アルゴリズム 30－19
(1992. 11. 21)

二分決定グラフとチューリング機械と組合せ論理回路の関係について

澤田 宏　　武永 康彦　　矢島 脩三

京都大学工学部情報工学教室

〒606-01 京都市左京区吉田本町

あらまし ある論理関数を表現する二分決定グラフは，入力変数の順序を変更することでその大きさが変化する．我々は，入力変数の順序づけを考慮した場合の，二分決定グラフの表現能力について議論する．そのため，入力テープの内容を並び替える機械を別に持つオンラインチューリング機械を定義し，二分決定グラフとの関係を明らかにする．また，セレクタのみで構成される回路を定義し，二分決定グラフとの関係について考える．さらに，平面回路と二分決定グラフの関係についても考える．

和文キーワード 二分決定グラフ，オンラインチューリング機械，平面回路，変数の順序づけ，計算複雑度

On the Relations between Binary Decision Diagrams , Turing Machines and Combinational Logic Circuits

Hiroshi SAWADA , Yasuhiko TAKENAGA and Shuzo YAJIMA

Department of Information Science, Faculty of Engineering, Kyoto University

Yoshida-honmachi Sakyo-ku Kyoto, 606-01

Abstract The size of the Binary Decision Diagram which represents a Boolean function depends on the ordering of input variables. We discuss the expressible power of Binary Decision Diagrams considering the ordering of input variables. For this purpose, we define a on-line Turing machine with an input ordering machine and clarify the relation between Binary Decision Diagrams. We also define a circuit which consists only of selectors and discuss the relation between Binary Decision Diagrams. Furthermore, the relation between Binary Decision Diagrams and planar circuits is discussed.

英文 key words binary decision diagram, on-line Turing machine, planar circuit, ordering of variables, computational complexity

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 space and there exist a unique canonical form of BDD for each Boolean function. Therefore, it is widely used for manipulating Boolean functions on computers.

However, there exist Boolean functions which can not be expressed by BDD's of feasible size, whereas can be expressed by other representation forms of feasible size. An example of this is the Boolean function which represents the n-th bit of the output of a n-bit binary multiplier. This Boolean function can not be expressed by the BDD of polynomial size, whereas can be expressed by the combinational circuit of $O(\log n)$-depth and polynomial size [2].

The reason for this is that each input variable of a Boolean function which a BDD expresses is tested at most once in some ordering. There exist other models of computation having such a limitation. For example, an on-line Turing machine can move the head of the input tape only in one direction. In order to clarify the computational power of a BDD, it is important to discuss the relations between BDD's and other models of computation having this limitation. Several studies have been made on this topic. Ishiura [3] showed that the class of languages which are accepted by uniform families of BDD's of polynomial size is equivalent to the class of languages which are accepted by logarithm space bounded on-line Turing machine. Ikekawa [4] showed that the class of languages which are accepted by uniform families of planar circuits of $O(S(n))$-depth ($S(n) \geq \log n$) is includes by the class of languages which are accepted by $O(S(n))$-space bounded on-line Turing machine.

In the above two results, the ordering of input variables was not considered. In this paper, we consider the ordering of input variables and discuss the relations between these models. In the practical use of BDD's, we can order the input variables such that the sizes of representation are reduced. For example, let us discuss about a language $L = \{ww | w \in \{0,1\}^*\}$. A Boolean function corresponds to a language $L \cap \{0,1\}^n$ can not be expressed by a BDD of polynomial size in the variable ordering $x_1, x_2, \cdots, x_{n-1}, x_n$, whereas can be expressed in the variable ordering $x_1, x_{n/2+1}, \cdots, x_{n/2}, x_n$.

In order to relate BDD's which can order the input variables to on-line Turing machines, we define a Π on-line Turing machine which has the ability to read the contents of the input tape at most once in any ordering. We discuss the relations between BDD's and Π on-line Turing machines. We also define a disjoint selector circuit which consists only of data selectors, and discuss the relations between BDD's. Furthermore, the relation between BDD's and planar circuits is discussed.

This paper is organized as follows. In section 2, we define a BDD, an Π on-line Turing machine, a disjoint selector circuit and a planar circuit. In section 3, we define the complexity classes of the above models and discuss the relations between these classes. In section 4, we conclude our discussion.

2 Models of Computation

2.1 Binary Decision Diagram (BDD)

A binary decision diagram (BDD) (Figure 1) [1] which represents a Boolean function of n variables $f(x_1, \cdots, x_n)$ 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 a set of constant nodes,
$init \in N_V$ is a 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 a set of nodes to a set of levels such that

$level(init) = 1$,
$1 \leq level(n) \leq n$, $level(v) < level(edge(v,b))$ ($b \in \{0,1\}$) if $v \in N_V$,
$level(v) = n+1$ if $v \in N_C$,

$\pi : \{1, \cdots, n\} \to \{1, \cdots, n\}$ is a bijection from a set of levels of variable nodes to a set of indexes of variables.

Each node v of a BDD expresses a Boolean function f_v defined as

$f_{c_0} = 0$ (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 $(N_V, N_C, init, edge, level, \pi)$ expresses a Boolean function f_{init}.

For $1 \leq l \leq n$, we define that the value

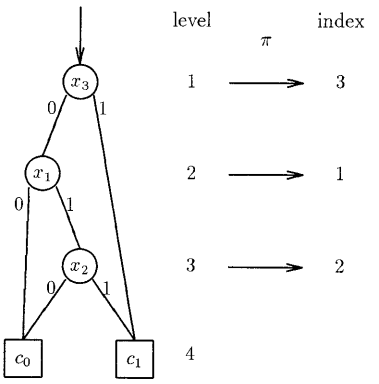

Figure 1: the BDD representing a Boolean function $x_1 \cdot x_2 + x_3$

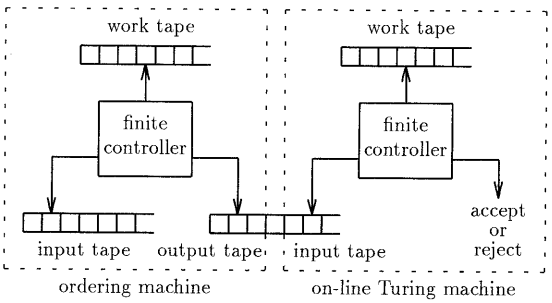

Figure 2: a Π on-line Turing machine

$$| \{ v \mid level(v) = l, v \in N_V \} | + | \{ edge(b,v) \mid level(v) < l < level(edge(b,v)), b \in \{0,1\}, v \in N_V \} |$$

is the width of a level l. Also we define that the value

$$max_{1 \leq l \leq n} \{ \text{ the width of a level } l \}$$

is a maximal width of a BDD.

In order to relate BDD's to on-line Turing machines, we define a family of BDD's and its uniformity [5].

A family $\{B_n\}$ of BDD's is a sequence $B_1, B_2, \cdots$, where B_n is a BDD representing a Boolean function of n variables. A family $\{B_n\}$ of BDD's is said to accept a language $L \subseteq \{0,1\}^*$ if and only if

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

A family $\{B_n\}$ of BDD's is $S(n)$-*uniform* if the function $n \rightarrow \overline{B_n}$ is computable with an $O(S(n))$-space bounded deterministic Turing machine, where $\overline{B_n}$ is an encoding over $\{0,1\}$ of B_n.

2.2 Π On-line Turing Machine

We define a Π on-line Turing machine (Figure 2). It consists of an input ordering machine and an on-line Turing machine [6]. An input ordering machine is a Turing machine with a two-way read-only input tape and a two-way work tape and a one-way output tape. An on-line Turing machine is a Turing machine with a one-way read-only input tape and a two-way work tape. The output tape of the input ordering machine and the input tape of the on-line Turing machine are assumed to be identical.

Let a input string of the Π on-line Turing machine be $b_1 \cdots b_n$ $(b_1, \cdots, b_n \in \{0,1\})$. The computation of the Π Turing machine is proceeds as follows. First, the input ordering machine start its computation with $b_1 \cdots b_n$ on

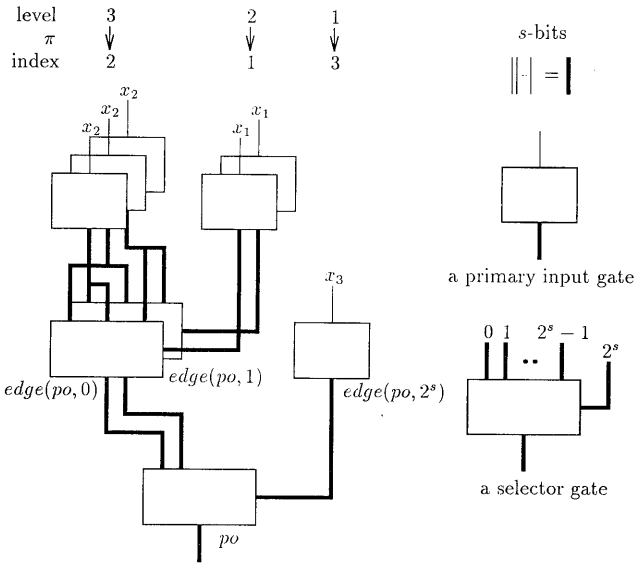

Figure 3: a s-bits disjoint selector circuit

the input tape and outputs $1^n\#b_{\pi(1)}\cdots b_{\pi(n)}$, where $\#$ is a special alphabet and $\pi : \{1,\cdots,n\} \longrightarrow \{1,\cdots,n\}$ is a bijection. Then, the on-line Turing machine start its computation with $1^n\#b_{\pi(1)}\cdots b_{\pi(n)}$ on the input tape. The Π on-line Turing machine accepts a language L if and only if the on-line Turing machine halts in an accepting state or in a rejecting state according as $b_1\cdots b_n \in L$ or $b_1\cdots b_n \notin L$.

This definition about the input tapes of an input ordering machine and an on-line Turing machine is slightly different from the ordinary one. We should define it like this because we discuss the relations between an Π on-line Turing machine , a family of BDD's, a family of disjoint selector circuits and a family of planar circuits. Note that the Turing machines defined above are deterministic one.

2.3 Disjoint Selector Circuit

We define a disjoint selector circuit (Figure 3) which represents a Boolean function of n variables $f(x_1,\cdots,x_n)$ as an 8-tuple $(s, G_{SEL}, G_{PI}, po, edge, data, level, \pi)$, where

s is a number of bit of the output of gates,
G_{SEL} is a set of selector gates,
G_{PI} is a set of primary input gates,
$po \in G_{SEL} \cup G_{PI}$ is a primary output gate,
$edge : G_{SEL} \times \{0,\cdots,2^s-1,2^s\} \rightarrow (G_{SEL} \cup G_{PI})$ represents the connections of gates,
$data : G_{PI} \times \{1,\cdots,s\} \rightarrow \{T,F,thru,not\}$ represents the output of primary input gates,
$level : G_{SEL} \cup G_{PI} \rightarrow 2^{\{1,\cdots,n\}}$ is a mapping from a set of gates to the power set of a set of levels such that

$|level(g)| = 1$ if $g \in G_{PI}$,
$level(g) = \bigcup_{0\le j\le 2^s} level(edge(g,j))$ if $g \in G_{SEL}$,
$\forall l_1 \in level(edge(g,2^s))$ and $\forall l_2 \in level(edge(g,j))$ $(\ 0 \le j \le 2^s-1\ )$, $1 \le l_1 < l_2 \le n$ if $g \in G_{SEL}$,

$\pi : \{1,\cdots,n\} \rightarrow \{1,\cdots,n\}$ is a bijection from a set of levels to a set of indexes of variables.

A selector gate has 2^s+1 s-bits inputs and one s-bits output. For $g \in G_{SEL}$ and $0 \le j \le 2^s$, $edge(g,j)$ represents a gate whose output connects to the j-th s-bits input of g. g selects one of the values of $edge(g,j)$ $(\ 0 \le j \le 2^s-1\ )$ according to the value of $edge(g,2^s)$. When the value of $edge(g,2^s)$ is the binary representation of k, the value of g is the value of $edge(g,k)$.

A primary input gate has one 1-bit input and one s-bits output. For $g \in G_{PI}$, the input of g is a variable $x_{\pi(level(g))}$ and for $1 \leq j \leq s$, j-th bit of the output is defined as follows

0(inconsistency) if $data(g,j) = F$,
1(tautology) if $data(g,j) = T$,
$x_{\pi(level(g))}$ if $data(g,j) = thru$,
$\overline{x_{\pi(level(g))}}$ if $data(g,j) = not$.

The first bit of the output of the primary output gate represents the value of the disjoint selector circuit.
For $g \in G_{SEL} \cup G_{PI}$ and for $l \in level(g)$, $\pi(l)$ is a index of variable on which the value of the output of g depends.
We define the size of a disjoint selector circuit as $|\ \{\ g \mid g \in G_{SEL} \cup G_{PI}\ \}\ |$.
A family of disjoint selector circuits and its uniformity can be defined in the same manner as those of BDD's.

2.4 Planar Circuit

A planar circuit [4] which represents a Boolean function of n variables $f(x_1, \cdots, x_n)$ is a 7-tuple $(G_{COMP}, G_{PI}, po, edge, type, level, \pi)$, where

G_{COMP} is a set of gates computing a 2-variable Boolean function,
G_{PI} is a set of primary input gates,
$po \in G_{COMP} \cup G_{PI}$ is a primary output gate,
$edge : G_{COMP} \times \{L, R\} \rightarrow G_{COMP} \cup G_{PI}$ represents the connections of gates satisfying the planar condition described below,
$type : G_{COMP} \rightarrow \{\ f_2 \mid f_2 : \{0,1\}^2 \rightarrow \{0,1\}\ \}$ represents Boolean functions of gates,
$level : G_{PI} \rightarrow \{1, \cdots, n\}$ is a mapping from a set of input gates to the set of levels,
$\pi : \{1, \cdots, n\} \rightarrow \{1, \cdots, n\}$ is a bijection from a set of levels of input gates to a set of indexes of variables.

A planar circuit is a combinational logic circuit whose underlying graph is a planar directed acyclic graph. A planar circuit can be embedded on the finite region of the plane bounded by a simple closed curve γ with no crossing edges. Let $pi_l \in G_{PI}$ be the input gate such that $level(pi_l) = l$ and po be the output gate. They assumed to be on the boundary γ and appear in the cyclic order $\langle pi_1, \cdots pi_n, po \rangle$. Note that pi_l is an input gate of variable $x_{\pi(l)}$.

The depth of a planar circuit is the maximum number of gates on the path from a input gate to the output gate.

A family of planar circuits and its uniformity can be defined in the same manner as those of BDD's.

3 On the Relations between the Models

We define the classes of languages which are discussed in this paper.

Definition 1 Let $U\text{-}BDDWIDTH(2^{S(n)})$ be the class of languages which are accepted by $S(n)$-$uniform$ families $\{B_n\}$ of BDD's whose maximal width are $2^{O(S(n))}$.

Let Π-1-$DSPACE(S(n))$ be the class of languages which are accepted by $O(S(n))$-space bounded Π on-line Turing machines.

Let $U\text{-}SELBITSIZE(S(n), 2^{S(n)})$ be the class of languages which are accepted by $S(n)$-$uniform$ families $\{CS_n\}$ of $O(S(n))$-bits disjoint selector circuits whose size are $2^{O(S(n))}$.

Let $U\text{-}PLADEPTH(S(n))$ be the class of languages which are accepted by $S(n)$-$uniform$ families $\{CP_n\}$ of $O(S(n))$-depth planar circuits. □

The main result in this paper is the relations between these classes.

Theorem 1 For $S(n) \geq \log_2 n$,
$U\text{-}PLADEPTH(S(n)) \subseteq$
$U\text{-}BDDWIDTH(2^{S(n)}) = U\text{-}SELBITSIZE(S(n), 2^{S(n)}) = \Pi\text{-1-}DSPACE(S(n))$.
[proof] It is derived from the following lemmas. □

The lemma described below is an extended result of Ishiura's [3].

Lemma 1 For $S(n) \geq \log_2 n$, Π-1-$DSPACE(S(n)) \subseteq U\text{-}BDDWIDTH(2^{S(n)})$.
[proof] Let M_1 be an $O(S(n))$-space bounded ordering machine and M_2 be an $O(S(n))$-space bounded on-line Turing machine. Let a Π on-line Turing machine consist of M_1 and M_2, and it be assumed to accept a language L.

Let the bijection which M_1 computes be π'.

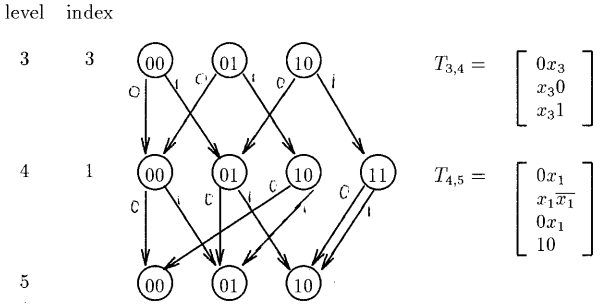

Figure 4: reachability vectors

Let a configuration of M_2 be a 4-tuple (q, l, u, j), where q is a state of the finite controller, l $(-n \leq l \leq n)$ is the position of the head of the input tape, the word u ($|u| = O(S(n))$) is the content of the work tape, and j $(1 \leq j \leq O(S(n)))$ is the position of the head of the work tape.

We design a BDD $B_n = (N_V, N_C, init, edge, level, \pi)$ accepting $L \cap \{0,1\}^n$. The bijection π of B_n is the inverse mapping π'^{-1} of π'. The nodes of B_n correspond to the configurations of M_2. The initial node of B_n is $(q_0, -n, 0^{O(S(n))}, 1)$, where q_0 is the initial state of M_2. Let a node v of B_n correspond to a configuration (q, l, u, j) of M_2. If q is a rejecting state or an accepting state of M_2, v is constant node c_0 or c_1, respectively, and $level(v) = n+1$. Otherwise, v is a variable node and $level(v) = l$ and $edge(v, b)$, $b \in \{0,1\}$ corresponds to (q', l', u', j') such that $\delta((q, l, u, j), b) = (q', l', u', j')$ where δ is a state transition function of M_2. Finally, we delete all the node v such that $edge(v, 0)$ and $edge(v, 1)$ are identical. Then, B_n is obtained.

Since M_2 is $O(S(n))$-space bounded it has at most $2^{O(S(n))}$ different configurations. Therefore the maximum width of B_n is $2^{O(S(n))}$.

Since π' is computable by the $O(S(n))$-space bounded Turing machine, the encoding of $\pi = \pi'^{-1}$ can be also generated by an $O(S(n))$-space bounded Turing machine. Since the state transition functions of M_2 are computable by the $O(S(n))$-space bounded Turing machine, the encoding of $N_V, N_C, init, edge, level$ can be generated by an $O(S(n))$-space bounded Turing machine. Therefore the family $\{B_n\}$ of BDD's is $S(n)$-uniform. □

We define a reachability of nodes of a BDD [7] for the next lemma. For nodes $v_1 \in N_V$ and $v_2 \in (N_V \cup N_C)$ of a BDD $(N_V, N_C, init, edge, level, \pi)$ of n variables, $v_1 \rightarrow_{\boldsymbol{b}} v_2$ means that v_2 is reachable from v_1 when an assignment to the variables $x_1, \cdots, x_n$ is $\boldsymbol{b} = b_1 \cdots b_n$, $(b_1, \cdots, b_n \in \{0,1\})$. The formal definition of reachability is as follows

$v_1 \rightarrow_{\boldsymbol{b}} v_2$ if $edge(b_{\pi(level(v_1))}, v_1) = v_2$,
$v_1 \rightarrow_{\boldsymbol{b}} v_2$ if $\exists v \in N_V$, $v_1 \rightarrow_{\boldsymbol{b}} v, v \rightarrow_{\boldsymbol{b}} v_2$.

Lemma 2 For $S(n) \geq \log_2 n$, $U\text{-}BDDWIDTH(2^{S(n)}) \subseteq U\text{-}SELBITSIZE(S(n), 2^{S(n)})$.

[proof] Let $W(n) = 2^{O(S(n))}$ and $w(n) = O(S(n))$. For a BDD B_n representing a Boolean function of n variable whose maximal width is $W(n)$, we design a disjoint selector circuits CS_n representing a same Boolean function.

We assume that B_n is quasi-reduced [7], that is, for $v \in N_V$ and $b \in \{0,1\}$, $level(v) + 1 = level(edge(v, b))$ are assumed to be satisfied. Note that the maximal width of B_n is unchanged even if B_n has converted to quasi-reduced. $\forall l_1, l_2$, $1 \leq l_1 < l_2 \leq n+1$, we define a reachability vector T_{l_1,l_2} (Figure 4). It is a $W(n)$ vector whose k-th $(0 \leq k \leq W(n)-1)$ element is a binary number of $w(n)$ bits. Let the nodes of level l_1 and level l_2 of B_n be $\{v_{l_1}^0, \cdots, v_{l_1}^{W(n)-1}\}$ and $\{v_{l_2}^0, \cdots, v_{l_2}^{W(n)-1}\}$, respectively. The k-th element t_{l_1,l_2}^k of T_{l_1,l_2} is defined as follows, where $encode(j)$ is a binary representation of j of $w(n)$ bits.

$$t_{l_1,l_2}^k(\boldsymbol{b}) = encode(j) \Leftrightarrow v_{l_1}^k \rightarrow_{\boldsymbol{b}} v_{l_2}^j.$$

Note that when $init = v_1^0$, $c_0 = v_{n+1}^0$ and $c_1 = v_{n+1}^1$, the 0-th element of $T_{1,n+1}$ is

$encode(0)$ if $f(\boldsymbol{b}) = 0$,
$encode(1)$ if $f(\boldsymbol{b}) = 1$.

$\forall l_1, l_2, l_3$, $1 \leq l_1 < l_2 < l_3 \leq n+1$, T_{l_1,l_3} can be computed from T_{l_1,l_2} and T_{l_2,l_3} as follows, where $decode(encode(j)) = j$ assumed to be satisfied.

$\forall k_1,\ 0 \leq k_1 \leq W(n)-1,$
$t^{k_1}_{l_1,l_3}(\boldsymbol{b}) = t^{k_2}_{l_2,l_3}(\boldsymbol{b}),\ k_2 = decode(t^{k_1}_{l_1,l_2}(\boldsymbol{b})).$

Let us denote this computation by $T_{l_1,l_3} = T_{l_1,l_2} * T_{l_2,l_3}$. $T_{1,n+1}$ can be computed as follows

$T_{1,n+1} = T_{1,2} * T_{2,3} * \cdots * T_{n,n+1}.$

We can compute the above formulas with a $w(n)$-bits disjoint selector circuit $CS_n = (w(n), G_{SEL}, G_{PI}, po, edge, data, level, \pi)$. Since a primary input gate can compute a element $t^k_{l_1,l_2}$, $W(n) \times n$ primary input gates are enough for computing $T_{1,2}, T_{2,3}, \cdots, T_{n,n+1}$. Since a selector gate can compute a formula $t^{k_1}_{l_1,l_3}(\boldsymbol{b}) = t^{k_2}_{l_2,l_3}(\boldsymbol{b}),\ k_2 = decode(t^{k_1}_{l_1,l_2}(\boldsymbol{b}))$, $W(n) \times n$ selector gates are enough for computing $T_{1,n+1} = T_{1,2} * T_{2,3} * \cdots * T_{n,n+1}$. Therefore, the size of CS_n is $2^{O(S(n))}$.

$G_{SEL}, po, edge, level$ of CS_n are independent from the form of B_n and the encoding of them can be computed with an $O(S(n))$-space bounded Turing machine. Since the family $\{B_n\}$ of the BDD's is $S(n)$-$uniform$, the encoding of $w(n), G_{PI}, data, \pi$ of CS_n can be computed with an $O(S(n))$-space bounded Turing machine. Therefore the family $\{CS_n\}$ of the disjoint selector circuits is $S(n)$-$uniform$. □

Lemma 3 For $S(n) \geq \log_2 n$, $U\text{-}SELBITSIZE(S(n), 2^{S(n)}) \subseteq \Pi\text{-}1\text{-}DSPACE(S(n))$.
[proof] Let $w(n) = O(S(n))$. We show the algorithm to compute the output of a $w(n)$-bits disjoint selector circuit $CS_n = (w(n), G_{SEL}, G_{PI}, po, edge, data, level, \pi)$ whose size is $2^{O(S(n))}$ with a $O(S((n))$-space bounded Π on-line Turing machine. Let M_1 be a $O(S(n))$-space bounded ordering machine and M_2 be a $O(S(n))$-space bounded on-line Turing machine. Let the Π on-line Turing machine consist of M_1 and M_2.

We use a simple recursive evaluation method like as [8] to compute the value of CS_n with M_1 and M_2. We start from the gate po and recursively evaluate its inputs.

We design M_1 to compute the reverse mapping π^{-1} of the bijection π of CS_n. Since CS_n is $S(n)$-uniform, $O(S(n))$-space is sufficient for computing π^{-1}.

The algorithm of M_2 is as follows, where val is the values of $w(n)$-bits and $decode$ is defined such that val is the binary representation of $decode(val)$.

```
function EVALUATE(g ∈ G_SEL ∪ G_PI);
begin
  if g ∈ G_PI then begin
    read the input tape to compute the w(n)-bits output val of g;
    EVALUATE = val;
  end else begin
    val = EVALUATE(sel(g));
    val = EVALUATE(data(g, decode(val)));
    EVALUATE = val;
  end
end.
```

Note that since CS_n is a disjoint selector circuit, M_2 can evaluate the output of CS_n reading the input string in the order $x_{\pi^{-1}(1)}, \cdots, x_{\pi^{-1}(n)}$.

Since CS_n is $S(n)$-uniform, $O(S(n))$-space is sufficient for generating the encoding of $w(n), G_{SEL}, G_{PI}, po, edge, data, level$.

M_2 has to memorize the gate number of g and the value of the $w(n)$-bits output of g. Since the number of gates of CS_n assumed to be $2^{O(S(n))}$ and the output of the gate of CS_n is $w(n)$-bits, $O(S(n))$-space is sufficient. □

The lemma described below is an extended result of Ikekawa's [4].

Lemma 4 For $S(n) \geq \log_2 n$, $U\text{-}PLADEPTH(S(n)) \subseteq \Pi\text{-}1\text{-}DSPACE(S(n))$.
[sketch of proof] We show the algorithm to compute the output of an $O(S(n))$-depth planar circuit $CP_n = (G_{COMP}, G_{PI}, po, edge, type, level, \pi)$ with an $O(S(n))$-space bounded Π on-line Turing machine. Let M_1 be an $O(S(n))$-space bounded ordering machine and M_2 be an $O(S(n))$-space bounded on-line Turing machine. Let the Π on-line Turing machine consist of M_1 and M_2.

We design M_1 to compute the reverse mapping π^{-1} of the bijection π of CP_n. Since CP_n is $S(n)$-uniform, $O(S(n))$-space is sufficient for computing π^{-1}.

A class of language accepted by a $S(n)$-$uniform$ family of planar circuits is included by a class of language accepted by a $O(S(n))$-space bounded on-line Turing machine [4]. We design M_2 as the above on-line Turing machine. □

We show the relations between the classes (Figure 5), where $poly(n)$ is a polynomial of n. NC^k, DL and NL are the class of languages which are accepted by a uniform family of combinational logic circuits of $O(\log^k n)$-depth, a $O(\log n)$-space bounded deterministic Turing machine and $O(\log n)$-space bounded nondeterministic Turing machine.

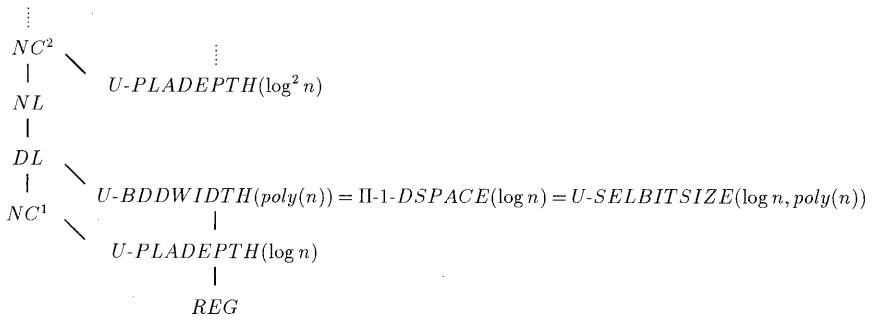

Figure 5: relations between classes

4 Conclusion

We defined a Π on-line Turing machine and a disjoint selector circuit, and we discussed the relations between BDD's, Π on-line Turing machines, disjoint selector circuits and planar circuits.

As future works, we want to find the concrete languages which can or can not be accepted by families of BDD's of $O(S(n))$-width ($\log_2 n \leq S(n)$).

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] 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.

[3] 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.

[4] M. Ikekawa. On depth-bounded planar circuits. *IEICE Trans. on Information and Systems*, E75-D(1):110–115, January 1992.

[5] W.L.Ruzzo. On uniform circuit complexity. *JCSS*, 22:365–383, 1981.

[6] H.Machida and T.Kasai. Space complexity in on-line computation. *JCSS*, 24:362–372, 1982.

[7] N. Ishiura. Synthesis of multi-level logic circuits from binary decision diagrams. In *Proc. Synthesis and Simulation Meeting and Int. Interchange (SASIMI '92)*, pages 74–83, April 1992.

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