7P－4

NANDゲートのみで論理回路を実現する一手法(MA法)*

宮腰 隆† 松田 秀雄†　増野 武裕‡　畠山 豊正†　山淵 龍夫†　中嶋 芳雄†
富山大学工学部§　京セラ（株）　富山大学工学部§

1 はじめに

NANDゲートのみによる論理回路の設計は構造の一様性から集積回路化に適し,重要である.(文献[1])では,一線入力多段NANDゲート回路の一設計法(以下MA法と略)を述べているが,そこでは,関数を積和項表現で与え様々な操作を行っている.今回は,積和項表現の代わりにBDD(二分決定グラフ),あるいは,SBDD(共有二分決定グラフ)を用いて効率化できるところは効率化し,多出力関数も扱えるように,拡張している.

2 基礎的事項

(2値)n変数論理関数$f(x_1,x_2,\ldots,x_n)$について考える.入力変数の肯定$x_i(i=1,2,\ldots,n)$およびその否定$\bar{x}_i$をリテラル,重複のないいくつかのリテラルの積を項といい,n個のリテラルからなる項を最小項という.$r(0\leq r\leq n)$個の肯定変数のみからなる項Pを許容項という(定数1も含める).このPは$2^{(n-r)}$個の最小項を含む.許容項が含む最小項の数で,許容項の大きさを定義する.

3 方法

関数F_sをマップで与えると,一つ一つの最小項は一つ一つのマップのセルに対応できる.MA法の概要を述べよう.まず,出力ゲートG_0に相当する許容項P_0=1があるものとする.そして,後出の手順TNANDをF_sについて呼ぶと,(a)真理値1(true)のセルをできるだけ大きな許容項(例えば,$P_1=x_3, P_2=x_2$)を生成して,被覆する.(b)もし,これらの許容項の中で真理値0(false)のセルを含むものがあれば(例えば,それをP_1としたら),それらの許容項についてだけTNANDを呼んで0を被覆する許容項を生成する(例えば,$P_3=x_3x_4, P_4=x_1x_2x_3$とする).もし,これらの許容項のいずれかが,1のセルを含めば,再び(a)のようにして,1のセルを被覆する許容項を生成し...というように,すべての許容項が1のセルと0のセルの混在しなくなるまで,TNANDを再帰的に呼んで(a)(b)の手順を交互に繰り返す.この手順の結果,図1(a)の例のように,許容項の木が形成される.例えば,許容項$P_3=x_3x_4$なら,この許容項を入力変数線x_3,x_4を持つNANDゲートG_3に置き換えるなどとすると,図1(b)のような,(1)初期回路が得られる.次に,(2)回路の接続関係を使って,簡単化を行うのが本手法である(本稿では(2)は省略).

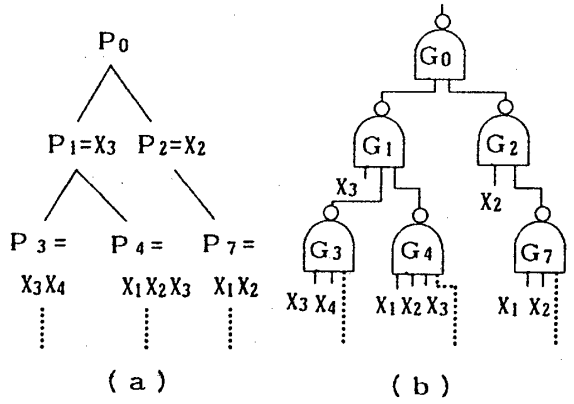

図1: 許容項の木とNAND回路

TNANDでは,許容項P_kより,式(1)のf_kに重きをおいて記述している.f_kは許容項P_kが(a)で生成されたものなら,P_kに含まれる真理値0をとるセルからなる関数を表し,(b)で生成されたものなら,P_kに含まれる真理値1のセルからなる関数

*A Method for Realizing Logic Circuits with NAND Gates alone (MA)

†Takashi MIYAGOSHI , Hideo MATSUDA , Toyomasa HATAKEYAMA , Tatsuo YAMABUCHI , Yoshio NAKAJIMA
‡Takehiro MASUNO
§Faculty of Engineering, Toyama University
¶KYOCERA Corporation

を表す. f_kを一つのＳＢＤＤに順次記憶していき, すでに, 記憶した関数 f'_kと一致したら,f'_k を出力するゲートあるいは部分回路ができているのだからこれを利用して {回路の接続} をして, 簡略化をはかる.

関数 $F_s = t_1 + t_2 + \ldots + t_m$が積和項表現で与えられているとしたら,$F_s$(およびその否定$\bar{F}_s$) を表すＳＢＤＤを作る. そして, $f = F_s$とおいて, 下記の手順 TNAND(f, r, e) を呼ぶ. 但し, 初め, r=1, e=1 とする. また, F_s^e は e が偶数なら F_s, e が奇数なら$\bar{F}_s$を, それぞれ, 表すとする. また, ある関数 FについてＳＢＤＤで表すと定数節点 0 だけになる (関数をマップで表すと, 真理値 1 をとるセルがない) とき, 関数 Fは ϕ (空) であるといおう.

『　　TNAND(f, r, e)

[許容項の生成法] により m 個の許容項 $P_r, P_{r+1}, P_{r+2}, \ldots, P_{r+m-1}$が生成したとする. 各許容項 $P_k(k = r, r+1, r+2, \ldots, r+m-1)$ について, 以下の操作を行う. 但し, [許容項の生成法] の中で論理照合の **B-2** での一致があった許容項については行わない.

$$f_k = P_k \cap F_s^e \qquad (1)$$

f_k が ϕ (空) なら RETURN,f_kがϕでなければ,

A : f_k をＳＢＤＤにして, 論理照合を行う.

A-1 論理照合で一致しなければ, f_k を実現するため G_kの後続ゲートを仮に番号未定の G_x としておく. そして, $r = r + m, e = e + 1$ とおいて, TNAND (f_k, r, e) を呼ぶ.

A-2 論理照合で一致すれば {回路の接続} をして, RETURN.　』

但し, [許容項の生成法] $j = r$とおく.

1) $f' = f_k$, $f = f_k$ とする.

2) fを表すＳＢＤＤで, 根から定数節点 1 にいたる道の中で 1 枝を通る回数の 1 ばん少ない道を選ぶ.

3) その道で 1 枝を通った変数の肯定リテラルの積項を許容項 P_j とする.

B : $f_j = f' \cap P_j$ として, f_j をＳＢＤＤにして, 論理照合を行う.

B-1 f' と論理照合で一致 $(f' = f_j)$ すれば,f_kは唯 1 個の許容項 P_jしか生成しない場合にあたるので, **A-1** で G_xとおいたゲートの番号を G_jときめる.

B-2 f'以外との関数と論理照合で一致すれば,f_kを実現する部分回路がすでに存在するので {回路の接続} を行う.

B-3 論理照合でどの関数とも一致しなければ P_j に対応するゲート G_j(G_kの後続ゲート) を持たせる.

$$f = f' \cap \bar{P}_j \qquad (2)$$

fがϕなら, 許容項の生成はこれで終わる. fがϕでなければ, **B-2** での一致がなければ, $j = j + 1$ として, また, **B-2** での一致があれば, $j = j$として, **2)** へいく.

以上が本方法のアルゴリズムである. 出力数が N_0なら, 各 $F_s(S = 1, 2, \ldots, N_0)$ について, 繰り返す.

4　実験結果

プログラム (FORTRAN) 化し実行した結果の 1 例を表 1 に示す. RAN1, RAN2, RAN3 はマップ上における真理値 1 の割合がそれぞれ 0.2, 0.35, 0.5 となるように乱数により発生した 3 個の関数の平均値である. 本方法の計算時間は積和項表現の手法の場合より 5 割程短縮する. 但し, 初期回路が得られるまでの比較である. MLP4, ADR4 は 4 ビット乗算回路および加算回路へ適用した例であるが, この場合は回路の簡単化を行っている.

プログラムの充実, 他の方法との比較などやるべき多くの課題が残っている.

表 1 : 実験結果

関数	入力数 出力数	ゲート数	計算時間 (S)
RAN1 (0.2)	12 1	3107	241 (491)
RAN2 (0.35)	12 1	3688	405 (801)
RAN3 (0.5)	12 1	3785	475 (943)
MLP4	8 8	213	29
ADR4	8 5	76	3

(0.2,0.35,0.5):真理値表濃度
(　) は積和項表現の手法による計算時間

参考文献

[1] 松田他,"多段ＮＡＮＤゲート回路の一設計法", 情処研報, DA-67, 1993.