設計自動化49－1
(1989.10.24)

多値SUM-PLA簡単化の一手法

A Minimization Technique for Multiple-Valued SUM-PLAs

井口幸洋　土井強　丸矢英之　向殿政男
Yukihiro IGUCHI, Tsuyoshi DOI, Hideyuki MARUYA and Masao MUKAIDONO
明治大学理工学部
School of Science and Technology, Meiji University

あらまし　多値ＰＬＡで実現されるMIN-SUM形多値論理式の簡単化手法を提案する．本手法は，積項を併合・整形することにより項数を順次削減するアプローチを採用している．従来の発見的手法に比べると，多くの場合に，最小解が得られる可能性のあることが計算機実験により明らかになった．

Abstract　We propose a minimization technique for the multi-valued MIN-SUM expressions, which can be realized on the Multiple-Valued Programmable Logic Arrays. This technique employs such an approach that reduces the number of terms by iterative merge and reshape of product terms. Preliminary experimental results indicate that this technique may achieve the absolute minimum more than the conventional heuristic methods.

1．はじめに

２値の多出力組合せ論理関数を実現するために，ＰＬＡ(Programmable Logic Array)がＶＬＳＩ内で多用されている[1]．２値のＰＬＡではＡＮＤ－ＯＲ演算が行われる．一方，多値の論理関数を実現する多値ＰＬＡ(Multiple-Valued PLA)が提案されている[2][3]．多値ＰＬＡを実現する場合，電流モードの多値回路，例えば，ＣＣＤ(Charge-Coupled Device)技術を用いると，このＰＬＡが実現する論理演算は，ＭＩＮ－ＳＵＭ演算となる[2]-[4]．このＰＬＡのコストを減らすためには多値論理式の簡単化を行えばよい．論理式の最簡形を求める手法が提案されているが，極めて多くの計算時間を必要とする[5]．そこで，いくつかの発見的な手法が提案されている[6]-[8]．しかし，これらの発見的な手法には，最簡形を導き出す割合が低いなどの問題がある[8]．

組合せ問題の解法の１つに繰返し改善法があるが，これは局所的最適解に陥ってしまうおそれがあることが知られており，いろいろな出発点から解を求める必要がある．ところで，最近，この局所的最適解に陥らずに解く新しい方法として，ｼﾐｭﾚｰﾃｯﾄﾞｱﾆｰﾘﾝｸﾞ法[9]が注目を集めている．また，ＳＨ法(Sequential Heuristic Method)[10]も提案されている．どちらも，解の良さを表す評価関数が悪くなる変更でも，ある条件のもとでその変更を採用し，問題を解いている．

本稿で提案する論理式の簡単化手法は，論理式の積項を併合・整形することにより項数を順次削減するアプローチを採用している．本手法でも，局所的最適解に陥らないように，削減過程の状態を制御し，項数が減少しない併合・整形も採用している．第２章で本稿で取り扱う論理式とその表記法について述べる．第３章では，論理式の簡単化手法を説明する．第４章では，計算機実験により，従来の発見的手法に比べると，本手法が多くの場合に最小解が得られる可能性のあることを示す．

2. 4値論理式と表記法

本稿で提案する簡単化手法は，一般にｒ値の論理関数にまで適用可能である．しかし，現在提案されている多値ＰＬＡ[2][3]が４値であるのと，簡単のために，本稿では４値を例にとり論ずる．この多値ＰＬＡは，２値のＰＬＡでのＡＮＤ演算が多値ＰＬＡでのＭＩＮ演算に，ＯＲ演算がＳＵＭ演算に対応している．

まず，４値論理関数を４値論理式で表現する方法を示す．

今，真理値０，１，２，及び，３からなる集合をＶ，

$V = \{0, 1, 2, 3\}$,

とする．また，$X = \{x_1, x_2, \cdots, x_n\}$ をｎ個の変数の集合とする．ここで取り扱う論理関数は，$V^n \rightarrow V$ への写像となっている．

２項演算ＭＩＮ，及び，ＳＵＭ演算は以下のように定義される．

$x_1 \cdot x_2 = MIN(x_1, x_2)$,

$x_1 \diamond x_2 = SUM(x_1, x_2)$

$= MIN(3, x_1 + x_2)$.

$x_1 \cdot x_2$は，x_1とx_2との小さい方の値を返す．$x_1 \diamond x_2$は，x_1とx_2との算術和を返すが，この値が３以上になる場合は３を返す．表１(a)(b)にＭＩＮ演算及びＳＵＭ演算の真理値表を示す．

X_2 \ X_1	0	1	2	3
0	0	0	0	0
1	0	1	1	1
2	0	1	2	2
3	0	1	2	3

(a) $x_1 \cdot x_2 = MIN(x_1, x_2)$

X_2 \ X_1	0	1	2	3
0	0	1	2	3
1	1	2	3	3
2	2	3	3	3
3	3	3	3	3

(b) $x_1 \diamond x_2 = SUM(x_1, x_2)$

表１　ＭＩＮ演算とＳＵＭ演算の真理値表

リテラル演算は以下のように定義される．

$$x^{ab} = 3 \quad (a \leqq x \leqq b \text{の場合})$$

$$= 0 \quad (\text{上記以外の場合})$$

この演算には，

x^{00}, x^{01}, x^{02}, x^{03}, x^{11}, x^{12}, x^{13}, x^{22}, x^{23}, x^{33}

の１０種類のパターンが存在する．

積項γΦとは，定数$\gamma \in \{1, 2, 3\}$とリテラルの集合であるΦ＝$\{x_1, x_2, \cdots, x_n\}$の各要素とをＭＩＮ演算で結んだ項である．この積項をＳＵＭ演算で結んで論理式が構成される．

例１． 表２の真理値表で示される１変数論理関数を表す論理式を求める．

$$f(x) = 1 x^{00} \diamond 3 x^{11} \diamond 2 x^{22} \diamond 3 x^{33}$$

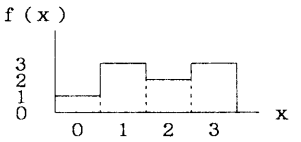
となり，これは４項で実現できたわけであるが，これは３項で実現できる（図１参照）．

$$f(x) = 1 x^{01} \diamond 2 x^{12} \diamond 3 x^{33}$$

図１(ｂ)からわかるように，ｘ＝１の部分で真理値１と２が加算されることで，まったく新しい３という真理値が作られている．これがＳＵＭ演算の特徴である．

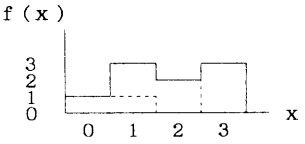
x	f(x)
0	1
1	3
2	2
3	3

表２



$f(x) = 1 x^{00} \diamond 3 x^{11} \diamond 2 x^{22} \diamond 3 x^{33}$

図１　（ａ）



$f(x) = 1 x^{01} \diamond 2 x^{12} \diamond 3 x^{33}$

図１　（ｂ）

例２． 表３の真理値表で示される２変数論理関数を表す論理式を求める．

$$f(x_1, x_2) = 1 x_1^{01} \cdot x_2^{01} \diamond 1 x_1^{12} \cdot x_2^{01} \diamond 3 x_1^{11} \cdot x_2^{33}$$

これ以降ＭＩＮを表す・を簡単のために省略する．

$$f(x_1, x_2) = 1 x_1^{01} x_2^{01} \diamond 1 x_1^{12} x_2^{01} \diamond 3 x_1^{11} x_2^{33}$$

X_2 \ X_1	0	1	2	3
0	1	2	1	0
1	1	2	1	0
2	0	0	0	0
3	0	3	0	0

表３

3. 4値論理式の簡単化

3.1 従来の簡単化手法

単出力の2値論理式の簡単化を考えてみよう．各積項は1番大きな積項である主項にまで，まず拡大する．次に，すべての主項の集合の中から，最も少ない個数で関数を被覆できる組合わせが最簡形であった．しかし，多値論理式では，1番大きな積項のみを組合わせても最簡形が求まらない，つまり，その内項までをも組合わせの考慮に入れなければ最簡形は求まらない場合がある[5]．これが，多値論理式を簡単化する問題を複雑にしている理由である．

多値論理式の代表的な簡単化手法が文献[5]で比較されている．これらは，

(1) 最簡形を求める手法(absolute minimization)

(2) 発見的な手法

(a) Pomper & Armstrong の手法[6]

(b) Besslich の手法[7]

(c) Dueck & Miller の手法[8]

である．

このうち(1)の手法は，最簡形が求まるが極めて多くの計算時間を必要としている．(2)の3つは発見的な手法であり，4値2変数論理関数という単純な関数であっても最簡形を導き出す割合が高くないことが実験的に示されている[5]．基本的には(1)(2)とも，与えられた関数から内項を選んで行き，すべてが被覆されるまで続けることで簡単化を行う．(1)と(2)の相違点は，(1)ではこの手続きを最簡形に含まれる可能性のある内項すべてについて行うが，(2)では内項の中から一定の基準を満たすものを発見的に選択する点である．

3.2 本稿で提案する簡単化手法

本稿で提案する方法は，基本的には積項数を評価関数にして，

(1) 2つの積項を1つに併合すること，

(2) 2つの積項を組合わせてそれを整形すること，

を繰返し行うことで解を得る方法である．

論理式の併合・整形操作

論理式の併合・整形操作は以下の通りである．これを繰り返して簡単化を行う．

①論理式fから2つの積項t_1，t_2を任意に選択する．

②t_1，t_2それぞれの内項が隣接または重複していれば，それを併合して新たな積項t_iを生成する．併合しうる内項の対が存在しない場合は手続きを終了，複数個存在するときは任意の対を選択する．

③t_1，t_2の内項の内で，t_iに含まれないものがある場合は，それらを最小項として記録する．

④fからt_1，t_2を削除，t_iと③で生成された最小項を付加する．

例3． 論理式$f(x)=1^{00}x \diamond 3^{11}x \diamond 2^{22}x \diamond 3^{33}x$に併合・整形操作を施した例を図2(a)(b)に示す．

① t_1，t_2として$1^{00}x$と$3^{11}x$を選択

② t_iとして$1^{01}x$を生成

③ t_iに含まれない，最小項$2^{11}x$を記録

④ $1^{00}x$，$3^{11}x$を削除，$1^{01}x$と$2^{11}x$を付加

結果は，$f(x)=2^{22}x \diamond 3^{33}x \diamond 1^{01}x \diamond 2^{11}x$となる．

①' t_1，t_2として$2^{22}x$と$2^{11}x$を選択

②' t_iとして$2^{12}x$を生成

③' t_iに含まれない，最小項は存在しない

④' $2^{22}x$，$2^{11}x$を削除，$2^{12}x$を付加

結果は，$f(x)=3^{33}x \diamond 1^{01}x \diamond 2^{12}x$となる．

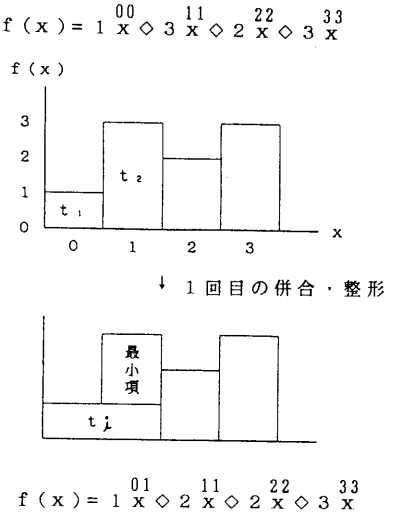


図2(a) 論理式の併合・整形の例

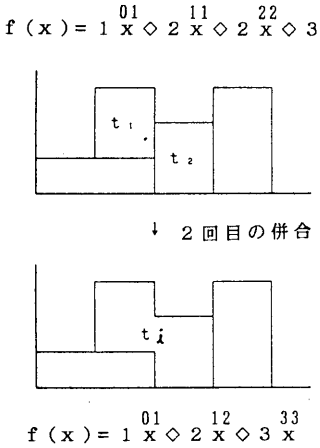


図２（ｂ）論理式の併合・整形の例

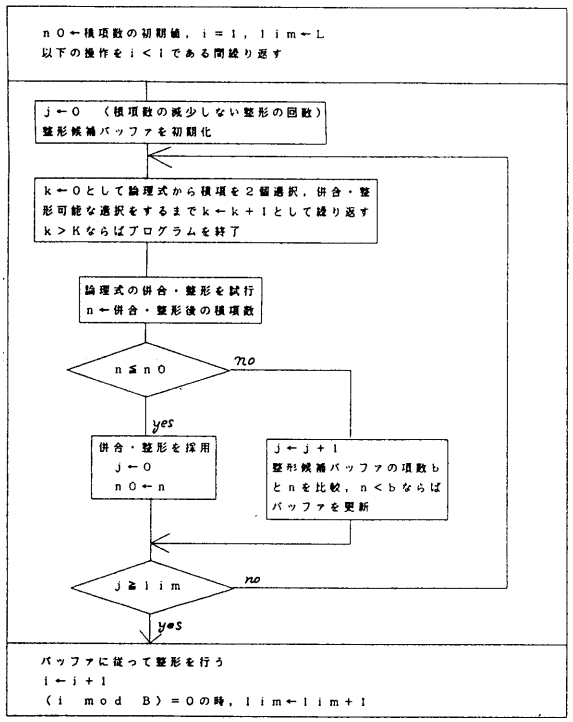


図３．論理式の簡単化の制御過程

論理式の簡単化制御過程

簡単化の過程を図３に示す．この時，必ずしも積項数が減少しない整形であっても，ある判定条件のもとでその整形を採用することに注意されたい．これは，局所的最適解に陥ることを防ぐ為である．実際には，limをlim←lim+1で増加させ，最初のうちは積項数が増加する変形でもどんどん採用し，あとになればなるほど悪くなる変形は採用されにくくなることを実現している．図中のパラメータB，I，K，及び，Lは，対象とする関数の入力等に応じて適宜設定する．実際のプログラムでは実行時間に制限を設けて，ある時間以上経過すると実行を打ち切っている．よって，実行の途中で求めた論理式で最小のものを記録しておく必要がある．本手法は，3.1の発見的な簡単化手法とは異なり，むしろ，集積回路内部の配置配線の改善問題等の組合わせ最適化問題に用いられる手法[9][10]とよくにている．

4．実験結果

本章では，第３章で提案した論理式簡単化手法の計算機実験の結果を示す．本章で示す実験は全てｓｕｎ３／６０ワークステーション上で行い，プログラムはＣ言語を用いて開発した．

従来手法との比較実験

図４は，２変数４値論理関数で，０でない出力値の値を１から１５まで，それぞれ５００個ランダムに生成した関数について簡単化した結果を示す．図４は，各々の手法で最簡な論理式が５００個中何個導けたかを示している．本図から明らかなように本論文で提案する手法が他の発見的な手法に比べ最簡形を導く可能性が高いことを示している．なお，この実験結果は，各パラメータをそれぞれＢ＝１００，Ｉ＝Ｋ＝１００００，Ｌ＝４として実行時間を６０秒以下に制限して得られたものである．

表４に参考のためにabsolute minimization に要した時間の平均値と本手法に要した時間の平均値を示す．ただし，本稿で提案する方法は，はじめに設定した制限時間内で簡単化を進めるという手法の性格から，プログラムの終了時ではなく最も積項数が減少したときのＣＰＵ時間を測定した．

本手法の特徴

本論文で提案する手法への各パラメータの影響について検討する．図５に４値３変数論理関数をＩ＝Ｋ＝１００００で，ＢとＬの値を変更して実験した結果を示す．

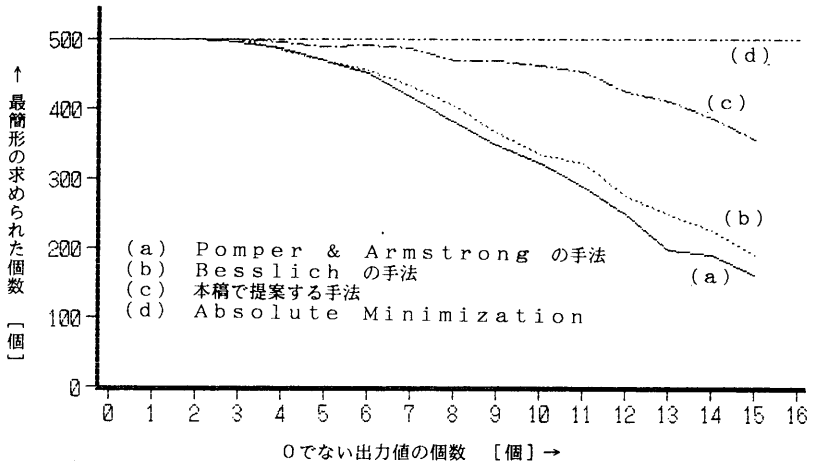


図４．関数の０以外の出力値の個数と最簡な論理式の求められた個数との対比

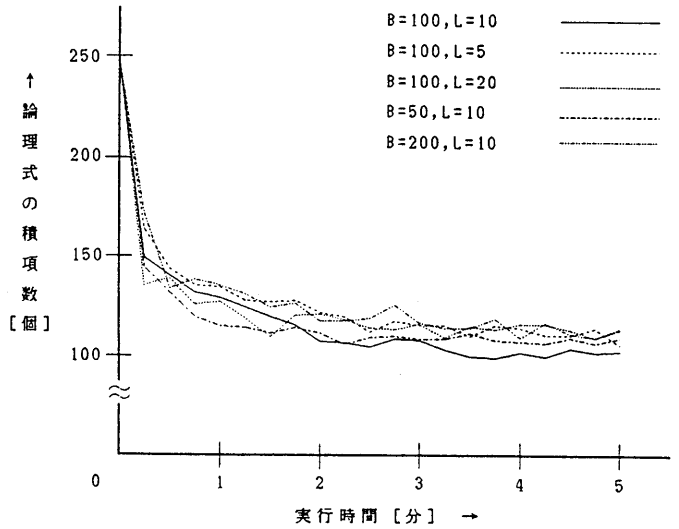


図５．パラメータを変更した際の積項数の推移の変化

０以外の出力値 [個]	簡単化に要した平均ＣＰＵ時間 [秒]	
	Absolute Minimizatinon	本論文で提案する手法
6	0．06	0．02
7	0．11	0．02
8	0．31	0．06
9	0．79	0．10
10	2．56	0．23
11	9．03	0．50
12	26．49	1．00
13	83．54	1．70
14	175．89	2．03
15	531．96	2．50
16	1108．43	2．37

表４．簡単化に要したＣＰＵ時間の比較

まず，Ｉ，及び，Ｋは終了条件であり，これが大きいほど論理式の整形回数が増加する．従って，十分大きい値を用いなければ，あまり簡単化が進まない内に終了することとなる．次に，Ｂについては大きな値を使用すると整形回数が増加するので積項数が減少するまでに時間を必要とし，逆に小さいと整形回数が減少し簡単化が十分に行われない可能性がある．Ｌについては，大きな値を用いると論理式の整形回数が減少し，小さな値を用いると増加する．ＢとＬについては実験的に値を決定する必要がある．

5．まとめ

多値論理式の簡単化の一手法を提案した．本手法では，数変数の多値論理式を実際的な時間でほぼ最簡形にまで簡単化できる．しかし，多値ＰＬＡで実現される論理式の簡単化を行うためには，数十変数まで処理できなければ実用にはならないと考えられる．よって，まったく異なったアプローチで考えることも必要である．

今後の課題としては，併合・整形を行うために積項の対をランダムに選ぶが，無駄な積項の選択をなるべく行わない方法を検討すること，積項をいくつかのグループに分け，問題を分割して高速化を図ることなどである．

謝辞　貴重な御意見をいただきました本学山田輝彦助教授に感謝いたします．なお，本稿の計算の一部は明治大学情報科学センターの計算機を使用した．

参考文献

[1] 笹尾勤，”ＰＬＡの作り方・使い方”，日刊工業新聞社 (1986).

[2] P. Tirumalai and J.T. Butler, "On the realization of multiple-valued logic functions using CCD PLA's", Proc. of 14th International Symposium on Multiple-Valued Logic, pp. 33-42 (1984).

[3] H.G. Kerkhoff and J.T. Butler, "Design of a high-radix programmable logic array using profiled peristaltic charge-coupled devices", Proc. of 16th ISMVL, pp. 128-136 (1986).

[4] E.A. Bender, J.T. Butler, and H.G. Kerkhoff, "Comparing the sum with the max for use in four-valued PLA's", 15th ISMVL, pp. 30-35 (1985).

[5] P. Tirumalai and J.T. Butler, "Analysis of minimization algorithms for multiple-valued programmable logic arrays", 18th ISMVL, pp. 226-236 (1988).

[6] G. Pomper and J.A. Armstrong, "Representation of multi-valued functions using the direct cover method", IEEE Trans. on Comput. pp. 674-679 (Sep. 1981).

[7] P.W. Besslich, "Heuristic minimization of MVL functions: a direct cover approach", IEEE Trans. on Comput., pp.134-144 (Feb. 1986).

[8] G.W. Dueck and D.M. Miller, "A direct cover MVL minimization using the truncated sum", Proc. of 17th ISMVL, pp. 221-225 (1987).

[9] S. Kirkpatrick, C.D. Gelatt,Jr., and M.P. Vecchi, "Optimization by simulated annealing", Science, Vol. 220, pp.671-680 (1983).

[10]宮下，小野沢，上田：”シミュレーティドアニーリング法と従来の配置手法との実験的比較”，VLD88-6，pp.41-48 (1988).

[11]井口，土井，向殿：”多値ＰＬＡの簡単化”，ＦＴＣ研究会，(Jan. 1989).