数理解析研究所講究録
第790巻 1992年 141-147

二分決定グラフによる論理関数処理の計算複雑さ

武永 康彦　　矢島 脩三
Yasuhiko TAKENAGA and Shuzo YAJIMA
京都大学工学部
Faculty of Engineering, Kyoto University

1 はじめに

二分決定グラフ (BDD : Binary Decision Diagram)[1] は、論理関数の表現方法の一種である。二分決定グラフは、多くの実際的な関数を比較的小さなサイズで表現できる、標準形が存在して変数の順序を固定すると関数の表現が一意に定まる、という特長をもつ。また、論理関数の論理和、論理積等の演算も、表現の大きさの積に比例した時間で実行できる。そのため、実際の組合せ回路に現れる論理式の多くに関して、その等価性判定を効率的に実行することができる。この性質を活かして、論理設計検証、テスト生成、記号シミュレーションなどの分野のアプリケーションにおいて、論理関数の内部表現として広く利用され、大規模な回路に対する高速な処理を可能にしてきた。最近では、より一層の二分決定グラフ処理の高速化をはかるため、ベクトル計算機 [2] や並列計算機 [3] 上でのアルゴリズムの研究もおこなわれている。

本稿は、二分決定グラフによる論理演算の並列計算量を明らかにすることを目的とする。また、その過程において、二分決定グラフに対する種々の基本的処理についてもその計算量を考察する。

本稿では、並列計算のモデルとして、論理回路を用いる。二分決定グラフの既約化、二分決定グラフによる論理演算、否定エッジの除去がともに、NC^2 に属することを示す。

NC^2 は、一般に効率のよい並列化が可能な問題のクラスとみなされており、以下のように定義される。

定義： NC^k, NC

$k \leq 0$ を整数とする。

$NC^k = UniformSIZE\text{-}DEPTH(n^{O(1)}, log^k n)$

$NC = \cup_{k>0} NC^k$

ただし、$UniformSIZE\text{-}DEPTH(n^{O(1)}, log^k n)$ は、多項式サイズ、段数 $O(log^k n)$ の一様回路族により計算される問題のクラスを表す。

2 二分決定グラフ

2.1 定義

二分決定グラフ (BDD)[1] は、論理関数を表現する有向非循環グラフである。ノードの集合は、変数ノードと定数ノードからなる。変数ノードには、入次数が0のノードが1個だけ含まれ、これをルートノードと呼ぶ。変数ノードの出次数は2であり、それぞれ、0エッジ、1エッジと呼ばれる。定数ノードの出次数は0である。

以下では、二分決定グラフは変数ノードを表す4つ組 $< i, x_i, low(i), high(i) >$ の集合およびルートノードの番号で与えられるものとする。i がノードの番号、$x_i \in \{1, 2, \cdots, N\}$(ただし N は表現する論理関数の変数の数) が対応する変数、$low(i)$、$high(i)$ がそれぞれ0エッジ、1エッジで指されるノードの番号を表す。ただし、定数ノードは定まったノード番号0,1を持つものとする。

変数ノード i が表す関数 $f(i)$ は、

$$f_i = x_i \cdot f_{high(i)} + \bar{x}_i \cdot f_{low(i)}$$

で定義される。二分決定グラフ A のルートノードを a とするとき、Aの表す関数 f_A は f_a に等しい。

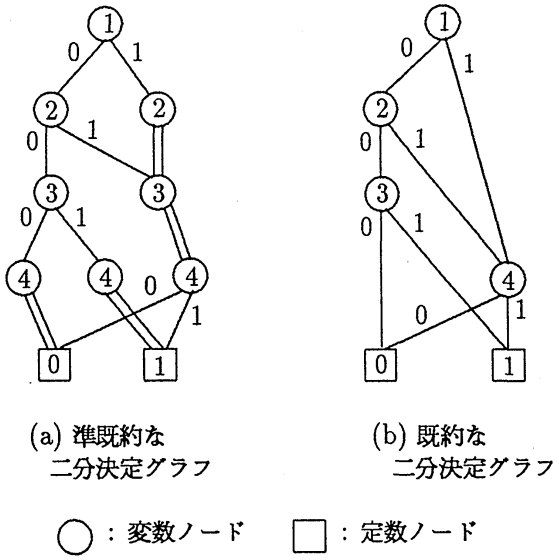

図 1: 準既約な二分決定グラフと既約な二分決定グラフ

二分決定グラフは、$\{1,2,\cdots,N\}$ 上のある置換 π に対して、$low(i)$ あるいは $high(i)$ が定数ノードとなる場合を除き、すべてのノード i が、

$\pi(x_i) < \pi(x_{low(i)})$,

$\pi(x_i) < \pi(x_{high(i)})$

を満たすものとする。$\pi(x_i)$ を $level(i)$ と記し、ノード i のレベル、あるいは変数 x_i のレベルと呼ぶ。

$level(i) < level(j)$ が成立するとき、i は j より上位にあり、j は i より下位にあると呼ぶ。ルートノードは最上位 (レベルが 1) にあたる。

2 個のノード i,j が、同じ関数を表すとき、i と j は等価であると呼び、$i \equiv j$ と記すものとする。i と j が等価である条件は、

$level(i) = level(j)$,

$high(i) \equiv high(j)$,

$low(i) \equiv low(j)$

が成立することである。また、

$high(i) = low(i)$

を満たすノードを冗長なノードと呼ぶ。

二分決定グラフのすべての変数ノードが、

$level(i) + 1 = level(low(i)) = level(high(i))$

を満たすとき、密な二分決定グラフと呼ぶ [6]。任意の二分決定グラフは、冗長なノードを付加することにより密な二分決定グラフに変換することができる。

密な二分決定グラフから、等価なノードをすべて 1 個にまとめたものを、準既約な二分決定グラフと呼ぶ。さらに、準既約な二分決定グラフから冗長なノードをすべて削除したものを、既約な二分決定グラフと呼ぶ。ただし、ノード i を削除するとき、i を指していたエッジはすべて $high(i)(= low(i))$ に置き換える。図 1 に、論理関数 $f = \bar{x}_1\bar{x}_2x_3 + (x_1 + x_2)x_4$ を表現する準既約な二分決定グラフと既約な二分決定グラフを示す。

以上で定義した二分決定グラフは、次のような特長をもつ。

- 論理関数を既約な二分決定グラフ、あるいは準既約な二分決定グラフで表した場合、各変数のレベルを固定すれば　その表現が一意に定まる。そのため、関数の一致判定をグラフの同型判定によりおこなうことができる。
- n 変数論理関数を表す二分決定グラフのノード数は最大で $O(2^n/n)$ となるが、多くの実際的な関数を比較的少ないノード数で表現できる。

2.2 従来の論理演算アルゴリズム

二分決定グラフによる論理演算の基本的アルゴリズムは、グラフを再帰的にたどることにより、下位の変数から順に新しい二分決定グラフのノードを生成する。

二分決定グラフA，Bのルートノードをそれぞれ a, b とすると、Cは a, b に対して以下のアルゴリズムを適用することにより求まる。ここでは、論理演算として論理積を例にとる。

論理関数 $f_r \cdot f_s$ を表すノードを $c_{r,s}$ とする。

[論理演算アルゴリズム 1]

$f_i \cdot f_j$ を表す二分決定グラフを求める。

1： $f_i = 0$、あるいは $f_j = 0$ のとき、 $f_{c_{i,j}} = 0$。

2： $f_i = 1$ のとき、

$$high(c_{i,j}) = high(j),\quad low(c_{i,j}) = low(j)$$

となるノード $c_{i,j}$ を生成する。$f_j = 1$ のとき、

$$high(c_{i,j}) = high(i),\quad low(c_{i,j}) = low(i)$$

となるノード $c_{i,j}$ を生成する。

3： それ以外の場合、ノード $c_{i,j}$ を生成し、 $f_{low(i)} \cdot f_{low(j)}$ を求め、

$$low(c_{i,j}) = c_{low(i),low(j)}$$

とする。$high(c_{i,j})$ についても同様。 □

ただし、新しいノードが冗長なノードである場合には新たなノードは生成しない。また、すでに等価なノードが生成されている場合にも、新たなノードは生成せず、等価なノードを共有する。

1: および 2: では、2個のノードの論理積が自明に求まる場合の処理であり、論理積以外の論理演算の場合には、演算の種類に応じた処理をおこなう。

再帰呼び出しに基づく逐次アルゴリズムは並列化に適さないため、[2, 3] の並列アルゴリズムは、上位の変数から順に次のレベルのノードを生成する方法を用いている。ただし、以下のアルゴリズムでは、入力として準既約な二分決定グラフを仮定している。

[論理演算アルゴリズム 2]

1： ノード $c_{a,b}$ を生成する。$lev = 1$。

2： $level(c_{i,j}) = lev$ となるすべてのノード $c_{i,j}$ に対して、

$$high(c_{i,j}) = c_{high(i),high(j)},$$
$$low(c_{i,j}) = c_{low(i),low(j)}$$

とする。

3： $lev = N$ でなければ、$lev = lev + 1$ として、2: へ。

4： そのレベルが lev に等しいノードに対して、冗長なノードの除去、等価なノードの共有をおこなう。

5： $lev=1$ でなければ、$lev=lev-1$ として、4: へ。

□

再帰呼び出しに基づくアルゴリズムでは、ノードの生成の際に等価性や冗長性の判定をおこなうため、解として必要なノードのみを生成する。それに対し、このアルゴリズムでは、1: および 2: において上位のノードから順に幅優先で処理をおこなうため、冗長なノードや等価なノードが生成される。3: および 4: において、下位から順に二分決定グラフの既約化をおこなう。

3 二分決定グラフ処理の計算複雑さ

本章では、二分決定グラフに対する基本的操作の計算複雑さを明らかにすることにより、二分決定グラフによる論理演算が NC^2 に属することを示す。

二分決定グラフのノード数を n とすると、入力のビット数は $O(nlogn)$ となるが、本章では、簡単のため n を基準に議論する。入力として複数の二分決定グラフを与える場合には、どちらに属するノードであるかを示す値を入力として与えるものとし、そのノード数の合計を n とする。また、本節で議論するすべての問題では、出力される二分決定グラフのノード数をあらかじめ計算することはできない。そのため、出力の各ノードに対し、それが解に含まれるか否かを示すビットを付加して出力するものとする。

3.1 二分決定グラフの既約化

定義：二分決定グラフの既約化問題

二分決定グラフAが与えられたとき、$f_A=f_B$ を満たす既約な二分決定グラフBを求める。

定理 3.1 二分決定グラフの既約化問題は NC^2 に属する。

定理 3.1 は、二分決定グラフの形の変換に関する、以下の 3 個の重要な補題から導かれる。ただし、これらの補題は、NC^2 への所属を示すことを目的としており、その素子数は減らせる可能性がある。

補題 3.2 任意の二分決定グラフから、等価で密な二分決定グラフを、段数 $O(log^2n)$、素子数 $O(n^3logn)$ の回路で計算できる。

補題 3.3 密な二分決定グラフから、等価で準既約な二分決定グラフを、段数 $O(log^2n)$、素子数 $O(n^6)$ の回路で計算できる。

補題 3.4 準既約な二分決定グラフから、等価で既約な二分決定グラフを、段数 $O(log^2n)$、素子数 $O(n^3logn)$ の回路で計算できる。

補題 3.2 の証明

まず、各ノードのレベルを求める。

任意の 2 個の変数 i と j に対し、$x_k=i$ かつ、$x_{high(k)}=j$、または $x_{low(k)}=j$ となるノード k が存在するかどうかを調べる。存在した場合、$level(i)<level(j)$ を満たす。

i と j を固定した場合、これを計算するには、すべてのノードに対して上記の条件を満たすかどうかをチェックすればよい。1 個のノードに対するチェックは、変数の比較をおこなうことにより、$O(loglogn)$ 段で可能である。これをすべてのノードに対して並列に計算し、その論理和を $O(logn)$ 段で求める (存在するとき 1 とする)。素子数は、1 個の i,j に対して $O(nlogn)$、全体で $O(n^3logn)$ となる。

以上により求められた、変数のレベルに関する半順序関係をノード数 N の有向グラフと考え、topological ソーティングをおこなう。この結果、小さい方から k 番目になった変数のレベルを k とする。Topological ソーティングは NC^2 に属することが知られている。例えば、以下の方法で計算できる。

まず、有向グラフの接続行列の推移的閉包を計算することにより、各レベルより下位にある変数が求まる。ブール行列の推移的閉包は $O(log^2n)$ 段の回路で計算できる。これをもとに、マージソートをおこな

う。ソートされた系列$\{a_1,\cdots,a_{k/2}\}$と$\{a_{k/2+1},\cdots,a_k\}$のマージを考える。$\{a_1,\cdots,a_{k/2}\}$の各要素に対しては、$\{a_{k/2+1},\cdots,a_k\}$のうち上位にない要素の数を計算し、その数だけ右にシフトする。$\{a_{k/2+1},\cdots,a_k\}$の各要素についても、下位にない要素の数にしたがって、左にシフトをおこなう。要素数kのソーティングは$O(logk)$段でおこなえる。この方法では、素子数は$O(n^3logn)$となる。

各ノードのレベルを求めると、次に密な二分決定グラフにするためのノードを補う。1本のエッジに対して、最大$N-1$個のノードを補うことが必要である。あるエッジに対し、レベルtのノードを補うかどうかは、両端のノードのレベルをtと比較することにより決められる。新しいノードの番号は、もとのエッジの出るノード番号、0エッジ、1エッジの別とノードのレベルの連接により定める。新しいノードを補ったエッジに関しては、そのエッジの指すノード番号を新しいノードに変更する。

レベルの比較は段数$O(loglogn)$、新しいノード1個について素子数$O(logn)$で構成できる。ノード番号の変更は、段数$O(1)$で可能である。したがって、ノードを補う回路は段数$O(loglogn)$、素子数$O(n^2logn)$で構成できる。 □

補題3.3の証明

$n(n-1)/2$個の、ノードの非順序対を考える。任意の2個のノードの対i,jに対して、iとjが等価になるか否かを求める。まず、iとjが等価になるために等価であることが必要なノードの対の集合$P(i,j)$を求める。すべての対に対し、それが$P(i,j)$に含まれるとき1となる信号線をもうけ、その値を計算する。この本数は各i,jに対して$O(n^2)$、合計で$O(n^4)$となる。ただし、レベルの異なるノードは等価にならないため、レベルを比較し$level(i)\neq level(j)$が成立すれば、その対についての結果は無視する。

$P_k(i,j)$を計算する回路P_kを考える。P_1では、すべての対に関して$<high(i),high(j)>$および$<low(i),low(j)>$を$P_1(i,j)$に加える。P_k $(2\leq k)$では、すべての$<p,q>\in P_{k-1}(i,j)$に対して、$P_{k-1}(p,q)$の要素を$P_k(i,j)$に加える。

回路P_1からP_{logN}を順に接続することにより、レベルの等しいi,jにおいて、$P(i,j)$の要素$<r,s>$のr,sは定数ノードとなる。このとき、$P(i,j)$の要素が$<0,0>$および$<1,1>$のみからなれば、iとjは等価である。

各P_k $(2\leq k)$の回路は、以下のように構成される。各$i,j,<p,q>,<r,s>$に対し、$<p,q>\in P_{k-1}(i,j)$かつ$<r,s>\in P_{k-1}(p,q)$が成立するかどうかを計算する。この回路は、段数$O(1)$、素子数$O(n^6)$で構成できる。$<r,s>$が$P_k(i,j)$に含まれるか否かは、すべての$<p,q>$について、この結果の論理和を計算することにより段数$O(logn)$、素子数は、各$i,j,<r,s>$につき$O(n)$、全体で$O(n^3)$で求まる。

レベルの比較は段数$O(loglogn)$、P_{logN}の出力から等価性を調べる回路は$O(1)$で構成できる。

次に、等価なノードの対をもとに、それらを代表するノードを求める。いま、$<i,j>$が$i<j$を満たすものとする。(実際の回路では、ノード番号の大小でなく、入力の位置関係が保たれている。)ノードの対$<t,j>$が等価になるようなノードtが存在すれば、ノードjは代表するノードとしない。この操作により、等価なノードの集合に対して、最もノード番号の小さなノードのみが残り、これを代表とする。代表するノードiと等価なノードの集合$E(i)$は、$<i,t>$が等価となるtの集合である。

最後に、不要なノードを削除し、さらに、子ノードが$E(i)$の要素であるものを、すべてノードiに置き換える。

代表となるノードを求める回路は、最大n個の論理和をとることにより計算できるため、段数$O(logn)$、素子数$O(n^2)$で構成できる。また、ノードの削除およびノード番号の書き換えは、段数$O(logn)$、素子数は子ノードの番号を一箇所訂正するのに、n回の一致判定が必要なことから$O(n^2)$、合計で$O(n^4)$となる。

以上より、補題3.2の回路は、段数$O(log^2n)$、素子数$O(n^6)$で構成できる。 □

補題3.4の証明

準既約な二分決定グラフから、冗長なノードの除去をおこなう。

以下の回路 R_k を考える。R_k では、ある整数 s に対し、

$level(i) = s2^k + 2^{k-1} + 1,\ 1 \leq level(i) \leq N$

となるすべてのノード i に対して、

$high(i) = low(i)$

となるかどうかを調べる。一致したノード i は除去し、i を指すエッジは i に代えてノード $r(= high(i) = low(i))$ を用いる。

R_k で冗長なノードの除去をおこなうレベルは、2進数で表したときその下位から k ビットのみから定まるため、このビットの比較により判定できる。したがって、冗長なノードの判定は段数 $O(logn)$、素子数 $O(n^2)$ の回路で実現できる。エッジのつけ替えは、各ノードにおいて除去される全ノードとの一致を調べることにより、段数 $O(logn)$、素子数 $O(n^3)$ で実現できる。

回路 R_1 から R_{logN} までを順に接続することにより、冗長なノードはすべて除去される。 □

以上の補題で示した回路を生成するには、n の多項式までの数を定数個数えれば十分であることは、上記の証明を詳細に追えば明らかであり、対数領域一様性をもつ。これは、次節以降に示すすべての回路についても同様である。

3.2 二分決定グラフによる論理演算

二分決定グラフの論理演算問題を以下のように定式化する。

定義：二分決定グラフの論理演算問題

各変数のレベルが等しい2個の二分決定グラフA，Bが与えられたとき、$f_A\ op\ f_B = f_C$ を満たす既約な二分決定グラフCを求める。ただし、op は任意の論理演算を示す。

性質 3.5 二分決定グラフの論理演算問題において、二分決定グラフCのノード数は $O(n^2)$ で抑えられる。

定理 3.6 二分決定グラフの論理演算問題は NC^2 に属する。

証明

まず、入力として与えられた二分決定グラフを補題 3.2 および補題 3.3 の回路により準既約な二分決定グラフに変換する。

ノード a_i, b_j をそれぞれ、準既約な二分決定グラフA，Bのノードとする。

$level(a_i) = level(b_j)$

を満たすとき、以下のようにCのノード $c_{i,j}$ を定める。

$level(a_i) = level(b_j) \neq N$ のとき、

$high(c_{i,j}) = c_{high(i),high(j)}$,

$low(c_{i,j}) = c_{low(i),low(j)}$。

$level(a_i) = level(b_j) = N$ のとき、

$high(c_{i,j}) = op(high(i), high(j))$,

$low(c_{i,j}) = op(low(i), low(j))$。

これを、A，Bのノードのすべての組合せに対して実行することにより、二分決定グラフCが得られる。ただし、ここで得られたノードには、ルートノードから到達できない不要なノードが含まれる。

上記の方法でCのノードを求めるための回路を構成する。入力のノードから2個を選んだすべての組合せに対してCのノードが作られるかどうかを調べる。これらのうち、2個のノードがそれぞれA，Bに含まれ、そのレベルが同じもののみからCのノードが生成される。新しいノード番号を、もとの2個のノードのノード番号を連接したものとすれば、$high(c_{i,j})$ および $low(c_{i,j})$ も容易に計算できる。

2個のノードの組合せに対して、Cのノードを作るか否かは、変数の比較により段数 $O(logn)$、素子数 $O(n)$ の回路で判定できる。したがって、全体での素子数は $O(n^3)$ となる。新しいノードの生成は、段数 $O(1)$ で実現できる。

上記の方法で求めた二分決定グラフから、レベル0のノードから到達できない不要なノードを除去する。この二分決定グラフに対し、接点接続行列の推移的閉包を求めることにより、ルートノードから到達できるノードの集合を求めることができる。ノード数が$O(n^2)$であることから、補題3.2の方法で段数$O(log^2 n)$でノードの除去がおこなえる。 □

4 おわりに

本稿の結果は、二分決定グラフ処理において、高度な並列化により有効な処理がおこなえる可能性を示した。特に、これまで考えられていたようにレベル内のノードに関して並列化し、レベル毎に処理をおこなうだけでなく、すべてのノードに対して並列に処理をおこなうことにより、二分決定グラフ上の論理演算の並列計算時間が、変数の数に対して線形時間以下になりうることを示した。

参考文献

[1] R. E. Bryant : "Graph-Based Algorithms for Boolean Function Manipulation", IEEE Trans. Comput. Vol.C35, No.8, pp.677-691 (1986).

[2] H. Ochi, N. Ishiura and S. Yajima: "Breadth-First Manipulation of SBDD of Boolean Functions for Vector Processing", Proc. 28th ACM/IEEE DAC, pp.413-416 (1991).

[3] S.Kimura and E.M.Clarke : "A Parallel Algorithm for Constructing Binary Decision Diagrams", Proc. IEEE ICCD'90, pp.220-223 (1990).

[4] R. E. Bryant : "On the Complexity of VLSI implementations and Graph Representations of Boolean Functions with Application to Integer Multiplecation", IEEE Trans. Comput, Vol.40, No.2, pp.205-213 (1991).

[5] 石浦菜岐佐 : "多項式サイズの二分決定グラフで表現可能な論理関数のクラス", 情処研報 AL20-5, pp.1-7 (1991).

[6] 石浦菜岐佐 : "二分決定グラフからの組合せ論理回路の合成", 情処研報 DA60-20, pp.155-161 (1991).

[7] 湊真一, 石浦菜岐佐, 矢島脩三 : "論理関数の共有二分決定グラフによる表現とその効率的処理手法", 情報処理学会論文誌, Vol. 32, No. 1, pp.77-85 (1991).