EasyChair Preprint

№ 1521

Two-Way Quantum and Classical Automata with Advice for Online Minimization Problems

Kamil Khadiev and Aliya Khadieva

September 14, 2019

Two-Way Quantum and Classical Automata with Advice for Online Minimization Problems

Kamil Khadiev[1,2], Aliya Khadieva[2,3]

[1] Smart Quantum Technologies Ltd., Kazan, Russia
[2] Kazan Federal University, Kazan, Russia
[3] University of Latvia, Riga, Latvia
kamilhadi@gmail.com, aliya.khadi@gmail.com

Abstract. We consider online algorithms. Typically the model is investigated with respect to competitive ratio. In this paper, we explore two-way automata as a model for online algorithms. We focus on quantum and classical online algorithms. We show that there are problems that can be solved more efficiently by two-way automata with quantum and classical states than classical two-way automata in the case of sublogarithmic memory (sublinear size) even if classical automata get advice bits.
Keywords: quantum computation, online algorithms, streaming algorithms, online minimization problems, two-way automata, automata

1 Introduction

Online algorithms are a well-known computational model for solving optimization problems. The peculiarity is that the algorithm reads an input piece by piece and should return an answer piece by piece immediately, even if the answer can depend on future pieces of the input. The algorithm should return an answer for minimizing (maximizing) an objective function (the cost of the output). There are different methods to define the effectiveness of online algorithms [13, 17], but the most standard is the competitive ratio [40, 28]. Typically, online algorithms have unlimited computational power. At the same time, it is quite interesting to solve online minimization problems in the case of a big input stream such that the stream cannot be stored completely in the memory. As the algorithms, we can consider Turing machines with restricted memory or two-way automata with non-constant size (a number of states). In the paper, we focus on two-way automata. Streaming algorithms or one-way automata as online algorithms were considered in [9, 26, 14, 33, 32, 34, 30]. We focus on *quantum online algorithms*. This model was introduced in [33] and discussed in [1]. In the case of one-way streaming algorithms, it is known that quantum online streaming algorithms can be better than classical ones [33, 30]. Another model that was considered by researchers is quantum online streaming algorithms with repeated test [43].

In this paper, we explore quantum online algorithms that have the only restriction on memory but have no restriction on access to already taken input variables. We mean two-way automata as online algorithms. This model is more

close to the general model of online algorithms comparing to online streaming algorithms or online streaming algorithms with repeated test. The question of comparing quantum and classical models was explored for streaming algorithms (OBDDs and one way automata)[25, 3, 4, 21, 37, 31, 6, 7, 36, 2, 23, 24, 22], and for two-way automata [8]. We use these results as a base for our results.

Moreover, we are interested in an *advice complexity* measure [35, 12, 15, 20, 19, 16]. In this case, an online algorithm gets some bits of advice about an input. A trusted *Adviser* sending these bits knows the whole input and has unlimited computational power. Deterministic and randomized online algorithms with advice are considered in [27, 35, 10]. If we consider online streaming algorithms with advice, then the quantum model can be better than classical ones [33, 32, 34]. We compare the power of quantum online algorithms and classical ones in the case of two-way automata. This question was not investigated before. Typically, term "Adviser" is used in online algorithms theory; and term "Oracle" in the case of other models.

We use the "Black Hats Method" for constructing hard online minimization problems[32, 34]. We present problems for a separation between the power of quantum and classical two-way automata using this method. Suppose that algorithms use only $o(\log n)$ bits of memory ($n^{o(1)}$ states) in the case of exponential expected working time and $o\left((\log n)^{0.5-\alpha}\right)$ bits of memory ($n^{o\left((\log n)^{-(0.5+\alpha)}\right)}$ states) in the case of polynomial expected working time, where n is the length of an input, $0 < \alpha < 0.5$. For both cases (exponential and polynomial working time), we have two results:

- There is a special online minimization problem that has a two-way automaton with classical and quantum states with better competitive ratio than any two-way classical (probabilistic or deterministic) automata, even if the classical ones have a non-constant number of advice bits.
- For the same problem, a two-way automaton with classical and quantum states has a better competitive ratio than any deterministic online algorithm with unlimited computational power has.

We consider problems that are based on "Black Hats Method" [32, 34]; *Palindrome* and *Unitary equality* languages from [8].

The paper is organized as follows. We present definitions in Section 2. Black Hats Method is described in Section 3. A discussion on two-way automata with quantum and classical states vs. classical ones is given in Section 4.

2 Preliminaries

An online minimization problem consists of a set $\mathcal{I}$ of inputs and a cost function. Each input $I \in \mathcal{I}$ is a sequence of requests $I = (x_1, \ldots, x_n)$. Furthermore, a set of feasible outputs (or solutions) is associated with each I; an output is a sequence of answers $O = (y_1, \ldots, y_n)$. The cost function assigns a positive real value $cost(I, O)$ to a feasible input I and a feasible output O. For each input I, we call any feasible output O for I that has the smallest possible cost (i.

e., that minimizes the cost function) an optimal solution for I. The goal is the searching for the optimal solution for I.

Let us define an online algorithm for this problem as an algorithm which gets requests x_i from $I = (x_1, \ldots, x_n)$ one by one and should return answers y_i from $O = (y_1, \ldots, y_n)$ immediately, even if an optimal solution can depend on future requests. **A deterministic online algorithm** A computes the output sequence $A(I) = (y_1, \ldots, y_n)$ such that y_i is computed from $x_1, \ldots, x_i$. We say that a deterministic online algorithm A is *c-competitive* if there exists a non-negative constant α such that, for every n and for any input I of size n, we have: $cost(I, A(I)) \leq c \cdot cost(I, Opt(I)) + \alpha$, where Opt is an optimal offline algorithm for the problem and c is the minimal number that satisfies the inequality. Also we call c the **competitive ratio** of A. If $\alpha = 0, c = 1$, then A is optimal.

An online algorithm A with advice computes an output sequence $A^{\phi}(I) = (y_1, \ldots, y_n)$ such that y_i is computed from $\phi, x_1, \ldots, x_i$, where ϕ is the message from the adviser, who knows the whole input. A is c-competitive with advice complexity $b = b(n)$ if there exists a non-negative constant α such that, for every n and for any input I of size n, there exists some ϕ such that $cost(I, A^{\phi}(I)) \leq c \cdot cost(I, Opt(I)) + \alpha$ and $|\phi| \leq b$; $|\phi|$ is a length of ϕ.

A randomized online algorithm R computes an output sequence $R^{\psi}(I) = (y_1, \ldots, y_n)$ such that y_i is computed from $\psi, x_1, \ldots, x_i$, where ψ is the content of the random tape, i. e., an infinite binary sequence, where every bit is chosen uniformly at random and independently of all the others. By $cost(I, R^{\psi}(I))$ we denote the random variable expressing the cost of the solution computed by R on I. R is c-competitive in expectation if there exists a constant $\alpha > 0$ such that, for every I, $\mathbb{E}\left[cost(I, R^{\psi}(I))\right] \leq c \cdot cost(I, Opt(I)) + \alpha$.

We use two-way automata for online minimization problems as online algorithms with restricted memory. Let us give definitions of automata.

A two-way deterministic automaton working on inputs of length/size $m \geq 0$ (2DA) D is a 6-tuple $D = (\Sigma, \Gamma, S, s_1, \delta, Result)$, where (i) Σ is an input alphabet; (ii) Γ is an output alphabet; (iii) $S = \{s_1, \ldots, s_d\}$ is the set of states (d can be a function in m), $s_1 \in S$ is the initial state; (iv) $Results : S \to \Gamma$ is a function that transforms a state to an output symbol; (v) $\delta : S \times \Sigma \to S \times \{\leftarrow, \downarrow, \rightarrow\}$ is a transition function. Any given input $u \in \Sigma^m$ is placed on a read-only tape with a single head as $¢u_1u_2 \ldots u_m\$\diamond$, where $u_i \in \Sigma$ is the i-th symbol of u, ¢ is a left end marker and $\$, \diamond$ are right end markers. When D is in $s \in S$ and reads $u_i \in \Sigma$ on the tape, the automaton switches to state $s' \in S$ and updates the head position with respect to $a \in \{\leftarrow, \downarrow, \rightarrow\}$ if $\delta(s, u_i) \to (s', a)$. If $a =$ " $\leftarrow$ " (" $\rightarrow$ "), then the head moves one square to the left (the right), and, it stays on the same square, otherwise. The transition function δ must be defined to guarantee that the head never leaves $¢u\$\diamond$ during the computation. Moreover, if the automaton reaches the second endmarker $\diamond$ in a state s, then D finishes the computation and returns $Result(s)$ as a result.

The probabilistic counterpart of 2DA, denoted 2PA, can choose from more than one transition in each step such that each transition is associated with a probability. Thus, 2PAs can be in a probability distribution over the determinis-

tic configurations (the state and the position of the head forms a configuration) during the computation. The total probability must be 1, i.e., the probability of outgoing transitions from a single configuration must be 1. Thus, a 2PA returns some result for each input with some probability. For $v \in \Gamma$, a 2PA returns a result v for an input, with bounded-error if the 2PA returns the result v with probability at least $1/2 + \varepsilon$ for some $\varepsilon \in (0, 1/2]$.

Let us use these models for online minimization problems. A **2DA for online minimization problems** A computes the output sequence $A(I) = (y_1, \ldots, y_n)$ where y_i is a result of computation A on the input $¢x_1, \ldots, x_i\$\diamond$, such that A starts from a state s that is the final state for computing y_{i-1}, and the input head observes x_1. A **2PA**, a **2DA with advice** and a **2PA with advice for online minimization problems** have similar definitions, but with respect to definitions of corresponding models of online algorithms.

Let us define the quantum counterparts of the models. You can read more about quantum computation and quantum automata in [7, 38, 8]. Quantum devices manipulate quantum states. A quantum state can be described by a 2^q-dimensional vector from Hilbert space over the field of complex numbers. Here q is a number of quantum bits (qubits). A unitary transformation is applying $2^q \times 2^q$ (left) unitary matrices of complex numbers. Let us describe the measurement process. Suppose that an automaton is in a distribution of quantum states $|\psi\rangle = (v_1, \ldots, v_{2^q})$ before a measurement and measures the i-th qubit. Suppose states with numbers $a^0_1, \ldots, a^0_{2^{q-1}}$ correspond to the 0 value of the i-th qubit, and states with numbers $a^1_1, \ldots, a^1_{2^{q-1}}$ correspond to the 1 value of the qubit. Then the result of the measurement of the qubit is 1 with probability $pr_1 = \sum_{j=1}^{2^{q-1}} |v_{a^1_j}|^2$ and 0 with probability $pr_0 = 1 - pr_1$. If the algorithm measures z qubits on the j-th step, then it gets number $\gamma \in \{0, \ldots, 2^z - 1\}$ as a result of the measurement.

A **quantum online algorithm** Q computes the output sequence $Q(I) = (y_1, \ldots, y_n)$ such that y_i depends on $x_1, \ldots, x_i$. The algorithm uses quantum memory, and can apply unitary transformations to quantum memory and measure qubits several times during a computation. Note that a quantum computation is a probabilistic process. Q is c-competitive in expectation if there exists a non-negative constant ξ such that, for every I, $\mathbb{E}\left[cost(I, Q(I))\right] \leq c \cdot cost(I, Opt(I)) + \xi$.

Let us consider a **two-way automaton with quantum and classical states** (2QCA), which is a 9-tuple $M = (Q, S, \Sigma, \Gamma, \theta, \delta, v_1, s_1, Result)$, where (i) Q and S are sets of quantum and classical states respectively; (ii) θ and δ are quantum and classical transition functions; (iii) $v_1 \in Q$ and $s_1 \in S$ are initial quantum and classical states; (iv) Σ is an input alphabet and Γ is an output alphabet; (v) $Results : S \to \Gamma$ is a function that obtain output symbol by a state. The function θ specifies the evolution of the quantum portion of the internal state: for each pair $(s, x) \in S \times \Sigma$, $\theta(s, x)$ is an action to be performed on the quantum portion of the internal state of M. Each action $\theta(s, x)$ corresponds to either a unitary transformation or an orthogonal measurement. The function δ specifies the evolution of the classical part of M (i.e., the classical part of the

internal state and the tape head). In a case $\theta(s,x)$ is a unitary transformation, $\delta(s,x)$ is an element of $S \times \{\leftarrow, \downarrow, \rightarrow\}$ specifying a new classical state and a movement of the tape head. In a case $\theta(s,x)$ is a measurement, $\delta(s,x)$ is a mapping from the set of possible results of the measurement to $S \times \{\leftarrow, \downarrow, \rightarrow\}$ (again specifying a new classical state and a tape head movement, this time one such pair for each outcome of the observation). It is assumed that δ is defined so that the tape head never moves left when scanning the left end-marker ¢, and never moves right when scanning the right end-marker $\diamond$. Other restrictions and behavior are similar to 2DA. We can define **2QCA for online minimization problems** in the same way as for 2DA for online minimization problems. The 2QCA model is similar to 2QCFA model from [8] but the size (the number of states) of 2QCA can depend on the length of the input m. The same difference between 2DA and 2DFA, 2PA and 2PFA.

In the paper we use the terminology for branching programs [41] and algorithms. We say that an automaton computes Boolean function f_m if for any input X of length m, the automaton returns result 1 iff $f(X) = 1$. Additionally, we use the terminology on memory from algorithms. We say that an automaton has s bits of memory if it has 2^s states.

3 Two-Way Automata for Black Hats Online Minimization Problem

Let us describe the "black hats method" from [32, 34] that allows us to construct hard online minimization problems. In the paper we discuss a Boolean function f, but in fact we consider a family of Boolean functions $f = \{f_1, f_2, \dots\}$, for $f_m : \{0,1\}^m \to \{0,1\}$. We use notation $f(X)$ for $f_m(X)$ if the length of X is m and it is clear from the context.

Suppose we have a Boolean function f and integers $k, r, w, t > 0$, where k mod $t = 0$. Then an online minimization problem $BH^t_{k,r,w}(f)$ is the following. We have k guardians and k prisoners. They stay one by one in a line like $G_1P_1G_2P_2\dots$, where G_i is a guardian, P_i is a prisoner. The prisoner P_i has an input X_i of length m_i and computes a function $f_{m_i}(X_i)$. The prisoner paints his hat black or white with respect to the result 1 or 0. Each guardian wants to know the parity of a number of following black hats. So, G_i wants to compute $f_{m_i}(X_i) \oplus \dots \oplus f_{m_k}(X_k)$. We split sequential guardians into t blocks. The cost of a block is r if all guardians of the block are right; and w, otherwise. Let us define the problem formally:

Definition 1 (Black Hats Method). *We have a Boolean function f. Then an online minimization problem $BH^t_{k,r,w}(f)$, for integers $k, r, w, t > 0$, where k mod $t = 0$, is the following. Suppose we have an input $I = (x_1, \dots, x_n)$ and k integers $m_1, \dots, m_k > 0$, where $n = \sum_{i=1}^{k}(m_i+1)$. Let $I = 2\ X_1\ 2\ X_2\ 2\ X_3\ 2\ \dots\ 2\ X_k$, where $X_i = (x^i_1, \dots, x^i_{m_i}) \in \{0,1\}^{m_i}$, for $i \in \{1, \dots, k\}$. Let O be a sequence of answers that corresponds to the input I. Let $O' = (y_1, \dots, y_k)$ be answer variables corresponding to input variables with value 2 (in other words, output variables for guardians). An answer variable y_j corresponds to an input variable x_{i_j}, where*

$i_j = j + \sum_{r=1}^{j-1} m_r$. *Let* $g_j(I) = \bigoplus_{i=j}^{k} f_{m_i}(X_i)$. *We separate all answer variables* y_i *to* t *blocks of length* $z = k/t$. *The cost of the* i*-th block is* c_i. *Here* $c_i = r$ *if* $y_j = g_j(I)$ *for* $j \in \{(i-1)z+1, \ldots, i \cdot z\}$; *and* $c_i = w$, *otherwise. The cost of the whole output is* $cost^t(I, O) = c_1 + \cdots + c_t$.

We can show that any 2DA using s bits of memory cannot solve $BH^t_{k,r,w}(f)$ if there is no 2DA computing the function f using s bits of memory.

Theorem 1. *Let* s *be a positive integer. Suppose a Boolean function* f *is such that no 2DA for* f *uses at most* s *bits of memory. Then there is no* c*-competitive 2DA for* $BH^t_{k,r,w}(f)$ *using* s *bits of memory, where* $c < w/r$.

Proof. Let us consider any 2DA A for the $BH^t_{k,r,w}(f)$ problem that uses at most s bits of memory. Suppose that A returns y_1 as an answer of the first guardian. Let us prove that there are two parts of the input $X_1^0, X_1^1 \in \{0,1\}^{m_1}$ such that A returns the same value y_2 for both, but $f(X_1^0) = 0, f(X_1^1) = 1$.

Assume that there is no such triple (y_2, X_1^0, X_1^1). Then, we can construct a 2DA A' that uses s bits of memory and has the following property: $A'(X_1') = A'(X_1'')$ iff $f(X_1') = f(X_1'')$, for any $X_1', X_1'' \in \{0,1\}^{m_1}$. The automaton A' emulates the automaton A. Therefore, A' computes f or $\neg f$. In the case of $\neg f$, we can construct A'' such that $A''(X_1) = \neg A'(X_1)$. It is a contradiction with the claim of the theorem. By the same way, we can show existence of similar triples (y_{i+1}, X_i^0, X_i^1) for $i \in \{2, \ldots, k\}$.

Let us choose $\sigma_i = y_i \oplus 1 \oplus \bigoplus_{j=i+1}^{k} \sigma_j$, for $i \in \{1, \ldots, k\}$. Let us consider an input $I_A = 2X_1^{\sigma_1}2 \ldots 2X_k^{\sigma_k}$. An optimal offline solution is $(g_1, \ldots, g_k)$ where $g_i = \bigoplus_{j=i}^{k} \sigma_j$. Let us prove that $g_i \neq y_i$ for each $i \in \{1, \ldots, k\}$. We have $\sigma_i = y_i \oplus 1 \oplus \bigoplus_{j=i+1}^{k} \sigma_j$. Therefore, $y_i = \sigma_i \oplus 1 \oplus \bigoplus_{j=i+1}^{k} \sigma_j = 1 \oplus \bigoplus_{j=i}^{k} \sigma_j = 1 \oplus g_i$, so $y_i = \neg g_i$. Hence, all answers are wrong and $cost^t(I_A, A(I_A)) = tw$. So the competitive ratio c cannot be less than $tw/(tr) = w/r$. □

The similar result holds for probabilistic two-way automata.

Theorem 2. *Let* s *be a positive integer. Suppose a Boolean function* f *is such that no 2PA uses at most* s *bits of memory and computes* f *with bounded error. Then there is no* c*-competitive in expectation 2PA using* s *bits of memory and solving* $BH^t_{k,r,w}(f)$ *with bounded error, where* $c < 2^{-z} + (1 - 2^{-z})w/r$.

Proof. (Sketch) The proof is similar to deterministic case but we can guess unknown bits with probability 0.5. □

There is a bound on the competitive ratio in the case of unlimited computational power for a deterministic online algorithm.

Theorem 3 ([32]). *Claim 1. There is no* c*-competitive deterministic online algorithm* A *computing* $BH^t_{k,r,w}(f)$, *for* $c < \left(\lfloor (t+1)/2 \rfloor \cdot w + (t - \lfloor (t+1)/2 \rfloor) \cdot r\right)/(tr)$.

Claim 2. There is no c*-competitive deterministic online algorithm* A *for* $BH^1_{k,r,w}(f)$, *for* $c < w/r$.

Theorem 4. *Let us consider a Boolean function f. Suppose we have a 2QCA R that computes f with bounded error ε using s classical bits and s quantum bits of memory, where $0 \leq \varepsilon < 0.5$. Then there is a 2QCA A for $BH^t_{k,r,w}(f)$ that uses at most $s + O(1)$ classical bits and at most $s + 1$ quantum bits of memory, and has expected competitive ratio $c \leq \left(0.5(1-\varepsilon)^{z-1} \cdot (r-w) + w\right)/r$.*

Let us consider the model with advice.

In the following properties of $BH^t_{k,r,w}(f)$ problem, we show that if the model has not enough memory, then the problem can be interpreted as the "String Guessing, Unknown History" ($2{-}SGUH$) problem from [11]. The problem is the following. On each step, an algorithm should guess the next input bit.

The following result for the $2-SGUH$ is known:

Lemma 1 ([11]). *Consider an input string of length k for $2{-}SGUH$, for some positive integer k. Any online algorithm that is correct in more than αk characters, for $0.5 \leq \alpha < 1$, needs to read at least $(1 + (1-\alpha)\log_2(1-\alpha) + \alpha \log_2 \alpha)\, k$ advice bits.*

Using this result for $2-SGUH$, we can show the following properties of $BH^t_{k,r,w}(f)$ problem with respect to two-way automata with advice for online minimization problems.

Theorem 5. *Let s be a positive integer. Suppose a Boolean function f is such that no 2DA uses at most s bits of memory and computes f. Then there is no c-competitive 2DA that uses $s-b$ bits of memory and b advice bits, and solves $BH^t_{k,r,w}(f)$, where $c < (hr + (t-h)w)/(tr)$, $h = \lfloor v/z \rfloor, z = k/t$, v is such that $b = (1 + (1-v/k)\log_2(1-v/k) + (v/k)\log_2(v/k))\, k$, $0.5 \cdot k \leq v < k$.*

We have a similar situation in a probabilistic case. We use a function $\delta_x : \mathbb{R} \to \{0,1\}$ in the claim of the following theorem: $\delta_x = 1$ iff $x \neq 0$.

Theorem 6. *Let s be a positive integer. Suppose a Boolean function f is such that no 2PA uses at most s bits of memory and computes f. Then there is no c-competitive in expectation 2PA that uses $s-b$ bits of memory and b advice bits, and solves $BH^t_{k,r,w}(f)$ with bounded error, where $c \geq (hr + \delta_u \cdot (2^{u-z}r + (1-2^{u-z})w) + (t-h-\delta_u)(2^{-z}r + (1-2^{-z})w))/(tr)$, for $h = \lfloor v/z \rfloor, z = k/t, u = v - hz$, v is such that $b = (1 + (1-v/k)\log_2(1-v/k) + (v/k)\log_2(v/k))\, k$, $0.5k \leq v < k$.*

Proof. (Sketch). The idea of the proof is similar to the proof of Theorem 5, but here we can guess all "unknown" guardians with probability 0.5. □

4 Application

Let us discuss applications of Black Hats Method. We present examples of problems that allow us to show benefits of quantum computing.

Exponential Expected Working Time. Let us consider exponential expected working time for two-way automata.

In this case, we analyze the *palindrome* Boolean function. The Boolean function $Pal : \{0,1\}^m \to \{0,1\}$ is the following. $Pal(X) = 1$ if $X = X^R$; and $Pal(X) = 0$, otherwise. Here $X = (x_1, \ldots, x_m)$, $X^R = (x_m, \ldots, x_1)$ is a reversed X. It is known that there is a 2QCFA that recognizes the palindrome language [8]. 2QCFA is 2QCA with a constant size of memory. At the same time, we can show a lower bound for 2PA that is based on lower bounds from [18, 29, 5, 35, 39]. Therefore, we have the following results:

Lemma 2. *The following two claims are true: 1. There is a 2QCA that uses quantum and classical memory of constant size that works in exponential expected time and computes Pal with bounded error. 2. No 2PA uses $o(\log n)$ bits of memory that works in exponential expected time and computes Pal with bounded error, where n is the length of the input.*

Proof. The first claim follows from the result for the language version of Pal [8], let us prove the second claim. It is known from [18, 29] that if a 2PA recognize a language or computes a Boolean function f, then the following property holds: $N(f) \leq (C_1 \cdot \log T)^{C_2 \cdot d^2 \log_2 d}$, where $C_1, C_2 = const$, T is expected time, d is the size (the number of states) of the automaton. $N(f)$ is a number of Myhill-Nerode classes in a language version and the number of subfunctions in a Boolean functions version. The number of subfunctions is analogue of number of Myhill-Nerode classes, you can read more in [35, 42, 41]. Additionally, it is easy to see that $N(Pal) \geq 2^{n/2}$.

The memory of the automaton is $o(\log n) = o(0.5 \log n - \log\log n)$. Therefore, $d = o(\sqrt{n/(\log n)^2}) = o\left(\frac{\sqrt{n/(\log n)}}{\log(n/\log n)}\right)$. Hence $d^2 \log d = o(n/\log n)$. If T is exponential, then we can replace $C_1 \log T$ by $C_3 \cdot n$ for some constant C_3. Finally, we obtain that $(C_1 \cdot \log T)^{C_2 \cdot d^2 \log_2 d} = 2^{o(n)} < 2^{n/2}$. Therefore, by lower bounds [18, 29], 2PAs with $o(\log n)$ bits of memory cannot compute the function. □

Let us consider the $BHPal^t_{k,r,w} = BH^t_{k,r,w}(Pal)$ problem. Recall that $BH^t_{k,r,w}(f)$ is a black hat problem for k guardians, t blocks of guardians, the cost r for a right answer of a block, the cost w for a wrong answer of a block, $z = k/t$ and $k \bmod t = 0$. Let us discuss the properties of the $BHPal^t_{k,r,w}$ problem:

Theorem 7. *Suppose $P^t = BHPal^t_{k,r,w}$, $t \in \{1, \ldots, k\}$, $k = (\log_2 n)^{O(1)}$, v is such that $b = (1 + (1 - v/k)\log_2(1 - v/k) + (v/k)\log_2(v/k))\, k$, $0.5k \leq v < k$, all automata work in exponential expected time; then*

1. There is no c-competitive 2DA that uses $s = o(\log n)$ bits of memory and b advice bits, and solves P^t, where $c < \mathcal{C}_1 = w/r$, $b = o(z/\log z)$.

2. There is no deterministic online algorithm with unlimited computational power computing P^1 that is c-competitive, for $c < \mathcal{C}_1 = w/r$.

3. There is no c-competitive in expectation 2PA that uses $o(\log n)$ bits of memory and solves P^t, where $c < \mathcal{C}_3 = 2^{-z} + (1 - 2^{-z})w/r$.

4. There is no c-competitive 2DA that uses $s = o(\log n)$ bits of memory and b advice bits, and solves P^t, where $c < \mathcal{C}_2 = \frac{hr+(t-h)w}{tr}$, $h = \lfloor v/z \rfloor$.

5. No 2PA uses $s = o(\log n)$ bits of memory, b advice bits and solves P^t that is c-competitive in expectation for $h = \lfloor v/z \rfloor, u = v - hz$,

$c < \mathcal{C}_4 = \frac{hr+\delta_u \cdot (2^{u-z}r+(1-2^{u-z})w)+(t-h-\delta_u)(2^{-z}r+(1-2^{-z})w)}{tr}$.

6. There is a 2QCA Q that uses a constant number of classical and quantum bits of memory and solves P^t. The algorithm Q has expected competitive ratio $c \leq ((1-\varepsilon)^{z-1} \cdot 0.5 \cdot (r-w)+w)/r < \mathcal{C}_1, \mathcal{C}_2, \mathcal{C}_3, \mathcal{C}_4$, for some ε: $0 < \varepsilon < 0.5$.

Proof. Let us consider Claim 1 of the theorem. Due to Lemma 2, no 2DA with $o(\log n)$ computes P^t. Hence, because of Theorem 5, Claim 1 is true. Claim 2 follows from Theorem 3. Let us consider Claim 3 of the theorem. Due to Lemma 2, no 2PA with $o(\log n)$ computes P^t with bounded error. Therefore, because of Theorem 2, Claim 3 is true. Claim 4 follows from Lemma 2 and Theorem 5. Claim 5 follows from Lemma 2 and Theorem 6. Claim 6 follows from Lemma 2 and Theorem 4. □

This theorem gives us the following important results. (i) There is a 2QCA with a constant size of memory for $BHPal^1_{k,r,w}$ that has a better competitive ratio than any 2DA or 2PA with sublogarithmic memory and sublogarithmic number of advice bits (Claims 1, 3, 4, 5 and 6 of Theorem 7); any deterministic online algorithm without restriction on memory (Claims 2 and 6 of Theorem 7). (ii) If we increase the number of advice bits for 2DA or 2PA for $BHPal^t_{k,r,w}$, then the competitive ratio becomes smaller, in the case of sublogarithmic memory and $1 < t \leq k/2$. At the same time, the competitive ratio is still larger than for a 2QCA (Claims 4, 5 and 6 of Theorem 7).

Polynomial Expected Working Time. Let us consider the polynomial expected working time for two-way automata. For this case, we analyze the *UEQ* Boolean function. The Boolean function $UEQ : \{0,1\}^m \to \{0,1\}$ is the following. $UEQ(X) = 1$ iff $\#_1(X) = \#_0(X)$, where $\#_j(X)$ is the number of symbols j in X. It is known that there is a 2QCFA that recognizes the language version of UEQ in polynomial time [8]. At the same time, we can show a lower bound that is based on lower bounds from [18, 29, 35]. Therefore, we have the following results:

Lemma 3. *The following two claims are true. 1.There is a 2QCA that uses quantum and classical memory of constant size that works in exponential expected time and computes UEQ with bounded error. 2.No 2PA uses $o\left((\log n)^{0.5-\alpha}\right)$ bits of memory that works in polynomial expected time and computes UEQ with bounded error, where n is the length of input, $0 < \alpha < 0.5$.*

Proof. The first claim follows from [8], the proof of the second claim is similar to the proof of Lemma 2. □

Let us consider the $BHUEQ^t_{k,r,w} = BH^t_{k,r,w}(UEQ)$ problem. Recall that the problem is a black hat problem for k guardians, t blocks of guardians, the cost r for a right answer of a block, the cost w for a wrong answer of a block, $z = k/t$ and $k \bmod t = 0$. Let us discuss the properties of the $BHUEQ^t_{k,r,w}$ problem:

Theorem 8. *Suppose $P^t = BHUEQ^t_{k,r,w}$, $t \in \{1, \ldots, k\}$, $k = (\log_2 n)^{O(1)}$, v is such that $b = (1 + (1 - v/k)\log_2(1 - v/k) + (v/k)\log_2(v/k))\, k$, $0.5k \le v < k$, all automata work in polynomial expected time, $0 < \alpha < 0.5$; then*

1. There is no c-competitive 2DA that uses $s = o\left((\log n)^{0.5-\alpha}\right)$ bits of memory and b advice bits, and solves P^t, where $c < \mathcal{C}_1 = w/r$, $b = o(z/\log z)$.

2. There is no deterministic online algorithm with unlimited computational power computing P^1 that is c-competitive, for $c < \mathcal{C}_1 = w/r$.

3. There is no c-competitive in expectation 2PA that uses $o\left((\log n)^{0.5-\alpha}\right)$ bits of memory and solves P^t, where $c < \mathcal{C}_3 = 2^{-z} + (1 - 2^{-z})w/r$.

4. There is no c-competitive 2DA that uses $s = o\left((\log n)^{0.5-\alpha}\right)$ bits of memory and b advice bits, and solves P^t, where $c < \mathcal{C}_2 = \frac{hr+(t-h)w}{tr}$, $h = \lfloor v/z \rfloor$.

5. No 2PA using $s = o\left((\log n)^{0.5-\alpha}\right)$ bits of memory, b advice bits and solving P^t that is c-competitive in expectation for $h = \lfloor v/z \rfloor, u = v - hz$,

$c < \mathcal{C}_4 = \frac{hr+\delta_u\cdot(2^{u-z}r+(1-2^{u-z})w)+(t-h-\delta_u)(2^{-z}r+(1-2^{-z})w)}{tr}$.

6. There is a 2QCA Q that uses a constant number of classical and quantum bits of memory and solves P^t. The algorithm Q has expected competitive ratio $c \le ((1-\varepsilon)^{z-1} \cdot 0.5 \cdot (r - w) + w)/r < \mathcal{C}_1, \mathcal{C}_2, \mathcal{C}_3, \mathcal{C}_4$, for some ε: $0 < \varepsilon < 0.5$.

Proof. The proof is similar to the proof of Theorem 7. The claims follow from Lemma 3 and all theorems from Section 3. □

This theorem gives us the following important results: (i) There is a 2QCA for $BHUEQ^1_{k,r,w}$ with constant size of memory and polynomial expected time that has a better competitive ratio than any 2DA or 2PA with the size of memory less than $o\left(\sqrt{\log_2 n}\right)$ and the number of advice bits less than $o\left(\sqrt{\log_2 n}\right)$, and works in polynomial time (Claims 1, 3, 4, 5 and 6 of Theorem 8); any deterministic online algorithm without restriction on memory (Claims 2 and 6 of Theorem 8). (ii) If we increase the number of advice bits for 2DA or 2PA for $BHUEQ^t_{k,r,w}$, then the competitive ratio becomes smaller, in the case of sublogarithmic memory and $1 < t \le k/2$. At the same time, the competitive ratio is still larger than for a 2QCA (Claims 4, 5 and 6 of Theorem 8).

Acknowledgements. This work was supported by Russian Science Foundation Grant 19-71-00149.

References

1. F. Ablayev, M. Ablayev, K. Khadiev, and A. Vasiliev. Classical and quantum computations with restricted memory. *LNCS*, 11011:129–155, 2018.

2. F. Ablayev, A. Ambainis, K. Khadiev, and A. Khadieva. Lower bounds and hierarchies for quantum memoryless communication protocols and quantum ordered binary decision diagrams with repeated test. *In SOFSEM, LNCS*, 10706:197–211, 2018.
3. F. Ablayev, A. Gainutdinova, M. Karpinski, C. Moore, and C. Pollett. On the computational power of probabilistic and quantum branching program. *Information and Computation*, 203(2):145–162, 2005.
4. F. Ablayev, A. Gainutdinova, K. Khadiev, and A. Yakarylmaz. Very narrow quantum OBDDs and width hierarchies for classical OBDDs. In *DCFS*, volume 8614 of *LNCS*, pages 53–64. Springer, 2014.
5. F. Ablayev and K. Khadiev. Extension of the hierarchy for k-OBDDs of small width. *Russian Mathematics*, 53(3):46–50, 2013.
6. A. Ambainis and A. Yakaryılmaz. Superiority of exact quantum automata for promise problems. *Information Processing Letters*, 112(7):289–291, 2012.
7. A. Ambainis and A. Yakaryılmaz. Automata and quantum computing. Technical Report 1507.01988, arXiv, 2015.
8. Andris Ambainis and John Watrous. Two–way finite automata with quantum and classical states. *Theoretical Computer Science*, 287(1):299–311, 2002.
9. L. Becchetti and E. Koutsoupias. Competitive analysis of aggregate max in windowed streaming. In *ICALP*, volume 5555 of *LNCS*, pages 156–170, 2009.
10. H.-J. Böckenhauer, J. Hromkovič, D. Komm, R. Královič, and P. Rossmanith. On the power of randomness versus advice in online computation. In *Languages Alive*, pages 30–43. Springer, 2012.
11. H.-J. Böckenhauer, J. Hromkovič, D. Komm, S. Krug, J. Smula, and A. Sprock. The string guessing problem as a method to prove lower bounds on the advice complexity. *Theoretical Computer Science*, 554:95–108, 2014.
12. J. Boyar, L.M Favrholdt, C. Kudahl, K.S. Larsen, and J.W. Mikkelsen. Online algorithms with advice: A survey. *ACM Computing Surveys*, 50(2):19, 2017.
13. J. Boyar, S. Irani, and K. S. Larsen. A comparison of performance measures for online algorithms. In *Proceedings of WADS2009*, volume 5664 of *LNCS*, pages 119–130. Springer, 2009.
14. J. Boyar, K. S. Larsen, and A. Maiti. The frequent items problem in online streaming under various performance measures. *International Journal of Foundations of Computer Science*, 26(4):413–439, 2015.
15. H.-J. Bckenhauer, D. Komm, R. Krlovi, R. Krlovi, and . Mmke. On the advice complexity of online problems. *In ISAAC 2009, LNCS*, 5878:331–340, 2009.
16. S. Dobrev, R. Královič, and D. Pardubská. How much information about the future is needed? In *SOFSEM*, pages 247–258. Springer, 2008.
17. R. Dorrigiv and A. López-Ortiz. A survey of performance measures for on-line algorithms. *SIGACT News*, 36(3):67–81, 2005.
18. C. Dwork and L. J. Stockmeyer. A time complexity gap for two-way probabilistic finite-state automata. *SIAM Journal on Computing*, 19(6):1011–1123, 1990.
19. Y. Emek, P. Fraigniaud, A. Korman, and A. Rosén. Online computation with advice. In *ICALP*, pages 427–438. Springer, 2009.
20. Y. Emek, P. Fraigniaud, A. Korman, and A. Rosén. Online computation with advice. *Theoretical Computer Science*, 412(24):2642–2656, 2011.
21. A. Gainutdinova. Comparative complexity of quantum and classical OBDDs for total and partial functions. *Russian Mathematics*, 59(11):26–35, 2015.
22. A. Gainutdinova and A. Yakaryılmaz. Unary probabilistic and quantum automata on promise problems. In *Developments in Language Theory*, pages 252–263. Springer, 2015.

23. A. Gainutdinova and A. Yakaryılmaz. Nondeterministic unitary OBDDs. In *CSR 2017*, pages 126–140. Springer, 2017.
24. A. Gainutdinova and A. Yakaryılmaz. Unary probabilistic and quantum automata on promise problems. *Quantum Information Processing*, 17(2):28, 2018.
25. D. Gavinsky, J. Kempe, I. Kerenidis, R. Raz, and R. de Wolf. Exponential separations for one-way quantum communication complexity, with applications to cryptography. In *STOC '07*, pages 516–525, 2007.
26. Y. Giannakopoulos and E. Koutsoupias. Competitive analysis of maintaining frequent items of a stream. *Theoretical Computer Science*, 562:23–32, 2015.
27. J. Hromkovic. Design and analysis of randomized algorithms: Introduction to design paradigms, 2005.
28. A. R Karlin, M. S Manasse, L. Rudolph, and D. D Sleator. Competitive snoopy caching. In *FOCS, 1986., 27th Annual Symposium on*, pages 244–254. IEEE, 1986.
29. K. Khadiev, R. Ibrahimov, and A. Yakaryılmaz. New size hierarchies for two way automata. *Lobachevskii Journal of Mathematics*, 39(7), 2018.
30. K. Khadiev and A. Khadieva. Quantum automata for online minimization problems. In *Ninth Workshop on NCMA 2017 Short Papaers*, pages 25–33. Institute fur Computersprachen TU Wien, 2017.
31. K. Khadiev and A. Khadieva. Reordering method and hierarchies for quantum and classical ordered binary decision diagrams. In *CSR 2017*, volume 10304 of *LNCS*, pages 162–175. Springer, 2017.
32. K. Khadiev, A. Khadieva, D. Kravchenko, A. Rivosh, and I. Yamilov, R.and Mannapov. Quantum versus classical online streaming algorithms with logarithmic size of memory. *Lobachevskii Journal of Mathematics*, 2017. (in print). arXiv:1710.09595.
33. K. Khadiev, A. Khadieva, and I. Mannapov. Quantum online algorithms with respect to space and advice complexity. *Lobachevskii Journal of Mathematics*, 39(9):1210–1220, 2018.
34. K. Khadiev, M. Ziatdinov, I. Mannapov, A. Khadieva, and R. Yamilov. Quantum online streaming algorithms with constant number of advice bits. *arXiv:1802.05134*, 2018.
35. Dennis Komm. *An Introduction to Online Computation: Determinism, Randomization, Advice*. Springer, 2016.
36. François Le Gall. Exponential separation of quantum and classical online space complexity. SPAA '06, pages 67–73. ACM, 2006.
37. M. Sauerhoff and D. Sieling. Quantum branching programs and space-bounded nonuniform quantum complexity. *Theoretical Computer Science*, 334(1):177–225, 2005.
38. A.C. Cem Say and Abuzer Yakaryılmaz. Quantum finite automata: A modern introduction. In *Computing with New Resources*, pages 208–222. Springer, 2014.
39. John C. Shepherdson. The reduction of two–way automata to one-way automata. *IBM Journal of Research and Development*, 3:198–200, 1959.
40. Daniel D Sleator and Robert E Tarjan. Amortized efficiency of list update and paging rules. *Communications of the ACM*, 28(2):202–208, 1985.
41. Ingo Wegener. *Branching Programs and Binary Decision Diagrams: Theory and Applications*. SIAM, 2000.
42. Sergey Vsevolodovich Yablonsky. *Introduction to Discrete Mathematics: Textbook for Higher Schools*. Mir Publishers, 1989.
43. Q. Yuan. *Quantum online algorithms*. UC Santa Barbara, 2009. PhD thesis.