Electronic Colloquium on Computational Complexity, Report No. 85 (2000)

Probabilistic OBDDs: on Bound of Width versus Bound of Error

Rustam Mubarakzjanov *

Abstract

Ordered binary decision diagrams (OBDDs) are well established tools to represent Boolean functions. There are a lot of results concerning different types of generalizations of OBDDs. The same time, the power of the most general form of OBDD, namely probabilistic (without bounded error) OBDDs, is not studied enough. In order to compare probabilistic OBDDs with other kinds of branching programs, we consider such OBDDs bounding their width by a constant. We show this computation model can be more powerful than polynomial size non-deterministic, probabilistic with bounded error OBDDs and non-deterministic read-once branching programs. We discuss also the possibilities to find functions being hard for probabilistic OBDDs with constant width.

Key words: Probabilistic branching programs, Boolean functions, complexity classes.

1 Introduction

In order to study the relationship between different complexity classes restricted models of computation are considered. Branching programs form one of most investigated in last years computation model (see [13] for a lot of references). In particular, read-once ordered branching programs (OBDDs) determine complexity classes relationships between which are successfully proven [3], [8], [9]. The same time OBDDs are convenient tools to represent Boolean functions because of the possibility to manipulate them efficiently [5]. Probabilistic OBDDs have the most general form of OBDDs.

*Dept. of Computer Science, University of Trier. Research partially supported by the Russia Fund for Basic Research 99-01-00163 and Fund "Russia Universities" 04.01.52. Email: **rustam@ti.uni-trier.de**

ISSN 1433-8092

We recall basic definitions. A *deterministic* branching program P for computing a Boolean function $h_n : \{0,1\}^n \rightarrow \{0,1\}$ is a directed acyclic multi-graph with a source node and two distinguished sink nodes: accepting and rejecting. The out-degree of each non-sink node is exactly 2. Each node is labeled by some variable x_i; then it is called an x_i-node. The two arcs outgoing from x_i-node are labeled by 0 and 1. The label "a" indicates that only inputs satisfying $x_i = a$ may follow this arc in the computation. The branching program P computes function h_n in the obvious way: for each $\mathbf{x} \in \{0,1\}^n$, $h_n(\mathbf{x}) = 1$ iff there is a directed path starting in the source and leading to the accepting node such that all labels along this path are consistent with $\mathbf{x} = x_1 x_2 \ldots x_n$. The branching program becomes *non-deterministic* if we allow "guessing nodes" that is nodes with two outgoing arcs being unlabeled. A non-deterministic branching program P outputs 1 on an input $\mathbf{x}$ iff there exists (at least one) computation on $\mathbf{x}$ starting in the source node and leading to the accepting node. A *probabilistic* branching program has in addition to its standard (deterministic) nodes specially designated nodes called random nodes. Each such a node corresponds to a random input y_i having values from $\{0,1\}$ with probabilities $\{1/2, 1/2\}$. We say that a probabilistic branching program *computes* (or $(1/2+\epsilon)$- *computes*) a function h if it outputs $h(\mathbf{x})$ with a probability greater than 1/2 (at least$(1/2+\epsilon)$) for any input $\mathbf{x}$ $(0 < \epsilon \leq 1/2)$. A probabilistic branching program B on n variables determines a function c_B: $\{0,1\}^n \rightarrow [0,1]$; $c_B(\mathbf{x})$ is the probability that B reaches the accepting sink on the input $\mathbf{x}$. We call this function the *characteristic function* of the branching program B.

For a branching program P we define the *complexity* of the branching program P as the number of its nodes. We denote the class of Boolean functions computable by polynomial size nondeterministic branching programs by $NP{-}BP$. The class $coNP{-}BP$ contains all Boolean functions the negations of which are computable by polynomial size non-deterministic branching programs. We say that a function belongs to the class $PP{-}BP$ iff there is a polynomial size probabilistic branching program computing this function. For a probabilistic computation with bounded error, i.e. an $(1/2+\epsilon)$-computation, $\epsilon > 0$, we use an other notation for the complexity class. Let $BPP_\epsilon{-}BP$ be the class of functions $(1/2+\epsilon)$-computable by polynomial size probabilistic branching programs. Furthermore, let $BPP - BP := \bigcup_{0<\epsilon\leq 1/2} BPP_\epsilon - BP$. For a restricted class of branching programs Q, we define analogous complexity classes using "$-Q''$ as a suffix to their notations.

A *read-once* branching program (BP1) is a branching program in which every variable is tested no more than once on each path. A BP1 is called *ordered* (or OBDD) if the variables have to be tested according to some fixed ordering π. An OBDD is called *oblivious* if it can be leveled, i.e. arcs connect only neighboring levels, and each level contains only x-nodes for some fixed variable x. *The width* of an OBDD is a maximum number of nodes belonging to a level. OBDDs having the width bounded by a constant (bwOBDD for short) are studied in this work. The restriction on the width was studied earlier for general deterministic branching programs [4]. We show in this work that probabilistic bwOBDD can be more powerful than non-deterministic BP1 and probabilistic with bounded error OBDD.

First off all we motivate that probabilistic branching programs can be interesting for a practical computation. We present a function from $PP - OBDD \setminus (BPP - OBDD \cup NP - BP1 \cup coNP - BP1)$ that could be computed with an arbitrary precision by a polynomial times repetition of the computation of a probabilistic OBDD.

In Section 3 we present basic lemmas helping to obtain different characteristic functions of some probabilistic bwOBDDs. Using these lemmas we present a probabilistic bwOBDD reading the variables in the prescribed order, $\{1, 2, \ldots, n\}$, and computing some function that is hard for nondeterministic OBDDs reading the variables in the same order. In Section 4, we present an explicit function computable by probabilistic bwOBDD being hard for non-deterministic read-once branching programs. It is known that a polynomial randomized branching program can be transformed to an equivalent polynomial one having such a form that it can be partitioned into a tree of probabilistic nodes at the top and a completely deterministic part at the bottom [16]. Our probabilistic OBDD shows that such transformation is impossible if the error of computations is not bounded. The only model of probabilistic branching program for which an exponential lower bound of computation is known is probabilistic OBDD with bounded error [1]. For probabilistic read-once branching program, exponential lower bounds are obtained only for fixed error [18], [15]. For the computations without bounded error analogous problem is more hard. We discuss in Section 5 the possibilities to find functions being hard for probabilistic OBDD with constant width and some extra restrictions.

2 Repetition of probabilistic computations

One can obtain an arbitrary precision by a repetition of a probabilistic computation when the error is bounded. The number of repetitions is a constant depending on the desired precision. This in not the case for probabilistic computations generally. But if a polynomial probabilistic branching program P has the characteristic function $c_P(\mathbf{x})$ then $|c_P(\mathbf{x}) - 1/2| \geq \epsilon_n$ for some function ϵ_n and any $\mathbf{x}, |\mathbf{x}| = n$. The following lemma holds.

Lemma 1 *Let a polynomial size probabilistic branching program P compute some function q. Then for any positive constant α and any values of input variables $\mathbf{x}$, $|\mathbf{x}| = n$ the majority result of $O(1/\epsilon_n)$-times repetition of the computation of the branching program P on $\mathbf{x}$ is equal to $q(\mathbf{x})$ with the probability at least $1 - \alpha$.*

Proof. Let P computing some function q run t times on some fixed input $\mathbf{x}$. We obtain independent Poisson trials $Y_1, \ldots, Y_t$ with $Y_i \equiv (P(\mathbf{x}) = q(\mathbf{x}))$ for $i = 1, t$. The probability $Pr(Y_i = 1)$ is at least $1/2 + \epsilon_n$. Because of the Chernoff bound [7] (see also Theorem 4.2 in [11]) the majority result of the t experiments on P is not equal to $q(\mathbf{x})$ with the probability $Pr(\sum_{i=1}^{t} Y_i < t/2) < exp(-\frac{(1/2+\epsilon_n)t\delta^2}{2})$ where $t/2 = (1-\delta)(\frac{1}{2} + \epsilon_n)t$. Then $\delta = \frac{2\epsilon_n}{1+2\epsilon_n}$. Therefore if $t \geq (\ln \frac{1}{\alpha})\frac{1+2\epsilon_n}{\epsilon_n^2}$ then $Pr(\sum_{i=1}^{t} Y_i < t/2) \leq \alpha$. ∎

There are functions computable by polynomial size probabilistic OBDDs with $1/\epsilon_n = O(poly(n))$ that are hard for randomized OBDDs and for non-deterministic OBDD as well. We present such a function q using ideas of [8]. Let X, Y, and Z be sets of n variables with pairwise empty intersection. Then the function q on $3n$ variables is exclusive OR of three functions on X, Y and Z respectively: $q(\mathbf{xyz}) = Perm(\mathbf{x}) + \neg Perm(\mathbf{y}) + f_n(\mathbf{z})$. Here $Perm : \{0,1\}^{m^2} \to \{0,1\}, n = m^2$, is the "permutation" function ([10]), i.e. $PERM(\mathbf{x}) = 1$ iff every row and every column of the matrix $\mathbf{x} = (x_{1,1}, x_{1,2}, \ldots, x_{m,m})$ contains exactly one 1. The function f_n is presented in [17]. For the sake of completeness, we recall the definition of f_n. Let p[n] be the smallest prime greater or equal to an integer n. For every integer s, $\omega_n(s)$ is defined as follows. For $j \equiv s(mod\ p[n])$, $1 \leq j \leq p[n]$, $\omega_n(s) = j$ if $j \leq n$ and $\omega_n(s) = 1$ otherwise. The Boolean function f_n is defined as $f_n(\mathbf{z}) = z_j$ for every $\mathbf{z} \in \{0,1\}^n$ where $j = \omega_n(\sum_{i=1}^{n} iz_i)$.

Theorem 1 *The function q, $q(\mathbf{xyz}) = Perm(\mathbf{x}) + \neg Perm(\mathbf{y}) + f_n(\mathbf{z})$, belongs to $PP - OBDD \setminus (BPP - OBDD \cup NP - BP1 \cup coNP - BP1)$. Moreover for any positive constant α, there is a polynomial size probabilistic OBDD such that the majority result of a polynomial times repetition of its computation is equal to q with the probability at least $1 - \alpha$.*

Proof. It is known that $Perm \in BPP - OBDD \setminus NP - BP1$ ([10], [15]) and $f_n \in NP - OBDD \setminus BPP - OBDD$ ([17], [1]). Therefore the function q belongs neither to $BPP - OBDD$, nor to $NP - BP1$, nor to $coNP - BP1$.

The functions $Perm(\mathbf{x})$ and $\neg Perm(\mathbf{y})$ are computed by probabilistic OBDDs with bounded error B_1 and B_2 ([15]). There is an OBDD non-deterministically computing $f_n(\mathbf{z})$. One can transform its non-deterministic selection to a random selection and obtain after some modifications a probabilistic OBDD B_3 computing $f_n(\mathbf{z})$ with $\epsilon_n = \frac{1}{cn}$ for some constant c. Following ideas of [8] the desired OBDD B computing q has three parts corresponding to B_1, B_2 and B_3. Informally speaking, the sinks of B_i are identified in the proper way with sources of copies of B_{i+1}, $i = 1, 2$. How it was shown in [8] B computes q with ϵ_n such that $1/\epsilon_n = O(n^2)$. Lemma 1 gives the statement of Theorem. ∎

3 Characteristic functions

It is known that the size of an OBDD depends heavily on the variable order. If one restricts the class of computations fixing ordering of reading variables higher lower bounds of an OBDD complexity can be obtained. An idea helping to obtain such lower bounds for the nondeterministic case is the following one. For any fixed ordering of reading variables, an OBDD computing some function h on n variables has to keep values of m input variables read at first. If this number m is big, say $m = n/2$, then the OBDD has the exponential complexity. For the prescribed order, typical such a function is a following one: $h(\mathbf{x}) = 1$ iff $\mathbf{x}$ is the concatenation of two equal words, i.e. $\mathbf{x} = \mathbf{yy}$. We show that this function is computable by probabilistic bwOBDD respecting the prescribed order of deterministic variables.

First of all we give some answers to the following question. What kind of functions can be the characteristic functions of some OBDDs or of some bwOBDDs? We omit details of proofs of following simple lemmas for the lack of space.

Lemma 2 *For any constant α, $0 \le \alpha \le 1$, if the binary representation of α has t positions then there exists a bwOBDD B of the width 2 with t levels consisting only of random nodes such that $c_B = \alpha$.*

Lemma 3 *Let c_{B_1} and c_{B_2} be the characteristic functions of bwOBDDs B_1 of the width w_1 and B_2 of the width w_2, respectively, reading deterministic variables of the same set X in the prescribed order. Then following functions are the characteristic functions of some bwOBDDs with the prescribed order: $1-c_{B_1}(x)$; $1/2(c_{B_1}(x)+c_{B_2}(x))$; $c_{B_1}(x)c_{B_2}(x)$. These OBDDs have the width w_1, $w_1 + w_2$, w_1w_2 respectively.*

Lemma 4 *There is a bwOBDD B_n reading deterministic variables in the prescribed order and accepting a word $x_1 x_2 ... x_n$, $x_i \in \{0,1\}$, $i = 1, \ldots, n$, with the probability which binary representation is $0, x_n x_{n-1} ... x_1$.*

The construction of B_n is based on a probabilistic automaton determined by Rabin [14] and is presented on the following picture.

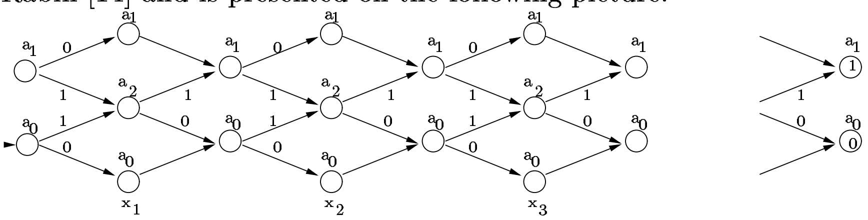

Figure 1. OBDD equivalent to Rabin's automaton. 2-nodes levels contain random nodes.
All nodes of a 3-nodes level are labeled by a variable written below.
An arc without a label denotes both arcs labeled by 0 and 1.

Theorem 2 *There exists probabilistic bwOBDD computing function the h ($h(\mathbf{x}) = 1$ iff $\mathbf{x} = \mathbf{yy}$) and reading deterministic variables in the prescribed order.*

Proof. We write $\mathbf{x}$ as $\mathbf{yz}$, $\mathbf{y} = y_1 \ldots y_n, \mathbf{z} = z_1 \ldots z_n$. Let $0, 1y_n \ldots y_1 = q$ and $0, z_n \ldots z_1 = p$. Then there exists a bwOBDD P such that $c_P(\mathbf{x}) = \frac{1}{2}qp + \frac{1}{2}q(1-q)$ (Lemmas 3 and 4). For a fixed p, if $q = 1/2p + 1/2$ (i.e. $\mathbf{y} = \mathbf{z}$) then this probability is maximal and equal to $r' = 1/8p^2 + 1/4p + 1/8$. Otherwise, it is at most $r' - \frac{1}{2^{2n+1}}$. There exists a bwOBDD P' that reaches the accepting node with the probability $1 - r'$ (Lemmas 3 and 4). Combining the bwOBDDs P and P' we obtain the bwOBDD P'' with $c_{P''}(\mathbf{x}) = 1/2(1 + c_P(\mathbf{x}) - r')$. Because of Lemma 2, there is a bwOBDD P''' with the characteristic function $\frac{1}{2} + \frac{1}{2^{2n+3}}$. The bwOBDD with a random source the arcs from which lead to the sources of P'' and P''' computes the function h. ∎

4 Probabilistic OBDD of constant width can do more than polynomial nondeterministic BP1

The author presented in [12] a function GE_n belonging to the class $Q = PP \setminus (BPP \cup NP \cup coNP)$ in the context of OBDD. The function GE_n is computable by a probabilistic branching program of the width bounded by a constant. The function $GE_n : \{0,1\}^n \to \{0,1\}$ ("greater or equal") is specified as follows. Let $n = 4l$. Say that even bit x_i, $i \in \{2, 4, \ldots, 4l\}$, has type 0 (1) if the corresponding previous odd bit x_{i-1} is 0 (1). For a sequence $\mathbf{x} \in \{0,1\}^{4l}$, denote by $\mathbf{x}^0$ ($\mathbf{x}^1$) subsequence of $\mathbf{x}$ that consists of all even bits of type 0 (1). For any word $\mathbf{x} = x_1 \dots x_m \in \{0,1\}^*$, let $b(\mathbf{x})$ be the binary number $0, x_m x_{m-1} \dots x_1$. Let $b(\mathbf{e}) = 0$ for the empty word $\mathbf{e}$. Then $GE_n(\mathbf{x}) = 1$ iff $b(\mathbf{x}^0) \geq b(\mathbf{x}^1)$.

Theorem 3 *There is a probabilistic bwOBDD $B(GE_{4l})$ computing the function GE_{4l}. There is no polynomial size nondeterministic OBDD and no polynomial size probabilistic with a bounded error OBDD computing GE_{4l}.*

We present now a new function $LePerm$. The function $LePerm : \{0,1\}^{m^2} \to \{0,1\}$ ("less or equal" than "permutation") is based on the function $Perm$ mentioned in Section 2 and is specified as follows. Let $m(m+1)/2 = M$ and $\mathbf{x} = (x_{1,1}, x_{1,2}, \dots, x_{m,m})$ then $LePerm(\mathbf{x}) = 1$ iff for any $i, 1 \leq i \leq m$, $|\{j | x_{i,j} = 1\}| = 1$ and $\sum_{i=1}^{m} j_i \leq M$ where $x_{ij_i} = 1$, $i = 1, m$. Note that if $Perm(\mathbf{x}) = 1$ then $LePerm(\mathbf{x}) = 1$.

Theorem 4 *A such probabilistic OBDD $B(LePerm)$ exists that it computes the function LePerm and has the width 5. There are no polynomial nondeterministic read-once branching program computing this function.*

Proof. Firstly we consider a bwOBDD B_1 presented on the following picture.

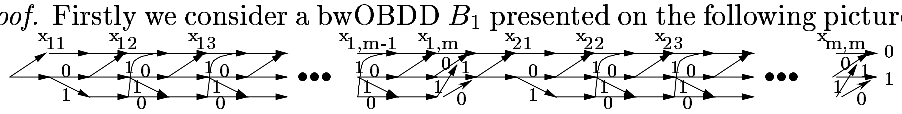

Figure 2.

This OBDD has fictitious nodes. These nodes have both outgoing arcs leading to the same node. These arcs correspond to an unlabeled arc on the picture. Each level contains only x-nodes for some variable x. We put the label of these nodes on the top of each "deterministic" level. Unlabeled levels contain random nodes. B_1 runs in the following manner. The variables are tested in the prescribed order. While going through a row of the matrix $(x_{1,1}, x_{1,2}, \dots, x_{m,m})$ the OBDD looks for a variable equal to 1. Before testing each deterministic variable, B_1 goes to the rejecting path (sink) with the probability 1/2. After finding an 1 in a row, the OBDD tests whether rest variables of the row are equal to 0. If it finds an other variable equal to 1 then the OBDD goes to the rejecting path.

The bwOBDD has the width 3 and accepts the word $(x_{1,1}, x_{1,2}, \dots, x_{m,m})$ with the probability $c_{B_1} = (1/2)^{M(\mathbf{x})}, M(\mathbf{x}) = \sum_{i=1}^{m} j_i$ where $x_{ij_i} = 1$, $i = 1, m$, only if $|\{j | x_{i,j} = 1\}| = 1$ otherwise it rejects the word.

There is a bwOBDD B_2 that reaches the accepting node with the probability $1 - 3/4(1/2)^M$ (Lemma 2).

The bwOBDD $B(LePerm)$ has a random source. The arcs from the source lead to the source of B_1 and to the source of B_2. To make $B(LePerm)$ oblivious we put in B_2 fictitious random and deterministic levels. If the equation $|\{j|x_{i,j} = 1\}| = 1$ does not hold for some i then $B(LePerm)$ reaches the accepting node with probability $1/2 - 3/4(1/2)^{M+1} < 1/2$.

If $|\{j|x_{i,j} = 1\}| = 1$, $i = 1, m$, then $B(LePerm)$ reaches the accepting node with probability $c_B = \frac{1}{2} - \frac{3}{4}\frac{1}{2^{M+1}} + \frac{1}{2^{M(\mathbf{x})+1}}$. If $M(\mathbf{x}) \leq M$ then $c_B > 1/2$. If $M(\mathbf{x}) \geq M + 1$ then $c_B < 1/2$. Therefore $B(LePerm)$ computes $LePerm$.

To find a lower bound of the complexity of a non-deterministic read-once branching program D computing $LePerm$, we use an idea of [10].

Let R be the set of such words w that $PERM(w) = 1$. For these words $LePerm(w) = 1$. Consider a part of D determined by accepting paths corresponding to the words of R. Let L be a set of nodes of D when exactly $m/2$ ones are read. For any two accepting paths in D, let w_1 and w_2 be corresponding words from R. Let a node a (b) from L correspond to w_1 (w_2). Let $m/2$ ones of w_1 (of w_2) reached at first be the values of variables $x_{i_1,j_1}, \ldots, x_{i_{m/2},j_{m/2}}$, $i_1 < i_2 < \ldots < i_{m/2}$ ($x_{i'_1,j'_1}, \ldots, x_{i'_{m/2},j'_{m/2}}$, $i'_1 < i'_2 < \ldots < i'_{m/2}$).

If $a = b$ then $i_l = i'_l$ for $l = 1, m/2$. Analogous property holds for the first indexes (i.e. i- and i'-indexes) of variables equal to 1 tested after a according to w_1 and to w_2. Because the words w_1 and w_2 correspond to permutation matrices the same properties can be written for second indexes (i.e. j- and j'-indexes) of corresponding variables. That means that for any node $a \in L$, there are at most $((m/2)!)^2$ words of R going through a. These words are obtained by fixing the order and values of i-indexes of variables equal to 1. Then values of j-indexes are fixed also but they can stay in different order. Therefore $|L| \geq |R|/((m/2)!)^2 = m!/((m/2)!)^2 \geq 2^m/(2\sqrt{m})$. ∎

5 Functions being hard for probabilistic OBDDs with some restrictions

There is no any known exponential lower bound of the complexity of a probabilistic computation. To obtain such lower bound we consider an extra restrictions for bwOBDD. First of all we can transform every bwOBDD to an other branching program having on each level the constant number of nodes. It does not disturb a generalization of our consideration but makes easier following definitions. We number nodes of every level of a bwOBDD. Each node v obtains a number $n(v)$. Nodes having the same number are called *corresponding*. For a bwOBDD, we call as *connections between two neighboring levels* L_1 and L_2 the following set M. For $v_1 \in L_1$ and $v_2 \in L_2$, if there is an arc (v_1, v_2) labeled by a then M contains $(n(v_1), n(v_2), a)$. A bwOBDD has fixed connections between levels iff there is such numbering of levels that the connections between any two neighboring levels are the same and moreover the number of levels between nearest deterministic levels are the same too. We use the notation fixOBDD for this OBDD. Although the class of fixOBDDs seems to form a very weak computation model some functions (one of them is GE_n presented in previous Section) being hard for non-probabilistic OBDDs are computable by probabilistic fixOBDDs [12].

For any Boolean function f, let $X^n(f)$ be the set of binary words $\mathbf{x}$ of length n such that $f(\mathbf{x}) = 1$.

Lemma 5 *Let F be the family of functions that are computable by probabilistic fixOBDDs of width c_1 with c_2 levels between any two nearest deterministic levels. Then there is a constant $c = c(c_1, c_2)$ such that the cardinality of $\{X_n(f) | f \in F\}$ is at most c for any integer n.*

Proof. Let P be a probabilistic fixOBDD. For any nearest deterministic levels L_1 and L_2, we can consider stochastic matrices $P(0)$ and $P(1)$: (i, j)-th element of $P(a)$ is a probability that P leads from the x'-node $v' \in L_1, n(v') = i$ to the x''-node $v'' \in L_2, n(v'') = j$ when $x' = a$. For given constant c_1 and c_2, matrices $P(0)$ and $P(1)$ have the size $c_1 \times c_1$ and each element of these matrices is equal to $k/2^{c_2}$ for some integer k, $0 \le k \le 2^{c_2}$. There are only a constant number of such matrices. Therefore there are only constant number of different fixOBDDs for any number n of variables. ∎

Let L_{n_0,n_1} be a language consisting of words that contain exactly n_0 zeros and n_1 ones. Let F be a set of functions f such that f on the words of length n is the characteristic function of some L_{n_0,n_1}, $n_0 + n_1 = n$, $n_0 > 0, n_1 > 0$.

Theorem 5 *For any constant numbers c_1 and c_2, there exists a function $f \in F$ that can not be computed by a probabilistic fixOBDD of the width c_1 with c_2 levels between any two nearest deterministic levels. Any function $f \in F$ is computable by deterministic OBDD of width n.*

Proof. For any n, $|\{L_{n_0,n_1} | n_0 + n_1 = n,\ n_0 > 0, n_1 > 0\}| = n - 1$. Thus it follows from Lemma 5 that not all functions from F are computable by probabilistic fixOBDD of width c_1 with c_2 levels between any two nearest deterministic levels. A deterministic OBDD computing a function from F calculates the number of O-s in the input string. It is possible if the OBDD has the width n. ∎

Unfortunately, we can not obtain analogous result for bwOBDDs.

Theorem 6 *Let a function f on the words of length n be the characteristic function of some language L. Let for any integer n exist numbers p_n and $\epsilon_n < 1$ having both in the binary representation $poly(n)$ positions and a probabilistic OBDD B of the width c with the following property. For a word $\mathbf{x}, |\mathbf{x}| = n$, if $\mathbf{x} \in L$ then $c_B(\mathbf{x}) = p_n$, otherwise $|c_B(\mathbf{x}) - p_n| > \epsilon_n$. Then f is computable by probabilistic OBDD of the width $c^2 + 2c + 2$.*

Proof. Let $p_n > 1/2$. Because of Lemma 2 and 3, there exists a bwOBDD B_1 of the width $c(c+2)$ with the characteristic function $c_{B_1} = c_B(\frac{1}{2}(1-c_B+p'))$, $p' = 2p_n - 1$. The function c_{B_1} has the maximum equal to $p'' = p_n^2/2$ when $c_B = p_n$.

If $p_n <= 1/2$ then there exists a bwOBDD B_1 of the width $c(c+2)$ with the characteristic function $c_{B_1} = (1 - c_B)(\frac{1}{2}(c_B + p'))$, $p' = 1 - 2p_n$. The function c_{B_1} has the maximum equal to $p'' = (1 - p_n)^2/2$ when $c_B = p_n$.

For the both cases if $c_B(\mathbf{x}) \neq p_n$, i.e. $\mathbf{x} \notin L$, then $c_{B_1} \leq p'' - \epsilon_n^2/2$. The binary representation of $p''' = 1 - p'' + \epsilon_n^2/4$ has $poly(n)$ positions and is equal to the characteristic function of some probabilistic bwOBBD of width 2. Therefore there exists a bwOBDD of the width $c(c+2)+2$ with the characteristic function $\frac{1}{2}(c_{B_1} + p''')$. This OBDD computes the function f. ∎

Corollary 1 *Any function $f \in F$ is computable by a probabilistic OBDD of the width 10.*

Indeed, there exists an OBDD of the width 2 with the characteristic function $(\frac{1}{2})^{|\{i|x_i=0\}|}$. This OBDD is even a fixOBDD.

The last statement says also that the bound on the number of levels between the nearest deterministic levels in Theorem 5 is essential. One can consider only the width of deterministic levels of OBDDs and fix connections between deterministic levels. That means informally that the probabilities leading from one deterministic node to a next deterministic node for any neighboring deterministic levels are fixed. Lemma 3 with some weak modifications holds for such OBDDs too. Lemma 2 does not give a fixOBDD but gives an OBDD with fixed connections between deterministic levels. To see it we put between fictitious deterministic nodes the only probabilistic OBDD B from Lemma 2. The deterministic levels have 3 nodes. The first node corresponds to the source of B. The second and third nodes correspond to sink-nodes of B. These nodes are connected "direct" with corresponding nodes of next deterministic level. Because there is a fixOBDD with the characteristic function from Lemma 4 the statement holds.

Corollary 2 *Any function $f \in F$ is computable by a probabilistic OBDD of a constant width having the same connections between any two neighboring deterministic levels.*

Acknowledgment
The author would like to thank Farid Ablayev and Christoph Meinel for helpful discussions on the subject of this paper.

References

[1] F. Ablayev, *Randomization and nondeterminism are incomparable for ordered read-once branching programs*, Proc. ICALP'97, LNCS 1256, Springer, 1997, pp. 195-202.

[2] F. Ablayev and M. Karpinski, *On the power of randomized ordered branching programs*, ECCC TR98-004, 1998 `http://www.ecc.uni-trier.de/eccc`

[3] F.Ablayev, M.Karpinski, and R.Mubarakzjanov, *On BPP versus NP $\cup coNP$ for Ordered Read-Once Branching Programs*, Proc. Randomized Algorithms, Brno, 1998.

[4] D.A.Barington, *Bounded-width polynomial-size branching programs recognize exactly those languages in NC^1*, Proc. of the 18st ASM STOC, 1986, 1-5.

[5] R.E. Bryant, *Graph-based algorithms for Boolean function manipulation*, IEEE Transactions on Computers 35, 1988, 677-691.

[6] A. Borodin, A. Razborov, and R. Smolensky, *On lower bounds for read-k-times branching programs*, Computational Complexity, 3, (1993), 1-18.

[7] H.Chernoff, *A measure of asymptotic efficiency for tests of a hypothesis based on the sum of observations*, Annals of Mathematical Statistics, 23:493-509, 1952.

[8] M.Karpinski and R.Mubarakzjanov, *Some separation problems on randomized OBDDs*, CSIT'99, Moscow, Russia, 1999 `http://msu.jurinfor.ru/CSIT99`

[9] M.Karpinski and R.Mubarakzjanov, *A note on Las Vegas OBDDs*, ECCC TR99-009, 1999 `http://www.ecc.uni-trier.de/eccc/`

[10] M.Krause, Ch.Meinel, and S.Waak, *Separating the eraser Turing machine classes* $L_e, NL_e, co-NL_e and P_e$, Proc. of MFCS, Springer-Verlag, 1988, pp. 405-413 (LNCS No. 324)

[11] R.Motwani and P.Raghafan, *Randomized Algorithms*, Cambridge University Press, 1995.

[12] R.Mubarakzjanov, *On probabilistic OBDDs with constant width*, accepted by CSIT'00, Ufa, Russia, 2000.

[13] A.A. Razborov, *Lower bounds for deterministic and nondeterministic branching programs*, Proc. Fundamentals in Computing Theory, Springer, Lecture Notes in Computer Science, 529, 47-60.

[14] M.O.Rabin,*Probabilistic Automata*, "Information and Control", 1963, 6, N3, p.230-245.

[15] M.Sauerhoff, *Randomness and Nondeterminism are incomparable for Read-Once Branching Programs*, ECCC TR98-018, 1998 http://www.ecc.uni-trier.de/eccc/

[16] M.Sauerhoff, *Complexity theoretical results for randomized branching programs* , PhD thesis, Univ. of Dortmund, Shaker 1999 http://ls2-www.informatik.uni-dortmund.de/ sauerhoff/

[17] P.Savicky and S.Zak, *A large bound for 1-branching programs,* Revision 01 of ECCC TR96-036, 1996 http://www.ecc.uni-trier.de/eccc/

[18] J. Thathachar, *On separating the read-k-times branching program hierarchy*, ECCC TR98-002, 1998 http://www.ecc.uni-trier.de/eccc/

ECCC ISSN 1433-8092
http://www.eccc.uni-trier.de/eccc
ftp://ftp.eccc.uni-trier.de/pub/eccc
ftpmail@ftp.eccc.uni-trier.de, subject 'help eccc'