数理解析研究所講究録
第 833 巻 1993 年 236-241

A Language Complete for the Families of Polynomial-Size Binary Decision Diagrams
多項式サイズの二分決定グラフの族に対する完全言語

Hiroshi SAWADA 澤田 宏 **Yasuhiko TAKENAGA** 武永 康彦 **Shuzo YAJIMA** 矢島 脩三

Faculty of Engineering, Kyoto University
京都大学 工学部

1 Introduction

A binary decision diagram (BDD) [1] is one of representation forms of Boolean functions. It can represent many practical Boolean functions by feasible size and there exists a unique canonical form for each Boolean function. Therefore it is widely used for manipulating Boolean functions on computers.

A BDD represents a Boolean function. On the other hand, a family of BDD's represents a language. The class of languages accepted by families of polynomial-size BDD's (*PolyBDD*) [2] can be seen as a complexity class which is computable by BDD's of feasible size. A family of symmetric functions, the language $\{0^n1^n\}$ and the language $\{ww|w \in \{0,1\}^*\}$ are examples of elements of *PolyBDD*.

In this paper, we show a complete language for *PolyBDD* under constant-depth circuit reducibility. It reflects the characteristics of *PolyBDD* and is the representative language of *PolyBDD*. To clarify the relations between *PolyBDD* and other complexity classes, it is sufficient to clarify the relations between a complete language for *PolyBDD* and other complexity classes.

This paper is organized as follows. In section 2, we define a BDD and *PolyBDD*. In section 3, we show a complete language for *PolyBDD* under constant-depth circuit reducibility. In section 4, we conclude our discussion.

2 Preliminaries

2.1 Binary Decision Diagram (BDD)

A binary decision diagram (BDD) (Figure 1) [1] which represents an n-variable Boolean function $f(x_1, \cdots, x_n)$ is a 6-tuple $(N_V, N_C, init, edge, level, \pi)$, where

N_V is a set of variable nodes,
$N_C = \{c_0, c_1\}$ is the set of constant nodes,
$init \in N_V$ is an initial node,
$edge : N_V \times \{0,1\} \to (N_V \cup N_C)$ is a set of edges,
$level : (N_V \cup N_C) \to \{1, \cdots, n+1\}$ is a mapping from the set of nodes to the set of levels such that
  $level(init) = 1$,
  $level(v) < level(edge(v,b))$ $(b \in \{0,1\})$ if $v \in N_V$,
  $level(v) = n+1$ if $v \in N_C$,
$\pi : \{1, \cdots, n\} \to \{1, \cdots, n\}$ is a permutation from the set of levels of variable nodes to the set of indexes of variables, called a variable order.

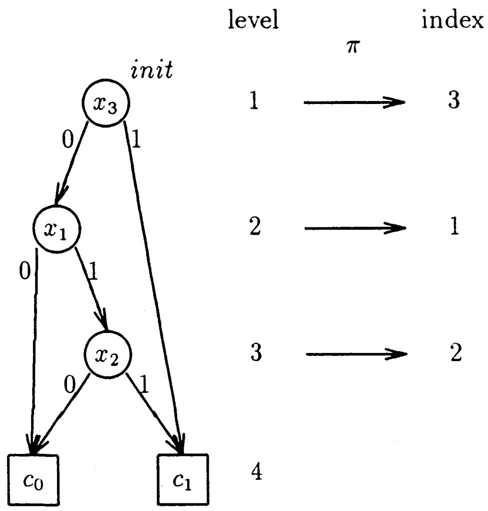


Figure 1: A BDD representing a Boolean function $x_1 \cdot x_2 + x_3$

Each node v of a BDD represents a Boolean function f_v defined as follows,

$f_{c_0} = 0$ (inconsistency),
$f_{c_1} = 1$ (tautology),
$f_v = \overline{x_{\pi(level(v))}} \cdot f_{edge(v,0)} + x_{\pi(level(v))} \cdot f_{edge(v,1)}$ if $v \in N_V$.

A BDD $(N_V, N_C, init, edge, level, \pi)$ represents the Boolean function f_{init}. A BDD is called reduced if there are not any node v such that $edge(v,0) = edge(v,1)$ and any pair of nodes v, v' such that $level(v) = level(v')$, $edge(v,0) = edge(v',0)$ and $edge(v,1) = edge(v',1)$. A reduced BDD represents a Boolean function with a variable order is unique up to isomorphism.

2.2 Family of Polynomial-Size BDD's

A BDD is one of representation forms of Boolean functions. On the other hand, a family of BDD's represents a language. A family $\{B_n\}$ of BDD's is a sequence $B_1, B_2, \cdots$ of BDD's, where $B_n = (N_V, N_C, init, edge, level, \pi)$ is a BDD representing an n-variable Boolean function. A family $\{B_n\}$ of BDD's is said to accept a language $L \subseteq \{0,1\}^*$ if and only if

$\forall n,\ b_1 \cdots b_n \in L \Leftrightarrow f_n(b_1, \cdots, b_n) = 1$, where f_n is the Boolean function which B_n represents.

We can regard a family of BDD's as a computational model which accept a language.

Definition 1 Let *PolyBDD* be the class of languages which are accepted by families of BDD's whose size are bounded by a polynomial of the number of the variables. □

In this paper, we do not consider the uniformity, the property that the function $n \rightarrow B_n$ is computable easily, of families $\{B_n\}$ of BDD's. In order to characterize nonuniform families of BDD's, we use nonuniform on-line Turing machines.

A nonuniform Turing machine is a Turing machine with a two-way read-only input tape, a two-way work tape and a two-way read-only advice tape. For an input $b_1 \cdots b_n$ $(b_1, \cdots, b_n \in$

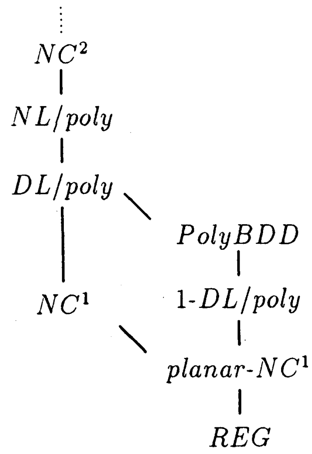


Figure 2: The relations among the classes

$\{0,1\}$), a nonuniform Turing machine start its computation with $b_1 \cdots b_n$ on the input tape and with $\alpha(n)$ on the advice tape, where $\alpha : \{1, 2, \cdots\} \to \{0,1\}^*$ is called an advice function. A nonuniform on-line Turing machine is a nonuniform Turing machine whose input tape is one-way.

Let *DL/poly*, *NL/poly* and 1-*DL/poly* be the class of languages accepted by logarithm-space bounded deterministic nonuniform Turing machines with polynomial advice, logarithm-space bounded nondeterministic nonuniform Turing machines with polynomial advice and logarithm-space bounded deterministic nonuniform on-line Turing machines with polynomial advice.

Let NC^k ($k = 1, 2, \cdots$) be the class of languages accepted by nonuniform families of constant fan-in logic circuits of $\log^k n$-depth and polynomial-size for n inputs.

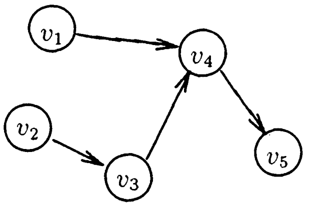
On the relations between *PolyBDD* and other classes, the following results are obtained (Figure 2), $planar\text{-}NC^1 \subseteq 1\text{-}DL/poly$ [3], $1\text{-}DL/poly \subset PolyBDD \subset DL/poly$ [2] [4], $NC^1 \subseteq DL/poly \subseteq NL/poly \subseteq NC^2$ [5], where REG is the class of regular languages and $planar\text{-}NC^1$ is the class of languages accepted by nonuniform families of constant fan-in planar circuits of $\log n$-depth and polynomial-size for n inputs.

3 A Complete Language for the class *PolyBDD*

3.1 Constant-Depth Circuit Reducibility

Let $L, L_c \subseteq \{0,1\}^*$. We say that L is constant-depth reducible to L_c (denote $L \leq_{cd} L_c$) if and only if there exists a function $f : \{0,1\}^* \to \{0,1\}^*$ computable by a family of constant fan-in logic circuits of constant-depth and polynomial-size such that for all $x \in \{0,1\}^*$, $x \in L \Leftrightarrow f(x) \in L_c$. Note that $|f(x)|$ is bounded by a polynomial of $|x|$ since the circuits are polynomial-size.

For a class $\mathcal{C}$, if $\forall L \in \mathcal{C}$, $L \leq_{cd} L_c$, we say that L_c is hard for $\mathcal{C}$ under constant-depth circuit reducibility. If L_c is hard for $\mathcal{C}$ and $L_c \in \mathcal{C}$, we say that L_c is complete for $\mathcal{C}$. If L_c is complete for $\mathcal{C}$ and $L_c \in NC^1$, it follows that $\mathcal{C} \subseteq NC^1$.



$$\begin{array}{c c} & \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} & \begin{bmatrix} 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \end{array}$$

Figure 3: A directed graph and its adjacency matrix

3.2 A Complete Language - Topologically Arranged Deterministic Graph Accessibility Problem

We show that Topologically Arranged Deterministic Graph Accessibility Problem (*TADGAP*) [6] is a complete language for *PolyBDD* under constant-depth circuit reducibility. Before defining the language *TADGAP*, we define a directed graph and its adjacency matrix (Figure 3). A directed graph is a 2-tuple (V, E), where

$V = \{ v_1, \cdots, v_{|V|} \}$ is a set of nodes,
$E \subseteq \{ (v_i, v_j) \mid v_i, v_j \in V \}$ is a set of directed edges.

The adjacency matrix (x_{ij}) of a directed graph $G = (V, E)$ is a $|V| \times |V|$ matrix and its element x_{ij}, $(1 \leq i, j \leq |V|)$ is defined as follows,

$$x_{ij} = \begin{cases} 0 & \text{if } \ (v_i, v_j) \notin E, \\ 1 & \text{if } \ (v_i, v_j) \in E. \end{cases}$$

We say that there exists a path from a node v_{i_1} to a node v_{i_2} in a directed graph (V, E) if there exist nodes $v_{j_1}, v_{j_2}, \cdots, v_{j_k} \in V$ such that (v_{i_1}, v_{j_1}), (v_{j_1}, v_{j_2}), $\cdots$, $(v_{j_k}, v_{i_2}) \in E$. We define that the outdegree of a node v_i is the value $| \{ v_j \mid (v_i, v_j) \in E \} |$. We say that a directed graph (V, E) is topologically sorted if for all $v_i, v_j \in V$, $(v_i, v_j) \in E \Rightarrow i \leq j$ is satisfied.

Definition 2 $TADGAP = \{ x_{11}x_{12} \cdots x_{1m} \cdots x_{mm} \mid$
(x_{ij}) is the adjacency matrix of a directed graph G such that
G is topologically sorted,
the outdegree of each node of G is 0 or 1,
there exists a path from v_1 to v_m of G. $\}$ □

Theorem 1 *TADGAP* is constant-depth complete for *PolyBDD*.
[proof] From the following two lemmas. □

Lemma 1 *TADGAP* $\in$ *PolyBDD*
[proof] We prove that *TADGAP* $\in$ 1-*DL/poly* ($\subseteq$ *PolyBDD*). Let M be a logarithm-space bounded nonuniform on-line Turing machine with polynomial advice. We design M to accept the language *TADGAP*. Let the input of M be $y \in \{0,1\}^*$ and y is the adjacency matrix of a directed graph G. The advice of M for the input of length n is the value of $m = \sqrt{n}$. M can check the following three conditions using the logarithm-space since M knows the value of $m = \sqrt{n}$, 1) G is topologically sorted, 2) the outdegrees of nodes of G is 0 or 1, 3) there exists a path from node v_1 to v_n satisfying the conditions above. Therefore *TADGAP* $\in$1-*DL/poly* $\subseteq$ *PolyBDD*. □

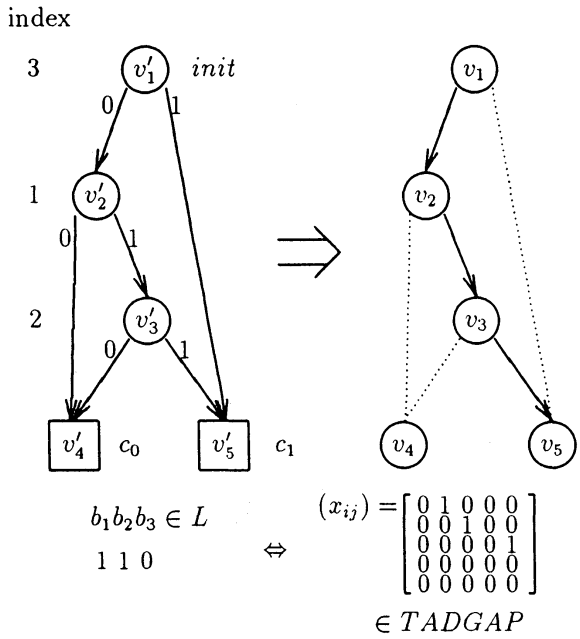


Figure 4: An example of reduction

Lemma 2 $\forall L \in PolyBDD,\ L \leq_{cd} TADGAP$
[proof] For each $L \in PolyBDD$, we consider the family $\{B_n\}$ of polynomial-size BDD's accepting the language L. Let the n-variable BDD of $\{B_n\}$ be $B_n = (N_V, N_C, init, edge, level, \pi)$ and $v'_i, v'_j \in (N_V \cup N_C)$ such that

$$\begin{aligned}
&v'_1 = init, \\
&v'_j = edge(v'_i, b) \Rightarrow i < j,\ (\forall v'_i, v'_j,\ \forall b \in \{0,1\}), \\
&v'_m = c_1\ (m = |N_V \cup N_C|).
\end{aligned}$$

For B_n and the input $b_1 \cdots b_n \in \{0,1\}^n$ of B_n, we consider the directed graph $G = (V, E)$ (figure4), where

$$\begin{aligned}
&V = (N_V \cup N_C) \text{ such that } v_i = v'_i\ (1 \leq i \leq m), \\
&E = \{\ (v', edge(v', b_{\pi(level(v'))}))\ |\ v' \in N_V\ \}.
\end{aligned}$$

The directed graph G has the outdegree of 0 or 1 and is topologically sorted. It seems to be clear that there exists a path from node v_1 to v_m of G if and only if $b_1 \cdots b_n \in L$. Hence, if we let the adjacency matrix of G be y,

$$b_1 \cdots b_n \in L \Leftrightarrow y \in TADGAP.$$

$|y|$ is bounded by a polynomial of n because m, the size of B_n, is bounded by a polynomial of n. An element x_{ij} $(1 \leq i, j \leq m)$ of y is computable by the formula

$$x_{ij} = \bigvee_{b \in \{0,1\}} (\ b_{\pi(level(v'_i))} = b \wedge v'_j = edge(v'_i, b)\ ).$$

Therefore y is computable by a constant-depth circuit from $b_1 \cdots b_n$. □

3.3 The Relation between *PolyBDD* and NC^1

It is known that $NC^1 \not\subseteq PolyBDD$ because the Boolean function of the n-th bit output of the n-bit binary multiplier can not be represented by a BDD of polynomial-size whereas can be represented by a logic circuit of logarithm-depth [7]. Here, we have a question whether $PolyBDD \subset NC^1$ or not (Figure 2). From the result of theorem 1, we have the following result.

Corollary 1 $TADGAP \in NC^1 \Leftrightarrow PolyBDD \subset NC^1$ □

We conclude that the necessary and sufficient condition for $PolyBDD \subset NC^1$ is $TADGAP \in NC^1$. However, the following result indicate that to clarify whether ($1\text{-}DL/poly \subseteq$) $PolyBDD \subset NC^1$ is as difficult as to clarify whether $DL/poly \subseteq NC^1$, which is one of the famous open problems.

Theorem 2 ([3]) $1\text{-}DL/poly \subseteq NC^1 \Leftrightarrow DL/poly \subseteq NC^1$ □

4 Conclusion

In this paper, we show that *TADGAP* is a complete language for *PolyBDD* under constant-depth circuit reducibility and that the necessary and sufficient condition for $PolyBDD \subset NC^1$ is $TADGAP \in NC^1$. However, to clarify whether $PolyBDD \subset NC^1$ is as difficult as to clarify whether $DL/poly \subseteq NC^1$, one of the famous open problems.

Acknowledgment

The authors would like to thank all the members of Yajima Lab. at Kyoto University for their valuable discussion and suggestions.

References

[1] R. E. Bryant. Graph-based algorithms for boolean function manipulation. *IEEE Trans. Comput.*, C-35(8):677–691, August 1986.

[2] N. Ishiura and S. Yajima. A class of logic functions expressible by a polynomial-size binary decision diagram. In *Proc. Synthesis and Simulation Meeting and Int. Interchange (SASIMI '90)*, pages 48–54, October 1990.

[3] 池川 将夫. 論理回路の複雑さに関する一考察. *LA シンポジウム*, July 1989.

[4] H. Sawada, Y. Takenaga and S. Yajima. On the relations between binary decision diagrams, Turing machines and combinational logic circuits. *IEICE, COMP92-66*, November 1992.

[5] Allan Borodin. On relating time and space to size and depth. *SIAM Journal on Computing*, 6(4):733–744, December 1977.

[6] J. Hartmanis and S. Mahaney. Languages simultaneously complete for one-way and two-way log-tape automata. *SIAM Journal on Computing*, 10(2):383–390, May 1981.

[7] R. E. Bryant. On the complexity of VLSI implementations and graph representations of boolean functions with application to integer multiplication. *IEEE Trans. Comput.*, 40(2):205–213, February 1991.