Log-space counter is useful for unary languages by help of a constant-size quantum register

Abuzer Yakaryılmaz

University of Latvia, Faculty of Computing, Raina bulv. 19, Rīga, LV-1586, Latvia
abuzer@lu.lv

Abstract. The minimum amount of resources to recognize a non-regular language is a fundamental research topic in theoretical computer science which has been examined for different kinds of resources and many different models. In this note, we focus on unary languages and space complexity on counters. Our model is two-way one-counter automaton with quantum and classical states (2QCCA), which is a two-way finite automaton with one-counter (2DCA) augmented with a fixed size quantum register or a two-way finite automaton with quantum and classical states (2QCF A) augmented with a classical counter. It is known that any 2DCA using a sublinear space on its counter can recognize only regular languages [DG82]. In this note, we show that bounded-error 2QCCAs can recognize a non-regular unary language by using logarithmic space on its counters for the members. Note that it is still an open problem whether bounded-error 2QCF A can recognize a non-regular unary language.

Keywords: automata theory, counter automata, quantum automata, unary languages, non-regular languages, pebble automata

1 Background

We assume the reader familiar with the basics of the quantum computation (see [NC00]). Some part of the background are taken from a previous related work [Yak13a], in which the same models are defined.

Throughout the paper, $\Sigma$ not containing $\epsilon$ and $\$$ denotes the input alphabet and $\bar{\Sigma} = \Sigma \cup \{\epsilon, \$$\}$. For a given string $w$, $|w|$ is the length of $w$ and $w_i$ is the $i^{th}$ symbol of $w$, where $1 \leq i \leq |w|$. The string $\epsilon w \$$ is represented by $\bar{w}$. Moreover, $\Theta$ is the set of $\{0, \pm\}$.

The models defined in the paper have a two-way infinite read-only input tape whose squares are indexed by integers. Any given input string, say $w \in \bar{\Sigma}^*$, is placed on the tape as $\bar{w}$ between the squares indexed by 1 and $|\bar{w}|$. The tape has a single head, and it can stay in the same position (↓) or move to one square to the left (←) or to the right (→) in one step. It must always be guaranteed that the input head never leaves $\bar{w}$. A counter is an infinite storage having two status, i.e. zero (0) or nonzero (±), and updated by a value from $\{-1, 0, 1\}$ in one step.
A two-way deterministic one-counter automaton (2DCA) is a two-way deterministic finite automaton with a counter. Formally, a 2DCA $D$ is a 6-tuple

$$D = (S, \Sigma, \delta, s_1, s_a, s_r),$$

where $S$ is the set of states, $s_1 \in S$ is the initial state, $s_a \in S$ and $s_r \in S$ ($s_a \neq s_r$) are the accepting and rejecting states, respectively, and $\delta$ is the transition function governing the behaviour of $D$ in each step, i.e.

$$\delta : S \times \tilde{\Sigma} \times \Theta \rightarrow S \times \{\leftarrow, \downarrow, \rightarrow\} \times \{-1, 0, 1\}.$$  

Specifically, $\delta(s, \sigma, \theta) \rightarrow (s', d_i, c)$ means that when $D$ is in state $s \in S$, reads symbol $\sigma \in \tilde{\Sigma}$, and the status of its counter is $\theta \in \Theta$, then it updates its state to $s' \in S$, the position of the input head with respect to $d_i \in \{\leftarrow, \downarrow, \rightarrow\}$, and the value of the counter by $c \in \{-1, 0, 1\}$.

At the beginning of the computation, $D$ is in state $s_1$, the input head is placed on symbol $\epsilon$, and the value of the counter is set to zero. A configuration of $D$ on a given input string is represented by a triple $(s, i, v)$, where $s$ is the state, $i$ is the position of the input head, and $v$ is the value of the counter. The computation is terminated and the input is accepted (rejected) by $D$ when it enters to $s_a$ ($s_r$).

A two-way finite state automaton with quantum and classical states [AW02] (2QCFA) is a two-way finite state automaton using a constant-size quantum register. Note that, a two-way probabilistic automaton (2PFA) is obtained if the quantum register is replaced by a random number generator. The main difference between a random number generator and a constant-size quantum register is that quantum register can keep some information by its (pure) quantum state

1 It was shown that 2QCFAs are more powerful than their probabilistic counterparts in the bounded-error setting by Ambainis and Watrous [AW02]. In fact, they can recognize many interesting languages in the bound error setting [YS10a,YS10b]. Currently, it is not known whether 2QCFAs using rational superoperators can be simulated by the original 2QCFAs using only rational unitary and measurement operators. For other kind of amplitudes such as algebraic, computable, unrestricted, etc. such simulation can be
easily obtained. In this paper, we restrict ourselves with only rational superoperators. Note that any such 2QCFA algorithm can be implemented by an original 2QCFA using algebraic unitary operators and integer projective measurements.

The most general quantum operator is a superoperator, which generalizes stochastic and unitary operators and also includes measurements. Formally, a superoperator $E$ is composed by a finite number of operation elements, $E = \{E_1, \ldots, E_k\}$, satisfying that

$$\sum_{i=1}^{k} E_i^\dagger E_i = I,$$

where $k \in \mathbb{Z}^+$ and the indices are the measurement outcomes. When a superoperator, say $E$, is applied to the quantum register in state $|\psi\rangle$, i.e. $E(|\psi\rangle)$, we obtain the measurement outcome $i$ with probability $p_i = \langle \tilde{\psi}_i | \tilde{\psi}_i \rangle$, where $|\tilde{\psi}_i\rangle$, the unconditional state vector, is calculated as $|\tilde{\psi}_i\rangle = E_i |\psi\rangle$ and $1 \leq i \leq k$. Note that using unconditional state vector simplifies calculations in many cases. If the outcome $i$ is observed ($p_i > 0$), the new state of the system is obtained by normalizing $|\tilde{\psi}_i\rangle$, which is $|\psi_i\rangle = \frac{|\tilde{\psi}_i\rangle}{\sqrt{p_i}}$. Moreover, as a special operator, the quantum register can be initialized to a predefined quantum state. This initialize operator, which has only one outcome, is denoted $\hat{E}$.

**Fig. 1.** The details of superoperators [Yak13b]

A 2QCFA $Q$ is a 8 tuple

$$(S, Q, \Sigma, \delta, s_1, s_q, s_r, q_1),$$

where, apart from a classical model, there are two different components: $Q$ is the state set of quantum register and $q_1$ is its initial state. Moreover, $\delta = (\delta_q, \delta_c)$, where $\delta_q$ governs the quantum part and $\delta_c$ governs the classical part. In each step, firstly, $\delta_q$ determines a superoperator depending on the current classical state ($s \in S$) and scanning symbol ($\sigma \in \tilde{\Sigma}$), i.e. $E_{s,\sigma}$, and then it is applied to the quantum register and one outcome, say $\tau$, is observed. Secondly, the classical part of $Q$ is updated depending on $s$, $\sigma$, and $\tau$, which is formally represented as $\delta_c(s, \sigma) \rightarrow (s', d_i)$, where $s' \in S$ is the new classical state and $d_i \in \{\leftarrow, \downarrow, \rightarrow\}$ is the update of the position of input tape. Note that $\delta_c$ must be defined for each possible $\tau$.

A two-way one-counter automaton with quantum and classical states (2QCCA) is a 2DCA augmented with a constant-size quantum register or a 2QCFA having a classical counter. The formal definition of a 2QCCA is exactly the same as a 2QCFA. So, the transition functions of a 2QCFA ($\delta_q$ and $\delta_c$) can be extended for a 2QCCA with the following modifications:

- The superoperator is determined by also the status of the counter ($\theta \in \Theta$), i.e. $E_{s,\sigma,\theta}$. 

– The classical part of $Q$ is updated depending on $s$, $\sigma$, $\theta$, and $\tau$, which is formally represented as $\delta_c(s, \sigma, \theta) \rightarrow (s', d_i, c)$, where $s' \in S$ is the new classical state, $d_i$ is the update of the position of input tape, and $c \in \{-1, 0, 1\}$ is the update on the counter.

2 Main result

Recently, Yakaryılmaz [Yak13a] introduce a new programming technique for 2QCCAs (and also for 2QCFAs with a pebble) and it was shown that

$$USQUARE = \{a^n^2 \mid n \geq 1\}$$

can be recognized by a 2QCCA for any error bound by using a $\sqrt{n}$-space on its counter for the members. The main idea is as follows. Let $w = a^m$ be the input, where $m \geq 1$. Otherwise, it is rejected immediately. The 2QCCA, say $USQUARE$, tries to mark the input from 1 to $m$ iteratively by using its (classical) counter in a FOR-LOOP. That is, on the $i^{th}$ iteration, the value of the counter is $i$ on the left end-marker. Then $USQUARE$ reads $i$ a’s by decrement the value of the counter by 1. It arrives on the $i^{th}$ a when the counter becomes empty. After that, $USQUARE$ firstly reads $i$ a’s again in reverse direction (by moving to the left end-marker) and then read the whole input, $m$ a’s, by moving to the right end-marker. In the first part, the counter value is set to $i$ again, and, in the second part, the value of counter does not changed. In fact, $USQUARE$ reads $a^i e a^m$. We know that 2QCFAs can recognize

$$SQUARE = \{a^n b a^2 \mid n \geq 1\}$$

for any error bound such that the members are accepted exactly and the non-members are rejected with high probability [YS10b]. Let $SQUARE$ be such 2QCFAs rejecting the non-members with a probability at least $\frac{2}{3}$. $USQUARE$ executes $SQUARE$ on $a^i e a^m$ to test whether $i^2$ is equal to $m$. $SQUARE$ needs to read $a^i e a^m$ many times and it can be provided by $USQUARE$ easily since the value of $i$ is stored on the counter. (Note that $SQUARE$ reads its input from left to right in an infinite loop and the computation terminates with probability 1 in exponential expected time.) $USQUARE$ passes to the next iteration only if $SQUARE$ gives the decision of “rejection”. So, if $i^2 = m$, then $USQUARE$ never passes to the $(i + 1)^{th}$ iteration since $SQUARE$ never gives the decision of “rejection”. $USQUARE$ terminates the FOR-LOOP if $SQUARE$ gives the decision of “acceptance”.

If $w$ is a member of $USQUARE$, then $USQUARE$ reaches the end-marker with a probability at least $(\frac{2}{3})^m$. At this point, $USQUARE$ rejects $w$. After termination of the FOR-LOOP, $USQUARE$ accepts the input with probability $(\frac{1}{3})^{2m}$. Thus,

– any member of $USQUARE$ is accepted with a probability $(\frac{1}{3})^{2m}$, and
– any non-member is rejected with a probability at least $4^m$ greater than the accepted probability.
So, by executing the above procedure in an infinite loop, \textit{USQUARE} accepts any member of \textit{USQUARE} exactly, and rejects any non-member with a probability at least $\frac{4}{5}$. By using conventional probability amplification techniques, the rejecting probability can be bounded below arbitrary close to 1. As can be easily verified, for the members, the counter value never exceeds $\sqrt{m} = i$. For the non-members, on the other hand, \textit{USQUARE} uses linear space.

Since the marking of the input described above can be implemented by a pebble as well, we can follow that \textit{USQUARE} can be recognized by a 2QCFA with a pebble for any error bound such that the pebble is moved only in $\sqrt{|w|}$ times for any $w \in \textit{USQUARE}$.

Similar to \textit{SQUARE}, \textit{POWER} = $\{a^n b^{2^n} \mid n \geq 1\}$ can also be recognized by 2QCFAs for any error bound \cite{YS10}. Some other languages recognized by bounded-error 2QCFAs listed in \cite{YS10} are as follows:

- \textit{TWIN} = $\{wcw \mid w \in \{a,b\}^*\}$
- \textit{MULT} = $\{x#y#z \mid x, y, z \in \{0,1\}^* \text{ and } x \times y = z\}$
- All polynomial languages \cite{Tur82} defined as
  \[\{a_1^{n_1} \cdots a_k^{n_k}b_1^{p_1(n_1,\ldots,n_k)} \cdots b_r^{p_r(n_1,\ldots,n_k)} \mid p_i(n_1,\ldots,n_k) \geq 0\},\]

  where $a_1, \ldots, a_k, b_1, \ldots, b_r$ are distinct symbols, and each $p_i$ is a polynomial with integer coefficients.

Actually, all these algorithms can be implemented by reading the input from left to right with a realtime head\footnote{The input head cannot stay on the same tape square more than a fixed number of steps.} in an infinite loop. We can call such models as restarting realtime finite automata with quantum and classical states (restarting rtQCFA). Knowledgeable readers can notice that restarting rtQCFA is a special case of rotating \cite{SS78,KKM12} 2QCFA, which is a special case of sweeping \cite{Sip80,KKM12} 2QCFA. Moreover, all these restarting rtQCFAs can be defined only with rational superoperators. We refer the reader also to \cite{Yak13a,Yak13b} for similar algorithms. Here, we will give a rational 2QCFA, say \textit{POWER}, (i.e., restarting rtQCFA) recognizing \textit{POWER} for any error-bound. After that, we describe a bounded-error 2QCCA for language \textit{UPOWER} = $\{a^2^n \mid n \geq 0\}$ such that the 2QCCA uses logarithmic space on its counter for the members.

Let $w \in \{a,b\}^*$ be the input. We can assume the input of the form $a^m b^n$, where $m, n > 0$. \textit{POWER} rejects the input immediately, otherwise. The quantum register has three states: $|q_1\rangle, |q_2\rangle, |q_3\rangle$. \textit{POWER} encodes $2^m$ and $n$ into amplitudes of $|q_2\rangle$ and $|q_3\rangle$, and then compare them by subtracting. If they are equal, then the resulting amplitude will be zero, and it is non-zero, otherwise.
Based on this amplitude, the input is rejected. Since we will use only rational numbers, we can bound this rejecting probability from the below when it is non-zero. Note that it is zero only for the members. Thus, by creating a carefully tuned accepting probability, the members can be only accepted and the non-members can be rejected with a probability sufficient greater than the accepting probability. The technical details are given below.

The following procedure is executed in an infinite loop. In each iteration (round), the input is read from left to right in realtime mood. At the beginning of the round, the quantum state is set to $|\psi_0\rangle = (1 \ 0 \ 0)^T$. In order to facilitate the calculations, the unconditional quantum state is traced as long as the current round is not terminated. When reading the left end-marker, $E_c = \{E_{c,1}, E_{c,2}\}$ is applied to the quantum register, i.e.

$$E_{c,1} = \frac{1}{2} \begin{pmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 2 \end{pmatrix} \text{ and } E_{c,2} = \frac{1}{2} \begin{pmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 2 & 0 \end{pmatrix},$$

where (i) the current round continues if outcome “1” is observed, and (ii) the current round is terminated without any decision if outcome “2” is observed.

Before reading $a$’s, the quantum state is

$$|\tilde{\psi}_0\rangle = \frac{1}{2} \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix}. $$

When reading an $a$, $E_a = \{E_{a,1}, E_{a,2}\}$ is applied to the quantum register, i.e.

$$E_{a,1} = \frac{1}{2} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 2 \end{pmatrix} \text{ and } E_{a,2} = \frac{1}{2} \begin{pmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix},$$

where (i) the current round continues if outcome “1” is observed, and (ii) the current round is terminated without any decision if outcome “2” is observed.

Before reading $b$’s, the quantum state is

$$|\tilde{\psi}_m\rangle = \left(\frac{1}{2}\right)^{m+1} \begin{pmatrix} 1 \\ 2^m \\ 0 \end{pmatrix}. $$

When reading a $b$, $E_b = \{E_{b,1}, E_{b,2}, E_{b,3}\}$ is applied to the quantum register, i.e.

$$E_{b,1} = \frac{1}{2} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix}, E_{b,2} = \frac{1}{2} \begin{pmatrix} 1 & 0 & -1 \\ 1 & 0 & 0 \\ 0 & 1 & 1 \end{pmatrix}, \text{ and } E_{b,3} = \frac{1}{2} \begin{pmatrix} 0 & 1 & -1 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix},$$

where (i) the current round continues if outcome “1” is observed, and (ii) the current round is terminated without any decision if outcome “2” or “3” is observed.
Before reading the right end-marker, the quantum state is

\[ |\tilde{\psi}_w\rangle = \left( \frac{1}{2} \right)^{m+n+1} \left( \frac{1}{2^m} \right). \]

When reading the right end-marker, \( \mathcal{E}_8 = \{ E_{8,1}, E_{8,2}, E_{8,3}, E_{8,4} \} \) is applied to the quantum register, i.e.

\[ E_{8,1} = \frac{1}{2k} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \quad E_{8,2} = \frac{1}{2k} \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 -k & -k \\ 0 & 0 -k & -k \end{pmatrix}, \quad E_{8,3} = \frac{1}{2k} \begin{pmatrix} k_1 & 0 & 0 \\ k_2 & 0 & 0 \\ k_3 & 0 & 0 \end{pmatrix}, \quad \text{and} \quad E_{8,4} = \frac{1}{2k} \begin{pmatrix} k_4 & 0 & 0 \\ 0 & k & k \\ 0 & k & k \end{pmatrix}, \]

where \( k \) can be any non-negative integer; \( k_1, k_2, k_3, \) and \( k_4 \) are integers satisfying \( k_1^2 + k_2^2 + k_3^2 + k_4^2 = 4k^2 - 1; \) and, the actions based on the measurement outcomes are as follows:

- the input is accepted if outcome “1” is observed,
- the input is rejected if outcome “2” is observed, and,
- the current round is terminated without any decision, otherwise.

Note that, we define \( k_i \)’s to have a well-defined quantum operator and, due to Lagrange’s four-square theorem, we know that each natural number can be represented as the sum of four integer squares, where \( 1 \leq i \leq 4. \) The analysis of the algorithm is as follows. If outcome “1” is observed, then the quantum state is

\[ |\tilde{\psi}_w\rangle_{w+1} = \left( \frac{1}{2} \right)^{m+n+1} \left( \frac{1}{k} \right). \]

That is, in a single round, the input is always accepted with probability

\[ \left( \frac{1}{4} \right)^{m+n} \left( \frac{1}{k^2} \right). \]

If outcome “2” is observed, then the quantum state is

\[ |\tilde{\psi}_w\rangle_{w+1} = \left( \frac{1}{2} \right)^{m+n+1} \left( \frac{1}{k} \right)^2 \left( \frac{k(2^m - n)}{k(2^m - n)} \right). \]

That is, in a single round, the input is rejected with a probability

\[ \left( \frac{1}{4} \right)^{m+n} \left( \frac{1}{k^2} \right)^2 \left( k(2^m - n) \right)^2, \]

which is
Thus, we can say that $\text{POWER}$ accepts any member exactly and rejects any non-member with a probability at least $\frac{2k^2}{2k^2+1}$. Thus, the error bound can be arbitrary close to zero by setting $k$ with an appropriate value.

A 2QCCA for $\text{UPOWER}$, say $\text{UPOWER}$, can be defined similar to $\text{USQUARE}$ (the 2QCCA given above for $\text{USQUARE}$). The pseudo-code of $\text{UPOWER}$ is given below. Let $w = a^m$ be the input.

```
FOR i = 1 TO m
  RUN $\text{POWER}$ on $w' = a^ib^m$
  IF $\text{POWER}$ accepts $w'$ THEN TERMINATE FOR-LOOP
  IF $\text{POWER}$ rejects $w'$ AND $i = m$ THEN REJECT the input
END FOR
ACCEPT $w$ with a nonzero probability at most $\left(\frac{1}{2k^2+1}\right)^m$
RESTART the algorithm
```

For the members of $\text{UPOWER}$, the decision of “rejection” is never given in FOR-LOOP. Therefore, they are accepted exactly. For the non-members, the input is rejected with a probability at least $\left(\frac{1}{2k^2+1}\right)^m$ at the end of a FOR-LOOP. Since the input can be accepted with a probability at most $\left(\frac{1}{2k^2+1}\right)^m$ after a FOR-LOOP, the rejecting probability is at least $k^2m$ times greater than the accepting probability after a FOR-LOOP. Therefore, any non-member is rejected with a probability at least $\frac{2k^2}{2k^2+1}$. It is clear that for the members, the counter value never exceeds $\log(|w|)$, so the space complexity is logarithmic for the members.

Once getting the details of the above algorithms, it is quite straightforward to show that each of the following languages can be recognized by 2QCCAs.

- $\text{POLY}(p) = \{a^{p(n)} \mid n \geq 1\}$,
- $\text{POWER}(m) = \{a^m \mid n \geq 1\}$, and
- $\text{POLY-POWER}(p, m) = \{a^{p(n)m^n} \mid n \geq 1\}$,

where $p$ is a polynomial such that $p(\mathbb{Z}^+) > 0$ and $m > 2$. Curious readers can also obtain their own combinations. As described before, the counter is used to mark the input iteratively in the above 2QCCA algorithms. Since such an iteration can be easily implemented by using a pebble, all languages above can be recognized by bounded-error 2QCFAs with a pebble. A counter can also iteratively mark the input as $1, n, n^2, n^3, \ldots$ for some integer $n \geq 2$. Thus, we can define new languages recognized by bounded-error 2QCCAs by replacing $n$ with $2^n$ in the languages given above. For example,

$$\left\{a^{2^n} \mid n \geq 1\right\}, \left\{a^{3^n} \mid n \geq 1\right\}, \left\{a^{2^n \cdot 2^n} \mid n \geq 1\right\}, \text{etc.}$$

All such languages can be recognized by 2QCCAs for any error bound such that the members are accepted exactly and the non-members are rejected with high probability. Currently, we do not know any bound-error 2QCFAs with a pebble for such languages.
3 Concluding remarks

In [DG82], it was shown that any unary language recognized by a two-way deterministic pushdown automaton using sublinear space on its stack is regular. Therefore, 2DCAs using sublinear space cannot recognize any nonregular unary languages. Currently, we do not known whether nondeterminism, alternation, or using random choices can help.

For automata without counters, it is still open whether 2QCFAs can recognize a non-regular unary language. But, it is known that 2PFAs cannot recognize a non-regular unary language with bounded-error [Ka91]. Moreover, USQUARE is a nonstochastic language [Tur81], not recognizable by 2PFAs with unbounded-error. Indeed, it is not known whether using a counter can help for 2PFAs.

4 Acknowledgements

We thank Alexander Okhotin, Holger Petersen, and Klaus Reinhardt for their answers to our questions on the subject matter of this paper.

References

AW02. Andris Ambainis and John Watrous. Two–way finite automata with quantum and classical states. Theoretical Computer Science, 287(1):299–311, 2002.
DG82. Pavol Duriš and Zvi Galil. On reversal-bounded counter machines and on pushdown automata with a bound on the size of their pushdown store. Information and Control, 54(3):217–227, 1982.
Ka91. Jānis Kārs. Regularity of one-letter languages acceptable by 2-way finite probabilistic automata. In Proceedings of the 8th International Symposium on Fundamentals of Computation Theory, pages 287–296, 1991.
KKM12. Christos A. Kapoutsis, Richard Králík, and Tobias Mömke. Size complexity of rotating and sweeping automata. Journal of Computer and System Sciences, 78(2):537–558, 2012.
NC00. Michael A. Nielsen and Isaac L. Chuang. Quantum Computation and Quantum Information. Cambridge University Press, 2000.
Sip80. Michael Sipser. Lower bounds on the size of sweeping automata. Journal of Computer and System Sciences, 21(2):195–202, 1980.
SS78. William J. Sakoda and Michael Sipser. Nondeterminism and the size of two way finite automata. In STOC’78: Proceedings of the tenth annual ACM symposium on Theory of computing, pages 275–286, 1978.
Tur81. Paavo Turakainen. On nonstochastic languages and homomorphic images of stochastic languages. Information Sciences, 24(3):229–253, 1981.
Tur82. Paavo Turakainen. Discrete Mathematics, volume 7 of Banach Center Publications, chapter Rational stochastic automata in formal language theory, pages 31–44. PWN-Polish Scientific Publishers, Warsaw, 1982.
Yak13a. Abuzer Yakaryılmaz. One-counter verifiers for decidable languages. In CSR, volume 7913 of LNCS, pages 366–377. Springer, 2013.
Yak13b. Abuzer Yakaryılmaz. Public qubits versus private coins. In The Proceedings of Workshop on Quantum and Classical Complexity, pages 45–60. Univeristy of Latvia Press, 2013. ECCC:TR12-130.
Yak13c. Abuzer Yakaryılmaz. Quantum alternation. In CSR, volume 7913 of LNCS, pages 334–346. Springer, 2013.

YS10a. Abuzer Yakaryılmaz and A. C. Cem Say. Languages recognized by nondeterministic quantum finite automata. Quantum Information and Computation, 10(9&10):747–770, 2010.

YS10b. Abuzer Yakaryılmaz and A. C. Cem Say. Succinctness of two-way probabilistic and quantum finite automata. Discrete Mathematics and Theoretical Computer Science, 12(2):19–40, 2010.

YS11. Abuzer Yakaryılmaz and A. C. Cem Say. Unbounded-error quantum computation with small space bounds. Information and Computation, 279(6):873–892, 2011.