論理関数発生プログラムについて (1)

松田　秀雄，五味　利彰，宮腰　　隆，
畠山　豊正，中嶋　芳雄

1．はじめに

論理設計の分野では，回路を少しでも簡単にするため，その回路の動作を記述する，論理関数を簡単化する手法の開発が重要であるとされてきた。我々もそのためのいくつかの手法を提案してきた[1),2)]が，その場合，各手法の性能を比較するため簡単に関数を発生できるプログラムが必要となった。もちろん，簡単化手法の性能を比較するためには，ベンチマークの関数を使う方法もあるが，これは特定のかぎられた関数に限定され，大量の関数を発生させて，統計的性質をみるといった場合には不向きである。それでも変数の数が小さい場合には，乱数を使って，真理値表濃度で設定した数だけのTrueとなる最小項を発生させる方法もあるが，これは関数を真理値表で設定することと同じであると考えられ，20変数，30変数といった多変数の関数であるとき，項数が膨大となって発生不可能である。

ここで我々が使ってきたプログラムでは，2値で100変数ぐらい，4値で50変数の関数が組織的に生成できる。項数と成分濃度を与えると種々の関数を生成する。本論文では，プログラムの概要とプログラムを利用するさいの便宜を与えるデータを掲げる。

2．関数発生プログラムについて

2.1　論理関数の表し方

例えば，

$$f_1 = X_1^{\{0,1,3\}} X_2^{\{0,1\}} \vee X_1^{\{1,2\}} X_2^{\{1,2\}} \vee X_1^{\{0,3\}} X_2^{\{2,3\}} \qquad (1)$$

と表される2変数四値入力二値出力関数について考える。X_i（$i=1$，2）は入力変数で，$P=\{0, 1, 2, 3\}$の値をとり得る。X_i^{α}，但し，$\alpha \subseteq P$，はリテラルと呼ばれ，$X_i^{\alpha} \in \alpha$のとき，$X_i=1$，$X_i \notin \alpha$のとき0となる。リテラルの積項$X_1^{\alpha} X_2^{\beta}$は項と呼ばれ，積項のリテラルのうち最小の値をその積項の値とする。従って，f_1の第1項，$X_1^{\{0,1,3\}} X_2^{\{0,1\}}$は$X_1$が0または1または3で，かつ$X_2$が0または1のとき1となり，$X_1$と$X_2$のその他の組み合わせでは0となる。$f_1$はこのような各項の論理和の値で決まる。しかし，計算機上で関数を表す場合は次のビット位置表現が適している。

$$f_1 = 1101\text{-}1100 \vee 0110\text{-}0110 \vee 1001 \vee 0011 \qquad (2)$$

項のリテラルは”－”（これは説明の便宜上，挿入したもの）で区切られ，4値入力の場合は4ビット（p値入力のとき，pビット）の2進数で左よりX_1X_2（一般にn変数では……X_nまで）の順に対応する。各項の4ビットの2進数を左より第1成分，第2成分（一般に第n成分まで）と呼ぼう。f_1の最初の項1101-1100について考える。各成分の4ビットは左の桁より順次入力変数X_iのとり得る

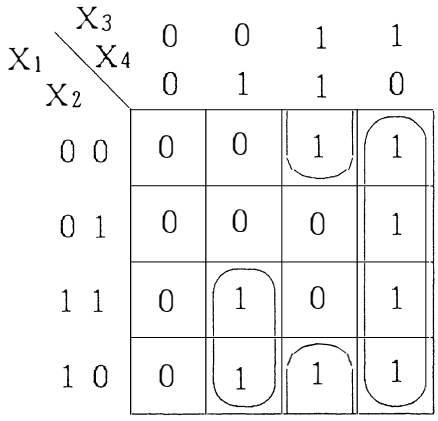


図１　二値入力４変数関数の例

```
REAL FUNCTION RANDOM(IY)
INTEGER*4 IY
IY=23*IY+11317
IY=IY-IY/32749*32749
R=IY
R=R/32749.0
RANDOM=R
RETURN
END
```

図２　一様乱数の生成

値０，１，２，３に対応し，例えば第１成分が1101なら，X_1 が０または１または３のときに，リテラル X_i^a が１になることを意味する。すなわち，$X_1^{\{0,1,3\}}$ を表す。同様に第２成分1100は X_2 が０または１のときのみ X_2^β は１となる。すなわち，$X_2^{\{0,1\}}$ を表している。

従って，1101-1100が $X_1^{\{0,1,3\}} X_2^{\{0,1\}}$ を表しているとみることができる。式(2)の第２項，第３項についても同様に考えれば，式(2)が式(1)を正しく表していることがわかる。

二値入力関数の場合は $P=\{0, 1\}$ で，$X_i^{\{0\}}=\bar{X}_i$（否定変数），$X_i^{\{1\}}=X_i$（肯定変数），$X_i^{\{0,1\}}=1$ は任意変数（陽に表れない変数）のことである。これらをビット位置表現で表すと，２値の場合は２ビットの２進数で，$\bar{X}_i$ は10で，X_i は01で，任意変数は11で表す。従って，二値入力関数は，

$$f_2=X_1\bar{X}_3X_4\vee X_3\bar{X}_4\vee\bar{X}_2X_3X_4 \tag{3}$$

$$f_2=01\text{-}11\text{-}10\text{-}01\vee 11\text{-}11\text{-}01\text{-}10\vee 11\text{-}10\text{-}01\text{-}01 \tag{4}$$

というビット位置表現となる。

式(4)を真理値表で表すと図１となる。四角います目はセルと呼ばれ，セルは入力変数の２進座標 X_1, X_2, X_3, X_4 で区別できる。関数は１と記したセルの座標の表す入力変数の組み合わせで True の値，０と記したセルの入力変数の組み合わせで False の値をとる。True の値をとるセルの数をセルの総数16個（n 変数では 2^n 個）で割った数値をこの関数の真理値表濃度 d という。真理値表濃度は関数を分類するための重要な指標の１つで論理関数簡単化データの比較のさい，よく使われる。４値入力変数の場合でみると，項の各成分は４ビットの２進数からなるので，１の数が１個か２個か３個か４個のいずれかであり，それぞれの生ずる確率を d_1, d_2, d_3, d_4（$d_1+d_2+d_3+d_4=1$）として，（d_1, d_2, d_3, d_4）で表し，成分濃度といおう。

２値入力変数の場合には項の各成分は１が１個（10か01）のとき（その確率が d_1）か２個（11）のとき（その確率が d_2）のときのいずれかで，成分濃度は（d_1, d_2）又は（$d_1 : d_2$）で表すことになる。

さて，ここで我々の関数生成プログラム CUBGEN では，変数の数 n と何値入力かを決める p の値と成分濃度（$d_1, d_2, \cdots, d_p$）と項の発生項数 COUNT を指定すると，そのような成分濃度を確率的にもった p 値入力 n 変数の項が COUNT 個からなる関数が発生するようになっている。普通，関数の各項の成分濃度は項ごとに異なるかもしれない。しかし，CUBGEN のように，各項が等しい成分濃度で生成されるとしても，十分に一般的な関数といえる。それは，もし，成分濃度を $d_1=1$，他の d_2 から d_p までを０とおけば，最小項のみからなる関数が発生できるし，d_1 から d_{p-1} までを０，$d_p=1$

とすれば，ユニバーサル項（マップ全体のセルからなる項で，これ1つで関数はトートロジーとなる）が発生でき，あとは成分濃度$d_1, d_2, \cdots, d_p$を適当に変えて，各種の関数が生成できるからである。

2.2 プログラム

プログラムはFORTRANでかかれている。

［乱数］ 乱数は図2のような関数副プログラムによって発生させている。すなわち，初期値ＩＹを与えると，0から1までの浮動小数点数の一様乱数が変数としてのRANDOMに求まり，次の乱数はいまのプログラムの実行で加工されたＩＹを使って副プログラムRANDOMを改めて呼んで別の乱数を生成することになる。

［成分ビットの発生］ いま，$p=4$値入力の場合で説明する。4値入力では各成分は4ビットからなり，1の個数の少ない順に（同一数のときには2進数として小さい順に）NEWSAMという手順でベクトルを配列 T(S, J) に生成する（図3）。1の数が1個のベクトルは2行目から5行目までの4つなので，このことを CHAN (1) = 2, CHANS (1) = 4 と記述し，1の数が2個のベクトルは6行目から10行目までの6つなので，CHAN (2) = 6, CHANS (2) = 6 と記述し，1の数が3個のベクトルは12行目から15行目までの4つなので，CHAN (12) = 4, CHANS (3) = 4，1の数が4個のベクトルは16行目ただ1つなので，CHAN (16) = 1, CHANS (4) = 1，と記述する。

S	T(S, J)			
1	0	0	0	0
2	1	0	0	0
3	0	1	0	0
4	0	0	1	0
5	0	0	0	1
6	1	1	0	0
7	1	0	1	0
8	0	1	1	0
9	1	0	0	1
10	0	1	0	1
11	0	0	1	1
12	1	1	1	0
13	1	1	0	1
14	1	0	1	1
15	0	1	1	1
16	1	1	1	1

左より右へ桁上がりするものとする

図3　2進ベクトル

成分ビット成分濃度d_1, d_2, d_3, d_4になるよう，発生するには次のようにしている。

（成分ビット発生の手順）

まず，ＩＹ＝1に設定して，

1）乱数を作る：R=RANDOM(IY)

2）Bを1成分4ビットの1の数とし，はじめB＝1とおく

3）Rとd_B（1の数がB個の成分濃度）を比較し，$R > d_B$なら，$R = R - d_B$とし，B=B+1として2）へ。$R \leqq d_B$なら，1の数がB個の成分ビットを生成するため次のステップ4）へ。

4）再びRANDOM(IY)を呼び，R=RANDOM(IY)，IR=100R，IRをCHANS(I)で割って，商と余りIAMARIを求め，配列T(S, J)のIROW={CHAN(I)+IAMARI}行目のベクトル：T(IROW, J)：J＝1から4までを成分ビットとする。

成分濃度（d_1, d_2, d_3, d_4）＝（0.5, 0.2, 0.2, 0.1）のとき，4値入力変数$n=8$変数の項をCOUNT＝8個生成する例で説明する。K＝1（最初の項）からK＝8（最後の項）まで以下のことをくり返すが，まずK＝1とする。$n=8$変数なので，成分ビットの手順を8回呼ぶ。1回目の手順1）でR＝0.34627，2）でB＝1のとき，3）で，$R > d_1 = 0.5$でないので，4）へいく。このとき，R＝0.30978が求まり，IR＝30，CHANS(1)＝4で割った余りはIAMARI＝2なので，IROW＝CHAN(1)＋2＝4で，配列T(S, J)の4行目のベクトル0010を成分ビットとする。このベクトルを最初の項のX_1成分として，項生成用の配列CUB(II, JJ)のCUB(1, 1)のはじめの左より4ビットに記憶する（但し，左より右へ桁上がりが生ずるものとする）。2回目の手順でR＝0.470518，2）でB＝1のとき，3）で$R > d_1 = 0.5$でないので，このときもB＝1で4）へいく。今度のR＝0.16748で，IR＝16，CHANS(1)＝4で割った余りIAMARI＝0，IROW＝CHAN(1)＝2であるから，配列T(S, J)の2行目の

	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8
II=1	0010-	1000-	0100-	1111-	0100-	1111-	1010-	1111
II=2	0010-	1110-	1000-	1011-	1000-	1000-	0010-	1110
II=3	1010-	0010-	1000-	0100-	0010-	1010-	1110-	1010
II=4	0111-	1100-	1101-	0010-	0100-	0010-	0001-	1011
II=5	0101-	1111-	0111-	0010-	0010-	1000-	1111-	1100
II=6	1111-	0100-	0100-	0011-	1000-	0010-	1100-	0001
II=7	0111-	1011-	0111-	0101-	0110-	0100-	1111-	1000
II=8	1011-	0010-	1101-	1110-	0001-	0001-	0111-	0001

図4　関数の発生例

ベクトル1000を成分ビットとする。このベクトルを最初の項の X_2 成分として，配列CUB(II, JJ)のCUB(1,1)の左より5から8桁目に入れる。以下，同様にして3回目から8回目までの成分ビットの手順で0100，1111，0100，1111，1010，1111が順次成分ビットとして求まり，CUB(1,1)の9桁目から順番につめると，CUB(1,1)の最初の n 変数の項が1つ求まる。続いて求まる8つの成分ビットはCUB(1,2)にたくわえ，その次の8つの成分ビットはCUB(1,3)……と順次CUB(1, K)がKは4から8になるまで，成分ビットの手順をくり返す。CUB(1,1)からCUB(1,8)に8個の項が生成される。関数 f はこれらの項からなる。図4はこのようにして得られた関数 f を印刷したものである。

$n=8$ 以上の場合は最初の項はCUB(1,1)，CUB(2,1)，CUB(3,1) と数ワードにまたがって記憶させ，その次の項はCUB(1,2)，CUB(2,2)，CUB(3,2) と数ワードにまたがって記憶させ……とすれば，いくらでも多変数に拡張できる。

3．実行結果

12変数の関数では，$2^{12}=4096$ のメモリがあれば真理値表で関数を与えることができるので，わざわざ本方法を用いなくてもよいのだが，変数の数 n を固定した場合の本方法の関数全体の特徴をみるために成分濃度（d_1：d_2）を0.8：0.2から0.2：0.8まで0.2刻みに d_1 は減少，d_2 は増加させて項の発生個数と発生関数の真理値表濃度との関係をプロットしたのが図5である。同図一番下の曲線は成分濃度（d_1：d_2）が0.8：0.2であるが，この程度の成分濃度ではCOUNTを相当大きくしなくては真理表濃度が1に近づかないので，（d_1：d_2）を0.8：0.2～0.95：0.05，COUNTを2000くらいまで大きくして，別図図6にプロットしてみた。

図7，図8は $n=16$ 変数関数の発生の場合で，項数は32個まで，曲線は上より，（d_1：d_2）が0.1：0.9から0.9：0.1まで，d_1 は0.1きざみに増加，d_2 は減少させたもので，0.7：0.3より下の方の曲線はほとんど重なっている。図9，図10は成分濃度（d_1：d_2）（それぞれ0.2：0.8および0.1：0.9）を一定にして，変数の数を12変数から32変数まで4変数ずつ増加して発生させた関数の特性である。項に含まれる最小項（セル）の数は各成分の1の数の積で表される。したがって，項の大小を項の含む最小項の数の大小であらわせば，d_2 が大きければ項は大きいといえる。図9，図10では項が非常に大きいので，16個程度で真理値表濃度がほとんど1になってしまう。同じく図11は $n=32, 40, 48$ と変えた関数の特性である。図11では成分濃度 d_1：$d_2=0.05$：0.95 一定となっている。図9，10，11の曲線はい

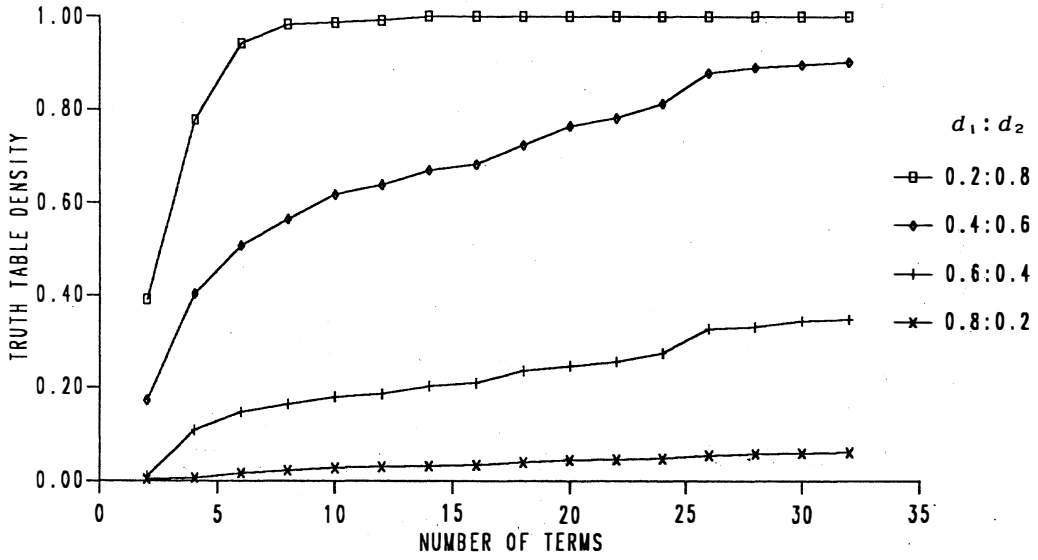


図５　12変数関数（各点２個の関数の平均）の発生項数と真理値表濃度との関係

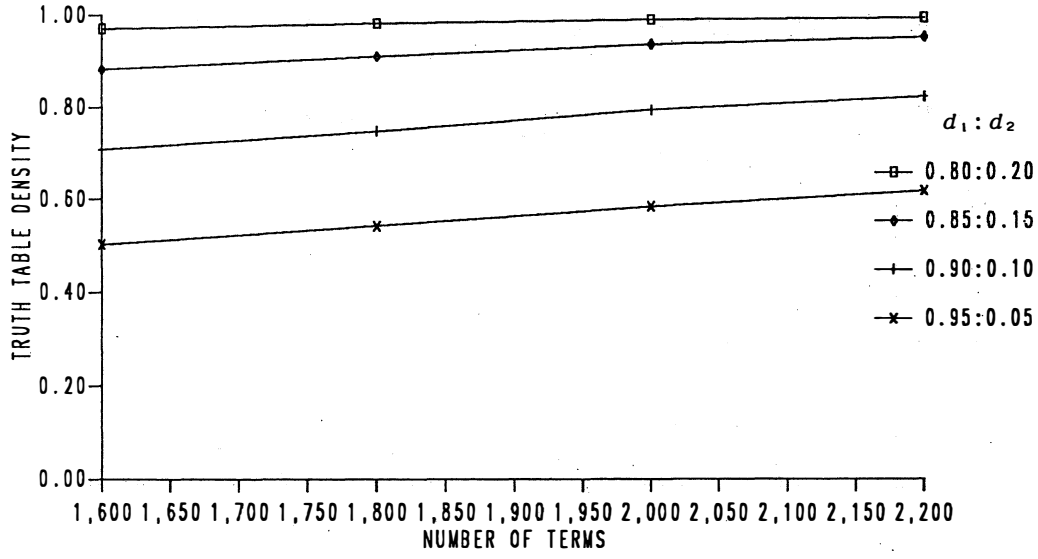
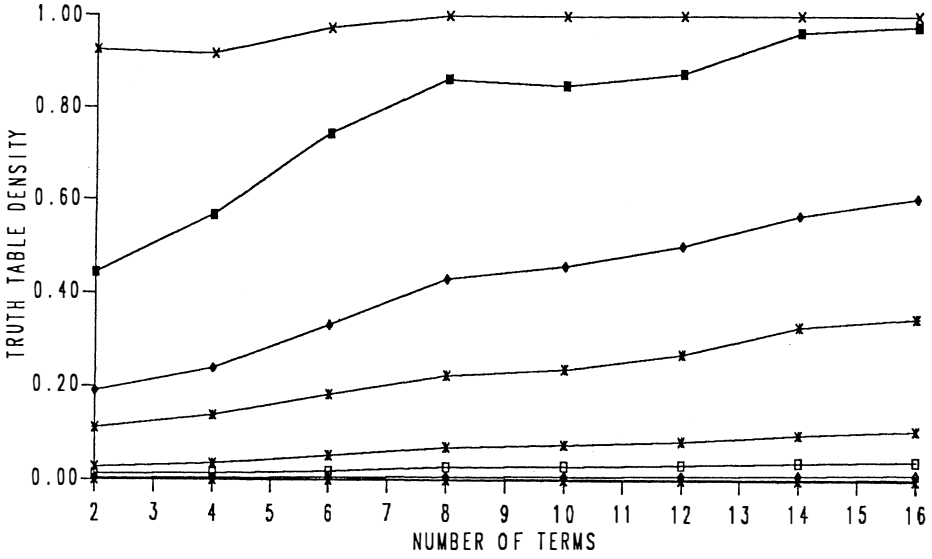
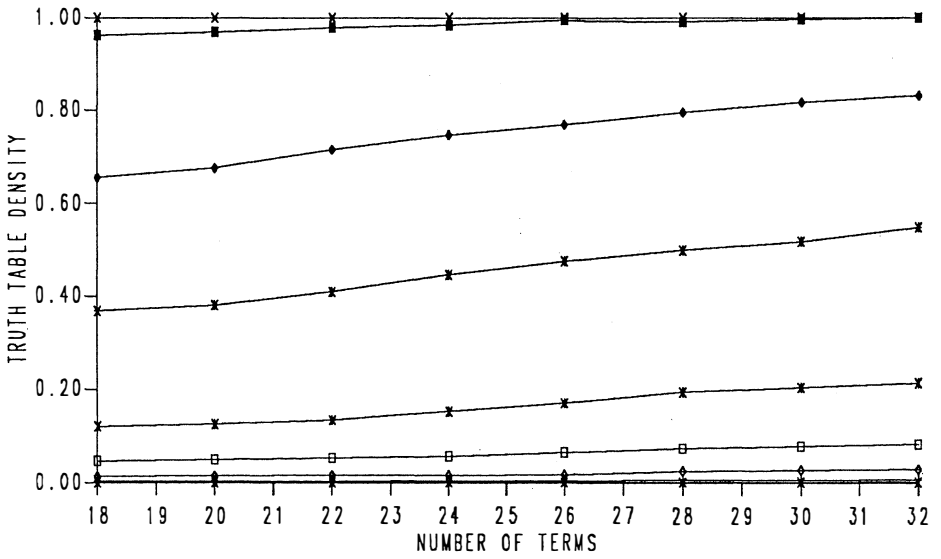


図６　12変数関数（各点１個の関数）の発生項数と真理値表濃度との関係



曲線は上より，（d_1：d_2）が0.1:0.9から0.9:0.1まで0.1きざみにd_1は増加，d_2は減少させたもの

図7　16変数関数（各点5個の関数の平均）の発生項数と真理値表濃度との関係



曲線は上より，（d_1：d_2）が0.1:0.9から0.9:0.1まで0.1きざみにd_1は増加，d_2は減少させたもの

図8　16変数関数（各点5個の関数の平均）の発生項数と真理値表濃度との関係

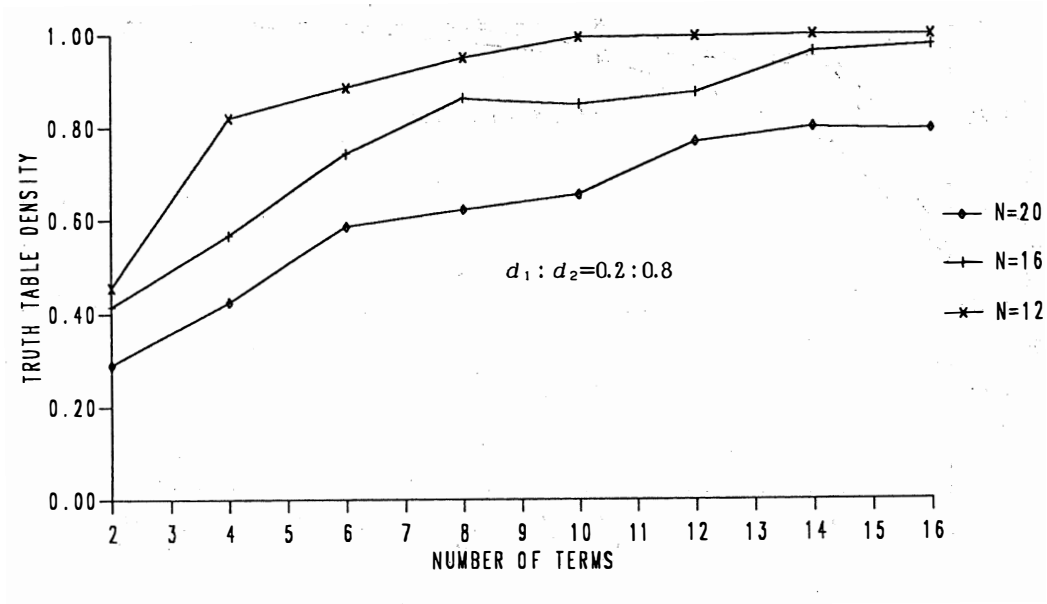


図９　12, 16, 20変数関数（各点５個の関数の平均）の発生項数と真理値表濃度との関係

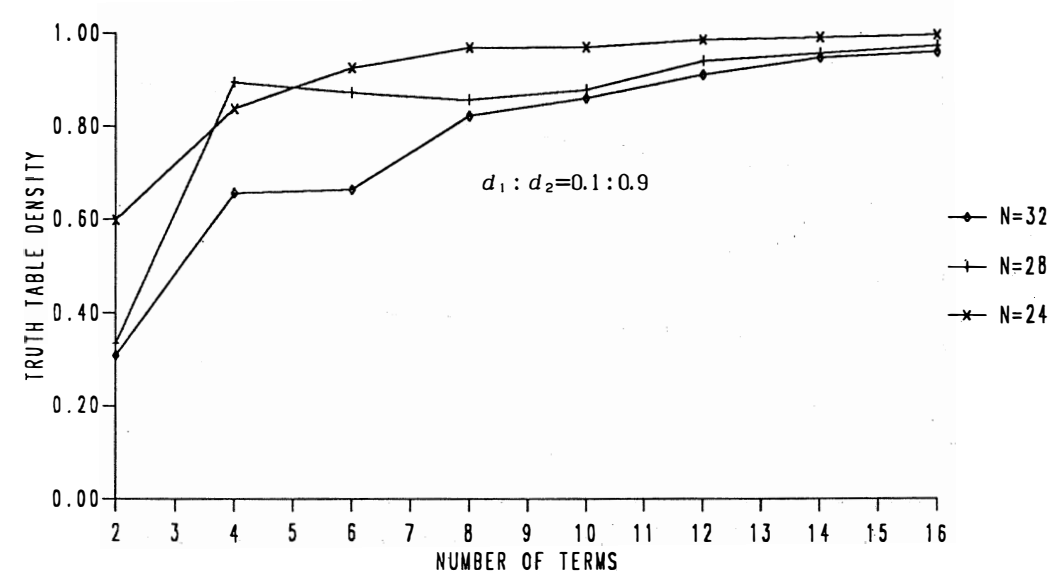


図10　24, 28, 32変数関数（各点５個の関数の平均）の発生項数と真理値表濃度との関係

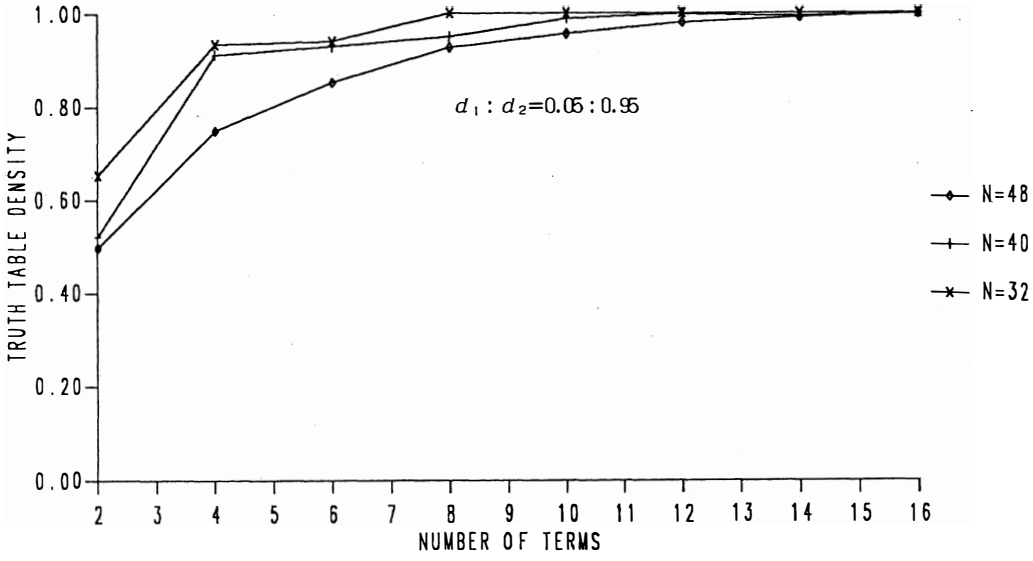


図11　32, 40, 48変数関数（各点5個の関数の平均）の発生項数と真理値表濃度との関係

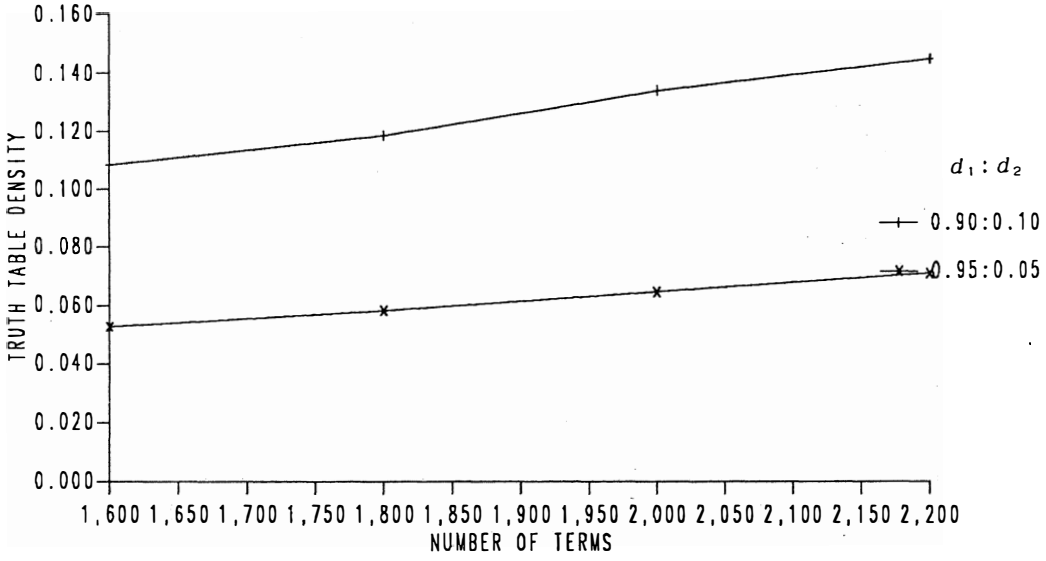


図12　16変数関数（各点1個の関数）の発生項数と真理値表濃度との関係

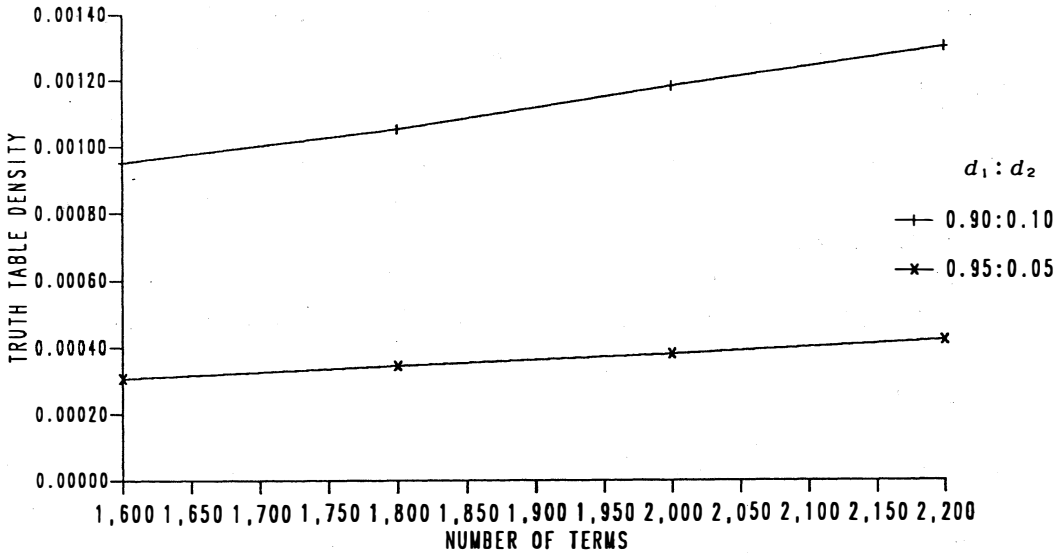


図13　24変数関数（各点１個の関数）の発生項数と真理値表濃度との関係

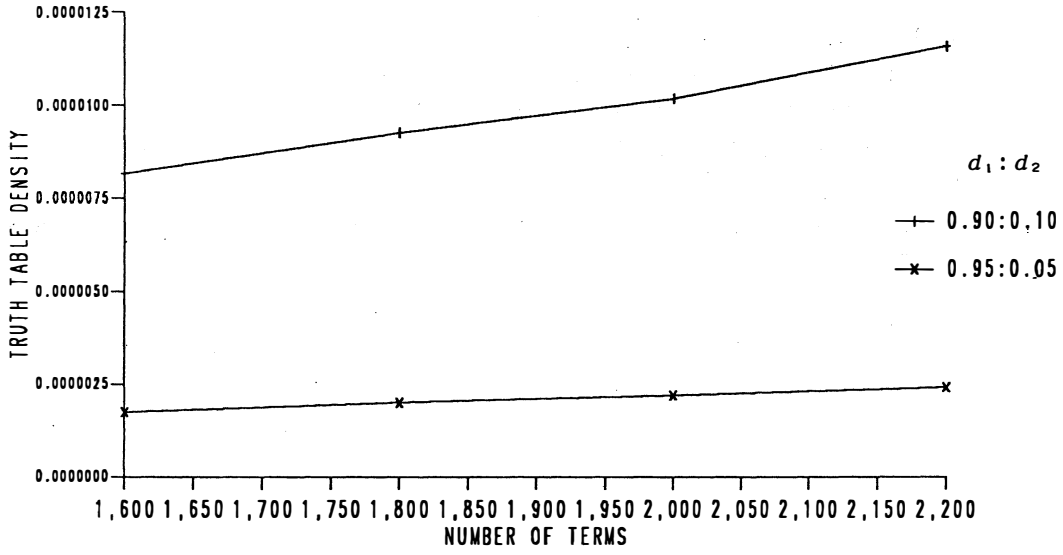


図14　32変数関数（各点１個の関数）の発生項数と真理値表濃度との関係

ずれも5個の関数の平均値をプロットしてある。図12，図13，図14は，今度はd_1が大きくd_2が小さい。すなわち，小さい項の場合で2000以上発生させても，真理値表濃度は1よりはるかに小さい。図12は$n=16$，図13は$n=24$，図14は$n=32$でそれぞれ1個の関数の値でプロットしている。

変数の値をきめて，すべての関数のグラフが求まるまでの平均時間はおおよそ，図5で5分，図6で31分，図7で2分，図8で33分，図9で16分，図10で44分，図11で16分，図12で52分，図13で53分そして図14で1時間8分であった。

12変数では成分濃度全般にわたって，また，項の個数も$2^{12}=4096$近くまでの関数が発生できるが，変数の数nが16以上になると，10数個の項で関数がトートロジィ（真理値表濃度1の関数のこと）になってしまう程大きい項で与えるとか，逆に最小項とあまり変わらないほど小さい項を数千個程度与えるといった関数しか発生できなくなる。これはメモリ容量および計算時間からの制約のためである（使用計算機；富山大学情報処理センター　IBM 4381-T92 2Proc. 8MIPS）。

なお，ここで各図の関数の真理値表濃度は，関数を一旦分離加法形に直して求めている[3)]。

4．ま と め

論理関数簡単化の研究で，他の簡単化の方法と性能比較するとき，手軽に多数の多変数関数が発生できるプログラムがあれば好都合であり，その一手法を示した。乱数を使うので，その使用パラメーターの初期値を設定すれば，誰でも，どこでも，同じ関数が順序よく発生できる。

変数の数，成分濃度，項の発生項数と発生関数の真理値表濃度の関係を示した。

本報告で二値入力変数一出力関数についてデータを示したが，多値入力二値多出力関数の発生も可能であるがこれらについては別途報告したい。なお，図2の乱数生成のプログラムは富士通などの計算機で使われているものである。

参考文献

1）宮腰　隆，松田秀雄：2値論理関数簡単化の一手法　—部分マップ法について—，電子情報通信学会論文誌D，**Vol. j71-D** No. 11，pp. 2259-2265（1988）.

2）宮腰　隆，松田秀雄，畠山豊正：論理式簡単化の一手法：MINI-LN，情報処理学会論文誌，**Vol. 34**，No. 12，pp. 2624-2635（1993）.

3）松田秀雄，宮腰　隆：論理式を分離加法形式で表現する一手法，情報処理学会論文誌，**Vol. 33**，No. 4，pp. 560-569（1992）.

Regarding the Program of Generating Functions with a Large Number of Variables (1)

Hideo matsuda, Toshiaki Gomi, Takashi Miyagoshi,
Toyomasa Hatakeyama and Yoshio Nakajima

There are many minimizaion procedures for Boolean functions. In the evaluation of those minimization algorithms, we are often under the necessity of generating a lot of Boolean functions. Here, we propose a method that functions are generated using random numbers so that preset component density (d_1, d_2), where $d_1+d_2=1$, $d_1 \geqq 0$ and $d_2 \geqq 0$, and number of terms (count). We can get many kinds of functions by trying all sorts of combinations of d_1 and d_2, and by varying the value of count. Data showing the relation between truth table density which is an important character of a Boolean function and number of generated terms are indicated graphically.

〔英文和訳〕

論理関数発生プログラムについて (1)

松田　秀雄，五味　利彰，宮腰　　隆，
畠山　豊正，中嶋　芳雄

ブール関数を簡単化するためには多くの方法がある。それらの方法を評価するとき，しばしば，大量の関数を生成する必要にせまられる。ここで，我々が提案する手法は乱数を用いて，あらかじめ設定した成分濃度（d_1, d_2），ここに，$d_1+d_2=1$，$d_1 \geqq 0$，$d_2 \geqq 0$，と項の個数（count）の関数が生成されるようにする方法である。d_1, d_2 の色々な組み合わせと，count の値を変えることにより，各種の関数が発生できる。ブール関数の性質の重要な指標である，真理値表濃度と関数の項の数との関係を示すデータをグラフとして与えている。