関数変換法によるプライムインプリカントの自動導出について

宮腰　隆・松田秀雄・野末　裕※

緒　　言

論理設計を行なう場合，与えられた関数のプライムインプリカント（以下PIと略記）を求める必要がある。このPIを計算機で求める通常の方法は，（Quine）McCluskey[1]法である。一方，変数の数が小さく手計算で求める場合は，視覚的にカルノー図上で行なうのが便利である。[2]

ここに報告する関数変換法は，カルノー図上で行なう操作にできるだけ立脚し，簡単な繰り返し計算を実行することにより短時間で解を得ることができる計算機向きアルゴリズムである。

関数変換法の処理過程は，論理積操作とPIの照合操作からなるが，このうち後者の軽減には照合配列の技法が効果的であり非常に優れた結果を得ることができる。

1．原　　理

1.1　基礎原理

関数Fが図1のカルノー図で与えられているとする。但し，ここで○印のセルでtrueを示す。又，各セルの数字は，表1の変換前の欄のように，座標ベクトル（x_1，x_2，x_3，x_4）で1の成分の少ない順に，もし同一の1の数を持つ場合には2進数字とみなして小さい順に並べた順位番号を表わす。

否定形の変数を含まない変数の積項で表わされるキューブを許容キューブというが，これらはセル番号で表現できる。許容キューブ1（カルノー図全体）はセル1に対応するものとし，$P(1)$と表わす。許容キューブx_4は座標（x_1，x_2，x_3，x_4）＝（0，0，0，1）のセル2に対応するものとして$P(2)$と表わす。許容キューブx_3は（0，0，1，0）のセル3に対応するものとして$P(3)$と表現する。以下同様に，積項が示す領域の最小のセル番号で許容キューブを表現するものとすれば許容キューブ$x_1x_2x_4$は$P(14)$となり，$x_1x_2x_3x_4$は$P(16)$となる。

このように約束すると$i<j$なら，$P(i)\geqq P(j)$となる。

但し，大小関係はキューブが含むセルの数の大小関係によって決める。なお，セルの数は2^N（Nは変数の数）あるが，丁度同じ数だけ許容キュー

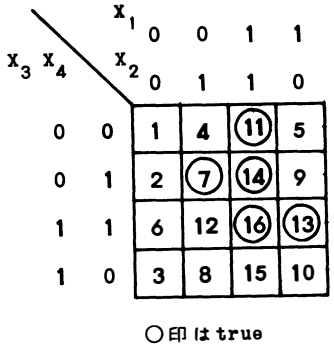
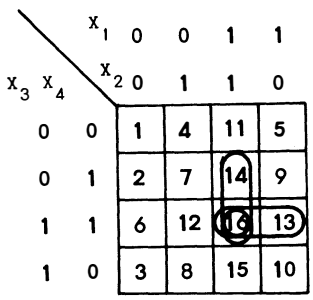


図1　関数Fの例



$P(13)$，$P(14)$および$P(16)$

図2

※　三ツ葉電機

ブもある。$P(i)$をセルiを通る許容キューブといおう。

さて，これらの許容キューブの1つ$P(i)$と関数Fとの論理積をとり

$$|\text{関数 } F| \cap |\text{許容キューブ} P(i)| = |\text{許容キューブ} P(i)| \qquad (1)$$

記号 $\cap$;　論理積

式(1)が成り立てば，$P(i)$はインプリカントである。ここで，更に$P(i)$が他のインプリカントに含まれなければPIであると判定できる。

先の操作を論理積（操作），あとの操作をPIの包含関係の照合（操作）と呼ぶことにする。

図1の関数Fに大きい許容キューブから順次$P(1)$，$P(2)$，……と式(1)の論理積をとっていくと，$P(13)$〔セルの集合で表わして$|13, 16|$〕，$P(14)$〔$|14, 16|$〕，及び$P(16)$〔$|16|$〕がインプリカントであることがわかる。（図2参照）

このうち$P(16)$は他のものに含まれるので除去すると$P(13)$と$P(14)$がPIとなる。

さて，この関数では$x_2 x_3' x_4$〔$|7, 14|$〕と$x_1 x_2 x_3'$〔$|11, 14|$〕もPIであるが，許容キューブではないので式(1)の操作だけでは求まらない。そこで関数変換の技法を用いる。

例えば，セル14の座標は（1，1，0，1）であるから，1をとる変数はそのまま，0をとる変数はその否定に

$$x_1 \to y_1, \quad x_2 \to y_2, \quad x_3' \to y_3, \quad x_4 \to y_4$$

（変換14）

と変数変換する。この変換で，表1のように各セルのベクトル座標が変わりセル番号も変わる。セル番号のこの変化は置換

表1　セル14による変換

変換前					変換後				
セル番号	2進ベクトル X_1	X_2	X_3	X_4	2進ベクトル Y_1	Y_2	Y_3	Y_4	セル番号
1	0	0	0	0	0	0	1	0	3
2	0	0	0	1	0	0	1	1	6
3	0	0	1	0	0	0	0	0	1
4	0	1	0	0	0	1	1	0	8
5	1	0	0	0	1	0	1	0	10
6	0	0	1	1	0	0	0	1	2
⑦	0	1	0	1	0	1	1	1	⑫
8	0	1	1	0	0	1	0	0	4
9	1	0	0	1	1	0	1	1	13
10	1	0	1	0	1	0	0	0	5
⑪	1	1	0	0	1	1	1	0	⑮
12	0	1	1	1	0	1	0	1	7
⑬	1	0	1	1	1	0	0	1	⑨
⑭	1	1	0	1	1	1	1	1	⑯
15	1	1	1	0	1	1	0	0	11
⑯	1	1	1	1	1	1	0	1	⑭

$$T_{14} = \begin{pmatrix} 1, & 2, & 3, & 4, & 5, & 6, & 7, & 8, & 9, & 10, & 11, & 12, & 13, & 14, & 15, & 16 \\ 3, & 6, & 1, & 8, & 10, & 2, & 12, & 4, & 13, & 5, & 15, & 7, & 9, & 16, & 11, & 14 \end{pmatrix}$$

によって表わされ，○印のセル番号が変わるので関数Fが変化を受ける。この変化をFのセル14による関数変換といい，$F_{14} = T_{14} \circ F$と表わす。図3はF_{14}を示す。このF_{14}に積項x_3（セル14で0をとる変数で図3参照）に含まれるセル（3，6，8，10，12，13，15，16）を通る各許容キューブを大きいものから順次$P(3)$，$P(6)$，……と式(1)の論理積をとっていく。すると，$P(12)$，$P(15)$，$P(16)$がインプリカントとなるが，$P(16)$は他に含まれ，$P(12)$，$P(15)$が残る。これをセル番号で表わすと，$|12, 16|$，$|15, 16|$でTの逆置換を行なうとFのインプリカント$|7, 14|$，$|11, 14|$となる。すでに得られているPIと包含関係の照合を行なうと，これがそのままPIであること が

わかる。

本例では，セル14による唯一回の変換だけでPIがすべて求まったが一般にはこのように簡単ではない。セル14と同様にしてセル i の変換が必要である。

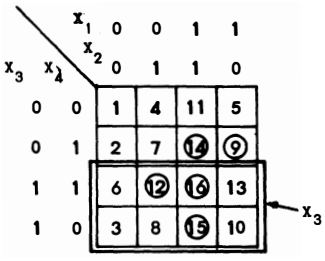


図3　F_{11}と積項x_3

1.2　アルゴリズムと諸性質

〔定義1〕セル i のベクトル座標で1をとる変数はそのまま，0をとる変数はその否定変数になるよう変数を変換する。

このセル iの変換でセル番号が変わり，これは置換 T_i で表わされる。T_iで関数Fの○印のセル番号が変わるが，この変化後の関数を$F_i = T_i \circ F$とおいて，Fのセル iによる変換と呼ぶ。

〔定義2〕セル iの座標ベクトルで0をとる変数の肯定形だけの積項で表わされるカルノー図上の部分を部分図 iと呼び，部分図 iのセルを小さい順にi_1，i_2，……，i_pとする。

従って，セルm＜セルnなら，部分図m≦部分図nの関係がある。

ここで関数変換法の手順は，大きい番号のセル iから順次関数を変換し，その都度そのセルに関する部分図の許容キューブと式(1)の論理積をとっていく。この場合論理積の総数は3^Nとなるが，次の諸性質によってその数を相当数減少させることができ，他のアルゴリズムに比肩するものとなる。

（性質1）○印のセルについてのみ関数変換を行えばよい。

（性質2）各変換ごとに部分図内における○印のセルを通る許容キューブとだけ論理積をとればよい。

（性質3）ある許容キューブがインプリカントなら，それに属するセルを通る許容キューブの論理積は不用である。

Fが○印のセルの少ない関数なら，（性質1）によって変換する回数が少なくなり，ひいては論理積数が減る。又，Fが○印のセルを多数もつ関数なら，（性質3）によ

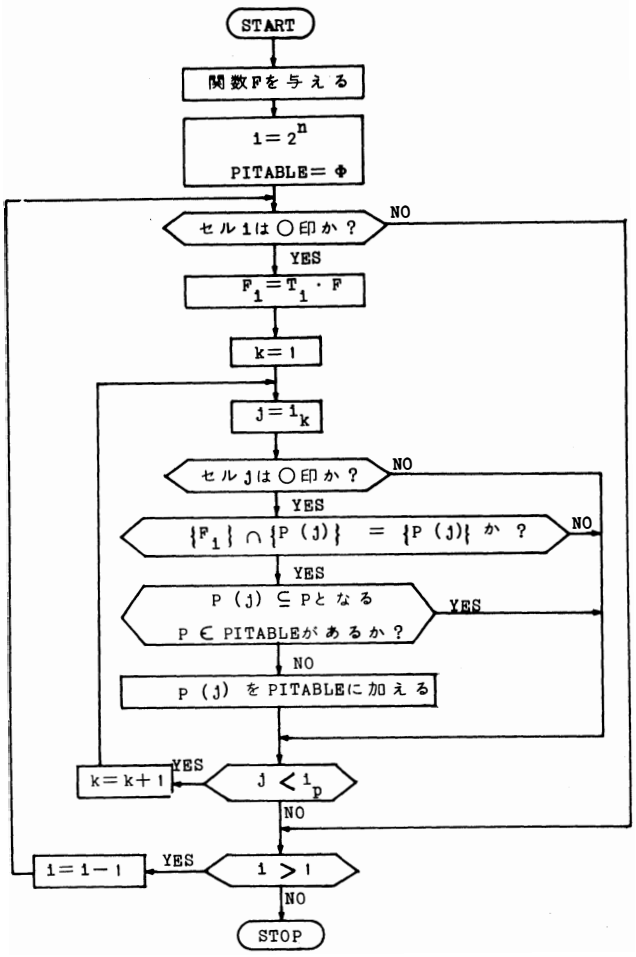


図4　関数変換法の流れ図

り論理積数が減る。

例えば，関数Fがただ1個のセルだけ○印をもつなら，（性質1）と（性質2）により論理積数は1回だけとなる。又，全セルが○印なら，（性質1）と（性質3）によって各部分図ごとに論理積数を1回づつ，計2^N回でよい。

図4は関数変換法の流れ図である。ごく大まかなもので，PIの照合前の逆置換などは省略してある。

2．結　　果

2．1　計算例による比較

関数F_{3k}を変数の数Nが$N=3k$（$k=1$，2，3，……）でそれを構成するすべてのキューブがそれぞれk個の肯定，否定，任意変数からなる論理積項で表わされるものとしよう。F_{3k}はN（$=3k$）変数関数のうちで最も多数のPIをもち，$N=9$の時，1680個にもなる。

この関数で計算時間を比較してみると，関数変換法（2分17秒97ミリ秒），McCluskey法（15分27秒249ミリ秒）である。関数変換法が非常に優れたアルゴリズムである一例である。ところが，各方法とも関数の形によって得手不得手がある。図5は関数の形によって計算時間が異ってくる様子を表わしている。$N=9$の場合で，カルノー図を左右に両分すると256個のセルからなるキューブが2つできる。ついで上下に2分すると128セルのキューブが4個できる。これを又左右に両分……，上下に両分……としていくと次第に小さくなり，最後に1個のセルだけからなる最小のキューブに到る。ここで，各分割ごとに等しい数のfalseのキューブとtrueのキューブにわけ，それぞれカルノー図上交互に配置するようにすると，9つの関数ができる。図5の横軸はキューブの大きさで，これらの関数を区別するよう目盛ったもので，縦軸はそれらの計算時間を示す。

これからわかるように，小さなキューブの関数ではMcCluskey法の方が速くPIが求まる。特に1セルキューブの関数では論理積数，PIの照合数共に0になるのに対し，関数変換法ではそれぞれ4921，及び，32640となる。McCluskey法のこの優位性もキューブが大きくなるにつれて失われ，256セルの

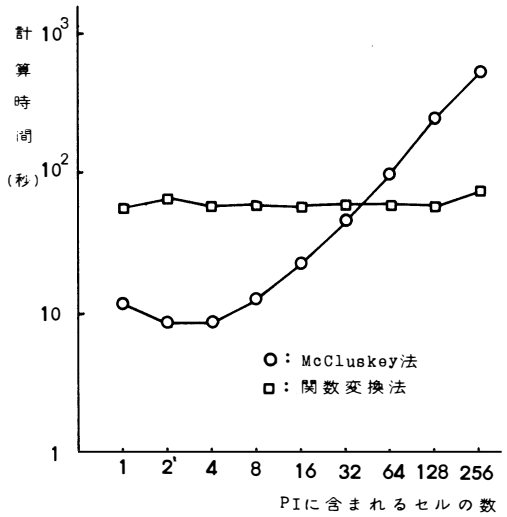


図5　PIの大きさと計算時間との関係
（$N=9$，trueのセルの割合0.5）

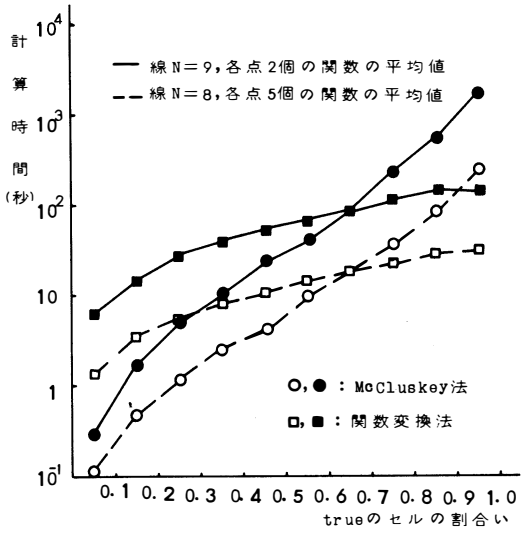


図6　trueのセルの割合いと計算時間

キューブの関数に到っては，関数変換法に比べ7倍以上も時間がかかる。この場合の論理積数，及び，PI照合数は1807760，及び，2770142であるのに対し，関数変換法ではそれぞれわずか256，及び，255にすぎない。この様にキューブが大きくなると論理積数，PIの照合数が共に著しく増大して不利になるのに対し，関数変換法ではこれら一連の関数ではきわだった差がないので計算時間はほぼ一定となる。

次に全体のセルの中でtrueの現われる確率をいろいろ変えて，McCluskey法と関数変換法との計算時間を対比させてみたのが図6である。各点はそれぞれ横軸の目盛りでtrueが表わされるよう乱数で作った関数の$N=8$では5回の平均，$N=9$では2回の平均時間である。確率が小さければPIのキューブも小さく，McCluskey法の方が速く求まるが，確率が大きくなるとキューブも大きくなり，計算時間は指数関数的増大する。関数変換法も多少とも計算時間の増加傾向はあるが，確率が大きい所で頭打ちとなる。これは，**1．2**で述べた(性質3)により論理積数の増加が防がれる為と考えられる。

以上で示したように，関数のPIとなるキューブの大きさ，trueのセルの割合いなど関数の形によって，各アルゴリズムの計算時間が異る。そこで多数の関数の平均計算時間で比較するのが妥当と考えられる。表2にこれを示す。4～7変数については無作為に選んだ100個の関数の平均値，8変数は50個の関数の平均値，9変数は20個の関数の平均値である。関数変換法は6変数以上で明らかに有利であると考えられる。

2．2　照合操作の軽減

関数変換法でPIを求める処理過程は論理積操作と，PIの照合操作からなることを先に述べた。これらの数がどのような大きさになるかを示したのが図7，図8である。

計算時間を決める要因は主として論理積数であるが，多変数でtrueのセルの割合いの多い関数ではPIの照合数も無視できない影響を与える。PIの照合数は，多変数の関数になると大きくなり，又同じ変数でもtrueのセルの割合いが多くなると増大し論理積数に匹敵あるいはしのぐ値となる。

表2　平均計算時間の比較

変数	McCluskey法 M S MS	関数変換法 M S MS
4	35	46
5	166	185
6	879	765
7	6 − 159	3 − 287
8	41 − 767	15 − 26
9	4 − 45 − 490	1 − 12 − 331

照合数がこのように増加するのは1つのPIが見つかると，それまで得られているものすべてと包含関係を照合しなくてはならないからである。

PIの数をLとすると，PI同志の照合数だけで$L(L-1)/2$となり，これに包含されて除かれるインプリカントも入れると，この数倍の大きさとなる。そこで，もしこのPIの照合操作を除去できるか，あるいは軽減できれば計算時間を短縮することが可能である。

筆者らが，照合配列と称しているものはこの目的に使うもので，原理を簡単の為，3変数の例で述べる。3変数の場合，キューブは3^3（N変数で3^N）個あるが，これらは包含関係で半順序集合となる。これはハッセの図形で書け，図9の様に併合された木構造となる。但し，ここでは関数$f=1$（常にtrue）は別に処理することにして，キューブ1を除外して考える。

図9のキューブに3進数を割り当てる。例えば肯定変数に2，否定変数に1，任意変数に0を割り当てる。図のキューブの下の数字はこれを示し，○印の数字は，更にこれを10進数に変換したものである。このようにして，すべてのキューブに10進数字を対応させることができる。

さて，一方ではDIMENSIONの大きさが3^N-1の配列$R(I)$を用意する。各キューブには10進数字が対応しているから，その数をそのキューブに対応した添字の値Iとする。配列$R(I)$を始め全部

0にしておく。今，あるキューブがPIとわかったら，このキューブに相当する添字の$R(I)$を1にする。それと共にハッセの図形をたどって，包含されているキューブに相当する添字の$R(I)$をすべて1に変える。このような探索は表3の順序で並んだ2進ベクトルを用いると比較的容易に行える。

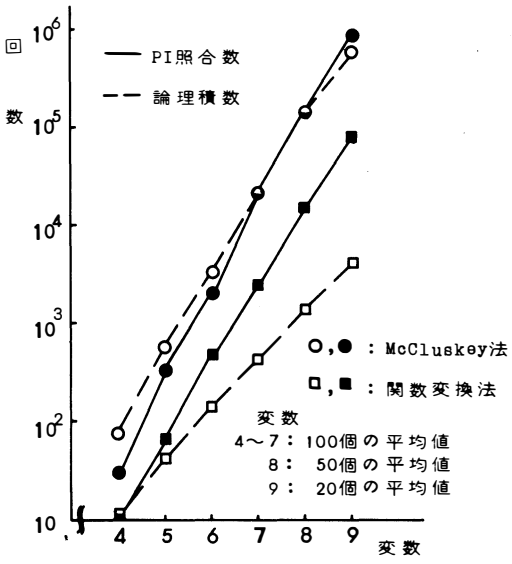


図7　変数の数とPIの照合数との関係

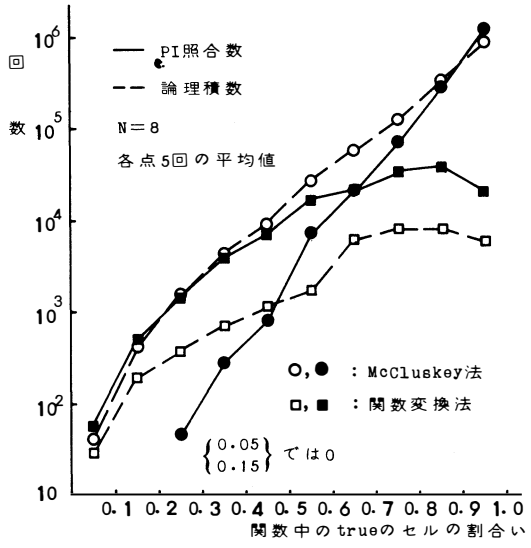


図8　trueのセルの割合いとPIの照合数との関係

表3は，2進ベクトルを1の数の少ない順に，又同一の1の数の場合は2進数とみて小さい順に並べたものである。

例えば，x_3がPIであるとわかったとする。この時，x_3に1のあるベクトルを表3から選ぶ。すると，（0，0，1），（0，1，1），（1，0，1），（1，1，1）の4つが該当する。これらの各ベクトルで，x_3以外で1をとる変数のそれぞれ肯定，否定をとるすべての組合わせからなる積項を考え，それらにx_3をかけたキューブがx_3に包含される全キューブとなる。即ち，（0，0，1）ではx_3以外に1がないのでx_3そのもの，（0，1，1）ではx_2x_3，$x_2'x_3$，（1，0，1）ではx_1x_3，$x_1'x_3$，（1，1，1）では$x_1x_2x_3$，$x_1'x_2x_3$，$x_1x_2'x_3$，$x_1'x_2'x_3$の8個がx_3に含まれる。つまり，$R(8)=R(5)=R(20)=R(11)=R(26)=R(17)=R(23)=R(14)=1$とおく。

セルiで変換される関数F_iについても全く同様に処理されるが，この場合，包含されるキューブの変数をいったんセルiで逆変換してからそのキューブに対応する10進数字の添字の$R(I)$を1に変えねばならない。このように処理された照合配列$R(I)$を用いると次のことがいえる。

キューブと関数との論理積をとる時，それに先立ってこのキューブに対応する配列$R(I)$が0かどうかを確め0なら論理積をとり，1なら次のキューブにいくようにすると，式(1)でインプリカントと判定されたキューブは他のPIに包含されることは決してない。即ち，PIの照合操作が割愛できる。

この方法は，他の方法にも適用でき，例えば，McCluskey法では同一のキューブが途中何度も現われるので，不用なキューブを取り除く操作（ここでは，これもPIの照合操作と呼ぼう）が必要になるが，$R(I)$と類似の原理でこの処理を割愛できる。図7，図8にはMcCluskey法の論理積（キューブ同志が更に大きなキューブに結合できるかどうかを見る操作）数とPIの照合数も示されている。

関数F_{3k}は，$N(=3k)$変数関数のうちで最も多数のPIをもち，$N=9$の時，1680個得られることを先に述べた。

表３　２進ベクトルの並べ方

セル番号	2進ベクトル x_1	x_2	x_3
1	0	0	0
2	0	0	1
3	0	1	0
4	1	0	0
5	0	1	1
6	1	0	1
7	1	1	0
8	1	1	1

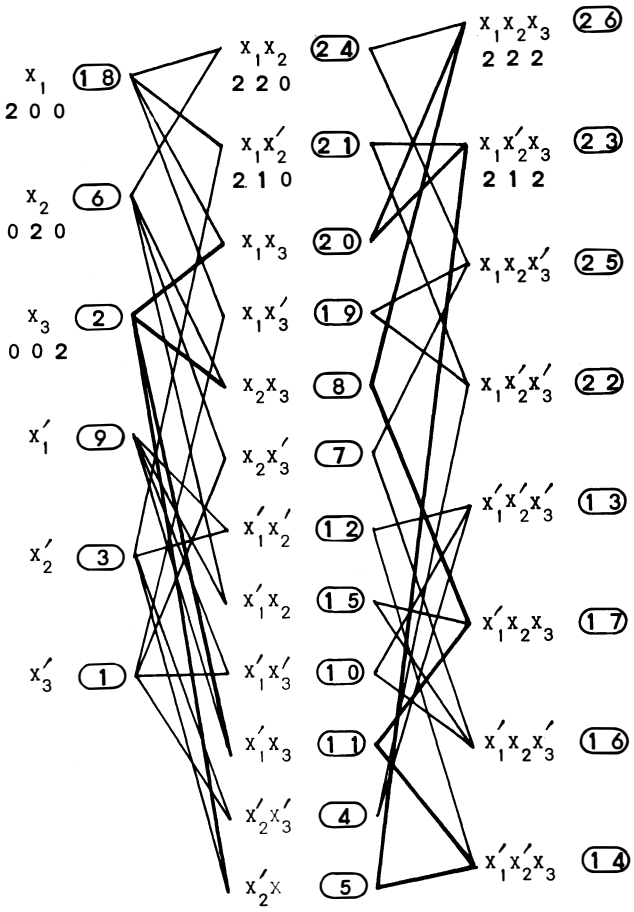


図９　３変数の場合のハッセの図

この関数について，照合配列の効果をみると，関数変換法で２分17秒97ミリ秒（照合配列なし）から１分17秒235ミリ秒（照合配列あり）に，約43.6%計算時間が短縮でき，McCluskey 法では15分27秒249ミリ秒（照合配列なし）から11分32秒337ミリ秒（照合配列あり）に，約25.3%短縮できた。８変数でtrueのセルの割合いの多い関数では，関数変換法，McCluskey 法とも10～15%計算時間を早めることができた。９変数では，更に計算時間を短縮させることができた。

結　　言

本方法は，プログラム化が容易であり占有記憶容量もMcCluskey法のようにPI表以外に中間段階のキューブの為の表を必要としないので少くて済む。

従来から最も標準的な方法として知られている（Quine）McCluskey法は，関数のPIとなるキューブが大きくなるにつれて計算時間の増加が著しい。又，trueのセルの割合いが多いと論理積数，PIの照合数が指数関数的に増大する為，それが計算時間に影響する。

これに対して本方法は，PIとなるキューブの大きさにはほとんど影響されないのでほぼ一定時間で解がえられること，trueのセルの割合いが多くなっても論理積数，PIの照合数が頭打ちとなりそれ程計算時間が増加しないので平均計算時間で優れている。

一方，照合数の軽減を目的とした照合配列の技法は，各アルゴリズムに適用することができその効

果は大きい。

使用計算機は，本学計算センター（FACOM230－45S）である。データをとるのにお世話になったセンターの皆様に感謝する。

参　考　文　献

1）E.J.McCluskey, Jr.; "Minimization of boolean functions", Bell Syst. Tech. J., **35**, P 1417, (1956).

2）M. Karnaugh; "The map method for synthesis of combinational logic circuits", Communication and Electronics, P 593, (Nov. 1953).

昭和53年　電気四学会北陸支部連合大会で一部発表

Automatic Determination of the Prime Implicants by the Method of Function Transformation

Takashi MIYAGOSHI, Hideo MATSUDA, Hiroshi NOZUE

We obtained a new method for automatic determination of the prime implicants of a given Boolean function.

The features can be summarized as follows.

(1) If there are true cells, the function is transformed, on the cells.

(2) On a considering cell, we can work out a partial Karnaugh map.

The results calculated by this method are compared with those applied to the Quine-McCluskey method that is generally known as computer algorithm.

This paper describes in detail the algorithm, the advantages of which are discussed in it.

(1978年10月24日受理)