1$とならないときだけ$x_i$を出力ゲートに結線できる．

2段目のゲートに対する入力: 2段目のゲートの出力が1になっている要素について出力関数は0の値をとる．従って0ベクトルを代表ベクトルに持つようなゲートを2段目に作成することにより出力関数を0にすることができる．しかし，その0ベクトルの下界に1ベクトルが含まれているときは2段目のゲートの出力からそれに対応する1要素を減らしてやらなければならない．そのために3段目のゲートを用いる．

3段目のゲートに対する入力: 3段目にその1ベクトルを代表ベクトルに持つゲートを作成してやり，それを2段めのゲートに結線すればその1ベクトルの下界に対する2段目のゲートの出力は0となる．

3 従来のアルゴリズムの概要

前節で述べた性質を利用した3段初期回路生成アルゴリズムの概要を示す．

Procedure LTMON

Step1: 求める関数に対する真理値表を束で表す．

Step2: 各入力変数x_iに対して$f=\overline{x_i}\cdot f$が成り立つかどうかを確かめ，もし成り立つならx_iを出力ゲートに結線する．

Step3: 束上で一番高い位置にあるマークされていない0ベクトルを求め，それを代表ベクトルに持つような2段目のゲートv_aを作成し，出力ゲートに結線する．

Step4: v_aの代表ベクトルの下界についてその全てのベクトルがマークされるまで幅優先探索で以下の操作を繰り返す．全ての下界がマークされたらStep5へ．

- もし0ベクトルなら0にマークする．
- もし1ベクトルならその1ベクトルの下界を全て1にマークし，その1ベクトルを代表ベクトルに持つような3段目のゲートを作成し，v_aの入力に結線する．

Step5: Step4の操作で1にマークされた全てのベクトルのマークを初期化する．

Step6: もしマークされていない0ベクトルがあればStep3 ～ Step5を行う．全ての0ベクトルがマークされたらStep7へ．

Step7: 多出力関数なら他の関数に対してStep1 ～ Step6を行う．

Step8: 入力段から順に同じゲート同士をマージしていく．

この手法は処理が高速であり，不完全指定の多出力関数に効果的であるが，真理値表を用いているために入力変数の数が増えると大量のメモリを消費するという欠点がある．そのため従来扱える変数の数は12ぐらいが限度であった．

Lattice Search Based 3-Level Initial Network Generation with Binary Decision Diagrams for Transduction Method
Sunao SAWADA*, Hiroyasu ISHIGAKI+, Yahiko KAMBAYASHI+
*Faculty of Engineering, Kyushu University
+Faculty of Engineering, Kyoto University

4 二分決定グラフを用いた改良アルゴリズム

前節で述べた3段初期回路生成アルゴリズムをSBDD向きに改良し，多入力変数に対応できるようにした．そのために現在の入力ベクトルの値に対して束の幅優先探索で次に当たる入力ベクトルを返す関数lattice_orderを用意する．この入力ベクトルをSBDDの入力として用いることによりメモリの節約を図る．

求める論理関数 f を表すのに2つのSBDD(f_0, f_1)を用いる．f との対応は以下の通りである．

f	f_0	f_1
0	1	1
1	0	1
*	0	0

このとき，入力変数 x_j に対して以下の判定式1が恒偽関数になれば，その x_j を用いずに求める論理関数を表すことができる．

$$f_0(x_j = 0) \cdot f_1(x_j = 1) + f_0(x_j = 1) \cdot f_1(x_j = 0) \quad (1)$$

この時 f_0 は $f_0(x_j = 0) + f_0(x_j = 1)$ に，f_1 は $f_1(x_j = 0) + f_1(x_j = 1)$ に置き換えられなければならない．

Procedure LTMON2

Step1: 求める論理関数 f を f_0, f_1 で与え，$flag$ という恒偽関数を用意する．変数表を用意し，全ての入力変数を記憶する．

Step2: 各入力変数 x_i に対して $f = \overline{x_i} \cdot f$ が成り立つかどうかを確かめ，もし成り立つならその x_i を出力ゲートに結線し，変数表から x_i を除き，f を $f(x_i = 0)$ に置き換える．

Step3: 変数表に含まれる各入力変数 x_j について，判定式1が恒偽関数になるかどうかを調べる．もし恒偽関数であるならば，変数表から x_j を除き，f_0, f_1 を所定の関数に置き換える．

Step4: 変数表に対してlattice_orderを呼び出し，$f0 \cdot \overline{flag}$ に対して変数の値を割り当て，1になるもの（まだカバーされていない0ベクトル）を探す．

Step5: Step4で求めたベクトルを代表ベクトルに持つような2段目のゲート v_a を作成し，出力ゲートに結線する．変数表で v_a に結線されなかった変数からなる一時変数表と，v_a に結線された変数を0に固定した f_1（これを f_{11} とする）と恒偽関数 tmp_flag を作る．

Step6: $f_{11} \cdot \overline{tmp\_flag}$ が恒偽関数になるまで以下の処理を行なう．一時変数表に対してlattice_orderを呼び出し，$f_{11} \cdot \overline{tmp\_flag}$ に対して変数を割り当て，1になるものを探す．1になるものがあればそのベクトルを代表ベクトルに持つような3段目のゲート v_b を作成し，Step5で作った v_a に結線する．tmp_flag とゲート v_b の出力の論理和を新しい tmp_flag にする．

Step7: ゲート v_a の関数を計算し，その関数と $flag$ の論理和を新しい $flag$ にする．

Step8: $f_0 \cdot \overline{flag}$ が恒偽関数になるまでStep4 ～ Step7を行う．

Step9: 多出力関数ならば他の関数に対してStep1 ～ Step8を行う．

Step10: 入力段から順に入力が同じゲートを1つにマージする．

5 実験結果

前節で述べた手続きをOMRON社のLUNA88K上にC言語を用いて実現し，各種ベンチマーク回路について他の手法と共に3段初期回路合成の実験を行なった．結果の一部を表1に，またその両者に著者らの開発したファンイン制限手続き[5]を適用した結果を表2に示す．ここでLTMONは本稿の手法による合成結果を，Espressoは文献[1]の手法による和積形論理出力を3段NOR回路に変換した結果を表している．gateはゲート数を，con.は結線数を，timeは処理時間(秒)を表し，lev.は回路の段数を表す．なおSBDDの処理には現NTT湊真一氏のSBDDパッケージ[4]を使用した．

この結果を見ると3段回路の段階ではEspressoの方がいい結果を出しているが，これにファンイン制限を加えると本手法の結果の方が改善の度合が大きいことが判る．これはEspressoが3段目を否定ゲートとしてしか用いていないのに対して本手法では複数入力を許すので3段回路の段階では3段目のゲート数が多くなることによると思われる．また，そのことによりトランスダクション法における回路の変形を行ない易くなるという効果も考えられる．

Network			LTMON			Espresso		
name	in	out	gate	con.	time	gate	con.	time
5xp1	7	10	116	483	0.2	80	370	1.8
b12	15	9	55	230	0.1	51	144	0.6
bw	5	28	115	426	0.2	58	287	1.6
e64	65	65	130	2210	4.3	258	2338	82.2
misex2	25	18	196	1131	1.9	88	293	9.3
rd73	7	3	158	1058	0.3	137	910	1.2
table3	14	14	1137	10105	100.1	297	3297	375.1
table5	17	15	1177	11085	243.5	318	3669	689.9

表 1: 3段NOR回路の合成結果

Network	LTMON			Espresso		
name	gate	con.	lev.	gate	con.	lev.
5xp1	107	277	7	164	418	9
b12	68	172	7	80	181	7
bw	127	328	7	144	380	7
e64	1472	3525	7	1486	3560	7
misex2	132	298	7	140	326	5
rd73	301	872	9	309	835	9
table3	2253	5705	11	2362	6787	11
table5	1996	5102	13	2524	7307	11

表 2: ファンイン制限の結果

6 むすび

SBDDを用いた束探索に基づく3段初期回路生成手法について述べてきた．本手法はトランスダクション法の初期回路としてだけではなく，部分回路置換による段数の削減などにも使用できる．今後の課題として各種初期回路についてトランスダクション法がどういう振舞いを見せるか，また各目的に応じてどういった初期回路を用いるのが良いかの検討が挙げられる．

謝辞

本手法について有益な御示唆を頂いたイリノイ大学の室賀三郎教授並びにSBDDパッケージを提供して頂いたNTTの湊真一氏に感謝致します．またプログラム開発において有益な助言を頂いた京都大学上林研究室の高田秀志氏に感謝致します．

参考文献

[1] R.K.Brayton, et al., "Logic Minimization Algorithms for VLSI Synthesis", Kluwer Academic, 1984.

[2] R.E.Bryant, "Graph-based Algorithms for Boolean Function Manipulation", *IEEE Trans. on Comput.*,C-35(8), August 1986.

[3] S.Muroga, et al., "The Transduction Method—Design of Logic Networks Based on Permissible Functions", *IEEE Trans. on Comput.*, Vol.38, No.10, October 1989.

[4] S.Minato, et al., "Shared Binary Decision Diagram with Attributed Edges for Efficient Boolean Function Manipulation", *Proc. 27th Design Automat. Conf.*, June 1990.

[5] S.Sawada, et al., "Generation of Fan-in Restricted Initial Networks for Transduction Method, *Proc. SASIMI'92*, April 1992.