計算機アーキテクチャ　98－5
設　計　自　動　化　65－5
(1993. 1. 21)

三分岐判定図を用いた主項の生成法について

笹尾 勤　　甘田 哲久

九州工業大学 情報工学部 電子情報工学科
〒820 飯塚市大字川津 680-4

あらまし

与えられた論理関数を表現する方法として種々の三分岐判定図 (TDD) を提案する．また，TDD の 1 パス (根から終端節点 1 に至る経路) が表す積和形論理式について考察する．AND TDD (ATDD) の 1 パスは，与えられた関数の内項を表現する．Prime TDD (PTDD) の 1 パスは，与えられた関数の主項を表現する．Reduced Ordered TDD (ROTDD) の 1 パスは，与えられた論理式の積項を表現する．変数の順序が固定された場合，ATDD や PTDD は，与えられた論理関数に対して一意的に定まる．また，ROTDD は，与えられた積和形論理式に対して一意的に定まる．PTDD を用いて与えられた論理関数の全ての主項を陰的に生成する方法を提案する．従来の分割統治法と比較し，ROTDD を用いた方法は大幅に高速である．また，論理関数を PTDD を用いて表現するために必要な節点数の評価を行う．

和文キーワード　論理関数の表現法，主項生成，二分岐判定図，三分岐判定図，BDD，TDD，論理式簡単化

A Method to Generate Prime Implicants using Ternary Decision Diagrams

Tsutomu Sasao　　Takahisa Amada

Department of Computer Science and Electronics
Kyushu Institute of Technology
Iizuka 820, Japan

Abstract

This paper presents various ternary decision diagrams (TDDs) to represent logic functions. We consider sum-of-products expressions represented by 1-paths (paths from the root node to the terminal representing constant 1). 1-paths for an AND TDD (ATDD) represent the implicants of the function. 1-paths for a Prime TDD (PTDD) represent the prime implicants of the function. 1-paths for a Reduced Ordered TDD (ROTDD) represents the given sum-of-products expression. When the order of the input variables is fixed, the ATDD and the PTDD are unique for the given function, and the ROTDD is unique for the given expression. An algorithm to generate all the prime implicants by using PTDDs is much faster than the one using a divide and conquer method. We also analize the number of the nodes in an ATDD to represent the function.

英文 key words　Representation of logic functions, Prime implicant generation, Binary decision diagram, Ternary decision diagram, Logic minimization

1 まえがき

積和形論理式の最小形式を求める方法として，Quine-McCluskey の方法が知られている [15]．この方法は二つのステップからなる．第一のステップでは全ての主項を生成する．第二のステップでは，主項と最小項の包含関係を示した主項表の最小被覆を求める．各々のステップを実行法として，種々の手法が考案されているが，本稿では，主項生成法について論ずる．

主項生成法としては種々のものが知られているが，主なものとして

1) 繰り返しコンセンサス法 [2],
2) Nelson の方法 [16],
3) Quine の方法 [17],
4) Tison の方法 [22],
5) Slagle の方法 [21],
6) Morreale の方法 [14],
7) Reusch の方法 [18],

等が知られている．これらの方法では，基本的には，主項をキューブ表現として計算機の内部に保持する必要があるので，n 変数関数の主項を生成するために必要なメモリは $n \times$ (主項の個数) に比例する．従って，主項の個数が非常に多い場合には，メモリの制約で主項の生成は困難であった．しかし，最近，Coudert-Madre[8] は，二分岐判定図 (BDD) を用いた主項の陰的生成法を考案し，通常のワークステーションを用いて主項数が 100 万を越える関数を取り扱えることを実証した．

本稿では，論理式の表現法として種々の三分岐判定図 (TDD) を提案する．また，TDD を用いることによって，主項を陰的に生成をする方法を示す．また，主項生成のために必要な TDD の節点数の解析を行い，シミュレーション結果を示す．その他，TDD の種々の性質を示す．

2 諸定義および基本的性質

特に断わらない限り，外部入力変数の個数を n で表す．f は論理関数を表し，$\mathcal{F}$ は積和形 (論理式) を表す．本論文で考える積和形は，同じ積項を二個以上含まないとする．また，積項の順序を変えて得られる積和形も同じものと見なす．

定義 2.1 BDD (Binary Decision Diagram) は根を持った有向グラフであり，節点集合 V をもつ．非終端節点 v は属性として $\mathrm{index}(v) \in \{1,\cdots,n\}$, と二つの子供 $\mathrm{low}(v)$, $\mathrm{high}(v) \in V$ をもつ．終端節点 v は属性として値 $\mathrm{value}(v) \in \{0,1\}$ をもつ．また，論理関数 f と BDD の関係を次のように定義する．

v が終端節点のとき，
$\mathrm{value}(v) = 1$ ならば $f_v = 1$,
$\mathrm{value}(v) = 0$ ならば $f_v = 0$.
v が非終端節点で $\mathrm{index}(v) = i$ のとき，

$$f_v(x_1, x_2, \cdots, x_n) = \bar{x}_i \cdot f_{\mathrm{low}(v)}(x_1, x_2, \cdots, x_n) \vee x_i \cdot f_{\mathrm{high}(v)}(x_1, x_2, \cdots, x_n).$$

x_i を節点 v の判定変数という．

定義 2.2 BDD で展開する変数の順序を固定したもので同形な部分グラフの共有化を可能な限り行ったものを QROBDD (Quesi Reduced Ordered Binary Decision Diagram) という．

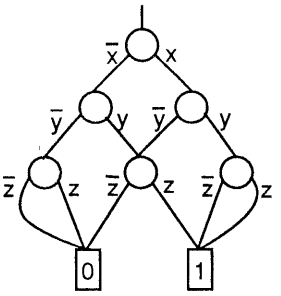

図 1: QROBDD の例

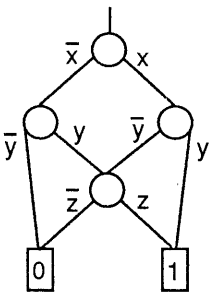

図 2: ROBDD の例

例題 2.1 論理関数 $f = xy \vee yz \vee zx$ の QROBDD を図 1に示す．

定義 2.3 BDD において根から終端節点 1 に至る経路を 1 パスという．

定理 2.1 任意の論理関数 f は QROBDD で表現できる．また，展開する変数の順序を固定すれば，QROBDD は一意的に定まる．任意の n 変数論理関数を QROBDD で表現するために必要十分な節点数は $O(2^n/n)$ である．QROBDD の 1 パスの集合は f の主加法標準形 (最小項展開) を表す．

(証明) QROBDD の表現の一意性は，ROBDD と同様に証明できる．節点数に関する証明は [12] を参照されたい．QROBDD の 1 パスは，すべての変数を通るので最小項に対応する．従って，1 パスの集合は f の主加法標準形を表す． □

定義 2.4 BDD で展開する変数の順序を固定したもので，冗長な節点の除去と同形な部分グラフの共有化を可能な限り行ったものを ROBDD (Reduced Ordered Binary Decision Diagram) という．

例題 2.2 論理関数 $f = xy \vee yz \vee zx$ の ROBDD を図 2に示す．

定理 2.2 任意の論理関数 f は ROBDD で表現できる．また，展開する変数の順序を固定すれば，ROBDD は一意的に定まる．任意の n 変数論理関数を表現するために必要十分な節点数は $O(2^n/n)$ である．ROBDD の 1 パスの集合は f の DSOP (分離的積和形: Disjoint Sum-Of-Products) を表す．

(証明) [7] と [1] を参照されたい． □

例題 2.3 図 1の 1 パスの集合は f の主加法標準形

$$\mathcal{F}_1 = \bar{x}yz \vee x\bar{y}z \vee xy\bar{z} \vee xyz$$

を表し，図 2の 1 パスの集合は f の分離的積和形

$$\mathcal{F}_2 = \bar{x}yz \vee x\bar{y}z \vee xy$$

を表す．

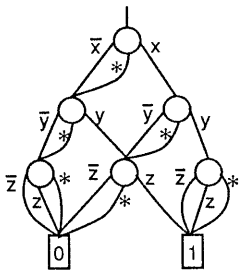

図 3: QROATDD の例

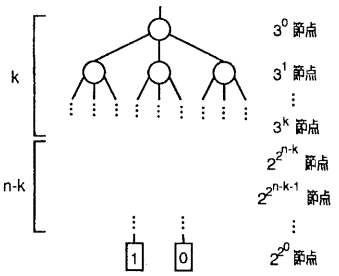

図 4: QRATDD の節点数の数え上げ

3 ATDD と内項の生成

定義 3.1 TDD (Ternary Decision Diagram) は根を持った, 有向グラフであり, 節点集合 V をもつ. 非終端節点 v は属性として $\mathrm{index}(v) \in \{1, \cdots, n\}$, と三つの子供 $\mathrm{low}(v)$, $\mathrm{high}(v)$, $\mathrm{and}(v) \in V$ をもつ. 終端節点 v は属性として値 $\mathrm{value}(v) \in \{0, 1\}$ をもつ.

定義 3.2 TDD のうち, 論理関数 f と TDD を次のように関係付けたものものを ATDD (AND Ternary Decision Diagram) という.
v が終端節点のとき,
$\mathrm{value}(v) = 1$ ならば $f_v = 1$,
$\mathrm{value}(v) = 0$ ならば $f_v = 0$.
v が非終端節点で $\mathrm{index}(v) = i$ のとき,

$$f_v(x_1, x_2, \cdots, x_n) = \bar{x}_i \cdot f_{\mathrm{low}(v)}(x_1, x_2, \cdots, x_n) \vee x_i \cdot f_{\mathrm{high}(v)}(x_1, x_2, \cdots, x_n).$$

$$f_{\mathrm{and}(v)}(x_1, x_2, \cdots, x_n) = f_{\mathrm{low}(v)}(x_1, x_2, \cdots, x_n) \cdot f_{\mathrm{high}(v)}(x_1, x_2, \cdots, x_n).$$

x_i を節点 v の判定変数という.
$f_{\mathrm{and}(v)} = f_{\mathrm{low}(v)} \cdot f_{\mathrm{high}(v)}$ が成立しない場合, このATDD に対応する論理関数は存在しない.

定義 3.3 ATDD で展開する変数の順序を固定し, 同形な部分グラフの共有化を可能な限り行ったものを QROATDD (Quesi Reduced Ordered AND Ternary Decision Diagram) という.

例題 3.1 論理関数 $f = xy \vee yz \vee zx$ の QROATDD を図3に示す.

補題 3.1 QROATDD の任意の節点 v を根とする部分グラフは, やはり QROATDD である.

(証明) QROATDD の帰納的な定義より明らかである. □

定理 3.1 任意の論理関数は QROATDD で表現できる. また, 展開する変数の順序を固定すれば, QROATDD は一意的に定まる.

(証明) 任意の論理関数は QROATDD で表現できることは明らかである. 一意性の証明は数学的帰納法を用いる. 0, 1 変数の TDD に対して定理が成立することは明らかである. k 変数以下の QROATDD が一意的に定まると仮定する. $k+1$ 変数関数 f の TDD が与えられたとき, TDD の定義より

$$f = \bar{x}_{k+1} f_0 \vee x_{k+1} f_1$$

と展開すると, 三分木の low, high, and の節点はそれぞれ $f_0, f_1, f_0 \cdot f_1$ を 表現する. $f_0, f_1, f_0 \cdot f_1$ はすべて, k 以下の変数の論理関数であるから, 各々の QROTDD は一意的に定まる. f の QROTDD は, $f_0, f_1, f_0 \cdot f_1$ の QROTDD の部分グラフの共有化を可能な限り行ったものでありやはり一意的に定まる. 従って f の TDD も一意的に定まる. □

補題 3.2 QROATDD において同じレベルにある (同じインデックスを持つ) 節点が表現する関数はすべて異なる.

(証明) 同じレベルにある二つの節点が同じ関数を表現するとき, 定理 3.1よりこの二つの節点を根とする二つの部分グラフは同形であり共有できる. □

補題 3.3 QROATDD の節点数は高々

$$\min\{\frac{3^{k+1}-1}{2} + 2^{2^{n-k}} + 2^{2^{n-k-1}} + 2^{2^{n-k-2}} + \cdots + 2^{2^1} + 2^{2^0}\}$$

である.

(証明) 与えられた n 変数関数を図 4のように上部ブロックに, k 変数の三分岐判定木を, 下部ブロックに $n-k$ 変数の三分岐判定図を用いて表現する.
(1) k 変数の完全三分岐判定木を考えると, 第 j レベルの節点数は 3^j 個である. 従って, 上部ブロックの節点総数は

$$\sum_{j=1}^{k} 3^j = \frac{3^{k+1}-1}{2}$$

となる.
(2) $n-k$ 変数の三分岐判定図を考える (図 4下部ブロック). 補題 3.2より, 同一レベルの節点が表す関数はすべて異なる (同じ関数を表す節点は存在しない). これより, 一番上のレベルの節点数は高々 $2^{2^{n-k}}$ である. 同様に, その次のレベルでは, 節点数は高々 $2^{2^{n-k-1}}$ となる. 以下同様にして, 最下位のレベルまで考えると, 下部のブロックの節点総数は高々

$$2^{2^{n-k}} + 2^{2^{n-k-1}} + 2^{2^{n-k-2}} + \cdots + 2^{2^1} + 2^{2^0}$$

となることがわかる.
(1), (2) より定理が成立する. □

定理 3.2 QROATDD の 1 パスの集合は f の全ての内項を表す.

(証明) 数学的帰納法を用いる. 0,1 変数の QROATDD に対して定理が成立することは明かである. k 変数以下の QROATDD に対して定理が成立すると仮定する. $k+1$ 変数関数 f が与えられたとき,

$$f = \bar{x}_{k+1} f_0 \vee x_{k+1} f_1$$

と展開できる. QROATDD の最上位 (根) の判定変数を x_{k+1} とすると, 根の子供である $\mathrm{low}(v), \mathrm{high}(v), \mathrm{and}(v)$ はそれぞれ $f_0, f_1, f_0 \cdot f_1$ に対応する. $\mathrm{low}(v), \mathrm{high}(v), \mathrm{and}(v)$ を根とする QROATDD の 1 パスは, 帰納法の仮定より各々の関数の全ての内項を表現する. 従って, f の QROATDD の 1 パスは,

$\bar{x}_{k+1} \cdot (f_0$ の全ての内項 $)$,
$x_{k+1} \cdot (f_1$ の全ての内項 $)$, および
$1 \cdot (f_0 \cdot f_1$ の全ての内項 $)$

を表現する. f の内項は, 上の三つのいずれかに含まれることは明かである. また, との ROATDD の 1 パスは f の内項以外の項は表現しない. 従って, 定理が成立する. □

例題 3.2 図 3の 1 パスの集合が表す論理式は

$$\begin{aligned} \mathcal{F}_3 &= x \cdot y \cdot (\bar{z} \vee z \vee 1) \vee x \cdot (\bar{y} \vee 1) \cdot z \vee (\bar{x} \vee 1) \cdot y \cdot z \\ &= xy\bar{z} \vee xyz \vee xy \vee x\bar{y}z \vee xz \vee \bar{x}yz \vee yz \end{aligned}$$

であり, 全ての内項を含む.

定義 3.4 ATDD で展開する変数の順序を固定したもので, 冗長な節点の除去と同形な部分グラフの共有を可能な限り行ったものを ROATDD (Reduced Ordered AND Ternary Decision Diagram) という.

例題 3.3 論理関数 $f = xy \vee yz \vee zx$ の ROATDD を図 5に示す.

定理 3.3 任意の論理関数は ROATDD で表現できる. また, 展開する変数の順序を固定すれば, ROATDD は一意的に定まる. ROATDD の 1 パスの集合は f の内項を表す.

(証明) ROATDD は QROATDD の冗長な節点を取り除いたものである. 従って定理が成立する. □

例題 3.4 図 4の 1 パスの集合が表す論理式は

$$\begin{aligned} \mathcal{F}_4 &= x \cdot y \vee x \cdot 1 \cdot z \vee (\bar{x} \vee 1) \cdot y \cdot z \\ &= xy \vee xz \vee \bar{x}yz \vee yz \end{aligned}$$

であり, これらはすべて f の内項である.

4 PTDD と主項生成

定義 4.1 関数 f の主項の集合を $PI(f)$ で表す.

定義 4.2 PTDD (Prime Ternary Decision Diagram) は, ROATDD に変更を加え PDD の 1 パスが f の主項のみを表現するようにした判定図であり, 次のように帰納的に定義する. ROATDD の節点 v が関数 f, $\mathrm{low}(v)$ が関数 f_0, $\mathrm{high}(v)$ が関数 f_1 を表現するとき, PTDD では,
v が $PI(f)$ を,
$\mathrm{and}(v)$ が $PI(f_0 \cdot f_1)$ を,
$\mathrm{low}(v)$ が $PI(f_0) - PI(f_0 \cdot f_1)$ を,
$\mathrm{high}(v)$ が $PI(f_1) - PI(f_0 \cdot f_1)$ を表現する.

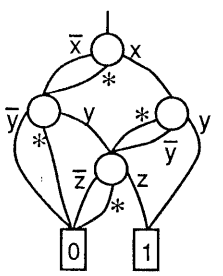

図 5: ROATDD の例

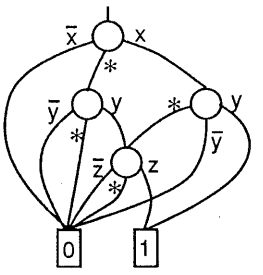

図 6: PTDD の例

例題 4.1 $f = xy \vee yz \vee zx$ の PTDD を図 6に示す.

定理 4.1 関数 f の全ての主項の論理和を表す積和形は一意的に定まる.

(証明) [6] を参照のこと. □

定理 4.2 任意の論理関数は PTDD で表現できる. また, 展開する変数の順序を固定すれば, PTDD は一意的に定まる.

(証明) 数学的帰納法を用いる. 0,1 変数の主項の集合が与えられたとき, PTDD は一意的に定まる. k 変数以下の主項の集合が与えられたとき, PTDD が一意的に定まると仮定する. $k+1$ 変数関数が与えられたとき,

$$f = \bar{x}_{k+1} f_0 \vee x_{k+1} f_1$$

と展開する. 根 v の子供 $\mathrm{low}(v), \mathrm{high}(v), \mathrm{and}(v)$ に対して
$\mathrm{low}(v) : PI(f_0) - PI(f_0 \cdot f_1)$,
$\mathrm{high}(v) : PI(f_1) - PI(f_0 \cdot f_1)$,
$\mathrm{and}(v) : PI(f_0 \cdot f_1)$
を対応させる. $\mathrm{low}(v), \mathrm{high}(v), \mathrm{and}(v)$ はそれぞれ, k 以下の変数の論理関数の主項の集合を表現する. 帰納法の仮定より, 各々の PTDD は一意的に定まる. f の PTDD は, $\mathrm{low}(v), \mathrm{high}(v), \mathrm{and}(v)$ の部分グラフの共有化を可能な限り 行ったグラフであり, やはり一意的に定まる. □

定理 4.3 論理関数 $f = \bar{x} f_0 \vee x f_1$ の主項は, 次式で与えられる.

$$\begin{aligned} PI(f) &= \bar{x} \cdot \{PI(f_0) - PI(f_0 \cdot f_1)\} \cup \\ &\quad x \cdot \{PI(f_1) - PI(f_0 \cdot f_1)\} \cup PI(f_0 \cdot f_1) \quad (1) \end{aligned}$$

(証明) A) f の主項はリテラルとして ①x を含む項, ②$\bar{x}$ を含む項, ③x も $\bar{x}$ も含まない項, のいずれかである. これらの主項は, それぞれ,
1) f_0 の主項に $\bar{x}$ をかけたもの,
2) f_1 の主項に x をかけたもの,
3) $g = f_0 \cdot f_1$ の主項
である. g の主項を p とすると, $x \cdot p$ や $\bar{x} \cdot p$ は f の主項ではない. 従って, (1) 式左辺は全ての主項を含む.
B) 次に, $\bar{x} \cdot \{PI(f_0) - PI(f_0 \cdot f_1)\}$ の任意の要素を $\bar{x} \cdot q$ とする. $\bar{x} \cdot q$ が主項でないと 仮定する. このとき, $\bar{x} \cdot q$ を包含する主項が存在する. その主項が $\bar{x}$ をリテラルとして含まない場合, それは g の主項でなければならない. しかし, これは, q の条件に反する. また, その主項がリテラル $\bar{x}$ を含むとき, q は f_0 の主項ではないことになる. しかし, これも矛

盾である．従って，$\bar{x}\cdot\{PI(f_0)-PI(f_0\cdot f_1)\}$ の要素はすべて f の主項である．同様にして，$x\cdot\{PI(f_1)-PI(f_0\cdot f_1)\}$ の要素もすべて f の主項であることを証明できる．次に $PI(f_0\cdot f_1)$ の任意の要素を r とする．r が f の主項でないと仮定すると，r を包含する別の f 主項が存在することになる．しかし，これは，r が $PI(f_0\cdot f_1)$ の要素であるという仮定に反する．従って，(1) 左辺の要素はすべて f の主項である．A, B より (1) が成立する． □

定理 4.4 PTDD の 1 パスの集合は f の全ての主項を表し，f の主項以外の項は表現しない．

(証明) 定義 4.1 と定理 4.3 より明らかである． □

推論 4.1 PTDD の節点数は $O(3^n/n)$ である．

(説明) PTDD は，QROATDD を縮約したものであり，QROATDD の節点数を越えないと考えることができる．補題 3.3 において，$k=n-\log_3 n$ とおくと，QROATDD の節点数は，$O(3^n/n)$ である．以上のことより，推論が成立する． □

5 ROTDD と積和形

定義 5.1 積和形に対して，以下の展開を再帰的に繰り返すことによって得られる三分岐判定木に対して，同形な部分グラフの共有を可能な限り行ったものを ROTDD (Reduced Ordered Ternay Decision Diagram) という．

$$\mathcal{F}=\bar{x}\cdot\mathcal{F}(|x=0)\vee x\cdot\mathcal{F}(|x=1)\vee 1\cdot\mathcal{F}(|x=*) \quad (2)$$

ここで，
$\mathcal{F}(|x=0)$: $\mathcal{F}$ の積項のうちで，$\bar{x}$ をリテラルとして含む積項からなる積和形で $\bar{x}$ を除いたもの．
$\mathcal{F}(|x=1)$: $\mathcal{F}$ の積項のうちで，x をリテラルとして含む積項からなる積和形で x を除いたもの．
$\mathcal{F}(|x=*)$: $\mathcal{F}$ の積項のうちで，x や $\bar{x}$ をリテラルとして含まない積項の積和形．
ただし，積和形が定数 1 になっても展開を終了せずに，全ての変数に関して展開し続けるものとする．

例題 5.1 積和形 $\mathcal{F}=xy\vee yz\vee zx$ を再帰的に展開すると

$$\begin{aligned}\mathcal{F} &= \bar{x}[0]\vee x[y\vee z]\vee 1[yz]\\ &= \bar{x}[0]\vee x[\bar{y}(0)\vee y(\bar{z}0\vee z0\vee 1)\vee 1(z)]\\ &\quad \vee 1[\bar{y}(0)\vee y(z)\vee 1(0)]\end{aligned} \quad (3)$$

となる．また図 7 に対応する ROTDD を示す．

定理 5.1 任意の積和形は ROTDD で表現できる．また，展開する変数の順序を固定すれば，ROTDD は一意的に定まる．

(証明) 積和形が ROTDD で表現できるのは，明らかである．一意性: 数学的帰納法を用いる．$n=0,1$ の時，ROTDD は一意的に決まる．k 変数以下の ROTDD が一意的に定まると仮定する．$k+1$ 変数積和形が与えられたとき，x で (3) の展開を考えると一意的に定まる．このとき，$\mathcal{F}(|x=0),\mathcal{F}(|x=1),\mathcal{F}(|x=*)$ は，k 変数以下の積和形

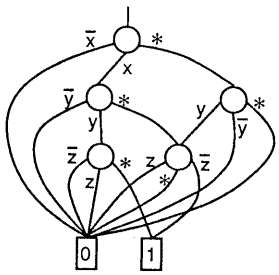

図 7: ROTDD の例

である．帰納法の仮定により，これらの式を表現する ROTDD は一意的に定まる．$\mathcal{F}$ を 表現する ROTDD は，これらの ROTDD の部分グラフの共有化を行ったものであり，やはり一意的に定まる． □

定理 5.2 ROTDD の 1 パスの集合は f の積和形を表す．

定理 5.3 与えられた n 変数論理関数の積和形 $\mathcal{F}$ の積項数を p とするとき，$\mathcal{F}$ を表現する ROTDD の節点数は高々 $n\cdot p+2$ である．

(証明) 数学的帰納法を用いる．0, 1 変数の積和形に対して定理が成立する．ここで，定数 0 と定数 1 を表現する終端節点を 2 個用いている．k 変数以下の積和形に対して定理が成立すると仮定する．$k+1$ 変数関数の積和形 $\mathcal{F}$ が与えられたとき，

$$\mathcal{F}=\bar{x}_{k+1}\mathcal{F}_0\vee x_{k+1}\mathcal{F}_1\vee 1\cdot\mathcal{F}_2$$

と展開できる．ここで，$\mathcal{F}_0,\mathcal{F}_1,\mathcal{F}_2$ は k 変数以下の積和形である．それぞれの積和形の積項数を p_0,p_1,p_2 とする．帰納法の仮定により，$\mathcal{F}_0,\mathcal{F}_1,\mathcal{F}_2$ はそれぞれ ROTDD で表現でき，各々の非終端節点数はそれぞれ，kp_0,kp_1,kp_2 以下である．$\mathcal{F}$ の ROTDD は，これらの三つの ROTDD を部分グラフとしてもつ ROTDD で表現でき，節点数は高々，

$$\begin{aligned}&1+kp_0+kp_1+kp_2+2\\ &= k(p_0+p_1+p_2)+3\\ &= kp+3\le(k+1)p+2\end{aligned}$$

となる．従って，$k+1$ 変数の積和形に対しても定理が成立する． □

n 変数積和形 $\mathcal{F}$ を表現する積和形の積項数を p とする．任意の n 変数関数は，積項数が高々 2^{n-1} 個の積和形で表現できるので，$p\le 2^{n-1}$ と仮定できる．従って，ROTDD の節点数の上界は $n\cdot 2^{n-1}$ となる．一方，ROBDD の節点数の上界は $O(2^n/n)$ なので，一般には，BDD の方が節点数を少なくできる．しかし，与えられた積和形の積項数 p が少ない場合には，ROTDD の方が少なくなることもある．例えば，$F=X_1\cdot X_2\vee X_3\cdot X_4\vee\cdots\vee X_{2P-1}\cdot X_{2P}$ などは，展開順序を $X_1,X_3,\cdots,X_{2P-1},X_2,X_4,\cdots,X_{2P}$ とすると，ROBDD の節点数は 2^p となるが ROTDD では $np+2$ でよい．

表 1: ベンチマーク問題に対する実験結果

Name	In/Out	Terms	SOP	Primes	BDD/ATDD/WORK	Time	Cou.	ROTDD/WORK	Time	s_t/s_c
add6	12/7	1092	355	8568	598/2298/10136	1.2	0.2	695/4616	0.2	1.89
adr4	8/5	255	75	397	123/279/803	0.5	0.2	178/752	0	2.66
alu1	12/8	19	19	780	1003/1553/5134	0.9		129/326	0	6.56
bc0	26/11	419	177	6596	4473/160531/453394	6.7	9.2	1646/4854	0.2	5.92
chkn	29/7	153	141	671	5279/8296/25974	2.0	1.0	869/5104	0.2	3.75
co14	14/1	14	14	14	29/29/56	0.5	0.2	29/211	0	1.87
dc1	4/7	15	9	22	26/49/82	0.5		26/67	0	3.85
dc2	8/7	58	39	173	275/463/1175	0.6		122/432	0	3.50
dist	8/5	255	121	401	212/697/2264	0.6	0.5	303/1378	0	2.80
f51m	8/8	255	76	561	511/1284/3617	0.6	0.3	165/851	0	2.43
gary	15/11	214	107	706	387/2272/7532	1.1	1.2	704/1968	0	5.58
in1	16/17	110	104	928	565/3589/9471	1.1	2.9	635/2421	0	4.88
in2	19/10	137	135	666	677/2358/7471	3.4	1.1	785/3626	0.1	4.40
in3	35/29	75	75	1114	26182/458555/942015	1.6	4.1	1319/3311	0.1	8.95
in4	32/20	234	212	3076	9294/209839/568851	5.3	4.2	1364/7125	0.4	3.53
in5	24/14	62	62	1067	3371/12130/34075	1.7	1.8	627/2156	0	6.74
in6	33/23	54	54	6174	148373/573794/1350885	19.5	2.7	1002/2313	0	9.99
in7	26/10	84	55	2112	2225/12050/37375	1.9	1.1	408/1806	0	4.72
jbp	36/57	166	126				204	1713/4948	0.2	6.34
misg	56/23	75	69				0.4	941/5334	0.2	5.09
mish	94/43	91	82				2.7	2241/8558	0.4	6.60
mlp4	8/8	225	126	606	337/841/2224	0.7	0.7	321/1442	0	2.85
opa	17/69	342	79	477	574/2402/5925	0.7	6.7	904/1927	0	7.63
radd	8/5	120	75	397	124/288/876	0.5	0.2	144/709	0	2.15
rckl	32/7	96	32	302	495/2999/12546	1.1	0.9	530/1585	0	9.46
rd53	5/3	31	31	51	21/30/61	0.5	0.2	31/204	0	1.27
rd73	7/3	147	127	211	36/61/140	0.5	0.2	61/934	0	0.56
risc	8/31	74	27	43	73/130/315	0.5		139/342	0	5.15
root	8/5	255	57	152	99/304/939	0.5	0.3	172/655	0	3.38
sqn	7/3	84	84	75	77/156/396	0.5		101/372	0	1.40
sqr6	6/12	63	51	205	127/381/794	0.5	0.3	139/507	0	3.18
ti	47/72	241	215				56	4087/13591	1.2	7.39
tial	14/8	640	579	7145	1717/22918/76391	11.7		1701/8906	0.4	2.65
vg2	25/8	110	110	1188	1762/5852/20091	1.3	0.4	215/1835	0.1	1.30
x1dn	27/6	112	110	1220	769/3370/11202	1.6	0.7	286/2413	0.1	1.65
x2dn	82/56	112	104				9.7	2528/9822	0.5	6.48
x6dn	39/5	121	81	916	362/2395/7068	1.0	0.9	1170/3182	0.1	7.22
x7dn	66/15	622	538				29	1720/17879	2.7	1.07
x9dn	27/7	120	120	1272	794/2877/10333	1.7	0.7	273/2901	0.1	1.45
z4	7/4	127	59	167	63/119/326	0.5	0.2	98/508	0	1.94
5xp1	7/10	75	67	390	255/822/1983	0.6	0.3	133/601	0	2.22
9sym	9/1	87	87	1680	62/61/122	0.5	0.2	198/874	0	2.45

Terms : 与えられた PLA の積項数

SOP : AND-OR で簡単化した時の積項数

Time : 単位は秒. 使用計算機 HP9000/720, 640MB メモリ

Cou. : Coudert-Madre の方法による実行時間. 単位は秒, 使用計算機 SUN SPARC2, 48MB メモリ

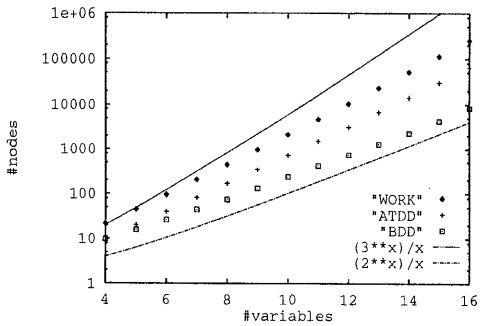

図 8: ROBDD,ATDD,WORK の節点数

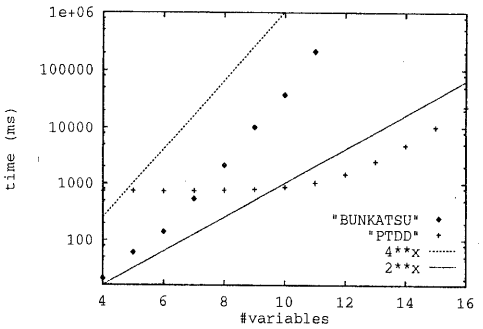

図 9: 分割統治法と PTDD による主項の生成時間の比較

6 実験結果

6.1 PTDD による主項生成

A. 主項の個数, ATDD と ROBDD の節点数, 最大使用節点数 最小項の個数が 2^{n-1} 個の真理値表を擬似乱数を用いて生成した. 主項を生成した ($n \leq 14$ のとき 50 個の関数を生成し, その平均値を求めた. $n = 15, 16$ は 1 個の関数を生成した, 以下同様). 図 8は, ROBDD の節点数, ATDD の節点数, 及び PTDD の作成に使用した節点数の最大値 (以下 WORK と呼ぶ) を示したものである. ROBDD の節点数は, ほぼ $2^n/n$ に比例している. また, ATDD の節点数は $3^n/n$ よりも少ない. 次に, ESPRESSO ベンチマーク関数 [4] の内, 完全記述関数に対して主項を生成した. 表 1は, ATDD(第 6 列真中), WORK の節点数 (第 6 列右) と計算時間 (第 7 列, HP720, 単位は ms) を示す. 何れの場合も比較的短時間で求まっている. Primes の列で空欄はメモリオーバフローのため計算を中止した事を示す. その他のベンチマーク関数では, 関数 mainpla(主項数 87692), misj(主項数 139103), prom1(主項数 9326), t1(主項数 15135) 及び ts10(主項数 524280) の主項の陰的生成法に成功した. これらの関数は, 従来の方法では, 主項の個数が非常に多く生成が困難であるといわれていた [19].

B. 分割統治法を用いたアルゴリズムとの比較 Morreale[14] の方法や Reusch[18] の方法は, 分割統治法 [4] とも呼ばれ, Coudert-Madre の方法が発表されるまでは, 最も能率が良いとされていた. ここでは, 多値入力論理式最小化用に開発した分割統治法を用いた主項生成プログラムと比較する. このプログラムは, 以前 FORTRAN を用いて開発したものであり, データ構造や計算機も異なるので, 計算時間の増加率で比較した. 図 9は, 上記乱数関数の主項生成のための計算時間をプロットしたものである. 分割統治法の場合, 計算時間の増加は 4^n に近いが, PTDD 法の場合は 2^n に近い. ただし, PTDD 法では, ROBDD から PTDD への変換時間を示す. 主項生成の場合は, 全ての 1 パスを列挙するための時間が必要である. 一方, 分割統治法では, 主項生成までの時間である.

C. 変数の順序の影響 8 変数の乱数関数に対して変数の順序付を変えた場合 (全部で 40320 通りある) の ROBDD, PTDD および WORK の節点数の分布を表 2に示す. 節点数は, 変数の順序を変化させると, 平均値に比べ 1 割程度変化する.

D. Coudert-Madre の方法との比較 本研究は, TDD の節点数などの統計的性質を調べることを目的としているので, PTDD のプログラムは, 特に高速化のための手法は用いていない. しかし, 従来の方法に比べ大幅に高速である. 今後, いくつかの高速化手法を取り込むことによって, さらに高速化が期待できる. 表 1, 第 8 列に Coudert-Madre の計算時間を示している. 計算時間に関しては, Coudert-Madre のプログラムの方が高速な場合が多い. 本手法で, メモリオーバフローのため解けなかった問題に対して Coudert-Madere の方法ではすべて主項の陰的生成に成功している. 本実験では, 否定枝などの技術は用いていないので, 今後, メモリ削減のための工夫が必要である.

表 2: ROBDD, PTDD, WORK の節点数の分布

	最小	平均	最大
ROBDD	69	74.78	79
PTDD	130	140.92	154
WORK	384	437.76	487

6.2 ROTDD による積和形の表現

ESPRESSO ベンチマーク関数に対して, 論理和形を ROTDD で表現した. ROTDD とキューブ表現に必要なメモリは次式で計算できる.

キューブ表現 (Sc):
積項数 ×(キューブ表現のバイト数 $=n+16$)

ROTDD(St): 節点数 ×(各節点のバイト数 =28)

表 1に, 積項数 (第 4 列) と節点数 (第 8 列左), 及び ($RATE = St/Sc$) を示す. rd73 を除き, キューブ表現の方が記憶容量は少なくて良い. その他のベンチマーク関数で RATE の値が 1 以下になったのは, パリティ関数, cordic, file, sym10, t481 などである.

7 あとがき

本論文では, 論理式や論理関数の表現法として種々の TDD(三分岐判定図) を提案した. 表 3にこれらの性質をまとめたものを示す. PTDD は 1 パスの集合が全ての主項の集合を表現するため, 主項生成法として利用できる. この表現法は Coudert の方法よりも直接的であり分かりやすい. また, 主項生成に必要な PTDD の節点数は $O(3^n/n)$ と推論できた. ROTDD は 1 パスの集合が, 与えられた積和

表 3: 種々 TDD とその性質

	QROBDD	ROBDD	QRATDD	PTDD	ROTDD
表現の一意性	一意的	一意的	一意的	一意的	異なる
節点数	$O(2^n/n)$	$O(2^n/n)$	$O(3^n/n)$	$O(3^n/n)$	$O(n2^n)$
1パスの表現する論理式	主加法標準形	分離的積和形	完全内項積和形	完全主項積和形	積和形
用途	関数表現	関数表現	内項生成	主項生成	積和表現

形を表現する. その他, TDD の第 3 番目の子供に $f_0 \oplus f_1$ を対応させた, EXOR-TDD [20] が考案されている. EXOR-TDD は, AND-EXOR 論理式の簡単化に利用できる.

謝辞

本研究は一部文部省科学研究費補助金による.

参考文献

[1] S. B. Akers, "Binary decision diagrams," *IEEE Trans. Comput.*, Vol. C-27. No. 6, June 1978, pp. 509-516.

[2] A. Blake, "Canonical expressions in Boolean algebra," Ph. D. Dissertation, Dept. of Mathematics, Univ. Chicago, 1937. Published by Univ. Chicago, Libraries, 1937.

[3] K. S. Brace, R. L. Rudell and R. E. Bryant, "Efficient implementation of a BDD package," *Proc. 27th Design Automation Conference*, June 1990, pp. 40-45.

[4] R. K. Brayton, G. D. Hachtel, C. T. McMullen, and A. L. Sangiovanni-Vincentelli, *Logic Minimization Algorithms for VLSI Synthesis*, Boston, MA. Kluwer, 1984.

[5] J. G. Bredeson and P. T. Hulina, "Generation of prime implicants by direct multiplications," *IEEE Trans. Comput.* Vol. C-20, April 1971, pp. 475-476.

[6] F. M. Brown, *Boolean Reasoning: The logic of Boolean equations*, Kluwer Academic Publishers, Boston, 1990.

[7] R. E. Bryant, "Graph-based algorithms for Boolean function manipulation," *IEEE Trans. Comput.* Vol. C-35, No. 8, Aug. 1986, pp. 677-691.

[8] O. Coudert and J. C. Madre, "A new graph based implicit prime and essential prime computation technique," *International Symposium on Logic Synthesis and Microprocessor Architecture*, ISKIT-92, pp. 124-131. "Implicit and incremental computation of Primes and Essential primes of Boolean functions," in *Proc. of DAC-92*, June 1992. Also, in Sasao (ed.), *Logic Synthesis and Optimization*, Kluwer Academic Publisher, pp. 33-57, 1992.

[9] S. R. Das, "A new algorithm for generating prime implicants," *IEEE Trans. Comput.* Vol. C-20, No. 12, 1971, pp. 1614-1615.

[10] M. Fujita, H. Fujisawa, and N. Kawato, "Evaluation and implementation of Boolean Comparison method base on binary decision diagrams," *ICCAD-88*, Nov. 1988, pp. 6-9.

[11] M. A. Harrison, *Introduction to Switching and Automata Theory*, McGraw-Hill, 1965.

[12] H-T. Liaw and C-S. Lin, "On the OBDD representation of generalized Boolean functions," *IEEE Transactions on Comput.* Vol. 41, No. 6, June 1992, pp. 661-664.

[13] S. Minato, N. Ishiura, and S. Yajima, "Shared binary decision diagram with attributed edges for efficient Boolean function manipulation," *Proc. 27th ACM/IEEE Design Automation Conf.*, June 1990, pp. 52-57.

[14] E. Morreale, "Recursive operators for prime implicant and irredundant normal form determination," *IEEE Trans. Comput.* Vol. C-19, No. 6. June 1970, pp. 504-509.

[15] S. Muroga, *Logic Design and Switching Theory*, John Wiley & Sons, 1979.

[16] R. J. Nelson, "Simplest normal truth functions," *J. Symb. Logic* June 1954, pp. 105-108.

[17] W. V. Quine, "A way to simplify truth functions," *Amer. Math. Mon.*, Vol. 62, November 1955, pp. 627-631.

[18] B. Reusch, "Generation of prime implicant from subfunctions and a unifying approach to the covering problem," *IEEE Trans. Comput.* Vol. C-24, No. 9, Sept. 1975, pp. 924-930.

[19] R. L. Rudell, "Multiple-valued logic minimization for PLA synthesis," *Technical Report*, University of California, Electronics Research Laboratory, Berkeley, CA. June 1986.

[20] T. Sasao, "Optimization of multiple-valued AND-EXOR expressions using multiple-place decision diagrams," *ISMVL-92*, May 1992, pp. 451-458. Also, in Sasao (ed.), *Logic Synthesis and Optimization*, Kluwer Academic Publisher, pp. 33-57, 1992.

[21] J. R. Slagle, C. L. Chang, and R. C. T. Lee, "A new algorithm for generating prime implicants," *IEEE Trans. Comput.* Vol. C-19, No. 4. April 1970, pp. 304-310.

[22] P. Tison, "Generalization of consensus theory and application to the minimization of Boolean functions," *IEEE Trans. Electron Comput.*, Vol. EC-16, August 1967, pp. 446-456.

[23] S. Yang, "Logic synthesis and optimization benchmark user guide, Version 3.0," MCNC, Jan. 1991.