Electronic Colloquium on Computational Complexity, Report No. 176 (2017)

Exponential Separation between Quantum and Classical Ordered Binary Decision Diagrams, Reordering Method and Hierarchies*

Kamil Khadiev
University of Latvia
Kazan Federal University
kamilhadi@gmail.com

Aliya Khadieva
University of Latvia
Kazan Federal University
aliya.khadi@gmail.com

Alexander Knop
UC San Diego
aaknop@gmail.com

November 15, 2017

Abstract

In this paper, we study quantum OBDD model, it is a restricted version of read-once quantum branching programs, with respect to "width" complexity. It is known that the maximal gap between deterministic and quantum complexities is exponential. But there are few examples of functions with such a gap. We present a method (called "reordering"), which allows us to transform a Boolean function f into a Boolean function f', such that if for f we have some gap between quantum and deterministic OBDD complexities for the natural order over the variables of f, then for any order we have almost the same gap for the function f'. Using this transformation, we construct a total function REQ such that the deterministic OBDD complexity of it is at least $2^{\Omega(n/\log n)}$, and the quantum OBDD complexity of it is at most $O(n^2)$. It is the biggest known gap for explicit functions not representable by OBDDs of a linear width. We also prove the quantum OBDD width hierarchy for complexity classes of Boolean functions. Additionally, we show that shifted equality function can also give a good gap between quantum and deterministic OBDD complexities.

Moreover, we prove the bounded error probabilistic OBDD width hierarchy for complexity classes of Boolean functions. And using "reordering" method we extend a hierarchy for k-OBDD of polynomial width, for $k = o(n/\log^3 n)$. We prove a similar hierarchy for bounded error probabilistic k-OBDDs of polynomial, superpolynomial and subexponential width.

1 Introduction

Branching programs are a well-known computation model for discrete functions. This model has been shown useful in a variety of domains, such as hardware verification, model checking, and other CAD applications [30].

One of the most important types of branching programs is oblivious read once branching programs, also known as Ordered Binary Decision Diagrams, or OBDD [30]. This model is suitable for studying of data streaming algorithms that are actively used in industry.

One of the most useful measures of complexity of OBDDs is "width". This measure is an analog of number of states in finite automaton and OBDDs can be seen as nonuniform finite automata (see for example [3]). As for many other computation models, it is possible to consider quantum OBDDs, and during the last decade they have been studied vividly [2, 4, 14, 23, 26, 27].

In 2005 Ablayev, Gainutdinova, Karpinski, Moore, and Pollett [5] have proven that for any total Boolean function f the gap between the width of the minimal quantum OBDD representing f and the width of the minimal deterministic OBDD representing f is at most exponential (however, this is not true for partial functions [1, 6, 12]). They have also shown that this bound could be reached for $\mathrm{MOD}_{p,n}$ function, that

*The extended abstract of this work was presented on International Computer Science Symposium in Russia, CSR 2017, Kazan, Russia, June 8 – 12, 2017 [18]

ISSN 1433-8092

takes the value 1 on an input iff number of 1s modulo p in this input is equal to 0; i.e. they have presented a quantum OBDD of width $O(\log p)$ for $\mathrm{MOD}_{p,n}$ (another quantum OBDD of the same width has been presented in [8]) and proven that any deterministic OBDD representing $\mathrm{MOD}_{p,n}$ has the width at least p. However, a lower bound for width of a deterministic OBDD that represents $\mathrm{MOD}_{p,n}$ is tight, and it was unknown if it is possible to construct a function with an exponential gap but an exponential lower bound for the size of a deterministic OBDD representing this function. It was shown that Boolean function PERM_n did not have a deterministic OBDD representation of width less than $2^{\sqrt{n}/2}/(\sqrt{n}/2)^{3/2}$ [21]. In 2005 Sauerhoff and Sieling [28] presented a quantum OBDD of width $O(n^2 \log n)$ representing PERM_n and three years later Ablayev, Khasianov, and Vasiliev [7] improved this lower bound and presented a quantum OBDD for this function of width $O(n \log n)$. But as in the previous case, this separation does not give us a truly exponential lower bound for deterministic OBDDs.

Nevertheless, if we fix an order of variables in the OBDD, it is possible to prove the desired statement. For example, it is known that *equality function*, or EQ, does not have an OBDD representation of the size less than 2^n for some order and it has a quantum OBDD of width $O(n)$ for any order [7]. Unfortunately, for some orders, the equality function has a small deterministic OBDDs.

Proving lower bounds for different orders is one of the main difficulties of proving lower bounds on width of OBDDs. In the paper, we present a new technique that allows us to prove such lower bounds. Using the technique, we construct a Boolean Function g from a Boolean function f such that if any deterministic OBDD representing f with the natural order over the variables has width at least $d(n)$, then any deterministic OBDD representing g has width at least $d(O(n/\log n))$ **for any order over the variables** and if there is a quantum OBDD of width $w(n)$ for f, then there is a quantum OBDD of width $O(w(\frac{n}{\log n}) \cdot \frac{n}{\log n})$ for the function g. It means that if we have a function with some gap between quantum OBDD complexity and deterministic OBDD complexity for some order, then we can transform this function into a function with almost the same gap but for all the orders. We call this transformation "reordering". The idea which is used in the construction of the transformation is similar to the idea of a transformation from [16, 20].

We prove five groups of results using the transformation. At first, we consider the result of the transformation applied to the equality function (we call the new function *reordered equality* or REQ_q). We prove that REQ_q does not have a deterministic OBDD representation of width less than $2^{\Omega(\frac{n}{\log n})}$ and there is a bounded error quantum OBDD of width $O(\frac{n^2}{\log^2 n})$. As a result, we get a more significant gap between width of quantum OBDDs and width of deterministic OBDDs than this gap for the PERM_n function, we prove such a gap for all the orders in contrast with a gap for EQ_n, and we prove a better lower bound for deterministic OBDDs than the lower bound for the $\mathrm{MOD}_{p,n}$ function.

Additionally, we considered *shifted equality function* (SEQ_n). We prove that SEQ_n does not have a deterministic OBDD representation of the width less than $2^{\Omega(n)}$ and there is a bounded error quantum OBDD with width $O(n^2)$. Note that the lower bound for the width of the minimal OBDD representing SEQ_n is better than for REQ_q but the upper bound for the width of the minimal QOBDD representation is much better.

Using properties of $\mathrm{MOD}_{p,n}$, REQ_q, and *mixed weighted sum function* (MWS) introduced by [25], we prove the width hierarchy for classes of Boolean functions computed by bounded error quantum OBDDs. We prove three hierarchy theorems:

1. the first of them and the tightest works for width up to $\log n$;
2. the second of them is slightly worse than the previous one, but it works for width up to n;
3. and finally the third one with the widest gap works for width up to $2^{O(n)}$.

Similar hierarchy theorems are already known for deterministic OBDDs [6], nondeterministic OBDDs [1], and k-OBDDs [2, 16, 17]. Additionally, we present similar hierarchy theorems for bounded error probabilistic OBDDs in the paper.

The fourth group of results is an extension of hierarchies by number of tests for deterministic and bounded error probabilistic k-OBDDs of polynomial size. There are two known results of this type:

- The first is a hierarchy theorem for k-OBDDs that was proven by Bollig, Sauerhoff, Sieling, and Wegener [11]. They have shown that $\mathbf{P}$-$(k-1)$-OBDD $\subsetneq$ $\mathbf{P}$-k-OBDD for $k = o(\sqrt{n}\log^{3/2} n)$;
- The second one was proven in [19] it states that $\mathbf{P}$-k-OBDD $\subsetneq$ $\mathbf{P}$-$(k \cdot r)$-OBDD for $k = o(n/\log^2 n)$ and $r = \omega(\log n)$.

We partially improve both of these results, proving that $\mathbf{P}$-k-OBDD $\subsetneq$ $\mathbf{P}$-$2k$-OBDD for $k = o(n/\log^3 n)$. Our result improves the first one because it holds for bigger k, and the second one, because of a smaller gap between classes. The proof of our hierarchy theorem is based on properties of the Boolean function called *reordered pointer jumping*, which is "reordering" of *pointer jumping* function defined in [11, 24].

Additionally, we partially improve a similar result of Hromkovich and Sauerhoff [13] for a more general model, for probabilistic oblivious k-BP. They have proven such a hierarchy for $k \leq \log \frac{n}{3}$. We show similar hierarchy for polynomial size bounded error probabilistic k-OBDDs with error at most 1/3 for $k = o(n^{1/3}/\log n)$.

Structure of the paper

Section 2 contains descriptions of models, classes, and other necessary definitions. Discussion about the reordering method and applications for quantum OBDDs is located in Section 3. Section 4 contains an analysis of properties of a function that guarantee existence of a small commutative OBDD representation of this function. In Section 5 we explore the gap between quantum and deterministic OBDD complexities. The width hierarchies for quantum and probabilistic OBDDs are proved in Section 6. Finally, Section 7 contains applications of the reordering method and hierarchy results for deterministic and probabilistic k-OBDDs.

2 Preliminaries

Ordered binary decision diagrams, or OBDDs, is a well-known way to represent Boolean functions. This model is a restricted version of Branching Program [30]. A branching program over a set $X = \{x_1, \ldots, x_n\}$ of n Boolean variables is a directed acyclic graph P with one source node s. Each inner node v of P is labeled by a variable $x_i \in X$, each edge of P is labeled by a Boolean value, for each node v labeled by a variable x_i, v has outgoing edges labeled by 0 or 1, and each sink of this graph is labeled by a Boolean value. A branching program P called *deterministic* iff for each inner node there are exactly two outgoing edges labeled by 0 and 1, respectively.

We say that a branching program P accepts $\sigma \in \{0,1\}^n$ iff there a exists a path, called *accepting* path, from the source to a sink labeled by 1, such that in the all nodes labeled by a variable x_i this path goes along an edge labeled by $\sigma(i)$. A branching program P represents a Boolean function $f : \{0,1\}^n \to \{0,1\}$ if for each $\sigma \in \{0,1\}^n$ $f(\sigma) = 1$ holds iff P accepts σ. The *size* of a branching program P is a number of nodes in the graph.

A branching program is *leveled* if the nodes can be partitioned into levels $V_1, \ldots, V_\ell$, and $V_{\ell+1}$ such that all the sinks belong to $V_{\ell+1}$, $V_1 = \{s\}$, and nodes in each level V_j with $j \leq \ell$ have outgoing edges only to nodes in the next level V_{j+1}. The *width* $w(P)$ of a leveled branching program P is the maximum of the number of nodes in levels of P, i.e. $w(P) = \max_{1 \leq j \leq \ell+1} |V_j|$. A leveled branching program is called *oblivious* if all the inner nodes of each level are labeled by the same variable.

A branching program is called a *read*-k branching program if each variable is tested on each path only k times. A deterministic oblivious leveled read once branching program is also called the ordered binary decision diagram. Note that OBDD reads variables on all the paths in the same order π. For a fixed order π we call an OBDD that reads in this order a π-OBDD. Let us also denote the natural order over the variables $\{x_1, \ldots, x_n\}$ as $\mathrm{id} = (1, \ldots, n)$. A Branching program is called k-OBDD if it is a read-k oblivious branching program that consists of k layers, such that each layer is a π-OBDD, possibly with many sources, for some order π.

Let $\mathrm{tr}_P : \{1, \ldots, n\} \times \{1, \ldots, w(P)\} \times \{0,1\} \to \{1, \ldots, w(P)\}$ be a transition function of an OBDD P. An OBDD is called *commutative* iff for any order π' we can construct an OBDD P' by only reordering of the transition function and P' still computes the same function. More formally, we call a π-OBDD commutative iff for any order π' a π'-OBDD P', defined by a transition function $\mathrm{tr}_{P'}(i,s,b) = tr_P(\pi^{-1}(\pi'(i)), s, b)$, represents the same function as P. Additionally, we call a k-OBDD commutative if each layer is a commutative OBDD.

Nondeterministic OBDD or NOBDD is a nondeterministic oblivious leveled read once branching program. Now let us define probabilistic OBDD or POBDD. POBDD over a set $X = \{x_1, \ldots, x_n\}$ is a nondetermenistic OBDD with a special mode of acceptance. We say that POBDD is a bounded error representation of a Boolean function $f : \{0,1\}^n \to \{0,1\}$ iff for every $\sigma \in \{0,1\}^n$ the following recursion procedure returns $f(\sigma)$ with probability at least $\frac{2}{3}$:

1. Initially it starts from the source of the POBDD;
2. If the current node is a sink, then it returns the value of its label;
3. Let the current node be labeled by x_i if there are no outgoing edges with label $\sigma(i)$ from the current node, then it returns 0;
4. Otherwise, it chooses randomly an edge labeled by $\sigma(i)$ from the current node to a node u, consider u as the current node, and it goes to the step 2.

Let us define quantum OBDDs or QOBDDs [4, 5]. For a given $n > 0$, a QOBDD P of a width w, is a 4-tuple $P = (T, q_0, \mathrm{Accept}, \pi)$, where

- $T = \left\{(G_i^0, G_i^1)\right\}_{i=1}^n$ is a sequence of pairs of (left) unitary matrices representing the transitions applying on the i-th step, where choice of G_i^0 or G_i^1 is determined by the corresponding input bit;
- $\mathrm{Accept} \subseteq \{1, \ldots, w\}$ is a set of accepting states;
- π is a permutation of $\{1, \ldots, n\}$ defining the order over the input variables.

For any given input $\sigma \in \{0,1\}^n$, the computation of P on σ can be traced by a vector from w-dimensional Hilbert space over the field of complex numbers. The initial one is $|\psi\rangle_0 = |q_0\rangle$. On each step j, we test the input bit $x_{\pi(j)}$ and then the corresponding unitary operator is applied: $|\psi\rangle_j = G_j^{x_{\pi(j)}}(|\psi\rangle_{j-1})$, where $|\psi\rangle_{j-1}$ and $|\psi\rangle_j$ represent the state of the system after the $(j-1)$-th and j-th steps, respectively. At the end of the computation, the program P measures qubits. The accepting probability of P on an input σ is $\sum_{i \in \mathrm{Accept}} v_i^2$, where $(v_1, \ldots, v_w) = |\psi\rangle_n$. We say that a function $f : \{0,1\}^n \to \{0,1\}$ has a bounded error QOBDD representation iff for any $\sigma \in \{0,1\}^n$ holds

- if $f(\sigma) = 1$, then the accepting probability of P is at least $\frac{2}{3}$ and
- if $f(\sigma) = 0$, then the accepting probability of P is at most $\frac{1}{3}$.

Similarly to commutative deterministic OBDDs we may define commutative QOBDDs. QOBDD P is called *commutative* iff for any permutation π' we can construct equivalent QOBDD P' by only reordering matrices G. Formally, it means that for any order π', $P' = (T', q_0, \mathrm{Accept}, \pi)$ is a bounded error representation of the same function as P where $T' = \left\{G^0_{\pi^{-1}(\pi'(i))}, G^1_{\pi^{-1}(\pi'(i))}\right\}$. We call a k-QOBDD commutative if each layer of this diagram is a commutative QOBDD.

3 Reordering Method

As it was mentioned before, one of the biggest issues in proving lower bounds on the OBDD complexity of a function is proving these lower bounds for different orders. In this section we suggest a method, called "reordering", which allows us to construct a transformation of a Boolean function $f : \{0,1\}^q \to \{0,1\}$ into a partial function $\mathrm{reordering}_f : \{0,1\}^n \to \{0,1\}$, such that

- $n = q\lceil \log q \rceil$ [1];
- If any π-OBDD representation of f has width at least $d(q)$, then any OBDD representation of $\mathrm{reordering}_f$ has width at least $d(q)$;
- If there is a bounded error commutative QOBDD representation of f of width $e(q)$, then there is a bounded error QOBDD representation of $\mathrm{reordering}_f$ of width $e(q)\cdot q$.

For this transformation we construct a function $\mathrm{reordering}_f$, such that for any permutation $\pi \in S_q$ there is a substitution ρ such that $f(x_{\pi(1)},\dots,x_{\pi(q)}) = \mathrm{reordering}_f|_\rho(x_1,\dots,x_q)$. In order to do it we consider

$$\mathrm{reordering}_f(z_{1,1},\dots,z_{1,l},\dots,z_{q,1},\dots,z_{q,l},y_1,\dots,y_q) = f(y_{\mathrm{bin}(z_{1,1},\dots,z_{1,l})+1},\dots,y_{\mathrm{bin}(z_{q,1},\dots,z_{q,l})+1}),$$

where $l = \lceil \log q \rceil$ and $\mathrm{bin}\,(a_1,\dots,a_l)$ is a natural number with binary representation $a_1\dots a_l$. The function $\mathrm{reordering}_f$ is defined on an input $(z_{1,1},\dots,z_{1,l},\dots,z_{q,1},\dots,z_{q,l},y_1,\dots,y_q)$ iff $\{\mathrm{bin}\,(z_{1,1},\dots,z_{1,l})+1,\dots,\mathrm{bin}\,(z_{q,1},\dots,z_{q,l})+1\} = \{1,\dots,q\}$.

Similarly, we define

$$\text{xor-reordering}_f(z_{1,1},\dots,z_{1,l},\dots,z_{q,1},\dots,z_{q,l},y_1,\dots,y_q) = f\left(y_{\mathrm{bin}\left(\bigoplus_{i=1}^{1} z_{i,1},\dots,\bigoplus_{i=1}^{1} z_{i,l}\right)+1},\dots,y_{\mathrm{bin}\left(\bigoplus_{i=1}^{q} z_{q,1},\dots,\bigoplus_{i=1}^{q} z_{q,l}\right)+1}\right).$$

Theorem 3.1. *Let k be an integer, θ be a permutation of $\{1,\dots,q\}$. If $f:\{0,1\}^q \to \{0,1\}$ is a Boolean function such that any θ-k-OBDD representation of $f(x_1,\dots,x_q)$ has width at least d, then any k-OBDD representation of* $\mathrm{reordering}_f(z_{1,1},\dots,z_{q,l},y_1,\dots,y_q)$ *(*$\text{xor-reordering}_f(z_{1,1},\dots,z_{q,l},y_1,\dots,y_q)$*) has width at least d.*

Proof. Proofs for $\mathrm{reordering}_f$ and xor-reordering_f are almost the same. Here we present only the proof for $\mathrm{reordering}_f$.

Let us assume that there is a π-k-OBDD representation P of $\mathrm{reordering}_f$ of width $d' < d$. Let ρ be a substitution to the variables $z_{1,1},\dots,z_{1,l},\dots,z_{q,1},\dots,z_{q,l}$, such that the variables $y_{\pi(1)},\dots,y_{\pi(q)}$ has addresses $\theta(1), \dots, \theta(q)$, respectively. Formally it means, that $\mathrm{bin}\left(\rho(z_{\pi(i),1}),\dots,\rho(z_{\pi(i),l})\right) = \theta(i)$.

It is easy to see that if we consider P' equal to $P|_\rho$ with all the variables $y_{\pi(i)}$ replaced by $x_{\theta(i)}$. P' is a θ-k-OBDD of width at most $d' < d$. This is a contradiction with the fact that any θ-k-OBDD that represents $f(x_1,\dots,x_q)$ has width at least d. $\square$

Theorem 3.2. *Let $f:\{0,1\}^q \to \{0,1\}$ be a Boolean function and k be a positive integer. If there is a commutative k-OBDD (bounded error commutative k-POBDD or commutative k-NOBDD) representation of f of width d, then there are k-OBDD (bounded error k-POBDD or k-NOBDD) representations of* xor-reordering_f *and* $\mathrm{reordering}_f$ *of width $d\cdot q$.*

Proof. Let P be a commutative deterministic k-OBDD of width d representing a Boolean function f. We construct a deterministic k-OBDDs P_1 and P_2 of width $q\cdot d$ representing $\mathrm{reordering}_f$ and xor-reordering_f, respectively. P_1 and P_2 read variables in the following order: $z_{1,1},\dots,z_{1,l},y_1,\dots,z_{q,1},\dots,z_{q,l},y_q$; both of them have $q\cdot d$ nodes on each level, each of them corresponds to a pair (i,j), where $i \in \{0,1\}^l$ and $s \in \{1,\dots,d\}$, and both of them have q stages. Let us describe computation on the stage i.

reordering: At the beginning of the stage P_1 is in the state (λ, s) for some s. While reading $z_{i,1}, \dots, z_{i,l}$ the decision diagram stores a read part in the first component of the state and after we have read all these bits, we reached the node (a,s), and if the transition function of P is such that $s' = tr_P(\pi^{-1}(\mathrm{bin}\,(a)+1), s, y_i)$, then we go to the node $(1,s')$.

[1] We use log to denote logarithms base 2.

In the case when all $\mathrm{bin}\,(\sigma(z_{i,1}),\ldots,\sigma(z_{i,l}))$ are different numbers from $\{1,\ldots,q\}$ the diagram P_1 just emulates the work of P_π which is constructed from P by permutation of the transition function of P with respect to the order

$$\pi = (\mathrm{bin}\,(\sigma(z_{1,1}),\ldots,\sigma(z_{1,l}))+1,\ \ldots,\ \mathrm{bin}\,(\sigma(z_{q,1}),\ldots,\sigma(z_{q,l}))+1)\,.$$

By the definition of the commutative k-OBDD the diagram P_π computes the same function as P. Therefore, P_1 returns the same result. And by the definition of the functions $\mathrm{reordering}_f$, P_1 computes $\mathrm{reordering}_f$.

xor-reordering: At the beginning of the stage P_2 is in the state (b,s) for some s. While reading $z_{i,1}$, …, $z_{i,l}$ the decision diagram stores xor of a read part and b in the first component of the state and after we have read all these bits, we reached the node (a,s), and if the transition function of P is such that $s' = tr_P(\pi^{-1}(\mathrm{bin}\,(a)+1), s, y_i)$, then we go to the node (a,s').

In the case when all $\mathrm{bin}\left(\bigoplus_{i=1}^{1}\sigma(z_{i,1}),\ldots,\bigoplus_{i=1}^{1}\sigma(z_{i,l})\right)+1$ are different numbers from $\{1,\ldots,q\}$ the diagram P_2 just emulates the work of P_π which is constructed from P by permutation of the transition function of P with respect to the order

$$\pi = \left(\mathrm{bin}\left(\bigoplus_{i=1}^{1}\sigma(z_{i,1}),\ldots,\bigoplus_{i=1}^{1}\sigma(z_{i,l})\right)+1,\ \ldots,\ \mathrm{bin}\left(\bigoplus_{i=1}^{q}\sigma(z_{i,1}),\ldots,\bigoplus_{i=1}^{q}\sigma(z_{i,l})\right)+1\right).$$

By the definition of commutative k-OBDD the diagram P_π computes the same function as P. Therefore, P_2 returns the same result. And by the definition of the functions xor-reordering_f, P_1 computes xor-reordering_f.

All other cases have the same proofs. $\square$

Theorem 3.3. *If there is a bounded error commutative* QOBDD *representation of a Boolean function* $f:\{0,1\}^q\to\{0,1\}$ *of width* w*, then there is a bounded error* QOBDD *representation of a partial Boolean function* xor-reordering_f *of width* $w\cdot q$.

Proof. Note that if there is a bounded error commutative QOBDD representation of f of width w, then there is a bounded error π-QOBDD representation P of f of the same width. For the description of a computation in P we use a quantum register $|\psi\rangle = |\psi_1\psi_2\ldots\psi_t\rangle$ where $t=\lceil\log w\rceil$.

Let us consider xor-reordering_f. We construct a bounded error QOBDD representation P' of xor-reordering_f with the following order: $z_{1,1}$, …, $z_{1,l}$, y_1, …, $z_{q,1}$, …, $z_{q,l}$, y_q. This program uses a quantum register of $\lceil\log w\rceil+\lceil\log q\rceil$ qubits, i.e. having $w\cdot q$ states. Let us denote this register as $|\phi\rangle = |\phi_1\phi_2\ldots\phi_l\psi_1\psi_2\ldots\psi_t\rangle$.

The part of the register $|\psi\rangle$ consisting of $|\psi_1\psi_2\ldots\psi_t\rangle$ qubits (we call it as a *computing* part) is modified when P' reads a value bit. Additional qubits $|\phi_1\phi_2\ldots\phi_p\rangle$ (we call this part an *address* part) is used to determine address of the value bit.

Program P' consists of q stages, i-th stage corresponds to its own block $z_{i,1},\ldots,z_{i,l},y_i$. Informally, when P' processes the block, it stores address in the address part by applying the parity function to address of the current block. After that, the program applies the modification on the computation part, with respect to the value bit.

Let us describe i-th stage, for $i\in\{1,\ldots,q\}$. In the first $\lceil\log q\rceil$ levels of the stage the program computes address $\mathrm{bin}\,(z_{i,1},\ldots,z_{i,l})$, it reads bits one by one, and for a bit $z_{i,j}$ it applies a unitary operator $U_j^{z_{i,j}}$ on the address part of the register $|\phi\rangle$, where $U_j^{z_{i,j}} = I\otimes I\otimes\ldots\otimes I\otimes A^{y_j^i}\otimes I\ldots\otimes I$, $A^0=I$, $A^1=\mathrm{NOT}$, I and NOT are 2×2 matrices such that I is a diagonal 1-matrix and NOT is an anti-diagonal 1-matrix. And we do not modify the computation part.

Note that after all these operations the address part of the register is equal to $z_{i,1}, \ldots, z_{i,l}$. On the last level we read y^i and transform the register $|\phi\rangle$ by an unitary $(w \cdot q \times w \cdot q)$-matrix D^{y_i} defined in the following way:

$$D^0 = \begin{pmatrix} G_1^0 & 0 & \cdots & 0 \\ 0 & G_2^0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & G_q^0 \end{pmatrix} \text{ and } D^1 = \begin{pmatrix} G_1^1 & 0 & \cdots & 0 \\ 0 & G_2^1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & G_q^1 \end{pmatrix},$$

where $\left\{(G_i^0, G_i^1)\right\}_{i=1}^q$ are unitary matrices transforming a quantum system in P.

It is easy to see, that width of P' equals $w \cdot q$. Let us prove that P' represents xor-reordering$_f$ with bounded error. Let us consider an input $\sigma \in \{0,1\}^n$ and let

$$\pi = \left(\text{bin}\left(\bigoplus_{i=1}^{1} \sigma(z_{i,1}), \ldots, \bigoplus_{i=1}^{1} \sigma(z_{i,l})\right) + 1, \ \ldots, \ \text{bin}\left(\bigoplus_{i=1}^{q} \sigma(z_{i,1}), \ldots, \bigoplus_{i=1}^{q} \sigma(z_{i,l})\right) + 1\right)$$

be an order over the value variables induced by σ. Since P is a commutative bounded error QOBDD representation of f, we can reorder unitary operators $\left\{(G_i^0, G_i^1)\right\}_{i=1}^q$ according to the order π and get a bounded error π-QOBDD P_π representation of f as well. It is easy to see that P' emulates exactly the computation of P_π. Therefore P' on σ gives us the same result as P_π on corresponding value bits. Hence, by the definition of xor-reordering$_f$ we prove that P' represents xor-reordering$_f$ with bounded error. $\square$

Corollary 3.1. *For any positive k, if there is a commutative bounded error k-QOBDD of width e representing a Boolean function $f : \{0,1\}^q \to \{0,1\}$, then there is a bounded error k-QOBDD of width $e \cdot q$ representing a partial Boolean function* xor-reordering$_f$.

The proof of this corollary is exactly the same as the proof of Theorem 3.3.

Corollary 3.2. *Let $f : \{0,1\}^q \to \{0,1\}$ be a Boolean function, k be a positive integer, and π be an order over $x_1, \ldots, x_q$ such that*

- *any π-k-OBDD representation of f has width at least d and*
- *there is a commutative k-OBDD (k-NOBDD) representation of f of width e.*

Then there is a total Boolean function $g : \{0,1\}^n \to \{0,1\}$ ($n = q(\lceil \log q \rceil + 1)$), such that

- *g is an extension of the partial function* reordering$_f$,
- *there is a k-OBDD (k-NOBDD) representation of g of the width $e \cdot q$, and*
- *any k-OBDD representation of g has width at least d.*

Proof. By Theorems 3.3 and 3.1 any k-OBDD representation of reordering$_f$ has width at least d and there is a k-OBDD representation P of reordering$_f$ of width $e \cdot q$. Let g be a total Boolean function such that $g(\sigma) = \text{reordering}_f(\sigma)$ if reordering$_f$ is defined on σ, otherwise let us define $g(\sigma)$ as $P(\sigma)$.

Let us note that any k-OBDD representation of g also represents reordering$_f$; as a result, has width at least d. Additionally, let us note that P represents f. $\square$

4 Commutative OBDDs

In this section we discuss a criterion of existence of a small commutative OBDD (bounded error QOBDD). We say that a function $f : \{0,1\}^n \to \{0,1\}$ has a $S_{w,q,\odot}$ representation if there is a sequence of integers $\{C_i\}_{i=1}^n$, such that

$$f(x_1, \ldots, x_n) = q\left(\bigodot_{i=1}^{n} C_i x_i \bmod w\right),$$

where $\odot$ is some commutative operation over the set $\{0,\dots,w-1\}$ and $q:\{0,\dots,w-1\}\to\{0,1\}$.

Let us show that if a function f has a $S_{w,q,\odot}(X)$ representation, then there is a commutative OBDD of width w representing f.

Theorem 4.1. *Let f be a Boolean function, such that f has a $S_{w,q,\odot}$ representation for some w, q, and $\odot$. Then there is a commutative* OBDD *of width w representing f.*

Proof. Let us construct such an OBDD with an order $x_1, \dots, x_n$. We create a vertex on level j for each possible value of $\bigodot_{i=1}^{j-1} C_i x_i \bmod w$. Then for each node corresponding to $z\in\{0,\dots,w-1\}$ from j-th layer there are 1-edge leads to $z\odot C_j \bmod w$ and 0-edge leads to $z\odot 0$. We use q as a function that marks accepting nodes on the last layer.

By the definition of $S_{w,q,\odot}$ and the OBDD this OBDD represents $f(x_1,\dots,x_n)$, and, due to the commutativity of $\odot$, this OBDD is commutative. □

Note that any characteristic polynomial, discussed in [9], has a $S_{w,q,\odot}$ representation for appropriate w, $\odot$, and q.

Let us present the definition of these polynomials. We call a polynomial $G(x_1,\dots,x_n)$ over the ring $\mathbb{Z}_w$ a characteristic polynomial of a Boolean function $f(x_1,\dots,x_n)$ if for all $\sigma\in\{0,1\}^n$, $G(\sigma)=0$ holds iff $f(\sigma)=1$.

Ablayev and Vasilev [9] proved, using the fingerprint technique, the following result.

Lemma 4.1 ([9]). *If a Boolean function f has a linear characteristic polynomial over $\mathbb{Z}_w$, then the function can be represented by a bounded error quantum* OBDD *of width $O(\log w)$.*

It is easy to see that by a linear characteristic polynomial we can construct $S_{w,q,+}(X)$ representation, where q converts 0 to 1 and other values to 0. Let us denote such a function q as q_0.

Note that in the contrast with Theorem 4.1, the quantum fingerprint technique gives us a commutative QOBDD of a logarithmic width. Unifying these techniques we can prove the following theorem.

Theorem 4.2. *If a Boolean function f has a $S_{w,q_0,+}$ representation for some w, then there is a commutative bounded error* QOBDD *representation of f of width $O(\log w)$.*

Proof. Let a Boolean function f has a $S_{w,q_0,+}$ representation for some w. It means that it has a linear characteristic polynomial over $\mathbb{Z}_w$. Then by Lemma 4.1 one may construct a bounded error quantum OBDD of width $O(\log w)$ representing f. □

5 Exponential Gap between Quantum and Classical OBDDs

As we discussed in the introduction, it is known that the maximal gap between quantum and deterministic OBDD complexities of Boolean functions is exponential.

Lemma 5.1 ([5]). *If there is a bounded error* QOBDD *representation of a Boolean function f of width w, then there is an* OBDD *representation of f of the width 2^w.*

But all the examples that achieve an exponential gap have sublinear width of a bounded error quantum OBDD representation. Known examples with a bigger width do not achieve this gap. We present results for two functions, based on equality function, that achieve almost exponential gap.

5.1 Application of Reordering Method

Let us apply the reordering method to *equality function* ($\mathrm{EQ}_n : \{0,1\}^{2n} \to \{0,1\}$) where $\mathrm{EQ}_n(x_1,\dots,x_n,y_1,\dots,y_n)=1$, iff $x_1=y_1, \dots, x_n=y_n$.

In the paper [7] was proven that there is a commutative QOBDD of width $O(n)$ representing EQ_n with bounded error. Hence, a partial function xor-reordering$_{\mathrm{EQ}_q}$ is representable with bounded error by a QOBDD of width $O(q^2)$, due to Theorem 3.3.

It is well-known that any id-OBDD representation of EQ_n has width at least 2^n. As a result, by Theorem 3.1 any OBDD representation of xor-reordering$_{\mathrm{EQ}_q}$ has width at least 2^q.

Theorem 5.1. *There is a bounded error quantum* OBDD *representation of a partial Boolean function* xor-reordering$_{\mathrm{EQ}_q}$: $\{0,1\}^n \to \{0,1\}$ *of width* $O\left(\frac{n^2}{\log^2 n}\right)$*; any deterministic* OBDD *representation of* xor-reordering$_{\mathrm{EQ}_q}$ *has width at least* $2^{\Omega\left(\frac{n}{\log n}\right)}$.

Let us define *reordered equality function* ($\mathrm{REQ}_q : \{0,1\}^n \to \{0,1\}$ where $n = 2q(\lceil \log 2q \rceil + 1)$). This is a total version of xor-reordering$_{\mathrm{EQ}_q}$. Let us consider

$$u(z_{1,1},\ldots,z_{2q,l},y_1,\ldots,y_{2q}) = \sum_{i:\mathrm{bin}(z_{i,1},\ldots,z_{i,l})\le q} 2^{\mathrm{bin}(z_{i,1},\ldots,z_{i,l})} y_i \bmod 2^q$$

and

$$v(z_{1,1},\ldots,z_{2q,l},y_1,\ldots,y_{2q}) = \sum_{i:\mathrm{bin}(z_{i,1},\ldots,z_{i,l})> q} 2^{\mathrm{bin}(z_{i,1},\ldots,z_{i,l})-q} y_i \bmod 2^q.$$

We define $\mathrm{REQ}_q(z_{1,1},\ldots,z_{2q,l},y_1,\ldots,y_{2q}) = 1$ iff $u(z_{1,1},\ldots,z_{2q,l},y_1,\ldots,y_{2q}) = v(z_{1,1},\ldots,z_{2q,l},y_1,\ldots,y_{2q})$. Note that it is possible to prove the following lemma.

Lemma 5.2. *Any* OBDD *representation of* REQ_q *has width at least* $2^{\frac{n}{2\lceil \log n+1\rceil}}$.

Proof. Note that REQ_q is an extension of xor-reordering$_{\mathrm{EQ}_q}$. Thus any OBDD representation of REQ_q also represents xor-reordering$_{\mathrm{EQ}_q}$. Hence, by Theorem 3.1 any OBDD representation of REQ_q has width at least $2^q \ge 2^{\frac{n}{2\lceil \log n+1\rceil}}$. □

Theorem 5.2. *There is a bounded error quantum* OBDD *representation of* REQ_q *of width* $O\left(n^2\right)$.

Proof. Let us interpret Lemma 4.1 in other words. If computing of a Boolean function is equivalent to checking the equality of $g(y_1,\ldots,y_{2q}) = c_1y_1 + \cdots + c_{2q}y_{2q}$ and 0, then we can construct a commutative QOBDD with one side error using the quantum finger printing technique.

By the definition, $\mathrm{REQ}_q(z_{1,1},\ldots,z_{2q,l},y_1,\ldots,y_{2q}) = 1$ iff

$$\sum_{i:\mathrm{bin}(z_{i,1},\ldots,z_{i,l})\le q} 2^{\mathrm{bin}(z_{i,1},\ldots,z_{i,l})} y_i - \sum_{i:\mathrm{bin}(z_{i,1},\ldots,z_{i,l})> q} 2^{\mathrm{bin}(z_{i,1},\ldots,z_{i,l})-q} y_i \equiv 0 \pmod 2^q.$$

Then we can choose the required coefficient using additional address qubits as in the reordering method and get QOBDD representing REQ with bounded error. □

5.2 Shifted Equality

In order to get another separation between quantum and classical OBDD complexities let us consider *shifted equality function* ($\mathrm{SEQ}_n : \{0,1\}^{2n+l} \to \{0,1\}$ where $l = \lceil \log n \rceil$), the function introduced by JaJa, Prasanna, and Simon [15]. The function is defined in the following manner: $\mathrm{SEQ}_n(x_1,\ldots,x_n,y_1,\ldots,y_m,s_1,\ldots,s_l) = 1$ iff for all $i \in \{1,\ldots,n\}$,

$$x_i = y_{(i+\mathrm{bin}(s_1,\ldots,s_l)) \pmod n}.$$

Using a lower bound for the best communication complexity of this function [15] and the well-known connection between OBDD and communication complexities we have the following property.

Lemma 5.3 (see for example [22])**.** *Any* OBDD *representation of* SEQ_n *has the size at least* $2^{\Omega(n)}$.

We can also construct a bounded error quantum OBDD representation of SEQ_n of a small width.

Lemma 5.4. *There is a bounded error quantum* OBDD *representation for* SEQ_n *of width* $O(n^2)$.

Proof. Let us construct a QOBDD P that reads an input in the following order: s, then x, and then y; also P uses a quantum register consisting of two parts: the first part $|\phi\rangle$ is for storing the value of the shift $(\mathrm{bin}\,(s_1,\ldots,s_l))$ and the second one $|\psi\rangle$ is called a computational part. The size of $|\phi\rangle$ is $\lceil \log n \rceil$ qubits and the size of $|\psi\rangle$ is $\log n + C$, for some constant C.

On the first $\lceil \log n \rceil$ levels, the program stores input bits into $|\phi\rangle$ using a storing procedure similar to procedure from the proof of Theorem 3.3.

Then we apply the fingerprint algorithm from [8,10], but use unitary matrices for y with shift depending on the state of $|\phi\rangle$.

After reading the last variable we measure $|\psi\rangle$ and get the answer.

The width of the program is $2^{\lceil \log n \rceil + \log n + C} = O(n^2)$. □

It is interesting to compare this separation and the separation obtained in the previous subsection. In this result the lower bound for OBDD width is $2^{\Omega(n)}$ but in the previous one it is $2^{\Omega(\frac{n}{\log n})}$. On the other hand, the upper bound for the width of QOBDD is also larger.

6 Hierarchy for Probabilistic and Quantum OBDDs

In this section we consider classes $\mathbf{BPOBDD}_d$ and $\mathbf{BQOBDD}_d$ of Boolean functions that can be represented by bounded error probabilistic and quantum OBDDs of width $O(d)$, respectively. We prove hierarchies with respect to d for these classes.

6.1 Hierarchy for Probabilistic OBDDs.

Before we start proving the hierarchy let us consider a Boolean function WS_n, or *weighted sum* function introduced by Savickỳ and Žák [29].

Let $n > 0$ be an integer and let $p(n)$ be the smallest prime greater than n. Let us define functions $s_n : \{0,1\}^n \to \{0,1\}$ and $\mathrm{WS}_n : \{0,1\}^n \to \{0,1\}$, such that $s_n(x_1,\ldots,x_n) = \left(\sum_{i=1}^{n} i \cdot x_i\right) \bmod p(n)$ and $\mathrm{WS}_n(x_1,\ldots,x_n) = x_{s_n(x_1,\ldots,x_n)}$. For the function WS_n it is known that any bounded error probabilistic OBDD representing WS_n has width at least $2^{\Omega(n)}$.

Let us modify the Boolean function WS_n using padding. We will denote this modified function as WS_n^b. Let $n > 0$ and $b > 0$ be integers, such that $b \le \frac{n}{3}$ and $p(b)$ be the smallest prime greater than b. We denote by $\mathrm{WS}_n^b : \{0,1\}^n \to \{0,1\}$ a function such that $\mathrm{WS}_n^b(x_1,\ldots,x_n) = x_{s_b(x_1,\ldots,x_n)}$. Using techniques similar to techniques from the paper [29] we can prove the following.

Lemma 6.1. *For any $b(n) = \omega(1)$, any bounded error probabilistic* OBDD *that represents* WS_n^b *has width at least $2^{\Omega(b)}$ and there is a bounded error probabilistic* OBDD *of width 2^b representing* WS_n^b.

Let us prove the hierarchy theorem for $\mathbf{BPOBDD}_d$ classes using these properties of the Boolean function WS_n^b.

Theorem 6.1. *If d and δ are functions such that $d(n) = o(2^n)$, $d(n) = \omega(1)$, and $\delta(n) = \omega(1)$, then* $\mathbf{BPOBDD}_{d^{1/\delta}} \subsetneq \mathbf{BPOBDD}_d$.

Proof. It is easy to see that $\mathbf{BPOBDD}_{d^{1/\delta}} \subseteq \mathbf{BPOBDD}_d$. Let us prove the inequality of these classes. Due to Lemma 6.1, the Boolean function $\mathrm{WS}_n^{\log d} \in \mathbf{BPOBDD}_d$. However, any bounded error probabilistic OBDD representing $\mathrm{WS}_n^{\log d}$ has width $2^{\Omega(\log d)}$ that is greater than $d^{1/\delta}$ since $d = \omega(1)$. Therefore, $\mathrm{WS}_n^{\log d} \notin \mathbf{BPOBDD}_{d^{1/\delta}}$. □

6.2 Hierarchy for Quantum OBDDs.

In this subsection we consider similar modifications of three well-known functions: REQ_n, $\mathrm{MOD}_{p,n}$, and MSW_n (defined in [26]). The function MSW_n may be defined in the following way: $\mathrm{MSW}_n(x_1,\dots,x_n) = x_z \oplus x_{r+n/2}$, where $z = s_{n/2}(x_1,\dots,x_{n/2})$ and $r = s_{n/2}(x_{n/2+1},\dots,x_n)$, if $r = z$ and $\mathrm{MSW}_n(x_1,\dots,x_n) = 0$ otherwise.

Let $\{f_n : \{0,1\}^n \to \{0,1\}\}_{n\in\mathbb{N}}$ be a family of Boolean functions and $b : \mathbb{N} \to \mathbb{N}$ be a function such that $b(n) \leq n$. We denote by $\left\{f_n^b : \{0,1\}^n \to \{0,1\}\right\}_{n\in\mathbb{N}}$ the family of Boolean functions such that $f_n^b(x_1,\dots,x_n) = f_{b(n)}(x_1,\dots,x_{b(n)})$.

Remark 6.1. *If for any OBDD (bounded error POBDD or QOBDD) representation of f_n has width at least $w(n)$, then OBDD (bounded error POBDD or QOBDD) representation of f_n^b has width at least $w(b(n))$. Moreover, if there is an OBDD (bounded error POBDD or QOBDD) representation of f_n of width $d(n)$, then there is an OBDD (bounded error POBDD or QOBDD) representation of f_n^b of width $d(b(n))$.*

In order to use this remark we need the following two lemmas.

Lemma 6.2 ([26])**.** *Any bounded error quantum OBDD representation of MSW_n has width at least $2^{\Omega(n)}$ and there is a bounded error quantum OBDD of width 2^n representing MSW_n.*

Lemma 6.3 ([5, 8])**.** *Any bounded error quantum OBDD representation of $\mathrm{MOD}_{p,n}$ (for $p \leq n$) has width at least $\lfloor \log p \rfloor$ and there is a bounded error quantum OBDD of width $O(\log p)$ representing $\mathrm{MOD}_{p,n}$.*

Now we are ready to prove the main theorem of this section.

Theorem 6.2. *Let $d : \mathbb{N} \to \mathbb{N}$ and $\delta : \mathbb{N} \to \mathbb{N}$ be functions such that $d(n) = \omega(1)$ and $\delta(n) = \omega(1)$.*

- *If $d(n) \leq \log n$ for all n, then $\mathbf{BQOBDD}_{\frac{d}{\delta}} \subsetneq \mathbf{BQOBDD}_d$;*
- *If $d(n) \leq n$ for all n, then $\mathbf{BQOBDD}_{\frac{d}{\log^2 d}} \subsetneq \mathbf{BQOBDD}_{d^2}$;*
- *If $d(n) \leq 2^n$ for all n, then $\mathbf{BQOBDD}_{d^{1/\delta}} \subsetneq \mathbf{BQOBDD}_d$.*

Proof. It is easy to see that for any $d' \leq d$, $\mathbf{BQOBDD}_{d'} \subseteq \mathbf{BQOBDD}_d$. Let us prove the inequalities.

Due to Lemma 6.3, the Boolean function $\mathrm{MOD}_{2^d,n} \in \mathbf{BQOBDD}_d$. However, width of any bounded error quantum OBDD representing $\mathrm{MOD}_{2^d,n}$ is at least $O(d)$. Therefore $\mathrm{MOD}_{2^d,n} \notin \mathbf{BQOBDD}_{d/\delta}$.

Due to Theorem 5.2, the Boolean function $\mathrm{REQ}_n^d \in \mathbf{BQOBDD}_{d^2}$. On the contrary by Theorem 5.2 and Remark 6.1 width of any bounded error quantum OBDD representing REQ_n^d is at least $\left\lfloor \frac{d}{\lceil \log d+1 \rceil} \right\rfloor$. Therefore, $\mathrm{REQ}_n^d \notin \mathbf{BQOBDD}_{\frac{d}{\log^2 d}}$.

Due to Lemma 6.2, the Boolean function $\mathrm{MSW}_n^{\log d} \in \mathbf{BQOBDD}_d$. However, width of any bounded error quantum OBDD representing $\mathrm{MSW}_n^{\log d}$ is at least $2^{\Omega(\log d)}$. Therefore, $\mathrm{MSW}_n^{\log d} \notin \mathbf{BQOBDD}_{d^{1/\delta}}$. □

7 Extension of Hierarchies for Deterministic and Probabilistic k-OBDDs

This section shows the separation between k-OBDDs and $2k$-OBDDs using the reordering method and a lower bound for a complexity of *pointer jumping* function also denoted as PJ [11, 24]

At first, let us present a version of the pointer jumping function which works with integer numbers. Let V_A and V_B be two disjoint sets of vertices with $|V_A| = |V_B| = m$ and $V = V_A \cup V_B$. Let $F^A = \left\{f^A : V_A \to V_B\right\}$, $F^B = \left\{f^B : V_B \to V_A\right\}$ and $f = (f^A, f^B) : V \to V$ defined by the following rule:

- if $v \in V_A$, then $f(v) = f^A(v)$ and
- if $v \in V_B$, then $f(v) = f^B(v)$.

For each $k \geq 0$ we define $f^{(k)}(v)$ such that $f^{(0)}(v) = v$ and $f^{(k+1)}(v) = f(f^{(k)}(v))$. Let $v_0 \in V_A$, The function we are interested in is $g_{k,m} : F^A \times F^B \to V$ such that $g_{k,m}(f^A, f^B) = f^{(k)}(v_0)$.

The Boolean function $\mathrm{PJ}_{t,n} : \{0,1\}^n \to \{0,1\}$ is a Boolean version of $g_{k,m}$ where we encode f^A as a binary string using $m \log m$ bits and f^B as well. The result of the function is the parity of bits of the binary representation for the resulted vertex.

We apply the reordering method to the $\mathrm{PJ}_{k,m}$ function and call the total version of it, obtained from Corollary 3.2, as $\mathrm{RPJ}_{k,m}$.

Note that to prove an upper bound for $\mathrm{RPJ}_{2k-1,m}$ it is necessary to construct a commutative $2k$-OBDD for $\mathrm{PJ}_{2k-1,m}$. In order to prove a lower bound for $\mathrm{RPJ}_{2k-1,m}$ it is necessary to prove a lower bound for $\mathrm{PJ}_{2k-1,m}$.

For proving the lover bound we need notion of communication complexity. Let $f : \{0,1\}^m \times \{0,1\}^n \to \{0,1\}$ be a Boolean function. We have two players called Alice and Bob, who have to compute $f(x, y)$. The function f is known by both of them. However, Alice knows only bits of x and Bob knows only bits of y. They have a two-sided communication channel. On each round of their communication one of them send a string and Alice and Bob are trying to minimize two parameters: total number of sent bits and number of rounds. For the formal definition see for example [22].

Additionally, we say that the k-round communication complexity with Bob sending first of a function $f(x_1, \dots, x_n, y_1, \dots, y_m)$ equals to c iff the minimal number of sent bits of k-round communication protocols with Bob sending first is equal to c. We denote this complexity as $C^{B,k}(f)$ if this protocol is deterministic and $C^{B,k}_{\epsilon}(f)$ for probabilistic one with bounded error ϵ.

Lemma 7.1 ([24]). *$C^{B,k}(\mathrm{PJ}_{k,m}) = \Omega(m - k \log m)$ for any k.*

Lemma 7.2 ([24]). *$C^{B,k}_{1/3}(\mathrm{PJ}_{k,m}) = \Omega(\frac{m}{k^2} - k \log m)$ for any k.*

Note that there is a well-know connection between communication complexity and OBDD complexity.

Lemma 7.3 (see for example [19]). *Let $f(x_1, \dots, x_n, y_1, \dots, y_m)$ be a Boolean function, π be an order over the variables $x_1, \dots, x_n, y_1, \dots, y_m$ such that y_i precedes x_j for any $i \in \{1, \dots, n\}$, and $j \in \{1, \dots, m\}$.*

If there is a π-k-OBDD representing f of width w, then there is a $(2k-1)$-round communication protocol for f of cost $\log w$ and Bob sending first.

The next corollary follows from the previous three lemmas.

Corollary 7.1. *For any positive integer k and order π, such that the variables encoding f_A precedes the variables encoding f_B,*

- *width of any k-π-POBDD representing $\mathrm{PJ}_{2k-1,m}$ with bounded error is at least $2^{\Omega(\frac{m}{k^2} - k \log m)}$ and*
- *width of any k-π-OBDD representing $\mathrm{PJ}_{2k-1,m}$ is at least $2^{\Omega(m - k \log m)}$.*

Lemma 7.4. *There is a commutative $2k$-OBDD representing $\mathrm{PJ}_{2k-1,m}$ of width $O(km^2)$.*

Proof. First of all, let us note that if a function $f : \{0,1\}^n \to T_1$ has a k'-OBDD and for all $t \in T_1$, a function $g_t : \{0,1\}^n \to T_2$ has a commutative OBDD, then a function $h : \{0,1\}^n \to T_2$ such that $h(x) = g_{f(x)}(x)$ has a commutative $(k'+1)$-OBDD.

Secondly, note that for any v there is a commutative OBDD representing $f(v)$ due to the fact that $f(v)$ has a $S_{m,\mathrm{id},+}$ representation and Theorem 4.1 . $\square$

Corollary 7.2. *There is a $2k$-OBDD representing $\mathrm{RPJ}_{2k-1,m}$ of width $O(km^3)$.*

Using this results we can extend the hierarchy for following classes: **P**-k-OBDD, $\mathbf{BPP}_{\beta}$-k-OBDD, **SUPERPOLY**-OBDD, $\mathbf{BSUPERPOLY}_{\beta}$-$k$-OBDD, $\mathbf{SUBEXP}_{\alpha}$-k-OBDD, and $\mathbf{BSUBEXP}_{\alpha,\beta}$-$k$-OBDD. These are classes of Boolean functions computed by the following models:

- **P**-k-OBDD and $\mathbf{BPP}_{\beta}$-k-OBDD are for polynomial width k-OBDD, the first one is for deterministic case and the second one is for bounded error probabilistic k-OBDD with error at least β.

- **SUPERPOLY**-k-OBDD and **BSUPERPOLY**$_\beta$-k-OBDD are similar classes for superpolynomial width models.
- **SUBEXP**$_\alpha$-k-OBDD and **BSUBEXP**$_{\alpha,\beta}$-k-OBDD are similar classes for width at most $2^{O(n^\alpha)}$, for $0 < \alpha < 1$.

Theorem 7.1. *1.* $\mathbf{P}$-k-OBDD $\subsetneq$ $\mathbf{P}$-$2k$-OBDD*, for* $k = o(n/\log^3 n)$*.*

2. $\mathbf{BPP}_{1/3}$-k-OBDD $\subsetneq$ $\mathbf{BPP}_{1/3}$-$2k$-OBDD*, for* $k = o(n^{1/3}/\log n)$*.*

3. **SUPERPOLY**-k-OBDD $\subsetneq$ **SUPERPOLY**-$2k$-OBDD*, for* $k = o(n^{1-\delta})$*,* $\delta > 0$*.*

4. **BSUPERPOLY**$_{1/3}$-k-OBDD $\subsetneq$ **BSUPERPOLY**$_{1/3}$-$2k$-OBDD*, for* $k = o(n^{1/3-\delta})$ *and* $\delta > 0$*.*

5. **SUBEXP**$_\alpha$-k-OBDD $\subsetneq$ **SUBEXP**$_\alpha$-$2k$-OBDD*, for* $k = o(n^{1-\delta})$*,* $1 > \delta > \alpha + \varepsilon$*, and* $\varepsilon > 0$*.*

6. **BSUBEXP**$_{\alpha,1/3}$-k-OBDD $\subsetneq$ **BSUBEXP**$_{\alpha,1/3}$-$2k$-OBDD*, for* $k = o(n^{1/3-\delta/3})$*,* $1/3 > \delta > \alpha + \varepsilon$*, and* $\varepsilon > 0$*.*

Proof. Proofs of all statements are the same, hence, here we present only proof of the first one.

Let us consider $\mathrm{RPJ}_{2k-1,n}$. Every k-OBDD representing the function has width at least

$$2^{\Omega(n/(k\log n)-\log(n/\log n))} \geq 2^{\Omega(n/(n\log^{-3} n\log n)-\log(n/\log n))} = 2^{\Omega(\log^2 n)} = n^{\Omega(\log n)},$$

due to Lemma 7.4. Therefore, it has more than polynomial width. Hence, $\mathrm{RPJ}_{2k-1,n} \notin \mathbf{P}$-$k$-OBDD and $\mathrm{RPJ}_{2k-1,n} \in \mathbf{P}$-$2k$-OBDD, due to Lemma 7.4. □

Acknowledgements.

Partially supported by ERC Advanced Grant MQC. The work is performed according to the Russian Government Program of Competitive Growth of Kazan Federal University. We thank Alexander Vasiliev and Aida Gainutdinova from Kazan Federal University and Andris Ambainis from University of Latvia for their helpful comments and discussions.

References

[1] F. Ablayev, A. Gainutdinova, K. Khadiev, and A. Yakaryılmaz. Very narrow quantum OBDDs and width hierarchies for classical OBDDs. *Lobachevskii Journal of Mathematics*, 37(6):670–682, 2016.

[2] Farid Ablayev, Andris Ambainis, Kamil Khadiev, and Aliya Khadieva. Lower bounds and hierarchies for quantum memoryless communication protocols and quantum ordered binary decision diagrams with repeated test. *LNCS, SOFSEM 2018: Theory and Practice of Computer Science*, 2018. arXiv preprint arXiv:1703.05015.

[3] Farid Ablayev and Aida Gainutdinova. Complexity of quantum uniform and nonuniform automata. In Clelia de Felice and Antonio Restivo, editors, *Developments in Language Theory, 9th International Conference, DLT 2005, Palermo, Italy, July 4-8, 2005, Proceedings*, volume 3572 of *Lecture Notes in Computer Science*, pages 78–87. Springer, 2005.

[4] Farid Ablayev, Aida Gainutdinova, and Marek Karpinski. On Computational Power of Quantum Branching Programs. In Rusins Freivalds, editor, *Fundamentals of Computation Theory, 13th International Symposium, FCT 2001, Riga, Latvia, August 22-24, 2001, Proceedings*, volume 2138 of *Lecture Notes in Computer Science*, pages 59–70. Springer, 2001.

[5] Farid Ablayev, Aida Gainutdinova, Marek Karpinski, Cristopher Moore, and Chris Pollett. On the computational power of probabilistic and quantum branching program. *Inf. Comput.*, 203(2):145–162, 2005.

[6] Farid Ablayev, Aida Gainutdinova, Kamil Khadiev, and Abuzer Yakaryilmaz. Very narrow quantum OBDDs and width hierarchies for classical OBDDs. In Helmut Jürgensen, Juhani Karhumäki, and Alexander Okhotin, editors, *Descriptional Complexity of Formal Systems - 16th International Workshop, DCFS 2014, Turku, Finland, August 5-8, 2014. Proceedings*, volume 8614 of *Lecture Notes in Computer Science*, pages 53–64. Springer, 2014.

[7] Farid Ablayev, Airat Khasianov, and Alexander Vasiliev. On complexity of quantum branching programs computing equality-like boolean functions. *ECCC*, 2010.

[8] Farid Ablayev and Alexander Vasiliev. On the computation of boolean functions by quantum branching programs via fingerprinting. *Electronic Colloquium on Computational Complexity (ECCC)*, 15(059), 2008.

[9] Farid M Ablayev and Alexander Vasiliev. Algorithms for quantum branching programs based on fingerprinting. *Int. J. Software and Informatics*, 7(4):485–500, 2013.

[10] Farid Mansurovich Ablayev and AV Vasilyev. On quantum realisation of boolean functions by the fingerprinting technique. *Discrete Mathematics and Applications*, 19(6):555–572, 2009.

[11] Beate Bollig, Martin Sauerhoff, Detlef Sieling, and Ingo Wegener. Hierarchy theorems for kOBDDs and kIBDDs. *Theoretical Computer Science*, 205(1):45–60, 1998.

[12] A. F. Gainutdinova. Comparative complexity of quantum and classical OBDDs for total and partial functions. *Russian Mathematics*, 59(11):26–35, 2015.

[13] Juraj Hromkovič and Martin Sauerhoff. The power of nondeterminism and randomness for oblivious branching programs. *Theory of Computing Systems*, 36(2):159–182, 2003.

[14] Rishat Ibrahimov, Kamil Khadiev, Krišjānis Prūsis, and Abuzer Yakaryılmaz. Zero-error affine, unitary, and probabilistic OBDDs. *arXiv preprint arXiv:1703.07184*, 2017.

[15] Joseph JáJá, Viktor K. Prasanna, and Janos Simon. Information transfer under different sets of protocols. *SIAM J. Comput.*, 13(4):840–849, 1984.

[16] K. Khadiev. Width hierarchy for k-OBDD of small width. *Lobachevskii Journal of Mathematics*, 36(2), 2015.

[17] Kamil Khadiev and Rishat Ibrahimov. Width hierarchies for quantum and classical ordered binary decision diagrams with repeated test. In *Proceedings of the Fourth Russian Finnish Symposium on Discrete Mathematics*, number 26 in TUCS Lecture Notes. Turku Centre for Computer Science, 2017.

[18] Kamil Khadiev and Aliya Khadieva. Reordering method and hierarchies for quantum and classical ordered binary decision diagrams. In *Computer Science – Theory and Applications: 12th International Computer Science Symposium in Russia, CSR 2017, Kazan, Russia, June 8-12, 2017, Proceedings*, pages 162–175, Cham, 2017. Springer International Publishing.

[19] Khadiev Khadiev. On the hierarchies for Deterministic, nondeterministic and probabilistic ordered read-k-times branching programs. *Lobachevskii Journal of Mathematics*, 37(6):682–703, 2016.

[20] Jan Krajícek. An exponential lower bound for a constraint propagation proof system based on ordered binary decision diagrams. *J. Symb. Log.*, 73(1):227–237, 2008.

[21] Matthias Krause, Christoph Meinel, and Stephan Waack. Separating the eraser turing machine classes le, nle, co-nle and pe. *Theoretical Computer Science*, 86(2):267–275, 1991.

[22] Eyal Kushilevitz and Noam Nisan. *Communication complexity*. Cambridge University Press, 1997.

[23] Masaki Nakanishi, Kiyoharu Hamaguchi, and Toshinobu Kashiwabara. Ordered quantum branching programs are more powerful than ordered probabilistic branching programs under a bounded-width restriction. In *COCOON*, volume 1858 of *LNCS*, pages 467–476. Springer, 2000.

[24] Noam Nisan and Avi Wigderson. Rounds in communication complexity revisited. *SIAM Journal on Computing*, 22(1):211–219, 1993.

[25] Martin Sauerhoff. Quantum vs. classical read-once branching programs. *arXiv preprint quant-ph/0504198*, 2005.

[26] Martin Sauerhoff. Quantum vs. classical read-once branching programs. In Matthias Krause, Pavel Pudlák, Rüdiger Reischuk, and Dieter van Melkebeek, editors, *Complexity of Boolean Functions*, number 06111 in Dagstuhl Seminar Proceedings, Dagstuhl, Germany, 2006. Internationales Begegnungs- und Forschungszentrum für Informatik (IBFI), Schloss Dagstuhl, Germany.

[27] Martin Sauerhoff and Detlef Sieling. Quantum branching programs and space-bounded nonuniform quantum complexity. *Theoretical Computer Science*, 334(1-3):177–225, 2005.

[28] Martin Sauerhoff and Detlef Sieling. Quantum branching programs and space-bounded nonuniform quantum complexity. *Theoretical Computer Science*, 334(1):177–225, 2005.

[29] Petr Savickỳ and S Žák. A read-once lower bound and a (1,+ k)-hierarchy for branching programs. *Theoretical Computer Science*, 238(1):347–362, 2000.

[30] Ingo Wegener. *Branching Programs and Binary Decision Diagrams: Theory and Applications*. SIAM, 2000.

ECCC ISSN 1433-8092

https://eccc.weizmann.ac.il