不確定入力に対する論理関数の評価法

向 殿 政 男
（明治大学・工学部）

あるn変数の２値論理関数fに対して、入力変数のうち、幾つかは０又は１に確定しているが、残りの幾つかの入力変数の値は不確定、即ち、０か１か不明であったとする。この時、この不確定な入力変数を含む場合の入力に対するfの値を計算する事を考える。 本論文では、この為の幾つかのアルゴリズムについて考察する。ただし、fの値は、次の様に定義されるものとする。即ち、不確定な入力変数が、いかなる確定した値をとったと仮定しても、fの値がすべて同じで１（又は０）ならば、その不確定入力に対するfの値は、１（又は０）とし、それ以外の場合は、不確定とする。

１． まえがき

ある入力に対して、論理関数の値を計算しようとする場合、入力変数のうち幾つかは０又は１に確定しているが、残りの幾つかの値は不確定、即ち、０か１か不明であるということがある。例えば、この様な典型的な例に次の様なものがある。ある順序回路に含まれる組み合せ回路の部分が、論理関数を用いた機能記述で与えられている時、ある入力の組み合わせについて論理シミュレーションをしなければならないとする。この時、この順序回路の初期状態が０か１か不明であれば、その出力を計算するには、順序回路の入力変数のうち、幾つかを不確定のままとしてその値を求めなければならない（図１）。

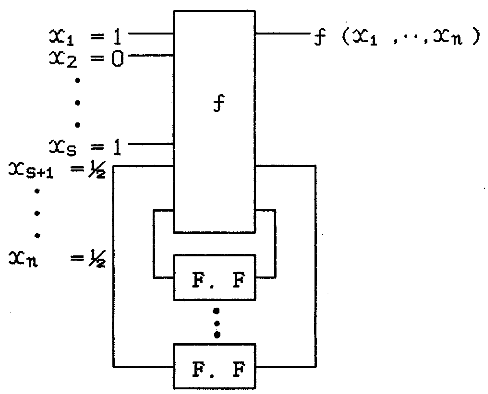


図１：不確定入力に対する論理関数の評価の例 （½は０か１か不明を表す。）

この時、次の二つの点が問題となる。一つは、不確定な入力変数を含む場合の出力の値をいかに定めるかであり、他は、その時の値を、いかに効率良く計算するかである。

本報告では、以上の様な不確定な入力を含む場合の論理関数の値の定め方を第２章で考察し、これらの問題を定式化する。続いて、第３章以降で、４つの異なった計算方法を提案し、それらの利害得失について論ずる。最後に、本手法の他の応用として、論理関数の簡単化などに良く用いられる、論理関数の各種の操作アルゴリズムへの応用にも言及する。なお、０か１か不明である事を表わす真理値として、（通常Xが用いられるが）本報告では便宜上、½を用いる。

２． 定式化

まず、不確定入力に対する論理関数の値を定義しよう。

真理値０と１からなる集合をB、０と½と１からなる集合をVと表わすことにする、即ち、

$B = \{0, 1\}$， $V = \{0, \frac{1}{2}, 1\}$。

n変数の論理関数fは、B^nからBへの写像

$f : B^n \rightarrow B$ －－－－－－－（1）

を表わしている。 $a = (a_1, \cdots, a_n)$を入力とすると、すべての入力変数の値a_iが確定していれば、aはB^nの一つの元として表現され、$f(a)$の値は０又は１となる。ところが、不確定な値を有する入力が一つでもある場合には、aは、V^nの一つの元として表わ

される事になる。この場合、f（a）の値は、0、1又は½のいずれかをとる事になり、fは3値論理関数とみなされなければならない。ある与えられた2値論理関数f（（1）式）に対して、上の様に拡張定義された3値論理関数を$\mathfrak{f}$と記すことにする。

即ち、

$$\mathfrak{f} : V^n \to V \qquad \text{-------}\ (2)。$$

任意に与えられた2値論理関数（1）式に対して、（2）式で表わされる3値論理関数$\mathfrak{f}$は一意的に定まらなければならない。これは、次の様に定義されるのが最も自然であろう。　即ち、V^n の元aに対して、aに現われる不確定な入力変数が確定したと仮定した時にとるB^n の各々のaに対して、f（a'）の値がすべて同じで1（0）ならば、aに対する$\mathfrak{f}$の値$\mathfrak{f}$（a）は1（0）とし、それ以外では½とする。これは、以下の様に定式化される。

今、aをV^n の一つの元とし、a^* をaにおけるすべての½を0又は1で置き換えて得られるB^n の元からなる集合を表わすとする。ここで、aにおける½の個数がS個とすると、a^* の要素の数は、2^S 個である。

〔例1〕　$a = (1, ½, ½)$ の時、

$$a^* = \{(1, 0, 0), (1, 0, 1), (1, 1, 0), (1, 1, 1)\}$$

2値論理関数fと、V^n の元aに対して、f（a^*）を次の様に定義する。

$$f(a^*) = \{f(a') \mid a' \in a^*\}$$

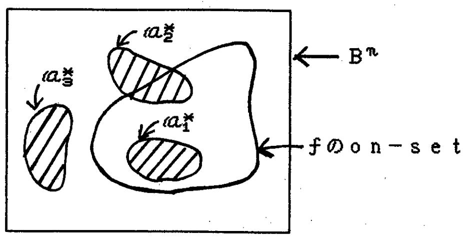
この時、fに対する、3値論理関数$\mathfrak{f}$は、つぎの様に定義される。

〔定義1〕

$\mathfrak{f}(a) = 0 \quad \text{if} \quad F(a^*) = \{0\}$,

$\mathfrak{f}(a) = 1 \quad \text{if} \quad F(a^*) = \{1\}$,

$\mathfrak{f}(a) = ½ \quad \text{if} \quad F(a^*) = \{0, 1\}$。



$\mathfrak{f}(a_1) = 1$, $\mathfrak{f}(a_2) = ½$, $\mathfrak{f}(a_3) = 0$

図2．fに対する$\mathfrak{f}$の値

〔例2〕　fを表1で表わされる2値論理関数、$a_1 = (1, 1, ½)$, $a_2 = (1, ½, ½)$, とすると $f(a_1^*) = \{1\}$, $f(a_2^*) = \{0, 1\}$。

よって、$\mathfrak{f}(a_1) = 1$, $\mathfrak{f}(a_2) = ½$である。

$a_1^* = (1, 1, ½)^*$

$a_2^* = (1, ½, ½)^*$

x_1 / x_2 \ x_3	0 0	0 1	1 1	1 0
0	0	0	1	0
1	1	0	1	1

表1：例題　f

次に、論理関数の表現方法について考えよう。論理関数は、通常、加法形式（論理和形、又は、加法標準形とも呼ばれる）といわれる論理式

$$f = \alpha_1 + \cdot \cdot + \alpha_m \quad \text{---}\ (3)$$

により表現される。ここでα_i　（$i = 1, \cdots, m$）は積項である。加法形式による表現方法は、例えば、論理関数が肯定で現われることを1、否定で現われることを0現われないことを－と置くと、論理関数のキューブ表現に相当している。

〔例3〕　$f = x_1 x_2 \overline{x}_3 + x_1 x_3 + \overline{x}_1 \overline{x}_2 x_3$

は加法形式であり、それに対応するキューブ表現は、表2のマトリックスとなる。

x_1	x_2	x_3
1	1	0
1	－	1
0	0	1

表2：

計算機を用いて論理関数の値を計算する場合、計算に要する時間は、論理関数の表現方法に大きく依存する。

本報告では、2値論理関数fは、すべて加法形式で表現されていると仮定する。

さて、ここで考察すべき問題は、

「n変数2値論理関数fが加法形式で与えられた時、

不確定な入力変数を含むある入力$\boldsymbol{a} \in V^n$に対して、

定義1の下での値$\underline{f}$（$\boldsymbol{a}$）を求める」

事である。

加法形式にも各種の表現方法があり、本報告では、加法形式に工夫を加える事に基づいて、各種の評価アルゴリズムを提案している。任意に与えられた論理関数に対して、対応する各種の加法形式を求める為の手順や、それに要する時間などについては、ここでは触れない事とし、それらは、既に与えられているものとして議論を進めて行く。これは、議論を統一的にする為であって、実際に、本報告の手法を応用しようとする場合には、対応する表現形式を求める時間等も考慮しなければならない。

3．定義に基づいた評価法

最も自然な計算方法は、定義1に従い、そのまま計算する事であろう。即ち、

（1）与えられた不確定入力$\boldsymbol{a} \in V^n$に対して、確定した入力の集合$\boldsymbol{a}^* \subseteq B^n$を求める。

（2）定義1に従い、すべての入力$\boldsymbol{a}' \in \boldsymbol{a}^*$に対して$f$（$\boldsymbol{a}'$）が1（0）ならば$\underline{f}$（$\boldsymbol{a}$）＝1（0）とし、$f$（$\boldsymbol{a}'$）＝0、$f$（$\boldsymbol{a}''$）＝1なる$\boldsymbol{a}'$, $\boldsymbol{a}''$が$\boldsymbol{a}^*$にある場合には$\underline{f}$（$\boldsymbol{a}$）＝½とする。

〔例4〕　表1の論理関数fが、加法形式　$f = x_1x_2 + \bar{x}_2x_3$で表現されていたとする。$\boldsymbol{a}$＝（1，½，1）なら、$\boldsymbol{a}^*$ ＝｛（1，1，1），（1，0，1）｝となり、f（111）＝f（101）＝1　より$\underline{f}$（$\boldsymbol{a}$）＝1　となる。

この場合には、fは任意の加法形式で表現されていてよい。しかし、f（$\boldsymbol{a}'$）の計算は、一般にfの積項の数に比例するので、簡単化された加法形式でfが与えられる事が望ましい。更に、f（$\boldsymbol{a}'$）＝0の確認には、fのすべての積項について調べなければならないが、f（$\boldsymbol{a}'$）＝1の確認の場合は、一般に、途中の積項で決定できる事がある。よって、文字数の少ない積項の順に並べておくと、計算時間が少なくて済む可能性が強い。

$\boldsymbol{a}$にある½の個数をS個とすると、$\underline{f}$（$\boldsymbol{a}$）＝1又は$\underline{f}$（$\boldsymbol{a}$）＝0の場合には、2^S個の確定した入力に対して　f（$\boldsymbol{a}'$）の値を計算しなければならない（$\underline{f}$（$\boldsymbol{a}$）＝½の場合には、途中で値を決定する事ができることがあるので）、本手法は½の個数が多い場合には、時間がかかるが、少ない場合には、有利である。

なお、この手法は、$\boldsymbol{a}$における確定した値をfに代入して得られた$\boldsymbol{a}$の½に対応する変数のみからなる2値論理関数$f_{/\boldsymbol{a}}$の真理値表を求めて解を得る事に対応している。よって、この場合には、$\underline{f}$（$\boldsymbol{a}$）＝1と$f_{/\boldsymbol{a}}$が恒真である事は等しい。

4．B－3値論理による方法1……素項展開法

fが3値論理関数であるので、fの加法形式を、そのまま、3値論理関数とみなして計算してしまうことが考えられる。　この場合、加法形式に現われる論理演算AND（・）、OR（＋）及び、NOT（￣）を、3値論理演算としてどの様に定義するかが問題となる。ここで、これをB－3値論理(1)の演算とみなして計算してみる。B－3値論理における論理演算は、表3で定義される。

x_2 \ x_1	0	½	1
0	0	0	0
½	0	½	½
1	0	½	1

AND：$x_1 \cdot x_2$

x_2 \ x_1	0	½	1
0	0	½	1
½	½	½	1
1	1	1	1

OR：$x_1 + x_2$

x	0	½	1
	1	½	0

NOT：$\bar{x}$

表3：B－3値論理

これは、通常の、0、1、Xの3値シミュレーションに用いられる真理値表と同じである(2)。加法形式で与えられた論理関数fを、B－3値論理関数として計算した値をf_Bと記すことにする。

〔例5〕 例3で与えられた加法形式を、
入力$\boldsymbol{a}=(1,\ 1,\ \frac{1}{2})$に対してB－3値論理関数として計算すると、

$$f_B(\boldsymbol{a}) = 1\cdot 1\cdot \overline{\tfrac{1}{2}} + 1\cdot \tfrac{1}{2} + \bar{1}\cdot \bar{1}\cdot \tfrac{1}{2}$$
$$= \tfrac{1}{2} + \tfrac{1}{2} + 0 = \tfrac{1}{2}$$

となる。

一般の加法形式fを、B－3値論理関数として評価すると、残念ながら、必ずしも、定義1による3値論理関数$\tilde{f}$とは一致しない。 即ち、一般に、

$$\tilde{f}(\boldsymbol{a}) \neq f_B(\boldsymbol{a})$$

である事がある。

〔例6〕 例4の加法形式において、fをB－3値論理関数として$\boldsymbol{a}=(1,\ \frac{1}{2},\ 1)$に対して計算すると

$$f_B(\boldsymbol{a}) = 1\cdot \tfrac{1}{2} + \tfrac{1}{2}\cdot 1 = \tfrac{1}{2}$$

となる。一方、例4より$\tilde{f}(\boldsymbol{a})=1$である。よって、この場合、$f_B(\boldsymbol{a}) \neq \tilde{f}(\boldsymbol{a})$。

ところが、fが、fの全ての素項(prime implicant)の和で表現された加法形式…素項展開…の場合には、両者は一致するという事が知られている。

〔定理1〕(1) $f=\alpha_1+\cdots\cdots+\alpha_m$ が、fのすべての素項の和ならば、V^nのすべての元$\boldsymbol{a}$に対して、

$$f_B(\boldsymbol{a}) = 0 \ \leftrightarrow\ \tilde{f}(\boldsymbol{a}^*) = \{0\},$$
$$f_B(\boldsymbol{a}) = 1 \ \leftrightarrow\ \tilde{f}(\boldsymbol{a}^*) = \{1\},$$
$$f_B(\boldsymbol{a}) = \tfrac{1}{2} \ \leftrightarrow\ \tilde{f}(\boldsymbol{a}^*) = \{0,\ 1\}$$

が成立する。

定義1及び定理1により、fが素項展開という特殊な形の加法形式で与えられていれば、 $\tilde{f}(\boldsymbol{a})$の値は、$f_B(\boldsymbol{a})$の値を計算する事により、直接求められる。

〔例7〕 表1の論理関数fに対する素項展開は、

$$f = x_1x_2 + x_1x_3 + \bar{x}_2x_3 \qquad \text{となる。}$$

この場合、今、$\boldsymbol{a}=(1,\ \frac{1}{2},\ 1)$とすると、

$$f_B(\boldsymbol{a}) = \tfrac{1}{2} + 1 + \tfrac{1}{2} = 1$$

となり、$f(\boldsymbol{a})$と一致する。

この手法の場合には、前回と異なりV^nのいかなる元$\boldsymbol{a}$に対しても、一回$f_B(\boldsymbol{a})$を計算するだけで$\tilde{f}(\boldsymbol{a})$の値を求める事ができる、という利点を有している。ただし、素項展開には、多くの積項が現われる為、変数が多い場合には、実用的ではなくなる。 前回と同様、$f_B(\boldsymbol{a})=1$の場合には、すべての積項を計算する必要がないことがあるので、fとしては、文字数の多い積項の順に並べておくと、計算時間を少なくできる可能性が強い。

5. B－3値論理による方法2……双対法

前節の方法では、積項数が多くなる為に、多変数では実用的でなくなってくる。そこで、これに改良を加えたのが本節の手法である。これは、次のB－3値論理の性質に基づいている。

〔定理2〕(1) $f=\alpha_1+\cdots\cdots+\alpha_m$ を任意の加法形式とする。V^nの任意の元$\boldsymbol{a}$に対して

$$f_B(\boldsymbol{a}) = 0 \ \leftrightarrow\ f(\boldsymbol{a}') = \{0\},$$
$$f_B(\boldsymbol{a}) = 1 \ \leftrightarrow\ f(\boldsymbol{a}') = \{1\}$$

が成立する。

上の定理により、fをB－3値論理関数f_Bとして計算し、もし$f_B(\boldsymbol{a})=0$又は1ならば、$\tilde{f}(\boldsymbol{a})=0$又は1であることが分かるが、 $f_B(\boldsymbol{a})=\frac{1}{2}$の時には、$\tilde{f}(\boldsymbol{a})=\frac{1}{2}$ か $\tilde{f}(\boldsymbol{a})=1$かの両方が有り得る事になる。よって、$f$の加法形式と共に$f$の否定$\bar{f}$の加法形式も用意しておけば、これを区別する事ができる。即ち、$f_B(\boldsymbol{a})=\frac{1}{2}$の時には、$\bar{f}$の加法形式をB－3値論理関数として計算して$\bar{f}_B(\boldsymbol{a})$を求める。この場合には$\bar{f}_B(\boldsymbol{a})=0$又は$\frac{1}{2}$のいずれかであるが、定理2より、$\bar{f}(\boldsymbol{a}^*)=\{0\}$ならば、即ち、$f(\boldsymbol{a}^*)=\{1\}$ならば、必ず、$\bar{f}_B(\boldsymbol{a})=0$となる。よって、次のアルゴリズムを得る。

（1）fをB－3値論理関数として評価し、$f_B(\boldsymbol{a})$を求める。

（2）$f_B(\boldsymbol{a})=1$（0）ならば$\mathfrak{f}$（$\boldsymbol{a}$）＝1（0）

（3）$f_B(\boldsymbol{a})=\frac{1}{2}$　ならば　$\bar{f}_B(\boldsymbol{a})$を求める。

（4）$\bar{f}_B(\boldsymbol{a})=0$　ならば　$\mathfrak{f}$（$\boldsymbol{a}$）＝1

それ以外は　$\mathfrak{f}$（$\boldsymbol{a}$）$=\frac{1}{2}$

〔例8〕　例2において

$$f=x_1x_2+\overline{x}_2x_3$$

$$\bar{f}=\overline{x}_1x_2+\overline{x}_2\overline{x}_3$$

として、fと$\bar{f}$の加法形式が与えられていたとする

$\boldsymbol{a}=(1,\ \frac{1}{2},\ 1)$とすると、$f_B(\boldsymbol{a})=\frac{1}{2}$　。

$\bar{f}_B(\boldsymbol{a})=0$　となり、　上記アルゴリズムより、

$\mathfrak{f}$（$\boldsymbol{a}$）＝1　であることが分かる。

この手法では、f及び$\bar{f}$は任意の加法形式で与えられていて良い。ただし、$f_B(\boldsymbol{a})$、$\bar{f}_B(\boldsymbol{a})$の計算は、積項数にほぼ比例するので、なるべく簡単化された加法形式を用いるのが良い。又、前と同様、文字数の少ない積項の順に並べておくと、速く計算できる可能性がある。この手法の利点は、いかなる入力$\boldsymbol{a}$に対しても高々２回論理式を計算すれば良い事である。欠点としては、肯定と共に、その否定を表わす加法形式も求めておかなければならないことである。

6．分離加法形式による方法

fを分離加法形式と呼ばれる特殊な加法形式で表現すると、$\mathfrak{f}$（$\boldsymbol{a}$）の計算は、かなり高速に実行できる。

〔定義2〕　加法形式

$$f=\alpha_1+\cdots\cdots+\alpha_m$$

において、各積項が互に分離的である時、即ち、

$\alpha_i\cdot\alpha_j=0$　がすべてのｉ，ｊ（$i\neq j$）について成立する時、分離加法形式であるといわれる。

fが分離加法形式で表現されているならば、fのｏｎ－ｓｅｔの要素数を求めるのは極めて容易である。これは、積項α_iのｏｎ－ｓｅｔの数（α_iに現われない変数の数をＳ個とすると　2^S個である）を$\|\alpha_i\|$と記す事にすると、fのｏｎ－ｓｅｔの数$\|f\|$は、

$$\|f\|=\sum_{i=1}^{m}\|\alpha_i\|$$

で求められるからである。この事実に基づき、次の様な$\mathfrak{f}$（$\boldsymbol{a}$）の値を求めるアルゴリズムが得られる。

ここで、不確定入力$\boldsymbol{a}$に対応する積項をα_aとする（定義3参照）と、$\alpha_i\cdot\alpha_a$のｏｎ－ｓｅｔは、α_iのｏｎ－ｓｅｔと$\boldsymbol{a}^*$との共通部分である。

（図３参照）

（1）$\boldsymbol{a}^*$の元の数を求める。

（2）$\boldsymbol{a}$に対応する積項をα_aとする。

$\alpha_a\cdot f$のｏｎ－ｓｅｔの総数$\|\alpha_a\cdot f\|$

$=\sum_{i=1}^{m}\|\alpha_i\cdot\alpha_a\|$　を求める。

（3）上式が0ならば$\mathfrak{f}$（$\boldsymbol{a}$）＝0，（1）の値と等しければ$\mathfrak{f}$（$\boldsymbol{a}$）＝1、それ以外は$\mathfrak{f}$（$\boldsymbol{a}$）$=\frac{1}{2}$。

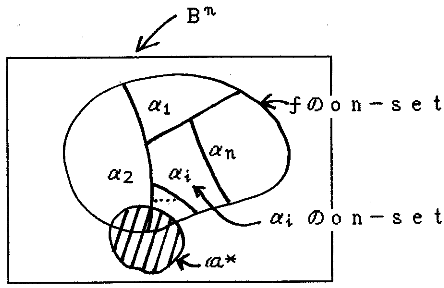


図３：分離加法形式のｏｎ－ｓｅｔ

〔例9〕　例3の表現形式は、分離加法形式である。

$\boldsymbol{a}=(1,\ \frac{1}{2},\ \frac{1}{2})$とする。この時　$\|\boldsymbol{a}^*\|=4$である。

$$\alpha_a=x_1$$

$$x_1x_2\overline{x}_3\cdot\alpha_a=x_1x_2\overline{x}_3$$

$$x_1x_3\cdot\alpha_a=x_1x_3$$

$$\overline{x}_1\overline{x}_2x_3\cdot\alpha_a=0$$

より　$\|x_2\overline{x}_3\|+\|x_3\|=1+2=3$

よって、$4\neq3\neq0$より、$\mathfrak{f}$（$\boldsymbol{a}$）$=\frac{1}{2}$　である。

この手法の特徴は、計算時間は、積項の数に正確に比例し、$\mathfrak{a}$における½の個数には、ほとんど依存しないことである。１回だけ計算すれば良いので、かなり高速に計算できる。ただし、欠点としては、分離加法形式は、一般に項数が多くなる事と、求めるのに手数を要する点である。

７．論理関数の操作アルゴリズムへの応用

ＰＬＡの簡単化は、実現すべき論理関数の加法形式による表現の簡単化に対応している。大規模な論理関数の実用的な簡単化アルゴリズムには、積項が拡大可能か否かの判定、必須項の判定、論理関数の等価性の判定(3)(4)や論理関数の恒真性の判定(5)などが重要な操作となる。これらの操作の基本は、全て、加法形式で表現された論理関数　$f=\alpha_1+\cdots\cdots+\alpha_m$　に対して、ある積項αがfに包含されるか否か、即ち、

$$f \geq \alpha$$

か否かの判定に帰着させられる。以上の問題は、次の様に本報告で考察している問題の特殊な場合に相当している。

〔定義３〕　積項　$\alpha=x_1^{a_1}\cdots\cdots x_n^{a_n}$　と　V^n の元 $\mathfrak{a}=(a_1,\cdots\cdots,a_n)$ とは、次の時、互に対応しているという。

$a_i=0 \leftrightarrow x_i^{a_i}=\overline{x_i}$

$a_i=1 \leftrightarrow x_i^{a_i}=x_i$

$a_i=½ \leftrightarrow x_i^{a_i}=1$（$\alpha$に変数$x_i$は存在していない）

〔例１０〕　$\mathfrak{a}=(1,½,0)$と$\alpha=x_1\cdot\overline{x_3}$とは互に対応している。

以上の定義から明らかな様に、$\mathfrak{a}$と$\alpha_{\mathfrak{a}}$とが互に対応している時

$$f \geq \alpha_{\mathfrak{a}}$$

と

$$\bar{f}(\mathfrak{a})=1$$

とは等価である（更に、$f\cdot\alpha_{\mathfrak{a}}=0$と$\bar{f}(\mathfrak{a})=0$とも等価である）。

以上により、論理関数の各種の操作アルゴリズムにも本報告の手法は使用できる事が分かる。

８．あとがき

不確定な入力変数を含む場合の、論理関数の値を求めるという、非常に素朴な問題を提案し、これに対する幾つかの手法を示した。本問題は、不確定状態を含む場合の論理シミュレーションのみならず、論理関数の各種の操作アルゴリズムにも関連していることを示した。

残された問題は、今後、これらの手法を、理論的に、又、実験的にも比較、検討してみることである。

謝辞

本報告の内容について、色々と御検討頂いた、本学井口幸洋氏に感謝致します。

参考文献

（１）向殿：Ｂ－３値論理関数について、信学会（Ｄ）Ｖｏｌ．５５－Ｄ，Ｎｏ．６（１９７２）

（２）M.A.Breuer: Note on Three-Valued Logic Simulation,IEEE Trans. Computers, April (1972)

（３）S.J.Hong, R.G.Cain and D.L.Ostapko:MINI, IBM J.Res.Develop., vol.18 (1974)

（４）R.K.Brayton et al.: A comparison of logic minimization strategies using ESPRESSO, proc. of 1982 International Symposium on Circuit and Systems, May (1982)

（５）T.Sasao: A fast complementation algorithm for sum-of-products expressions of multiple-valued input binary functions, proc. of 13th International Symposium on Multiple-Valued Logic, May (1983)