Quantum computation with devices whose contents are never read*

Abuzer Yakaryılmaz†
Boğaziçi University, Department of Computer Engineering,
Bebek 34342 İstanbul, Turkey
abuzer@boun.edu.tr

Rūsiņš Freivalds
Institute of Mathematics and Computer Science, University of Latvia,
Raiņa bulvāris 29, Riga, LV-1459, Latvia
Rusins.Freivalds@mii.lu.lv

A. C. Cem Say
Boğaziçi University, Department of Computer Engineering,
Bebek 34342 İstanbul, Turkey
say@boun.edu.tr

Ruben Agadzanyan
Institute of Mathematics and Computer Science, University of Latvia,
Raiņa bulvāris 29, Riga, LV-1459, Latvia
agrubi@gmail.com

Keywords: quantum computation, write-only memory, quantum finite automata, counter automata, quantum Fourier transform

Abstract

In classical computation, a "write-only memory" (WOM) is little more than an oxymoron, and the addition of WOM to a (deterministic or probabilistic) classical computer brings no advantage. We prove that quantum computers that are augmented with WOM can solve problems that neither a classical computer with WOM nor a quantum

*A preliminary version of this work was presented in the 9^{th} International Conference on Unconventional Computation (UC2010).

†Corresponding author

computer without WOM can solve, when all other resource bounds are equal. We focus on realtime quantum finite automata, and examine the increase in their power effected by the addition of WOMs with different access modes and capacities. Some problems that are unsolvable by two-way probabilistic Turing machines using sublogarithmic amounts of read/write memory are shown to be solvable by these enhanced automata.

Introduction

It is well known that many physical processes that violate human "common sense" are in fact sanctioned by quantum theory. Quantum computation as a field is interesting for precisely the fact that it demonstrates that quantum computers can perform resource-bounded tasks which are (in some cases, provably) beyond the capabilities of classical computers. In this paper, we demonstrate that the usage of "write-only memory" (WOM), a computational component that is used exclusively for being written to, and never being read, (which is little more than a joke in the classical setup,) can improve the power of quantum computers significantly. We prove that a quantum computer using a WOM can solve problems that neither a classical computer with a WOM nor a quantum computer without a WOM can solve, when all machines are restricted to use a constant amount of read/write memory; in fact, we show that certain tasks that cannot be achieved by classical probabilistic Turing machines, even if they are allowed to use sublogarithmic amounts of read/write memory, can be performed by quantum finite automata augmented with write-only memory (QFA-WOMs).

In the rest of the paper, we first present the basic notation and terminology that will be employed. After a review of the automaton variants whose powers will be compared with those of the machines with write-only memory, we give a formal definition of the automata with WOM. Following a thorough examination of the power of a QFA-WOM variant with severe restrictions on the way in which it can access its WOM, we look at less restricted models. The interesting question of just how much WOM is sufficient to endow an otherwise finite-state machine with the capability of recognizing nonregular languages is addressed, and a way of looking at our results as contributions about quantum function computation is presented.

Preliminaries

For a given vector v, $v[i]$ is the i^{th} entry of v and for a given string w, $|w|$ is the length of w.

Σ (*input alphabet*): Σ is a finite set of symbols, i.e. $\Sigma = \{\sigma_1, \ldots, \sigma_{|\Sigma|}\}$. As a convention, Σ never contains the symbol # (*the blank symbol*), ¢ (*the left input end-marker*), and \$ (*the right input end-marker*). $\tilde{\Sigma}$ denotes the set $\Sigma \cup \{¢, \$\}$. Additionally, $\tilde{w}$ denotes the string $¢w\$$, for any given input string $w \in \Sigma^*$.

Q (*the set of internal states*): Q is a finite set of internal states, i.e. $Q = \{q_1, \ldots, q_{|Q|}\}$. Unless otherwise specified, q_1 is *the initial state*. Moreover, depending on the context, $Q_a \subseteq Q$ (resp., $Q \setminus Q_a$) is the set of accepting (resp., rejecting) states.

$\langle\rangle$ (*the set of head directions*): $\langle\rangle$ is the set $\{\leftarrow, \downarrow, \rightarrow\}$, where "$\leftarrow$" means that the (corresponding) head moves one square to the left, "$\downarrow$" means that the head stays on the same square, and "$\rightarrow$" means that the head moves one square to the right. As a special case, $\triangleright$ is the set $\{\downarrow, \rightarrow\}$.

Θ (*the counter status*): Θ is the set $\{0, 1\}$, where 1 means that the counter value is nonzero, and 0 means that the counter value is zero.

δ (*the transition function*): The behavior of a machine is specified by its transition function. The domain and the range of a transition function may vary with respect to the capabilities of the model.

$f^a_{\mathcal{M}}(w)$ (*acceptance probability*): For a given machine $\mathcal{M}$ and an input string $w \in \Sigma^*$, $f^a_{\mathcal{M}}(w)$, or shortly $f_{\mathcal{M}}(w)$, is the probability that w will be accepted by $\mathcal{M}$. Moreover, $f^r_{\mathcal{M}}(w)$ will be used in order to represent *the rejection probability* of w by $\mathcal{M}$.

The language $L \subset \Sigma^*$ recognized by machine $\mathcal{M}$ with (strict) cutpoint $\lambda \in \mathbb{R}$ is defined as

$$L = \{w \in \Sigma^* \mid f_{\mathcal{M}}(w) > \lambda\}. \tag{1}$$

The language $L \subset \Sigma^*$ recognized by machine $\mathcal{M}$ with nonstrict cutpoint $\lambda \in \mathbb{R}$ is defined as (Blondel et al., 2005)

$$L = \{w \in \Sigma^* \mid f_{\mathcal{M}}(w) \geq \lambda\}. \tag{2}$$

The language $L \subset \Sigma^*$ is said to be recognized by machine $\mathcal{M}$ with unbounded error if there exists a cutpoint $\lambda \in \mathbb{R}$ such that L is recognized by $\mathcal{M}$ with strict or nonstrict cutpoint λ.

The language $L \subset \Sigma^*$ recognized by machine $\mathcal{M}$ with error bound ϵ ($0 \leq \epsilon < \frac{1}{2}$) is defined as

- $f^a_{\mathcal{M}}(w) \geq 1 - \epsilon$ when $w \in L$,

- $f^r_{\mathcal{M}}(w) \geq 1 - \epsilon$ when $w \notin L$.

This situation is also known as recognition with bounded error.

The language $L \subset \Sigma^*$ is said to be recognized by machine $\mathcal{M}$ with (positive) one-sided bounded error if there exists a $p \in (0, 1]$ such that

- $f_{\mathcal{M}}(w) \geq p$ when $w \in L$ and
- $f_{\mathcal{M}}(w) = 0$ when $w \notin L$.

Equivalently, it can be said that $L \subset \Sigma^*$ is recognized by machine $\mathcal{M}$ with (positive) one-sided error bound ϵ, where $\epsilon = 1 - p$ (and so $\epsilon \in [0, 1)$).

The language $L \subset \Sigma^*$ is said to be recognized by machine $\mathcal{M}$ with negative one-sided bounded error if there exists a $p \in (0, 1]$ such that

- $f_{\mathcal{M}}(w) = 1$ when $w \in L$ and
- $f_{\mathcal{M}}(w) \leq 1 - p$ when $w \notin L$.

Equivalently, it can be said that $L \subset \Sigma^*$ is recognized by machine $\mathcal{M}$ with negative one-sided error bound ϵ, where $\epsilon = 1 - p$ (and so $\epsilon \in [0, 1)$).

Conventional computation models

In this section, we will review the conventional computational models (i.e. those not involving write-only memory) to be used in the paper. The reader is assumed to be familiar with the standard definitions (involving a read-only input tape and one read/write work tape) (Arora and Barak, 2009) for deterministic and probabilistic Turing machines (TM and PTM, respectively). Our definition of quantum Turing machine (QTM) is a modification of the one found in (Watrous, 1998)[1]. Technically, we allow the additional finite register, which is observed after each step of the computation to decide whether to accept, reject, or continue, to have multiple symbols in its alphabet corresponding to each of these alternatives, and to be refreshed to its initial symbol after each observation[2]. The addition of this finite register, which will be explained in greater detail in the context of the definition of quantum counter automata below, allows our QTM's to implement general quantum operations, and therefore to simulate their classical counterparts precisely and efficiently. This result was shown for QTM's with classical tape head position by Watrous (Watrous, 2003).

Our discussion will focus almost entirely on realtime computation, where the input is consumed one symbol per step in a single left-to-right pass, and

the decision is announced immediately upon the reading of the right endmarker. A realtime $k \in \mathbb{Z}^+$ counter automaton (RT-kCA) is a realtime finite state automaton augmented with k counters which can be modified by some amount from $\Diamond = \{-1, 0, +1\}$ ("-1" means that the value of the counter is decreased by 1, "0" means that the value of the counter is not changed, and "$+1$" means that the value of the counter is increased by 1), and where the signs of these counters are also taken into account during transitions.

For a given input string $w \in \Sigma$ ($\tilde{w}$ is on the input tape), a configuration of a RT-kCA is composed of the following elements:

- the current internal state,
- the position of the input head, and
- the contents of the counters.

The computation begins with the initial configuration, in which the internal state is q_1, the value(s) of the counter(s) is (are) zero(s), and the input head is placed on symbol ¢.

Formally, a realtime probabilistic k-counter automaton (RT-PkCA) $\mathcal{M}$ is a 5-tuple[3]

$$\mathcal{P} = (Q, \Sigma, \delta, q_1, Q_a). \tag{3}$$

The transitions of $\mathcal{M}$ are specified by δ as follows: $\delta(q, \sigma, \bar{\theta}, q')$ is the probability that $\mathcal{M}$ will change its state to $q' \in Q$, if it is originally in state $q \in Q$, scanning the symbol $\sigma \in \tilde{\Sigma}$ on the input tape, and sensing $\bar{\theta} \in \Theta^k$ in its counter(s) (i.e. $\bar{\theta}[i]$ is the status of the i^{th} counter, where $1 \leq i \leq k$). In each transition, the input tape head moves one square to the right, and the counters are updated with respect to $\bar{c} = D_c(q')$, where D_c is a function from Q to $\Diamond^k$, (i.e. the value of the j^{th} counter is updated by $\bar{c}[j]$, where $1 \leq j \leq k$).

The input string is accepted by a RT-PkCA if the computation ends in an accepting state.

A realtime deterministic k-counter automaton (RT-DkCA) is just a RT-PkCA in which all transitions with nonzero probability have probability 1.

A realtime nondeterministic k-counter automaton (RT-NkCA) is just a RT-PkCA that is interpreted to recognize a language with cutpoint 0, that is, the language it recognizes is the set of all and only the strings that it accepts with nonzero probability.

For the quantum case, we will only be concerned with one-counter automata. A realtime quantum 1-counter automaton (RT-Q1CA)[4] is a 6-tuple

$$\mathcal{M} = (Q, \Sigma, \Omega, \delta, q_1, Q_a). \tag{4}$$

When $\mathcal{M}$ is in state $q \in Q$, reading symbols $\sigma \in \tilde{\Sigma}$ on the input tape, and sensing $\theta \in \Theta$ on the counter, it changes its state to $q' \in Q$, updates the value of the counter with respect to $c \in \{\diamondsuit$, moves the input tape head one square to the right, and writes $\omega \in \Omega$ in the finite register with transition amplitude $\delta(q, \sigma, \theta, q', c, \omega) \in \mathbb{C}$, satisfying the well-formedness condition to be described below.

As seen above, the quantum machines in this paper will be distinguished from their classical counterparts by the presence of the item Ω (*the finite register alphabet*) in their definitions. This register, whose incorporation in a classical machine would not make any change to its computational power, is an essential part of our quantum models. In realtime computation, the usage of the finite register can be simplified so that the intermediate observations, mentioned above for the case of general QTMs, are not required, and a single measurement of the internal state at the end suffices for our purposes (Yakaryılmaz, 2010; Yakaryılmaz and Say, 2010b). Q is partitioned into two disjoint sets, Q_a, and $Q_r = Q \setminus Q_a$. In each transition, the quantum machine goes through the following phases:

1. *pre-transition phase*: reset the register to its initial symbol "ω_1";

2. *transition phase*: update the content of the register, in addition to the changes in the configuration components normally associated by the transitions of the corresponding classical machine.

On the right end-marker, the projective measurement

$$P = \{P_{\tau \in \{a,r\}} \mid P_\tau = \sum_{q \in Q_\tau} |q\rangle\langle q|\} \tag{5}$$

is performed. The result associated with P_τ is simply τ, and the input is accepted if "a" is observed. Note that this just means that the acceptance probability is the sum of the squares of the moduli of the amplitudes of the accepting states at the end of the computation.

Since we do not consider the register content as part of the configuration, the register can be seen as the "environment" interacting with the "principal system" that is the rest of the quantum machine (Nielsen and Chuang, 2000). δ therefore induces a set of configuration transition matrices, $\{E_{\omega \in \Omega}\}$, where the $(i,j)^{th}$ entry of E_ω, the amplitude of the transition from c_j to c_i by writing $\omega \in \Omega$ on the register, is defined by δ whenever c_j is reachable from c_i in one step, and is zero otherwise. The $\{E_{\omega \in \Omega}\}$ form an operator $\mathcal{E}$.

Let $\mathcal{C}$ be the set of configurations that can be attained by the machine for a given input string. According to the modern understanding of quantum

computation (Aharonov et al., 1998), a quantum machine is said to be *well-formed* if $\mathcal{E}$ is a superoperator (selective quantum operator), i.e.

$$\sum_{\omega\in\Omega} E_\omega^\dagger E_\omega = I. \tag{6}$$

$\mathcal{E}$ can be represented by a $|\mathcal{C}||\Omega| \times |\mathcal{C}|$-dimensional matrix E (Figure 1) by concatenating each $E_{\omega\in\Omega}$ one under the other. It can be verified that $\mathcal{E}$ is a superoperator if and only if the columns of E form an orthonormal set.

Figure 1: Matrix E

	c_1	c_2	$\ldots$	$c_{\lvert\mathcal{C}\rvert}$
c_1				
c_2				
$\vdots$		E_{ω_1}		
$c_{\lvert\mathcal{C}\rvert}$				
c_1				
c_2				
$\vdots$		E_{ω_2}		
$c_{\lvert\mathcal{C}\rvert}$				
c_1				
c_2				
$\vdots$		$\vdots$		
$c_{\lvert\mathcal{C}\rvert}$				
c_1				
c_2				
$\vdots$		$E_{\omega_{\lvert\Omega\rvert}}$		
$c_{\lvert\mathcal{C}\rvert}$				

(7)

We define a realtime quantum finite automaton (RT-QFA) as just a RT-Q1CA that never updates its counter. The class of languages recognized with bounded error by RT-QFAs equals the class of regular languages (Bozapalidis, 2003; Jeandel, 2007; Ambainis and Yakaryılmaz, 2010).

Lemma 1. *Any classical automaton can be simulated by a quantum automaton of the corresponding type exactly, such that the simulating and simulated machines agree on the value of the acceptance probability of any string.*

Proof. See (Paschen, 2000; Hirvensalo, 2008; Watrous, 2009; Ambainis and Yakaryılmaz, 2010; Yakaryılmaz, 2010; Yakaryılmaz and Say, 2010b). □

The next lemma demonstrates a useful programming trick about counters. (We will show this for the quantum case. It is well known that the same method can also be employed for classical counter machines.) For any counter automaton model A, let A(m) be a machine of type A with the additional ability of updating each of its counters with an increment from the set $\{-m, \ldots, m\}$, where $m > 1$, in a single step.

Lemma 2. *For any RT-Q1CA(m) $\mathcal{M}$, there exists a corresponding RT-Q1CA $\mathcal{M}'$ such that*

$$f_{\mathcal{M}}(w) = f_{\mathcal{M}'}(w), \tag{8}$$

for all $w \in \Sigma^*$, where $m > 1$.

Proof. Let $\mathcal{M} = (Q, \Sigma, \Omega, \delta, q_1, Q_a)$. We construct $\mathcal{M}' = (Q', \Sigma, \Omega, \delta', q_1', Q_a')$. Q' contains m states for each state of $\mathcal{M}$, that is, the states of $\mathcal{M}'$ are of the form

$$\langle q, i \rangle \in Q \times \{0, \ldots, m-1\}. \tag{9}$$

Moreover, $q_1' = \langle q_1, 0 \rangle$, and

$$Q_a' = \{\langle q, i \rangle \mid q \in Q_a, i \in \{0, \ldots, m-1\}\}. \tag{10}$$

Let

$$\varphi : \mathbb{Z} \to \mathbb{Z} \times \{0, \ldots, m-1\} \tag{11}$$

be a bijection such that

$$\varphi(x) = \left(\left\lfloor \frac{x}{m} \right\rfloor, (x \mod m) \right). \tag{12}$$

Hence, we can say that the counter values of $\mathcal{M}$, say $x \in \mathbb{Z}$, can be equivalently represented by $\varphi(x)$, based on which we will construct $\mathcal{M}'$, where $\varphi(x)[1]$ is stored by the counter, and $\varphi(x)[2]$ is stored by the internal state. That is, for any configuration of $\mathcal{M}$, say (q, x),

$$\left(\langle q, \varphi(x)[2] \rangle, \varphi(x)[1] \right) \tag{13}$$

is an equivalent configuration of $\mathcal{M}'$. Moreover, the transitions of $\mathcal{M}'$ can be obtained from those of $\mathcal{M}$ in the following way: for any $i \in \{-m, \ldots, m\}$ and $j \in \{0, \ldots, m-1\}$, the part of the transition

$$(q, \sigma) \xrightarrow{\delta} \alpha(q', i, \omega) \tag{14}$$

of $\mathcal{M}$ is replaced by transition

$$(\langle q, j\rangle, \sigma) \xrightarrow{\delta'} \alpha \left(\langle q', j + i \mod m)\rangle, \left\lfloor \frac{j+i}{m} \right\rfloor, \omega \right) \tag{15}$$

in $\mathcal{M}'$, where $q \in Q$, $\sigma \in \tilde{\Sigma}$, $\omega \in \Omega$, and $\alpha \in \mathbb{C}$ is the amplitude of the transition. Since φ is a bijection, the configuration matrix of $\mathcal{M}$ is isomorphic to that of $\mathcal{M}'$ for any input string $w \in \Sigma^*$. Therefore, they carry out exactly the same computation on a given input string, say $w \in \Sigma^*$ and so

$$f_{\mathcal{M}}(w) = f_{\mathcal{M}'}(w). \tag{16}$$

□

An r-reversal RT-kCA (Chan, 1981), denoted as r-rev-RT-kCA, is a RT-kCA where the number of alternations from increasing to decreasing and vice versa on each counter is restricted by r, where r is a nonnegative integer.

A RT-kCA with blind counters, denoted a RT-kBCA, is a RT-kCA that never checks the status of its counter(s), (and so the component Θ^k is completely removed from δ,) and accepts its input only if all counters are zero, and the processing of the input has ended in an accept state.

Fact 1. *(Freivalds, 1979) For every k, if L is recognized by a deterministic RT-kBCA (RT-DkBCA), then for every $\epsilon \in (0, \frac{1}{2})$, then there exists a probabilistic RT-1BCA (RT-P1BCA) recognizing L with negative one-sided error bound ϵ.*

We can generalize this result to probabilistic RT-kBCAs (RT-PkBCAs), where $k > 1$.

Lemma 3. *Let $\mathcal{P}$ be a given RT-PkBCA and $\epsilon \in (0, \frac{1}{2})$ be a given error bound. Then, there exists a RT-P1BCA(R) $\mathcal{P}'$ such that for all $w \in \Sigma^*$,*

$$f_{\mathcal{P}}(w) \leq f_{\mathcal{P}'}(w) \leq f_{\mathcal{P}}(w) + \epsilon(1 - f_{\mathcal{P}}(w)), \tag{17}$$

where $R = 2^{\lceil \frac{k}{\epsilon} \rceil}$.

Proof. Based on the probabilistic method described in Figure 2, we can obtain $\mathcal{P}'$ by making the following modifications on $\mathcal{P}$:

1. At the beginning of the computation, $\mathcal{P}'$ equiprobably chooses a number r from the set $\{1, \ldots, R\}$.

Figure 2: Probabilistic zero-checking of multiple counters by one counter

In this figure, we review a method presented by Freivalds in (Freivalds, 1979): Given a machine with $k > 1$ counters, say $C_1, \ldots, C_k$, whose values can be updated using the increment set $\{-1, 0, 1\}$, we can build a machine with a single counter, say C, whose value can be updated using the increment set $\{-R, \ldots, R\}$ ($R = 2^{\lceil \frac{k}{\epsilon} \rceil}$), such that all updates on $C_1, \ldots, C_k$ can be simulated on C in the sense that (i) if all values of $C_1, \ldots, C_k$ are zeros, then the value of C is zero; and (ii) if the value of at least one of $C_1, \ldots, C_k$ is nonzero, then the value of C is nonzero with probability $1 - \epsilon$, where $\epsilon \in (0, \frac{1}{2})$. The probabilistic method for this simulation is as follows:

- Choose a number r equiprobably from the set $\{1, \ldots, R\}$.
- The value of C is increased (resp., decreased) by r^i if the value of C_i is increased (resp., decreased) by 1.

2. For each transition of $\mathcal{P}$, in which the values of counters are updated by $(c_1, \ldots, c_k) \in \{-1, 0, 1\}^k$, i.e., the value of the i^{th} counter is updated by c_i ($1 \leq i \leq k$), $\mathcal{P}$ makes the same transition by updating its counter values by $\sum_{i=1}^{k} r^i c_i$.

Hence, (i) for each accepting path of $\mathcal{P}$, the input is accepted by $\mathcal{P}'$, too; (ii) for each rejecting path of $\mathcal{P}$, the input may be accepted by $\mathcal{P}'$ with a probability at most ϵ. By combining these cases, we obtain the following inequality for any input string $w \in \Sigma^*$:

$$f_{\mathcal{P}}(w) \leq f_{\mathcal{P}'}(w) \leq f_{\mathcal{P}}(w) + \epsilon(1 - f_{\mathcal{P}}(w)) \tag{18}$$

□

Theorem 1. *If L is recognized by a RT-PkBCA with error bound $\epsilon \in (0, \frac{1}{2})$, then L is recognized by a RT-P1BCA with error bound ϵ' ($0 < \epsilon < \epsilon' < \frac{1}{2}$). Moreover, ϵ' can be tuned to be arbitrarily close to ϵ.*

Proof. Let $\mathcal{P}$ be a RT-PkBCA recognizing L with error bound ϵ. By using the previous lemma (Lemma 3), for any $\epsilon'' \in (0, \frac{1}{2})$, we can construct a RT-P1BCA(R), say $\mathcal{P}''$, from $\mathcal{P}$, where $R = 2^{\lceil \frac{k}{\epsilon''} \rceil}$. Hence, depending on the value of ϵ, we can select ϵ'' to be sufficiently small such that L is recognized by $\mathcal{P}''$ with error bound $\epsilon' = \epsilon + \epsilon''(1-\epsilon) < \frac{1}{2}$. Since for each RT-P1BCA(m), there is an equivalent RT-P1BCA for any $m > 1$, L is also recognized by a RT-P1BCA with bounded error ϵ', which can be tuned to be arbitrarily close to ϵ. □

Corollary 1. *If L is recognized by a RT-PkBCA with negative one-sided error bound $\epsilon \in (0, 1)$, then L is recognized by a RT-P1BCA with negative one-sided error bound ϵ' ($0 < \epsilon < \epsilon'$). Moreover, ϵ' can be tuned to be arbitrarily close to ϵ.*

Models with write-only memory

We model a WOM as a two-way write-only tape having alphabet $\Gamma = \{\gamma_1, \ldots, \gamma_{|\Gamma|}\}$. Γ contains # and ε (*the empty string* or *the empty symbol*).

In a general two-way WOM, a transition "writing" ε on the tape causes the tape head to move in the specified direction without changing the symbol in the square under the original location of the head. If we restrict the tape head movement of a WOM to $\triangleright$, i.e. one-way, we obtain a "push-only stack" (POS). In the case of machines with POS, we assume that the write-only tape head does not move if a ε is written, and it moves one square to the right if a symbol different than ε is written. A special case of the POS setup is the "increment-only counter" (IOC) (Say et al., 2010), where Γ includes only ε and a single *counting* symbol (different than #).

For any standard machine model, say, M, we use the name M-WOM to denote M augmented with a WOM component. A TM-WOM, for instance, just has an additional write-only tape. The computational power of the PTM-WOM is easily seen to be the same as that of the PTM; since the machine does not use the contents of the WOM in any way when it decides what to do in the next move, and the probability distribution of the partial machine configurations (not including the WOM) is not dependent on the WOM content at any time, every write-only action can just as well be replaced with a write-nothing action. The following lemma shows this more formally for the models that will come under focus in this paper.

Lemma 4. *The computational power of any realtime classical finite automaton is unchanged when the model is augmented with a WOM.*

Proof. For a given machine $\mathcal{M}$ and an input string w, consider the tree $\mathcal{T}$ of states, where the root is the initial state, each subsequent level corresponds to the processing of the next input symbol, and the children of each node N are the states that have nonzero-probability transitions from S with the input symbol corresponding to that level. Each such edge in the tree is labeled with the corresponding transition probability. The probability of node N is the product of the probabilities on the path to N from the root. The acceptance probability is the sum of the probabilities of the accept states at the last level.

Now consider attaching a WOM to $\mathcal{M}$, and augmenting its program so that every transition now also specifies the action to be taken on the WOM. Several new transitions of this new machine may correspond to a single transition of $\mathcal{M}$, since, for example, a transition with probability p can be divided into two transitions with probability $\frac{p}{2}$, whose effects on the internal state are identical, but which write different symbols on the WOM. It is clear that many different programs can be obtained by augmenting $\mathcal{M}$ in this manner with different WOM actions. Visualize the configuration tree $\mathcal{T}_{new}$ of any one of these new machines on input w. There exists a homomorphism h from $\mathcal{T}_{new}$ to $\mathcal{T}$, where h maps nodes in $\mathcal{T}_{new}$ to nodes on the same level in $\mathcal{T}$, the configurations in $h^{-1}(N)$ all have N as their states, and the total probability of the members of $h^{-1}(N)$ equals the probability of N in $\mathcal{T}$, for any N. We conclude that all the machines with WOM accept w with exactly the same probability as w, so the WOM does not make any difference. □

As stated, our aim is to show that WOMs do increase the power of QTMs. We will focus on several variants of realtime quantum finite automata with WOM (RT-QFA-WOMs), which are just QTM-WOMs which do not use their work tapes, and move the input tape head to the right in every step.

More precisely, we will examine the power of RT-QFAs that are augmented with WOM, POS, or IOC, namely, the models RT-QFA-WOM, RT-QFA-POS, or RT-QFA-IOC (0-rev-RT-Q1CA), respectively. Note that RT-QFA-WOMs, RT-QFA-POSs, and RT-QFA-IOCs are special cases of quantum realtime Turing machines, pushdown automata, and one-counter automata, respectively.

Formally, a RT-QFA-WOM $\mathcal{M}$ is a 7-tuple

$$(Q, \Sigma, \Gamma, \Omega, \delta, q_1, Q_a). \tag{19}$$

When in state $q \in Q$ and reading symbol $\sigma \in \tilde{\Sigma}$ on the input tape, $\mathcal{M}$ changes its state to $q' \in Q$, writes $\gamma \in \Gamma$ and $\omega \in \Omega$ on the WOM tape and the finite register, respectively, and then updates the position of the WOM tape head with respect to $d_w \in <>$ with transition amplitude $\delta(q, \sigma, q', \gamma, d_w, \omega) = \alpha$, where $\alpha \in \mathbb{C}$ and $|\alpha| \leq 1$.

In order to represent all transitions from the case where $\mathcal{M}$ is in state $q \in Q$ and reading symbol $\sigma \in \tilde{\Sigma}$ together, we will use the notation

$$\delta(q, \sigma) = \sum_{(q', \gamma, d_w, \omega) \in Q \times \Gamma \times <> \times \Omega} \delta(q, \sigma, q', \gamma, d_w, \omega)(q', \gamma, d_w, \omega), \tag{20}$$

where

$$\sum_{(q', \gamma, d_w, \omega) \in Q \times \Gamma \times <> \times \Omega} |\delta(q, \sigma, q', \gamma, d_w, \omega)|^2 = 1. \tag{21}$$

A configuration of a RT-QFA-WOM is the collection of

- the internal state of the machine,
- the position of the input tape head,
- the contents of the WOM tape, and the position of the WOM tape head.

The formal definition of the RT-QFA-POS is similar to that of the RT-QFA-WOM, except that the movement of the WOM tape head is restricted to $\triangleright$, and so the position of that head does not need to be a part of a configuration. On the other hand, the definition of the RT-QFA-IOC can be simplified by removing the Γ component from (19):

A RT-QFA-IOC $\mathcal{M}$ is a 6-tuple

$$(Q, \Sigma, \Omega, \delta, q_1, Q_a). \tag{22}$$

When in state $q \in Q$, and reading symbol $\sigma \in \tilde{\Sigma}$ on the input tape, $\mathcal{M}$ changes its state to $q' \in Q$, writes ω in the register, and updates the value of its counter by $c \in \triangle = \{0, +1\}$ with transition amplitude $\delta(q, \sigma, q', c, \omega) = \alpha$, where $\alpha \in \mathbb{C}$ and $|\alpha| \leq 1$.

In order to show all transitions from the case where $\mathcal{M}$ is in state $q \in Q$ and reads symbol $\sigma \in \tilde{\Sigma}$ together, we use the notation

$$\delta(q, \sigma) = \sum_{(q', c, \omega) \in Q \times \triangle \times \Omega} \delta(q, \sigma, q', c, \omega)(q', c, \omega), \tag{23}$$

where

$$\sum_{(q', c, \omega) \in Q \times \triangle \times \Omega} |\delta(q, \sigma, q', c, \omega)|^2 = 1. \tag{24}$$

A configuration of a RT-QFA-IOC is the collection of

- the internal state of the machine,
- the position of the input tape head, and
- the value of the counter.

Increment-only counter machines

We are ready to start our demonstration of the superiority of quantum computers with WOM over those without WOM. We will examine the capabilities of RT-QFA-IOCs in both the bounded and unbounded error settings, and show that they can simulate a family of conventional counter machines, which are themselves superior to RT-QFAs, in both these cases.

Bounded error

The main theorem to be proven in this subsection is

Theorem 2. *The class of languages recognized with bounded error by RT-QFA-IOCs contains all languages recognized with bounded error by conventional realtime quantum automata with one blind counter (RT-Q1BCAs).*

Before presenting our proof of Theorem 2, let us demonstrate the underlying idea by showing how RT-QFA-IOCs can simulate a simpler family of machines, namely, deterministic automata with one blind counter. Define a RT-QFA-IOC(m) as a RT-QFA-IOC with the capability of incrementing its counter by any value from the set $\{0, \ldots, m\}$, where $m > 1$.

Lemma 5. *If a language L is recognized by a RT-D1BCA, then L can also be recognized by a RT-QFA-IOC with negative one-sided error bound $\frac{1}{m}$, for any desired value of m.*

Proof. We will build a RT-QFA-IOC(m) that recognizes L, which is sufficient by Lemma 2.

Throughout this proof, the symbol "i" is reserved for the imaginary number $\sqrt{-1}$. Let the given RT-D1BCA be $\mathcal{D} = (Q, \Sigma, \delta, q_1, Q_a)$, where $Q = \{q_1, \ldots, q_n\}$. We build $\mathcal{M} = (Q', \Sigma, \Omega, \delta', q_{1,1}, Q'_a)$, where

- $Q' = \{q_{j,1}, \ldots, q_{j,n} \mid 1 \leq j \leq m\}$,
- $Q'_a = \{q_{m,i} \mid q_i \in Q_a\}$, and
- $\Omega = \{\omega_1, \ldots, \omega_n\}$.

$\mathcal{M}$ splits the computation into m paths, i.e. path_j $(1 \leq j \leq m)$, with equal amplitude on the left end-marker ¢. That is,

$$\delta'(q_{1,1}, ¢) = \underbrace{\frac{1}{\sqrt{m}}(q_{1,t}, 0, \omega_1)}_{\mathsf{path}_1} + \cdots + \underbrace{\frac{1}{\sqrt{m}}(q_{m,t}, 0, \omega_1)}_{\mathsf{path}_m}, \tag{25}$$

whenever $\delta(q_1, ¢, q_t, 0) = 1$, where $1 \leq t \leq n$. Until reading the right end-marker \$, path_j proceeds in the following way: For each $\sigma \in \Sigma$ and $s \in \{1, \ldots, n\}$,

$$\mathsf{path}_j : \delta'(q_{j,s}, \sigma) \;\; = \;\; (q_{j,t}, c_j, \omega_s) \tag{26}$$

whenever $\delta(q_s, \sigma, q_t, c) = 1$, where $1 \leq t \leq n$, and

- $c_j = j$ if $c = 1$,
- $c_j = m - j + 1$ if $c = -1$, and
- $c_j = 0$, otherwise.

To paraphrase, each path separately simulates[5] the computation of $\mathcal{D}$ on the input string, going through states that correspond to the states of $\mathcal{D}$, and incrementing their counters whenever $\mathcal{D}$ changes its counter, as follows:

- path_j increments the counter by j whenever $\mathcal{D}$ increments the counter by 1,
- path_j increments the counter by $m - j + 1$ whenever $\mathcal{D}$ decrements the counter by 1, and
- path_j does not make any incrementation, otherwise.

On symbol \$, the following transitions are executed (note that the counter updates in this last step are also made according to the setup described above):
If $q_t \in Q_a$,

$$\mathsf{path}_j : \delta'(q_{j,s}, \$) = \frac{1}{\sqrt{m}} \sum_{l=1}^{m} e^{\frac{2\pi i}{m} jl} (q_{l,t}, c_j, \omega_s) \tag{27}$$

and if $q_t \notin Q_a$,

$$\mathsf{path}_j : \delta'(q_{j,s}, \$) = (q_{j,t}, c_j, \omega_s), \tag{28}$$

whenever $\delta(q_s, \$, q_t, c) = 1$, where $1 \leq t \leq n$.

The essential idea behind this setup, where different paths increment their counters with different values to represent increments and decrements performed by $\mathcal{D}$ is that the increment values used by $\mathcal{M}$ have been selected carefully to ensure that the counter will have the same value in all of $\mathcal{M}$'s paths at any time if $\mathcal{D}$'s counter is zero at that time. Furthermore, all of $\mathcal{M}$'s paths are guaranteed to have different counter values if $\mathcal{D}$'s counter is nonzero[6].

For a given input string $w \in \Sigma^*$,

1. if $\mathcal{D}$ ends up in a state not in Q_a (and so $w \notin L$), then $\mathcal{M}$ rejects the input in each of its m paths, and the overall rejection probability is 1;
2. if $\mathcal{D}$ ends up in a state in Q_a, all paths make an m-way QFT (see Figure 3) whose distinguished target is an accepting state:

Figure 3: N-way quantum Fourier transform

Let $N > 1$ be a integer. The N-way QFT is the transformation

$$\delta(d_j) \rightarrow \alpha \sum_{l=1}^{N} e^{\frac{2\pi i}{N} jl}(r_l), \quad 1 \leq j \leq N, \tag{29}$$

from the *domain* states $d_1, \ldots, d_N$ to the *range* states $r_1, \ldots, r_N$. r_N is the *distinguished* range element. α is a real number such that $\alpha^2 N \leq 1$. The QFT can be used to check whether separate computational paths of a quantum program that are in superposition have converged to the same configuration at a particular step. Assume that the program has previously split to N paths, each of which have the same amplitude, and whose state components are the d_j). In all the uses of the QFT in our algorithms, one of the following conditions will be satisfied:

1. The WOM component of the configuration is different in each of the N paths: In this case, the QFT will further divide each path to N subpaths, that will differ from each other by the internal state component. No interference will take place.
2. Each path has the same WOM content at the moment of the QFT: In this case, the paths that have $r_1, \ldots, r_{N-1}$ as their state components will destructively interfere with each other (Yakaryılmaz and Say, 2009), and $\alpha^2 N$ of the probability of the N incoming paths will be accumulated on a single resulting path with that WOM content, and r_N as its state component.

(a) if the counter of $\mathcal{D}$ is zero (and so $w \in L$), all paths have the same counter value, that is, they will interfere with each other, and so $\mathcal{M}$ will accept with probability 1;

(b) if the counter of $\mathcal{D}$ is not zero (and so $w \notin L$), there will be no interference, and each path will end by accepting w with probability $\frac{1}{m^2}$, leading to a total acceptance probability of $\frac{1}{m}$, and a rejection probability of $1 - \frac{1}{m}$.

□

Proof of Theorem 2. Given a RT-Q1BCA that recognizes a language L $\mathcal{M}$ with error bound $\epsilon < \frac{1}{2}$, we build a RT-QFA-IOC(m) $\mathcal{M}'$, using essentially the same construction as in Lemma 5: $\mathcal{M}'$ simulates m copies of $\mathcal{M}$, and these copies use the set of increment sizes described in the proof of Lemma 5 to mimic the updates to $\mathcal{M}$'s counter. Unlike the deterministic machine of that lemma, $\mathcal{M}$ can fork to multiple computational paths, which is handled by modifying the transformation of Equation 26 as

$$\mathsf{path}_j : \delta'(q_{j,s}, \sigma, q_{j,t}, c_j, \omega) = \alpha \tag{30}$$

whenever $\delta(q_s, \sigma, q_t, c, \omega) = \alpha$, where $1 \leq t \leq n$, and $\omega \in \Omega$, and that of Equation 27 as

$$\mathsf{path}_j : \delta'(q_{j,s}, \$, q_{l,t}, c_j, \omega) = \frac{\alpha}{\sqrt{m}} e^{\frac{2\pi i}{m} jl}, \text{ for } l \in \{1, \ldots, m\} \tag{31}$$

whenever $\delta(q_s, \$, q_t, c, \omega) = \alpha$, where $1 \leq t \leq n$ and $\omega \in \Omega$; causing the corresponding paths of the m copies of $\mathcal{M}$ to undergo the m-way QFTs associated by each accept state as described above at the end of the input.

We therefore have that the paths of $\mathcal{M}$ that end in non-accept states do the same thing with the same total probability in $\mathcal{M}'$. The paths of $\mathcal{M}$ that end in accept states with the counter containing zero make $\mathcal{M}'$ accept also with their original total probability, thanks to the QFT. The only mismatch between the machines is in the remaining case of the paths of $\mathcal{M}$ that end in accept states with a nonzero counter value. As explained in the proof of Lemma 5, each such path will contribute $\frac{1}{m}$ of its probability to acceptance, and the rest to rejection.

For any given input string $w \in \Sigma^*$:

- If $w \in L$, we have $f^a_{\mathcal{M}}(w) \geq 1 - \epsilon$ and $f^r_{\mathcal{M}}(w) \leq \epsilon$, then

$$f^a_{\mathcal{M}'}(w) = f^a_{\mathcal{M}}(w) + \frac{1}{m} f^r_{\mathcal{M}}(w) \geq 1 - \epsilon. \tag{32}$$

- If $w \notin L$, we have $f^a_{\mathcal{M}}(w) \leq \epsilon$ and $f^r_{\mathcal{M}}(w) \geq 1 - \epsilon$, then

$$f^a_{\mathcal{M}'}(w) = f^a_{\mathcal{M}}(w) + \frac{1}{m} f^r_{\mathcal{M}}(w) \leq \epsilon + \frac{1}{m}(1 - \epsilon). \tag{33}$$

Therefore, by setting m to a value greater than $\frac{2-2\epsilon}{1-2\epsilon}$, L will be recognized by $\mathcal{M}'$ with error bound $\epsilon' = \epsilon + \frac{1}{m}(1 - \epsilon) < \frac{1}{2}$. Moreover, by setting m to sufficiently large values, ϵ' can be tuned to be arbitrarily close to ϵ. □

Corollary 2. *If L is recognized by a RT-Q1BCA (or a RT-P1BCA) $\mathcal{P}$ with negative one-sided error bound $\epsilon < 1$, then L is recognized by a RT-QFA-IOC $\mathcal{M}$ with negative one-sided error bound ϵ', i.e. $\epsilon < \epsilon' < 1$. Moreover, ϵ' can be tuned to be arbitrarily close to ϵ.*

For a given nonnegative integer k, L_{eq-k} is the language defined over the alphabet $\{a_1, \ldots, a_k, b_1, \ldots, b_k\}$ as the set of all strings containing equal numbers of a_i's and b_i's, for each $i \in \{1, \ldots, k\}$.

Fact 2. *(Freivalds, 1979) For any nonnegative k, L_{eq-k} can be recognized by a RT-P1BCA with negative one-sided bounded error ϵ, where $\epsilon < \frac{1}{2}$.*

Corollary 3. *RT-QFA-IOCs can recognize some non-context-free languages with bounded error.*

We have therefore established that realtime quantum finite automata equipped with a WOM tape are more powerful than plain RT-QFAs, even when the WOM in question is restricted to be just a counter.

L_{eq-1}'s complement, which can of course be recognized with positive one-sided bounded error by a RT-QFA-IOC by the results above, is a deterministic context-free language (DCFL). Using the fact (Alt et al., 1992) that no nonregular DCFL can be recognized by a nondeterministic TM using $o(\log(n))$ space, together with Lemma 1, we are able to conclude the following.

Corollary 4. *QTM-WOMs are strictly superior to PTM-WOMs for any space bound $o(\log(n))$ in terms of language recognition with positive one-sided bounded error.*

Unbounded error

The simulation method introduced in Lemma 5 turns out to be useful in the analysis of the power of increment-only counter machines in the unbounded error mode as well:

Theorem 3. *Any language recognized by a nondeterministic realtime automaton with one blind counter (RT-NQ1BCA) is recognized by a RT-QFA-IOC with cutpoint $\frac{1}{2}$.*

Proof. Given a RT-NQ1BCA $\mathcal{N}$, we note that it is just a RT-Q1BCA recognizing a language L with positive one-sided unbounded error (Yakaryılmaz and Say, 2010a), and we can simulate it using the technique described in the proof of Theorem 2. We set m, the number of copies of the RT-Q1BCA to be parallelly simulated, to 2. We obtain a RT-QFA-IOC(2) $\mathcal{M}$ such that

1. paths of $\mathcal{N}$ that end in an accepting state with the counter equaling zero lead $\mathcal{M}$ to accept with the same total probability;

2. paths of $\mathcal{N}$ that end in an accepting state with a nonzero counter value contribute half of their probability to $\mathcal{M}$'s acceptance probability, with the other half contributing to rejection; and

3. paths of $\mathcal{N}$ that end in a reject state cause $\mathcal{M}$ to reject with the same total probability.

Finally, we modify the transitions on the right end-marker that enter the reject states mentioned in the third case above, so that they are replaced by equiprobable transitions to an (accept,reject) pair of states. The resulting machine recognizes L with "one-sided" cutpoint $\frac{1}{2}$, that is, the overall acceptance probability exceeds $\frac{1}{2}$ for the members of the language, and equals $\frac{1}{2}$ for the nonmembers. □

We now present a simulation of a classical model with non-blind counter.

Theorem 4. *If L is recognized by a realtime deterministic one-reversal one-counter automaton (1-rev-RT-D1CA), then it is recognized by a RT-QFA-IOC with cutpoint $\frac{1}{2}$.*

Proof. We assume that the 1-rev-RT-D1CA $\mathcal{D} = (Q, \Sigma, \delta, q_1, Q_a)$ recognizing L is in the following canonical form:

- the counter value of $\mathcal{D}$ never becomes nonnegative;
- the transition on ¢ does not make any change ($\delta(q_1, ¢, 0, q_1) = 1$, and $D_c(q_1) = 0$);
- Q is the union of two disjoint subsets Q_1 and Q_2, i.e.
  1. until the first decrement, the status of the counter is never checked – this part is implemented by the members of Q_1,
  2. during the first decrement, the internal state of $\mathcal{D}$ switches to one of the members of Q_2, and
  3. the computation after the first decrement is implemented by the members of Q_2;
- once the counter value is detected as zero, the status of the counter is not checked again.

We will construct a RT-QFA-IOC $\mathcal{M} = (Q', \Sigma, \Omega, \delta', q_1, Q'_a)$, to recognize L with cutpoint $\frac{1}{2}$, where

- $Q' = \{q_1\} \cup \{q_{j,i} \mid j \in \{1, \ldots, 4\}, i \in \{1, \ldots, |Q|\}\}$,
- $Q'_a = \{q_{j,i} \mid j \in \{1, 2, 3\}, q_i \in Q_a\} \cup \{q_{4,i} \mid q_i \in Q_r\}$, and
- $\Omega = \{\omega_i \cup \omega'_i \mid i \in \{1, \ldots, |Q|\}\}$.

Figure 4: The details of the transition function of the RT-QFA-IOC presented in the proof of Theorem 4 (I)

In the following, "$*$" means that the corresponding transition does not depend on the status of the counter.

(i) On symbol ¢:

$$\delta'(q_1, ¢) = \underbrace{\frac{1}{\sqrt{2}}(q_{1,1}, 0, \omega_1)}_{\mathsf{path}_1} + \underbrace{\frac{1}{\sqrt{2}}(q_{2,1}, 0, \omega_1)}_{\mathsf{path}_2} \tag{34}$$

(ii) On symbol $\sigma \in \Sigma$: for each $q_i \in Q_1$, if $\delta(q_i, \sigma, *, q_j) = 1$ and $q_j \in Q_1$, then

$$\begin{aligned} \mathsf{path}_1 : \delta'(q_{1,i}, \sigma) &= \underbrace{(q_{1,j}, D_c(q_j), \omega_i)}_{\mathsf{path}_1} \\ \mathsf{path}_2 : \delta'(q_{2,i}, \sigma) &= \underbrace{(q_{2,j}, 0, \omega_i)}_{\mathsf{path}_2} \end{aligned} . \tag{35}$$

(iii) On symbol \$: for each $q_i \in Q_1$, if $\delta(q_i, \$, *, q_j) = 1$ and $q_j \in Q_1$, then

$$\begin{aligned} \mathsf{path}_1 : \delta'(q_{1,i}, \sigma) &= \underbrace{(q_{1,j}, 0, \omega_i)}_{\mathsf{path}_1} \\ \mathsf{path}_2 : \delta'(q_{2,i}, \sigma) &= \underbrace{(q_{2,j}, 0, \omega_i)}_{\mathsf{path}_2} \end{aligned} . \tag{36}$$

and the details of δ' are given in Figures 4 and 5.

$\mathcal{M}$ starts by branching to two paths, path_1 and path_2, with equal amplitude. These paths simulate $\mathcal{D}$ in parallel according to the specifications in Figure 4 until $\mathcal{D}$ decrements its counter for the first time. From that step on, path_1 and path_2 split further to create new offshoots (called path_3 and path_4,) on every symbol until the end of the computation, as seen in Figure 5. Throughout the computation, path_1 (resp., path_2) increments its counter whenever $\mathcal{D}$ is supposed to increment (resp., decrement) its counter. Since $\mathcal{M}$'s counter is write-only, it has no way of determining which transition $\mathcal{D}$ will make depending on its counter sign. This problem is solved by assigning different paths of $\mathcal{M}$ to these branchings of $\mathcal{D}$: path_1 and path_2 (the "pre-zero paths") always assume that $\mathcal{D}$'s counter has not returned to zero yet by being decremented, whereas path_3s and path_4s (the "post-zero paths") carry out their simulations by assuming otherwise. Except for path_4s, all paths imitate $\mathcal{D}$'s decision at the end of the computation. path_4s, on the other hand, accept if and only if their simulation of $\mathcal{D}$ rejects the input.

If $\mathcal{D}$ never decrements its counter, $\mathcal{M}$ ends up with the same decision as $\mathcal{D}$ with probability 1. We now focus on the other cases. As seen in Figure 5, the pre-zero paths lose some of their amplitude on each symbol in this

Figure 5: The details of the transition function of the RT-QFA-IOC presented in the proof of Theorem 4 (II)

(iv) On symbol $\sigma \in \Sigma$: for each $q_i \in Q$, if $\delta(q_i, \sigma, 1, q_j) = 1$ and $q_j \in Q_2$, then

$$\begin{aligned} \mathsf{path}_1 : \delta'(q_{1,i}, \sigma) &= \underbrace{\frac{1}{\sqrt{3}}(q_{1,j}, 0, \omega_i)}_{\mathsf{path}_1} + \underbrace{\frac{1}{\sqrt{3}}(q_{3,j}, 0, \omega_i')}_{\mathsf{path}_3} + \underbrace{\frac{1}{\sqrt{3}}(q_{4,j}, 0, \omega_i')}_{\mathsf{path}_4} \\ \mathsf{path}_2 : \delta'(q_{2,i}, \sigma) &= \underbrace{\frac{1}{\sqrt{3}}(q_{2,j}, c_2, \omega_i)}_{\mathsf{path}_2} + \underbrace{\frac{1}{\sqrt{3}}(q_{3,j}, c_2, \omega_i')}_{\mathsf{path}_3} - \underbrace{\frac{1}{\sqrt{3}}(q_{4,j}, c_2, \omega_i')}_{\mathsf{path}_4} \end{aligned}, \quad (37)$$

and if $\delta(q_i, \sigma, 0, q_j) = 1$, then

$$\begin{aligned} \mathsf{path}_3 : \delta'(q_{3,i}, \sigma) &= \underbrace{(q_{3,j}, 0, \omega_i)}_{\mathsf{path}_3} \\ \mathsf{path}_4 : \delta'(q_{4,i}, \sigma) &= \underbrace{(q_{4,j}, 0, \omega_i)}_{\mathsf{path}_4} \end{aligned}, \quad (38)$$

where $c_2 = 1$ only if $D_c(q_j) = -1$.

(iii) On symbol \$: for each $q_i \in Q$, if $\delta(q_i, \$, 1, q_j) = 1$ and $q_j \in Q_2$, then

$$\begin{aligned} \mathsf{path}_1 : \delta'(q_{1,i}, \sigma) &= \underbrace{\frac{1}{\sqrt{3}}(q_{1,j}, 0, \omega_i)}_{\mathsf{path}_1} + \underbrace{\frac{1}{\sqrt{3}}(q_{3,j}, 0, \omega_i)}_{\mathsf{path}_3} + \underbrace{\frac{1}{\sqrt{3}}(q_{4,j}, 0, \omega_i)}_{\mathsf{path}_4} \\ \mathsf{path}_2 : \delta'(q_{2,i}, \sigma) &= \underbrace{\frac{1}{\sqrt{3}}(q_{2,j}, c_2, \omega_i)}_{\mathsf{path}_2} + \underbrace{\frac{1}{\sqrt{3}}(q_{3,j}, c_2, \omega_i)}_{\mathsf{path}_3} - \underbrace{\frac{1}{\sqrt{3}}(q_{4,j}, c_2, \omega_i)}_{\mathsf{path}_4} \end{aligned}, \quad (39)$$

and if $\delta(q_i, \$, 0, q_j) = 1$, then

$$\begin{aligned} \mathsf{path}_3 : \delta'(q_{3,i}, \sigma) &= \underbrace{(q_{3,j}, 0, \omega_i)}_{\mathsf{path}_3} \\ \mathsf{path}_4 : \delta'(q_{4,i}, \sigma) &= \underbrace{(q_{4,j}, 0, \omega_i)}_{\mathsf{path}_4} \end{aligned}, \quad (40)$$

where $c_2 = 1$ only if $D_c(q_j) = -1$.

stage by performing a QFT to a new pair of post-zero paths. The outcome of this transformation depends on the status of $\mathcal{D}$'s counter at this point in the simulation by the pre-zero paths:

- If $\mathcal{D}$'s counter has not yet returned to zero, then path_2's counter has a smaller value than path_1's counter, and so they cannot interfere via the QFT. The newly created post-zero paths will contribute equal amounts to the acceptance and rejection probabilities at the end of the compu-

tation.

- If path_1 and path_2 have the same counter value as a result of this transition, this indicates that $\mathcal{D}$ has performed exactly as many decrements as its previous increments, and its counter is therefore zero. The paths interfere, the target path_4's cancel each other, and path_3 survives after the QFT with a probability that is twice that of the total probability of the ongoing pre-zero paths.

As a result, it is guaranteed that the path that is carrying out the correct simulation of $\mathcal{D}$ will dominate $\mathcal{M}$'s decision at the end of the computation: If $\mathcal{D}$'s counter ever returns to zero, the path_3 that is created at the moment of that last decrement will have sufficient probability to tip the accept/reject balance. If $\mathcal{D}$'s counter never returns to zero, then the common decision by the pre-zero paths on the right end-marker will determine whether the overall acceptance or the rejection probability will be greater than $\frac{1}{2}$. □

Consider the following language (Nasu and Honda, 1971):

$$L_{NH} = \{a^x b a^{y_1} b a^{y_2} b \cdots a^{y_t} b \mid x, t, y_1, \cdots, y_t \in \mathbb{Z}^+ \text{ and } \exists k \ (1 \leq k \leq t), x = \sum_{i=1}^{k} y_i\} \quad (41)$$

L_{NH} is recognizable by both 1-rev-RT-D1CAs and RT-N1BCAs[7] (and so RT-NQ1BCAs). It is known (Nasu and Honda, 1971; Freivalds and Karpinski, 1994; Li and Qiu, 2008; Yakaryılmaz and Say, 2010b) that neither a RT-QFA nor a $o(\log(\log(n)))$-space PTM can recognize L_{NH} with unbounded error. We therefore have the following corollary.

Corollary 5. *QTM-WOMs are strictly superior to PTM-WOMs for any space bound $o(\log(\log(n)))$ in terms of language recognition with unbounded error.*

Machines with push-only stack

We conjecture that allowing more than one nonblank/nonempty symbol in the WOM tape alphabet of a QFA increases its computational power. We consider, in particular, the language $L_{twin} = \{wcw \mid w \in \{a, b\}^*\}$:

Theorem 5. *There exists a RT-QFA-POS that recognizes the language L_{twin} with negative one-sided error bound $\frac{1}{2}$.*

Figure 6: The transitions of the RT-QFA-POS of Theorem 5

On symbol ¢:

$$\delta(q_1, ¢) = \underbrace{\frac{1}{\sqrt{2}}(q_1, \varepsilon, \omega_1)}_{\mathsf{path}_1} + \underbrace{\frac{1}{\sqrt{2}}(p_1, \varepsilon, \omega_1)}_{\mathsf{path}_2} \tag{42}$$

On symbols from Σ:

$$\mathsf{path}_1 : \begin{cases} \delta(q_1, a) = (q_1, a, \omega_1) \\ \delta(q_2, a) = (q_2, \varepsilon, \omega_1) \\ \delta(q_1, b) = (q_1, b, \omega_1) \\ \delta(q_2, b) = (q_2, \varepsilon, \omega_1) \\ \delta(q_1, c) = (q_2, \varepsilon, \omega_1) \\ \delta(q_2, c) = (q_3, \varepsilon, \omega_1) \\ \delta(q_3, a) = (q_3, \varepsilon, \omega_2) \\ \delta(q_3, b) = (q_3, \varepsilon, \omega_2) \\ \delta(q_3, c) = (q_3, \varepsilon, \omega_2) \end{cases} \tag{43}$$

$$\mathsf{path}_2 : \begin{cases} \delta(p_1, a) = (p_1, \varepsilon, \omega_1) \\ \delta(p_2, a) = (p_2, a, \omega_1) \\ \delta(p_1, b) = (p_1, \varepsilon, \omega_1) \\ \delta(p_2, b) = (p_2, b, \omega_1) \\ \delta(p_1, c) = (p_2, \varepsilon, \omega_1) \\ \delta(p_2, c) = (p_3, \varepsilon, \omega_1) \\ \delta(p_3, a) = (p_3, \varepsilon, \omega_2) \\ \delta(p_3, b) = (p_3, \varepsilon, \omega_2) \\ \delta(p_3, c) = (p_3, \varepsilon, \omega_2) \end{cases} \tag{44}$$

On symbol \$:

$$\mathsf{path}_1 : \begin{cases} \delta(q_1, \$) = (q_1, \varepsilon, \omega_1) \\ \delta(q_2, \$) = \frac{1}{\sqrt{2}}(q_2, \varepsilon, \omega_1) + \frac{1}{\sqrt{2}}(q_3, \varepsilon, \omega_2) \\ \delta(q_3, \$) = (q_3, \varepsilon, \omega_1) \end{cases} \tag{45}$$

$$\mathsf{path}_2 : \begin{cases} \delta(p_1, \$) = (p_1, \varepsilon, \omega_1) \\ \delta(p_2, \$) = \frac{1}{\sqrt{2}}(q_2, \varepsilon, \omega_1) - \frac{1}{\sqrt{2}}(q_3, \varepsilon, \omega_2) \\ \delta(p_3, \$) = (q_3, \varepsilon, \omega_1) \end{cases} \tag{46}$$

Proof. We construct a RT-QFA-POS $\mathcal{M} = (Q, \Sigma, \Gamma, \Omega, \delta, q_1, Q_a)$, where $Q = \{q_1, q_2, q_3, p_1, p_2, p_3\}$, $Q_a = \{q_2\}$, $\Omega = \{\omega_1, \omega_2\}$, and $\Gamma = \{\#, a, b, \varepsilon\}$. The transition details are shown in Figure 6.

1. The computation splits into two paths, path_1 and path_2, with equal amplitude at the beginning.

2. path_1 (resp., path_2) scans the input, and copies w_1 (resp., w_2) to the POS if the input is of the form $w_1 c w_2$, where $w_1, w_2 \in \{a, b\}^*$.

   (a) If the input is not of the form $w_1 c w_2$, both paths reject.

   (b) Otherwise, path_1 and path_2 perform a QFT at the end of the computation, where the distinguished range element is an accept state.

The configurations at the ends of path_1 and path_2 interfere with each other, i.e., the machine accepts with probability 1, if and only if the input is of the form wcw, $w \in \{a, b\}^*$. Otherwise, each of path_1 and path_2 contributes at most $\frac{1}{4}$ to the overall acceptance probability, and the machine accepts with probability at most $\frac{1}{2}$. □

Lemma 6. *No PTM (or PTM-WOM) using $o(\log(n))$ space can recognize L_{twin} with bounded error.*

Proof. Any PTM using $o(\log(n))$ space to recognize L_{twin} with bounded error can be used to construct a PTM recognizing the palindrome language L_{pal} with bounded error using the same amount of space. (One would only need to modify the L_{twin} machine to treat the right end-marker on the tape as the symbol c, and switch its head direction when it attempts to go past that symbol.) It is however known (Freivalds and Karpinski, 1994) that no PTM using $o(\log(n))$ space can recognize L_{pal} with bounded error. □

We are now able to state a stronger form of Corollary 4, which referred only to one-sided error:

Corollary 6. *QTM-WOMs are strictly superior to PTM-WOMs for any space bound $o(log(n))$ in terms of language recognition with bounded error.*

Open Problem 1. *Can a probabilistic pushdown automaton recognize L_{twin} with bounded error?*

Machines using two-way WOM tape

In this section, we present a bounded-error RT-QFA-WOM that recognizes a language for which we currently do not know a RT-QFA-POS algorithm, namely,

$$L_{rev} = \{wcw^{\mathrm{r}} \mid w \in \{a, b\}^*\}, \tag{47}$$

where w^{r} is the reverse of string w. Note that this language can also be recognized by a deterministic pushdown automaton.

Theorem 6. *There exists a RT-QFA-WOM that recognizes L_{rev} with negative one-sided error bound $\frac{1}{2}$.*

Proof. (sketch) We will use almost the same technique presented in the proof of Theorem 5. The computation is split into two paths (path_1 and path_2) with equal amplitude at the beginning of the computation. Each path checks whether the input string is of the form $w_1 c w_2$, where $w_1, w_2 \in \{a, b\}^*$ and rejects with probability 1 if it is not. We assume that the input string is of the form $w_1 c w_2$ in the rest of this proof. Until the c is read, path_1 copies w_1 to the WOM tape, and path_2 just moves the WOM tape head one square to the right at each step, without writing anything. After reading the c, the direction of the WOM tape head is reversed in both paths. That is, path_1 moves the WOM tape head one square to the left at each step, without writing anything, while path_2 writes w_2 in the reverse direction (from the right to the left) on the WOM tape. When the right end-marker is read, the paths make a QFT, as in the proof of Theorem 5. It is easy to see that the two paths interfere if and only if $w_1 = w_2^{\mathrm{r}}$, and the input string is accepted with probability 1 if it is a member of L_{rev}, and with probability $\frac{1}{2}$ otherwise. □

By an argument similar to the one used in the proof of Lemma 6, L_{rev} can not be recognized with bounded error by any PTM using $o(\log(n))$ space, since the existence of any such machine would lead to a PTM that recognizes the palindrome language using the same amount of space.

Open Problem 2. *Can a RT-QFA-POS recognize L_{rev} with bounded error?*

Small amounts of WOM can be useful

It is easy to see that a WOM of constant size adds no power to a conventional machine. All the algorithms we considered until now used $\Omega(n)$ squares of the WOM tape on worst-case inputs. What is the minimum amount of WOM that is required by a QFA-WOM recognizing a nonregular language? Somewhat less ambitiously, one can ask whether there is any nonregular language recognized by a RT-QFA-WOM with sublinear space. We answer this question positively for middle-space usage (Szepietowski, 1994), that is, when we are only concerned with the space used by the machine when the input is a member of the language.

Let $(i)_2^{\mathrm{r}}$ be the reverse of the binary representation of $i \in \mathbb{N}$. Consider the language

$$L_{rev-bins} = \{a(0)_2^{\mathrm{r}}a(1)_2^{\mathrm{r}}a \cdots a(k)_2^{\mathrm{r}}a \mid k \in \mathbb{Z}^+\}. \tag{48}$$

Theorem 7. *$L_{rev-bins}$ can be recognized by a RT-QFA-WOM $\mathcal{M}$ with negative one-sided error bound $\frac{3}{4}$, and the WOM usage of $\mathcal{M}$ for the members of $L_{rev-bins}$ is $O(\log n)$, where n is the length of the input string.*

Proof. It is not hard to modify the RT-QFA-POS recognizing L_{twin} to obtain a new RT-QFA-POS, say $\mathcal{M}'$, in order to recognize language $L_{twin'} = \{(i)_2^{\mathrm{r}}a(i+1)_2^{\mathrm{r}} \mid i \geq 0\}$ with negative one-sided error bound $\frac{1}{2}$. Our construction of $\mathcal{M}$ will be based on $\mathcal{M}'$. The main idea is to use $\mathcal{M}'$ in a loop in order to check the consecutive blocks of $\{0,1\}^+a\{0,1\}^+$ between two a's. In each iteration, the WOM tape head reverses direction, and so the previously used space can be used again and again. Note that, whenever $\mathcal{M}'$ executes a rejecting transition, $\mathcal{M}$ enters a path which will reject the input when it arrives at the right end-marker, and whenever $\mathcal{M}'$ is supposed to execute an accepting transition (except at the end of the computation), $\mathcal{M}$ enters the next iteration. At the end of the input, the input is accepted by $\mathcal{M}$ if $\mathcal{M}'$ accepts in its last iteration.

Let w be an input string. We assume that w is of the form

$$a\{0,1\}^+a\{0,1\}^+a \cdots a\{0,1\}^+a. \tag{49}$$

(Otherwise, it is rejected with probability 1.) At the beginning, the computation is split equiprobably into two branches, branch_1 and branch_2. (These will never interfere with each other.) branch_1 (resp., branch_2) enters the block-checking loop after reading the first (resp., the second) a. Thus, at the end of the computation, one of the branches is in the middle of an iteration, and the other one has just finished its final iteration. The branch whose iteration is interrupted by reading the end-marker accepts with probability 1.

If $w \in L_{rev-bins}$, neither branch enters a reject state, and the input is accepted with probability 1. On the other hand, if $w \notin L_{rev-bins}$, there must be at least one block $\{0,1\}^+a\{0,1\}^+$ that is not a member of $L_{twin'}$, and so the input is rejected with probability $\frac{1}{2}$ in one branch. Therefore, the overall accepting probability can be at most $\frac{3}{4}$.

It is easy to see that the WOM usage of this algorithm for members of $L_{rev-bins}$ is $O(\log n)$. □

Conclusion

In this paper, we showed that write-only memory devices can increase the computational power of quantum computers. We considered quantum finite automata augmented with WOMs, and demonstrated several example languages which are known to be unrecognizable by conventional quantum computers with certain restrictions, but are recognizable by a quantum computer employing a WOM under the same restrictions. The QFA-WOM models under consideration were also shown to be able to simulate certain classical machines that employ linear amounts of memory, and are therefore much more powerful than finite automata. We also showed that merely logarithmic amounts of WOM can be useful in the sense of enabling the recognition of nonregular languages.

A close examination of our algorithms reveals that quantum computers using WOM are able to avoid the argument in the proof of Lemma 4 thanks to their use of negative (and complex) transition amplitudes in the QFT, which enables two configurations with the same WOM value to cancel each other altogether, when they have suitable amplitudes.

If one changes the RT-QFA-POS model so that the POS is now an output tape, the machine described in Theorem 5 becomes a realtime quantum finite state transducer (RT-QFST) (Freivalds and Winter, 2001) computing the function (Say and Yakaryılmaz, 2010)

$$f(x) = \begin{cases} w, & \text{if } x = wcw, \text{ where } w \in \{a,b\}^* \\ \text{undefined}, & \text{otherwise} \end{cases}, \tag{50}$$

with bounded error. The arguments leading to Corollary 6 can then be rephrased in a straightforward way to show that conventional QTMs are strictly superior to PTMs in function computation for any common space bound that is $o(\log(n))$.

Finally, assume that we make another change to the RT-QFST described in the paragraph above, so that it prints the symbol c when it is about to accept. The resulting constant-space QTM is easily seen to be computing a reduction from L_{twin} to the language $L_1 = \{\{a,b\}^*c\}$ with bounded error. But no PTM $\mathcal{P}$ using $o(\log n)$ space can compute this reduction, since we could otherwise build a PTM for deciding L_{twin} with the same error bound by composing $\mathcal{P}$ with the finite automaton recognizing L_1. The detailed examination of how and to what extent quantum reductions outperform probabilistic reductions with common restrictions is an interesting topic.

Note that it is already known that adding a WOM to a reversible classical computer may increase its computational power, since it enables one to embed irreversible tasks into "larger" reversible tasks by using the WOM

as a trashcan. As a simple example, reversible finite automata (RFAs) can recognize a proper subset of regular languages (Pin, 1987), but RFA's with WOM can recognize exactly the regular languages, and nothing more. In the quantum case, WOM can also have a similar effect. For example, the computational power of the most restricted type of quantum finite automata (MCQFAs) (Moore and Crutchfield, 2000) is equal to RFAs, but it has been shown (Paschen, 2000; Ciamarra, 2001) that MCQFAs with WOM can recognize all and only the regular languages, attaining the power of the most general quantum finite automata (QFA) without WOM. In all these examples, the addition of WOM to a specifically weak model raises it to the level of the most general classical (deterministic) automaton. On the other hand, in this work, we show that adding WOM to the most general type of QFA results in a much more powerful model that can achieve a task that is impossible for all sublogarithmic space PTMs.

Some remaining open problems related to this study can be listed as follows:

1. Does a WOM add any power to quantum computers which are allowed to operate at logarithmic or even greater space bounds?

2. How would having several separate WOMs, each of which would contain different strings, affect the performance?

3. Is there a nontrivial lower bound to the amount of WOM that is useful for the recognition of nonregular languages by QFA-WOMs?

Acknowledgements

Yakaryılmaz and Say were partially supported by the Scientific and Technological Research Council of Turkey (TÜBİTAK) with grant 108142. Freivalds and Agadzanyan were partially supported by Grant No. 09.1570 from the Latvian Council of Science and by Project 2009/0216/1DP/1.1.2.1.2/09/IPIA /VIA/004 from the European Social Fund.

Notes

[1]The QTM model is appropriate for studying the effect of space bounds on computational power. See (Yao, 1993) for an alternative model of quantum computation.

[2]Unlike (Watrous, 1998), we also allow efficiently computable irrational numbers as transition amplitudes in our QTM's. This simplifies the description of some algorithms in the remainder of this paper.

[3]The reader may find it useful to consult the descriptions of the common notational items in the discussion, given immediately after the introduction.

[4]Note that our definition of quantum counter automata is more general than the previous ones, (Kravtsev, 1999; Bonner et al., 2001; Yamasaki et al., 2002, 2005) since it is based on general quantum operators.

[5]Note that each transition of $\mathcal{M}$ in Equation 26 writes a symbol determined by the source state of the corresponding transition of $\mathcal{D}$ to the register. This ensures the orthonormality condition for quantum machines described earlier.

[6]This idea has been adapted from an algorithm by Kondacs and Watrous for a different type of quantum automaton, whose analysis can be found in (Kondacs and Watrous, 1997).

[7]RT-N1BCAs can also recognize $L_{center} = \{ubv \mid u, v \in \{a, b\}^*, |u| = |v|\}$, and the languages studied in (Freivalds et al., 2010), none of which can be recognized by RT-QFAs with unbounded error.

References

D. Aharonov, A. Kitaev, and N. Nisan. Quantum circuits with mixed states. In *STOC'98: Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing*, pages 20–30, 1998.

H. Alt, V. Geffert, and K. Mehlhorn. A lower bound for the nondeterministic space complexity of context-free recognition. *Information Processing Letters*, 42(1):25–27, 1992.

A. Ambainis and A. Yakaryılmaz. *Automata: from Mathematics to Applications*, chapter Automata and quantum computing. 2010. (In preparation).

S. Arora and B. Barak. *Computational Complexity: A Modern Approach*. Cambridge University Press, 2009.

V. D. Blondel, E. Jeandel, P. Koiran, and N. Portier. Decidable and undecidable problems about quantum automata. *SIAM Journal on Computing*, 34(6):1464–1473, 2005. ISSN 0097-5397.

R. Bonner, R. Freivalds, and M. Kravtsev. Quantum versus probabilistic one-way finite automata with counter. In *SOFSEM 2007: Theory and Practice of Computer Science*, volume 2234 of *Lecture Notes in Computer Science*, pages 181–190, 2001.

S. Bozapalidis. Extending stochastic and quantum functions. *Theory of Computing Systems*, 36(2):183–197, 2003.

T. Chan. Reversal complexity of counter machines. In *STOC'81: Proceedings of the thirteenth annual ACM symposium on Theory of computing*, pages 146–157, 1981.

M. P. Ciamarra. Quantum reversibility and a new model of quantum automaton. In *FCT'01: Proceedings of the 13th International Symposium on Fundamentals of Computation Theory*, pages 376–379. Springer-Verlag, 2001.

R. Freivalds. Fast probabilistic algorithms. In *Mathematical Foundations of Computer Science 1979*, volume 74 of *Lecture Notes in Computer Science*, pages 57–69, 1979.

R. Freivalds and M. Karpinski. Lower space bounds for randomized computation. In *ICALP'94: Proceedings of the 21st International Colloquium on Automata, Languages and Programming*, pages 580–592, 1994.

R. Freivalds and A. Winter. Quantum finite state transducers. In *SOFSEM 2001: Theory and Practice of Informatics*, pages 233–242, 2001.

R. Freivalds, A. Yakaryılmaz, and A. C. C. Say. A new family of nonstochastic languages. *Information Processing Letters*, 110(10):410–413, 2010.

M. Hirvensalo. Various aspects of finite quantum automata. In *DLT'08: Proceedings of the 12th international conference on Developments in Language Theory*, pages 21–33, 2008.

E. Jeandel. Topological automata. *Theory of Computing Systems*, 40(4): 397–407, 2007. ISSN 1432-4350.

A. Kondacs and J. Watrous. On the power of quantum finite state automata. In *FOCS'97: Proceedings of the 38th Annual Symposium on Foundations of Computer Science*, pages 66–75, 1997.

M. Kravtsev. Quantum finite one-counter automata. In *SOFSEM'99: Theory and Practice of Computer Science*, volume 1725 of *Lecture Notes in Computer Science*, pages 431–440, 1999.

L. Li and D. Qiu. Determining the equivalence for one-way quantum finite automata. *Theoretical Computer Science*, 403(1):42–51, 2008. ISSN 0304-3975.

C. Moore and J. P. Crutchfield. Quantum automata and quantum grammars. *Theoretical Computer Science*, 237(1-2):275–306, 2000. ISSN 0304-3975.

M. Nasu and N. Honda. A context-free language which is not acceptable by a probabilistic automaton. *Information and Control*, 18(3):233–236, 1971.

M. A. Nielsen and I. L. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, 2000.

K. Paschen. Quantum finite automata using ancilla qubits. Technical report, University of Karlsruhe, 2000. Available at http://digbib.ubka.uni-karlsruhe.de/volltexte/1452000.

J.-È. Pin. On the language accepted by finite reversible automata. In *ICALP'87: Proceedings of the 14th International Colloquium, on Automata, Languages and Programming*, pages 237–249. Springer-Verlag, 1987.

A. C. C. Say and A. Yakaryılmaz. Quantum function computation using sublogarithmic space, 2010. (Poster presentation at QIP2010 - available at arXiv:1009.3124).

A. C. C. Say, A. Yakaryılmaz, and Ş. Yüzsever. Quantum one-way one-counter automata. In R. Freivalds, editor, *Randomized and quantum computation*, pages 25–34, 2010. Satellite workshop of MFCS and CSL 2010.

A. Szepietowski. *Turing Machines with Sublogarithmic Space*. Springer-Verlag, 1994.

J. Watrous. *Space-bounded quantum computation*. PhD thesis, University of Wisconsin - Madison, USA, 1998.

J. Watrous. On the complexity of simulating space-bounded quantum computations. *Computational Complexity*, 12(1-2):48–84, 2003. ISSN 1016-3328.

J. Watrous. Quantum computational complexity. In R. A. Meyers, editor, *Encyclopedia of Complexity and Systems Science*, pages 7174–7201. Springer, 2009.

A. Yakaryılmaz. *Classical and Quantum Computation with Small Space Bounds*. PhD thesis, Boğaziçi University, 2010. (Submitted).

A. Yakaryılmaz and A. C. C. Say. Efficient probability amplification in two-way quantum finite automata. *Theoretical Computer Science*, 410(20): 1932–1941, 2009.

A. Yakaryılmaz and A. C. C. Say. Languages recognized by nondeterministic quantum finite automata. *Quantum Information and Computation*, 10 (9&10):747–770, 2010a.

A. Yakaryılmaz and A. C. C. Say. Unbounded-error quantum computation with small space bounds. Technical Report arXiv:1007.3624, 2010b.

T. Yamasaki, H. Kobayashi, Y. Tokunaga, and H. Imai. One-way probabilistic reversible and quantum one-counter automata. *Theoretical Computer Science*, 289(2):963–976, 2002.

T. Yamasaki, H. Kobayashi, and H. Imai. Quantum versus deterministic counter automata. *Theoretical Computer Science*, 334(1-3):275–297, 2005.

A. C.-C. Yao. Quantum circuit complexity. In *SFCS'93: Proceedings of the 1993 IEEE 34th Annual Foundations of Computer Science*, pages 352–361, 1993.