Theoretical Computer Science 410 (2009) 1952–1959

Contents lists available at ScienceDirect

Theoretical Computer Science

journal homepage: www.elsevier.com/locate/tcs

Mathematical logic and quantum finite state automata

Ilze Dzelme-Bērziņa

Institute of Mathematics and Computer Science, University of Latvia, Raiņa bulvāris 29, LV–1459, Riga, Latvia

A R T I C L E   I N F O

Keywords:
Quantum finite-state automata
First-order logic
Modular logic
Measure-once quantum finite-state automata
Measure-many quantum finite-state automata

A B S T R A C T

This paper is a review of the connection between formulas of logic and quantum finite-state automata in respect to the language recognition and acceptance probability of quantum finite-state automata. As is well known, logic has had a great impact on classical computation, it is promising to study the relation between quantum finite-state automata and mathematical logic. After a brief introduction to the connection between classical computation and logic, the required background of the logic and quantum finite-state automata is provided and the results of the connection between quantum finite-state automata and logic are presented.

1. Introduction

The connection between automata theory and logic dates back to the early sixties to the work of Büchi [8] and Elgot [13], who showed that the finite automata and monadic second-order logic (interpreted over finite words) have the same expressive power, and that the transformation from logical monadic second-order formulas to finite-state automata and vice versa are effective. Later, the equivalence between finite-state automata and monadic second-order logic over infinite words and trees were shown in the works of Büchi [9], McNaughton [18], and Rabin [23]. The next important step in the connection between automata theory and logic was Pnueli's work [22]. It was proposed to use temporal logic for reasoning about continuously operating concurrent programs. In the eighties, temporal logic and fixed-point logic took the role of specification languages and more efficient transformations from logic formulas to automata were found. This led to powerful algorithms and software systems for the verification of finite-state programs ("model-checking"). The research on the equivalence between automata theory and logic formalism also influenced language theory itself. For example, automata classes were described in terms of logic.

The logical description of the computation models' behavior also influenced complexity theory. In 1974, Fagin [14] gave a characterization of nondeterministic polynomial time as the set of properties expressible in the second-order existential logic. Later, Immerman [15] and Vardi [26] characterized polynomial time as the set of properties expressible in a first-order inductive definition, which is defined by adding a least-point operator to the first-order logic. In the similar way, polynomial space has also been characterized [16]. These results led to the development of a new field – Description complexity – a subfield of computational complexity theory and mathematical logic, which seeks to characterize complexity classes by the type of logic needed to express the language in them.

A mixture of techniques and results from automata theory, logic, and complexity was achieved in circuit complexity theory, which studies the computational power of Boolean circuits, regarding restrictions in their size, depth, and types of allowed gates. Natural families of circuits can be described by generalized models of finite state automata as well as by appropriate systems of first-order logic.

A natural model of classical computing with finite memory is a finite-state automaton, likewise a quantum finite-state automaton is a natural model of quantum computation. Different notations of quantum finite-state automata are used. The

E-mail address: ilze.dzelme@gmail.com.

0304-3975/$ – see front matter © 2009 Elsevier B.V. All rights reserved.
doi:10.1016/j.tcs.2009.01.030

two most popular notations of quantum finite-state automata are quantum finite-state automata introduced by Moore and Crutchfield [19] (measure-once quantum finite-state automata) and quantum finite-state automata introduced by Kondacs and Watrous [17] (measure-many quantum finite-state automata). They have a seemingly small difference, in the first definition a quantum finite-state automaton performs the measurement only at the end of the computation, but in the second definition a quantum finite-state automaton performs the measurement at every step of the computation. Measure-once quantum finite-state automata and measure-many quantum finite-state automata with bounded error recognize only the subset of regular languages. Besides these two models of quantum finite-state automata there are also such models of quantum finite-state automata as "enhanced" quantum finite-state automata [21], Latvian quantum finite-state automata [2], 1-way quantum finite-state automata with control language [7], quantum finite-state automata with mixed states (introduced by Aharonov, Kitaev and Nisan [1]) and quantum finite-state automata with quantum and classical states (introduced by Ambainis and Watrous [6]) and others.

Quantum finite state automata have their strengths and weaknesses in comparison to their classical counterparts. The strength of quantum finite state automata is in the fact that quantum finite state automata can be exponentially more effective [4], but the main weakness is caused by necessity that a quantum process has to be reversible, that makes the most of quantum finite state automata notations with bounded error unable to recognize all regular languages. And for many notations of quantum finite state automata the problem to describe the class of the languages recognizable by the quantum finite state automata is still open. As logic has had a large impact on computation theory, it seems to be useful to look at the languages recognizable of quantum finite state automata in the terms of logic, as well as to look at the properties of quantum finite automata in the terms of logic. Using logic, we might characterize language classes or subclasses recognized by quantum finite state automata (especially for those notations of quantum finite state automata for whom the problem to describe the class of recognizable languages is still open) or find out other properties of quantum finite state automata. Logic properties could be useful tool for solving open issues in the quantum computation.

We give a survey on results on connection between quantum finite automata and logic. At first, we give a brief introduction to the logic, monoids and quantum finite-state automata. The third section contains the known results in the connection between logic and measure-once quantum finite-state automata. The fourth section is dedicated to the connection between first-order logic and measure-many quantum finite-state automata with respect to language recognition and acceptance probability.

2. Preliminaries

2.1. Monoids and groups

A *monoid* is a set M with an associative binary operation $*$ and an identity element. Let M, N be monoids. The mapping $\phi : M \to N$ is a *homomorphism* iff:

- $\phi(x *_M y) = \phi(x) *_N \phi(y)$
- $\phi(1_M) = 1_N$.

Let M be a monoid, A be a finite alphabet, and $\phi : A^* \to M$ a homomorphism. Every subset N of M defines a subset of A^*:

$$\phi^{-1}(N) := \{\omega \in A^* \mid |\phi(\omega) \in N\}.$$

The language $L \subseteq A^*$ is *accepted* by M iff there is a $N \subseteq M$ and a homomorphism $\phi : A^* \to M$ such that $L = \phi^{-1}(N)$.

The *syntactic monoid* of a language $L \subseteq A^*$ is the quotient monoid $M(L) = A^*/\sim_L$, where $\sim_L$ is the syntactic congruence over A^* defined by

$$w \sim_L w' \text{iff } \forall u, v \in A^*(uwv \in L \Longleftrightarrow uw'v \in L).$$

The *syntactic morphism* of L is the projection μ_L of A^* onto $M(L)$. A *finite aperiodic monoid M* is a monoid for which there is $n \geq 1$ such that $m^{n+1} = m^n$ holds for all $m \in M$.

A *group* is a set G with a binary operation $*$, that satisfies the four properties of closure, associativity, the identity property, and the inverse property. The *commutator of group G* elements $g, h \in G$, denoted $[g, h]$, is defined as $[g, h] = g^{-1}h^{-1}gh$, and for any two subgroups $H, K \leq G$ we write $[H, K]$ to denote the subgroup of G generated by all commutators $[h, k]$ with $h \in H$ and $k \in K$. The *derived subgroup* of G is $G' = [G, G]$, and we write $G^{(0)} = G, G^{(j)} = (G^{(j-1)})'$, for $j \geq 1$. A group is said to be *solvable* if $G^{(m)} = 1$ (the group consisting of just one element) for some value of m.

2.2. Logic and classical automata

Let A be a finite alphabet and let $\omega = a_1a_2 \dots a_n$ be a word over the alphabet A. The corresponding *word model* for the word ω is represented by the relational structure

$$\underline{\omega} = (dom(\omega), <, (Q_a)_{a\in A})$$

where $dom(\omega) = \{1, 2, \dots, n\}$ is the set of the letters "positions" of ω (the "domain" of ω), $<$ is the order relation on $dom(\omega)$, and $Q_a = \{i \in dom(\omega) \mid a_i = a\}$ ("position carries letter a").

We consider word models over the finite alphabet A. The corresponding *first order language FO*$[<]$ has variables $x, y, \dots$ ranging over positions in the word models, and is built from atomic formulas of the form

$$x = y, Q_a(x), x < y$$

by means of the connectives $\neg, \vee, \wedge, \rightarrow, \leftrightarrow$ and quantifiers $\exists$ and $\forall$. We may also use the successor relation $S(x, y)$, which can be expressed as first-order formula $x < y \wedge \neg\exists z(x < z \wedge z < y)$. The notation $\varphi(x_1, x_2, \dots, x_n)$ indicates that in the formula φ at most the variables $x_1, x_2, \dots, x_n$ are free, i.e. they are not in the scope of a quantifier. A *sentence* is a formula with no free variables. If $p_1, p_2, \dots, p_n$ are positions from $dom(\omega)$ then $(\underline{\omega}, p_1, p_2, \dots, p_n) \models \varphi(x_1, x_2, \dots, x_n)$ means that φ is satisfied in the word model $\underline{\omega}$ when $p_1, p_2, \dots, p_n$ serve as an interpretation of $x_1, x_2, \dots, x_n$. The language defined by the sentence φ is $L(\phi) = \{\omega \in A^* \mid \underline{\omega} \models \varphi\}$. Languages defined by such sentences are the first-order $FO[<]$ languages. For example, the sentence $\forall x(Q_a(x))$ over the alphabet $A = \{a, b\}$ defines the language containing all words having only letters a. This language is a $FO[<]$ language. The classical equivalence result of the first-order logic is results by Schützenberger [24]:

Theorem 2.1. *For a language $L \in A^*$ the following are equivalent*

(1) *L is star-free (the smallest class that satisfies the following: all finite languages over A belong to star-free languages, if languages L_1, L_2 are star-free then so are $L_1 \cdot L_2$, $L_1 \cup L_2$, $L_1 \cap L_2$ and $\bar{L}_1 = A^* \setminus L$).*
(2) *L is recognizable by a finite aperiodic monoid.*
(3) *L is defined by a first-order formula.*

We will consider a new quantifier $\exists^{m,n}x\varphi(x)$ that means $\varphi(x)$ is true for a number of x equal to *nmodm*. It is called a *modular quantifier* [25]. Let us denote by $MOD[<]$ the class of languages defined by first-order atomic formulas ($x = y$, $Q_a(x)$, $x < y$) by means of the connectives $\neg, \vee, \wedge, \rightarrow, \leftrightarrow$ and the modular quantifier. An example of $MOD[<]$ is the language containing all words that has even number of letter "a"s, the corresponding formula for this language is $\exists^{2,0}xQ_a(x)$. Let $MOD[S, <]$ be a class of languages defined by first-order atomic formulas $x = y$, $Q_a(x)$, $x < y$, $S(x, y)$ by means of the connectives $\neg, \vee, \wedge, \rightarrow, \leftrightarrow$ and the modular quantifier.

Theorem 2.2 ([25]). *Let $L \subseteq A^*$ be a regular language, then $L \in MOD[<]$ if and only if the syntactic monoid of language L is a solvable group.*

A logical framework can also be extended with generalized quantifiers; they have been introduced by Mostowski [20]. One of such quantifiers is Lindström quantifier.

Definition 2.2.1. Consider a language L over the alphabet $\Sigma = (a_1, a_2, \dots, a_s)$. Let $\bar{x}$ be a k-tuple of variables (each ranging from 1 to the input length n). In the following, we assume the lexical ordering on $\{1, 2, \dots, n\}^k$, and we write $X_1, X_2, \dots, X_{n^k}$ for this sequence of the potential values taken on by $\bar{x}$. Let $\phi_1(\bar{x}), \phi_2(\bar{x}), \dots, \phi_{s-1}(\bar{x})$ be s-1 Γ-formulas for some alphabet Γ. The

$$Q_L\bar{x}[\phi_1(\bar{x}), \phi_2(\bar{x}), \dots, \phi_{s-1}(\bar{x})]$$

holds on string $\omega = \omega_1\omega_2 \dots \omega_n$, iff the word of length n^k whose i-th letter ($1 \le i \le n^k$) is

$$\begin{cases} a_1 & \text{if } \omega \models \phi_1(X_i), \\ a_2 & \text{if } \omega \models \neg\phi_1(X_i) \wedge \phi_2(X_i), \\ a_3 & \text{if } \omega \models \neg\phi_1(X_i) \wedge \neg\phi_2(X_i) \wedge \phi_3(X_i), \\ \dots & \\ a_s & \text{if } \omega \models \neg\phi_1(X_i) \wedge \neg\phi_2(X_i) \wedge \dots \wedge \neg\phi_{s-1}(X_i), \end{cases}$$

belongs to L.

Consider alphabet $\Sigma = \{0, 1\}$ and a language L which is defined by regular expression$(0, 1)^*0(0, 1)^*$, then the formula $Q_L(\phi(x))$ is equal to the classical first-order existential quantifier applied to some quantifier-free formula ϕ with free variable x, i.e. $\exists x\phi(x)$. It is easy to see, that the formula will be true if there is at least one position of x for which $\phi(x)$ will be true.

2.3. Quantum finite automata

Definition 2.3.1. A measure-once quantum finite-state automaton (MO-QFA) is defined by a tuple as follows [19]

$$A_{MO-QFA} = (Q; \Sigma; \delta; q_0; Q_{acc}; Q_{rej})$$

where

(1) Q is a finite set of states,
(2) Σ is an input alphabet and $\Gamma = \Sigma \cup \{ \sharp; \$ \}$ is working alphabet of A_{MO-QFA}, and $\sharp$ and \$ are the left and the right end-markers,
(3) $q_0 \in Q$ is a initial state,
(4) $Q_{acc} \subseteq Q$ and $Q_{rej} \subseteq Q$ are sets of accepting and rejecting states ($Q_{acc} \cap Q_{rej} = \emptyset$),
(5) δ is the transition function $\delta : Q \times \Gamma \times Q \rightarrow C_{[0,1]}$, which represents the amplitudes that follows from the state q to the q' after reading symbol σ.

For all states $q_1, q_2, q' \in Q$ and symbols $\sigma \in \Gamma$, the function δ must be unitary, thus the function satisfies the condition

$$\sum_{q'} \overline{\delta(q_1, \sigma, q')}\delta(q_2, \sigma, q') = \begin{cases} 1 \ (q_1 = q_2) \\ 0 \ (q_1 \neq q_2) \end{cases}.$$

And it is assumed that an input word starts with the left end-marker and ends with the right end-marker.

The linear superposition of the automaton's A_{MO-QFA} states is represented by a n-dimensional complex unit vector, where $n = |Q|$. The vector is denoted by $|\phi\rangle = \sum_{i=1}^{n} \alpha_i |q_i\rangle$, where $\{|q_i\rangle\}$ is the set orthonormal basis vectors corresponding to the states of the automaton A_{MO-QFA}.

The transition function δ is represented by a set of unitary matrices $\{V_\sigma\}_{\sigma\in\Gamma}$, where V_σ is the unitary transition of the automaton A_{MO-QFA} after reading the symbol σ and is defined by $V_\sigma(|q\rangle) = \sum_{q'\in Q} \delta(q, \sigma, q') \left|q'\right\rangle$.

A computation of A_{MO-QFA} on input $\sharp\sigma_1\sigma_2 \dots \sigma_n\$$ proceeds as follows. It starts in superposition $|q_0\rangle$. Then a transformation corresponding to the left end-marker $\sharp$, the letters of the input word and the right end-marker \$ are performed. After reading the right end-marker \$ the final superposition is observed with respect to E_{acc} and E_{rej} where $E_{acc} = span\{|q\rangle : q \in Q_{acc}\}$ and $E_{rej} = span\{|q\rangle : q \in Q_{rej}\}$. It means if the final superposition is $\psi = \sum_{q_i\in Q_{acc}} \alpha_i |q_i\rangle + \sum_{q_j\in Q_{rej}} \beta_j \left|q_j\right\rangle$ then the measure once quantum finite-state automaton A_{MO-QFA} accepts the input word with probability $\sum \alpha_i^2$ and rejects $\sum \beta_j^2$.

Definition 2.3.2. A measure-many quantum finite-state automaton (MM-QFA) is defined by a 6-tuple as follows [17]

$$A_{MM-QFA} = (Q;\ \Sigma;\ \delta;\ q_0;\ Q_{acc};\ Q_{rej})$$

where

(1) Q is a finite set of states,
(2) Σ is an input alphabet and $\Gamma = \Sigma \cup \{\ \sharp;\ \$\ \}$ is working alphabet of A_{MM-QFA}, where $\sharp$ and $\$(\notin \Sigma)$ are the left and the right end-markers,
(3) δ is the transition function $\delta : Q \times \Gamma \times Q \to C_{[0,1]}$, which represents the amplitudes that flow from the state q to the state q' after reading symbol σ,
(4) $q_0 \in Q$ is the initial state,
(5) $Q_{acc} \subseteq Q$ and $Q_{rej} \subseteq Q$ are sets of accepting and rejecting states ($Q_{acc} \cap Q_{rej} = \emptyset$).

The states in Q_{acc} and Q_{rej} are halting states and the states in $Q_{non} = Q \setminus (Q_{acc} \cup Q_{rej})$ are non-halting states.

For all states $q_1, q_2, q' \in Q$ and symbols $\sigma \in \Gamma$, the function δ must be unitary. And it is assumed that an input word starts with the left end-marker and ends with the right end-marker.

The linear superposition of the automaton's A_{MM-QFA} states is also represented by a n-dimensional complex unit vector, where $n = |Q|$. The vector is denoted by $|\phi\rangle = \sum_{i=1}^{n} \alpha_i |q_i\rangle$, where $\{|q_i\rangle\}$ is the set orthonormal basis vectors corresponding to the states of the automaton A_{MM-QFA} and the transition function δ is represented by a set of unitary matrices $\{V_\sigma\}_{\sigma\in\Gamma}$, where V_σ is the unitary transition of the automaton A_{MM-QFA} after reading the symbol σ and is defined by $V_\sigma(|\ q\rangle) = \sum_{q'\in Q} \delta(q, \sigma, q') \mid q'\rangle$.

A computation of the automaton A_{MM-QFA} on the input word $\sharp\sigma_1\sigma_2 \dots \sigma_n\ \$$ proceeds as follows. It starts in the superposition $|q_0\rangle$, then a transition corresponding to the current input letter is performed. After every transition, the automaton A measures its state with respect to the observable $E_{acc} \oplus E_{rej} \oplus E_{non}$ where $E_{acc} = span\{|q\rangle : q \in Q_{acc}\}$, $E_{rej} = span\{|q\rangle : q \in Q_{rej}\}$ and $E_{non} = span\{|q\rangle : q \in Q_{non}\}$. If the observed state of the automaton A_{MM-QFA} is in E_{acc} subspace, then it accepts the input; if the observed state of A_{MM-QFA} is in E_{rej} subspace, then it rejects the input, otherwise the computation continues. After every measurement, the superposition collapses to the measured subspace. A measurement is represented by a diagonal zero-one projection matrices P_{acc}, P_{rej} and P_{non} which projects the vector onto E_{acc}, E_{rej} and E_{non}.

Since the automaton A_{MM-QFA} can have a non-zero probability of halting, it is useful to keep a track of the cumulative accepting and rejecting probabilities. Therefore, the state of the automaton A_{MM-QFA} is represented by a triple (ϕ, p_{acc}, p_{rej}), where p_{acc} and p_{rej} are the cumulative probabilities of accepting and rejecting. The transition of A_{MM-QFA} on reading the symbol σ is denoted by $(P_{non}\left|\phi'\right\rangle, p_{acc} + \left\|P_{acc}\phi'\right\|^2, p_{rej} + \left\|P_{rej}\phi'\right\|^2)$, where $\phi' = V_\sigma\phi$.

A measure-once quantum finite-state automaton $A_1 = (Q_1;\ \Sigma;\ \delta_1;\ q_1;\ Q_{acc_1};\ Q_{rej_1})$ with n states can be easily simulated by a measure-many quantum finite-state automaton $A_2 = (Q_2;\ \Sigma;\ \delta_2;\ q_1;\ Q_{acc_2};\ Q_{rej_2})$, where $Q_2 = \{q_1, \dots, q_{2n}\}$, states $q_1, \dots, q_n$ are non-halting states, the state q_{n+i} is accepting if the state q_i of the automaton A_1 is accepting, otherwise $q_{n+i} \in Q_{rej_2}$, the transition function δ_2 is defined as follows:

- $V_{2_\sigma}(|q\rangle) = V_{1_\sigma}(|q\rangle)$ for $q \in \{q_1, \dots, q_n\}$ and $\sigma \in \Sigma \cup \{\sharp\}$
- $V_{2_\sigma}(|q\rangle) = |q\rangle$ for $q \in \{q_{n+1}, \dots, q_{2n}\}$ and $\sigma \in \Sigma \cup \{\sharp\}$
- $V_{2_\$}(|q\rangle) = \sum_{q_{n+j}\in\{q_{n+1},\dots,q_{2n}\}} \delta(q, \$, q_{j+n}) \left|q_{j+n}\right\rangle$ if
  $V_{1_\$}(|q\rangle) = \sum_{q_j\in Q_1} \delta(q, \$, q_j) \left|q_j\right\rangle$ for $q \in \{q_1, \dots, q_n\}$.

Definition 2.3.3. A quantum finite-state automaton A is said to accept (recognize) language L with a probability p if it accepts every word in L with a probability at least p, and rejects every word not in L with a probability at least p.

Definition 2.3.4. A quantum finite state automaton A is said to accept a language L with isolated cut-point λ if for all $x \in L$ the probability of A accepting x is greater than λ and for all $x \notin L$ the probability of A accepting x is at most λ.

Definition 2.3.5. A quantum finite-state automaton A is said to accept a language L with bounded error if there exists an $\epsilon > 0$ such that for all $x \in L$ the probability of A accepting x is greater than $\lambda + \epsilon$ and for all $x \notin L$ the probability of A accepting x is less than $\lambda - \epsilon$.

3. Measure-once quantum finite-state automata and logic

In this section, the known results in the connection between measure-once quantum finite-state automata and logic have been studied. The most popular notations of quantum finite-state automata with bounded error recognize only regular languages but not all regular languages. The logical description of these language classes should be weaker than monadic second-order logic described by Büchi, which follows from the theorem of Büchi. The first intention was to study "natural" subclasses of MSO. The most "natural" subclass of monadic second-order logic is $FO[<]$. In [10], the following theorem has been proved:

Theorem 3.1. *If a language in alphabet Σ can be recognized by a measure-once quantum finite automaton and it is $FO[<]$ definable, then it is trivial, i.e. an empty language or Σ^*.*

Afterwards, the connection between modular logic and measure-once quantum finite-state automata was observed. If we consider languages in a single letter alphabet, it is easy to see that all such languages accepted by a measure-once quantum finite-state automaton can be defined by modular logic. But it is not true for larger alphabets, in fact, there exist languages that can be recognized by measure-once quantum finite-state automata, but cannot be defined by modular logic and there are also languages that cannot be recognized by measure-once quantum finite automata, but are definable by modular logic. In [11], the formulas, for which it is possible to construct a measure-once quantum finite-state automaton recognizing the language defined by the formula, are defined.

MO-QFA recognizable languages could not be described by using these "natural" subclasses of MSO, so less standard logic should be considered, one of extensions could be the use of generalized quantifiers. Using Lindström quantifier, the following theorem has been proved:

Theorem 3.2 (*[10]*). *A language can be recognized by a measure-once quantum finite automaton if and only if this language can be described by Lindström quantifier formula corresponding to the group languages (languages recognized by deterministic finite reversible automata) using atomic formulas $Q_a(x)$.*

4. Measure-many quantum finite-state automata and logic

4.1. Measure-many quantum finite-state automata and $FO[<]$

In [12], the formulas, for which it is possible to construct a measure-many quantum finite-state automaton recognizing the language defined by the formula, are defined. But in this subsection, we study the connection between accepting probabilities of quantum finite-state automata and $FO[<]$ considering accepting probability of automata. From the fact that intersection of languages recognized by measure-once quantum finite-state automata and languages defined by $FO[<]$ follows that there are languages recognized by measure-many quantum finite-state automata which cannot be defined by $FO[<]$. However there are $FO[<]$ languages recognized by measure-many quantum finite-state automata with probability 1.

Let us look at the language class L_1, which contains all languages defined by the following rules:

(1) $a_i \in \Sigma$, $\{a_1 a_2 \dots a_k \Sigma^*\} \in \underline{L_1}$, $k \in N$
(2) if $L_i \in L_1$ and $L_j \in L_1$, then $\overline{L_i} \in L_1$, $L_i \cup L_j \in L_1$ and $L_i \cap L_j \in L_1$.

These languages can be defined by the $FO[<]$ formula. $FO[<]$ formula describing the language can be constructed as follows:

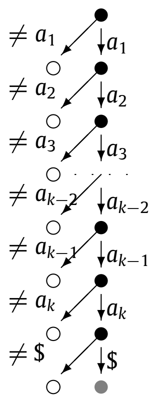
(1) $\forall x_1, x_2, \dots, x_k (first(x_1) \rightarrow Q_{a_1}(x_1) \wedge S(x_1, x_2) \rightarrow Q_{a_2}(x_2) \wedge S(x_2, x_3) \rightarrow \cdots \rightarrow Q_{a_{k-1}}(x_{k-1}) \wedge S(x_{k-1}, x_k) \rightarrow Q_{a_k}(x_k)\{\wedge last(x_k)\}^*)$
(2) if ϕ_i defines language $L_i \in L_1$ and ϕ_j recognizes the language $L_j \in L_1$, then $\neg\phi_i$ recognizes $\overline{L_i}$, $\phi_i \vee \phi_j$ - $L_i \cup L_j \in L_1$, and $\phi_i \wedge \phi_j$ - $L_i \cap L_j \in L_1$.

Lemma 4.1. *The languages in L_1 can be recognized by a measure-many quantum finite-state automaton with probability 1.*

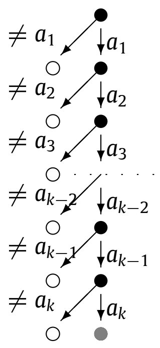
Proof. Let us look at the language $L_i \in L_1$ to construct the measure-many quantum finite-state automaton for the language L_i, we need to represent the language L_i in the tree view. The representation tree is constructed in the following way:

- *The representation tree for the language $\{a_1 a_2 \dots a_k\}$.* The representation tree has $k + 2$ levels. The edges of the tree are labeled with the letters from the alphabet $\Sigma \cup \$$. The nodes of the representation tree are colored in three colors - white, gray, and black. White and gray nodes are leaves. Each black node has $|\Sigma| + 1$ children, one outgoing edge for each letter. The parent node of the tree is black, and it is the first level of the tree. A node of level $1 < j \leq k + 1$ is colored black if the ingoing edge is labeled with a_{j-1} and white otherwise. A node of level $k+2$ is colored gray if the ingoing edge is labeled with \$ and white otherwise.

Representation tree for $\{a_1a_2...a_k\}$



Representation tree for $\{a_1a_2...a_k\Sigma^*\}$



- *The representation tree for the language* $\{a_1a_2\ldots a_k\Sigma^*\}$. The representation tree has $k+1$ levels. The edges of the tree are labeled with the letters from the alphabet $\Sigma\cup\$$. The nodes of the representation tree are painted in three colors - white, gray, and black. White and gray nodes are leaves. Each black node has $|\Sigma|+1$ children, one outgoing edge for each letter. The parent node of the tree is black, and it is the first level of the tree. A node of level $1 < j \leq k$ is colored black if the ingoing edge is labeled with a_{j-1} and white otherwise. A node of level $k+1$ is colored gray if the ingoing edge is labeled with a_k and white otherwise.
- For $\overline{L_i}$ the L_i tree gray nodes are colored white and the white nodes in gray.
- The tree of $L_i \cup L_j$ is a union of the trees for L_i and L_j. If the edge with label a from the level k to $k+1$ in one of the trees goes to white leaf, then the subtree of the other tree is chosen in the final tree, if the leaf in the one of the trees is gray, then the final tree will have this edge.
- The tree of $L_i \cap L_j$ is intersection of the trees for L_i and L_j. If the edge with label a from the level k to $k+1$ in one of the trees goes to white leaf, then the final tree will have this edge, if the leaf in the one of the trees is gray, then the subtree of the other tree is chosen in the final tree.

The measure-many quantum finite-state automaton accepting the language L_i with probability 1 is the automaton $A = (Q; \Sigma; \delta; q_0; Q_a; Q_r)$, where $Q = \{q_0, q_1, \ldots, q_n\}$, where n is the count of nodes in the representation tree of L_i. Q_a contains the states which correspond to nodes colored in gray, Q_r contains the states which correspond to nodes colored white. The transition functions are defined by the representation tree. The edge (i, j, a_l) defines the transition $|q_i\rangle = |q_j\rangle$ for letter a_l. As each node has exactly one ingoing edge, the transition function of the automaton is unitary. □

From the above lemma follows:

Theorem 4.1. *FO*$[<]$ *languages contains languages which can be recognized by a measure-many quantum finite-state automaton with probability 1.*

Let us look at the languages that can be recognized by MM-QFA with accepting probability 1 and which cannot be recognized by measure-once quantum finite-state automata. Currently we have shown a specific class of languages which can be recognized by MM-QFA with probability 1 and which are *FO*$[<]$ definable. Naturally, a question arises:

- Are there other *FO*$[<]$ languages which can be recognized by MM-QFA with probability 1, but are not in the language class L_1?

The answer to which is yes, for example, a language ab^*a can be recognized by MM-QFA with probability 1 and it is *FO*$[<]$ definable. Another question for further investigation is:

- Is it possible to give a characteristics of the languages which can be recognized by measure-many quantum finite-state automata with probability 1, but which cannot be recognized by measure-once quantum finite-state automata and are not *FO*$[<]$ definable? It is clear that such languages exist, for example, $a^{2k}b$ where $K \in Z$.

But now let us look at the other issue - can we present *FO*$[<]$ languages which cannot be recognized by a measure-many quantum finite-state automaton with probability 1? *FO*$[<]$ languages contain such languages which can only be recognized by a measure-many quantum finite automaton with probability less than 1. One of examples is the language a^*b^*.

Theorem 4.2. *There is no such probability p greater than* $\frac{1}{2}$ *for which the following holds:*

- *any measure-many quantum finite automaton accepts FO*$[<]$ *languages (recognized by MM-QFA) at least with probability p.*

Proof. Let us assume that it is possible to provide such probability p. We can express p as $\frac{1}{2}+l$. It is possible to find such n so that $l>\frac{3}{\sqrt{n-1}}$. At the same time it is known [3], that language L_n (L_n is defined as $a_1^*a_2^*a_3^*\dots a_n^*$) cannot be recognized with probability greater then $\frac{1}{2}+\frac{3}{\sqrt{n-1}}$. We obtained a contradiction which means that such p does not exist. □

4.2. Measure-many quantum finite-state automata and modular logic

In this section, we consider the connection between accepting probabilities of quantum finite automata and the modular logic. At first, let us look at the languages in $MOD[<]$.

Theorem 4.3. *Languages in the language class $MOD[<]$ can be recognized by a measure-many quantum finite-state automaton with probability 1.*

Proof. Let us suppose that it is false, that there is a language L in the language class $MOD[<]$ and it cannot be recognized with a measure-many quantum finite-state automaton with probability 1. From the Theorem 2.2 follows a monoid M ($\phi:\Sigma^*\to M, L=\phi^{-1}(N)$) recognizing the language L is a solvable group. We can construct a finite automaton A recognizing the language from the monoid M. The automaton A is defined as $(M,\Sigma,1_M,\delta,N)$, where $\delta(m,\sigma)=m*\phi(\sigma)$. If the finite automaton A is reversible then we can construct a measure-many quantum finite-state automaton recognizing the language with probability 1. As L cannot be recognized by MM-QFA with probability 1, then A must not be reversible. It means, the automaton A has a state j, a letter a and transactions $\delta(i_1,\sigma)=j$ and $\delta(i_2,\sigma)=j$ $(i_1\neq i_2)$. As the monoid M is a group, it has the inverse property (every element of the set M has an inverse element). Let us consider the following equations:

$$i_1*\phi(\sigma)*\phi(\sigma)^{-1}=j*\phi(\sigma)^{-1}=i_2*\phi(\sigma)*\phi(\sigma)^{-1},$$
$$i_1*1_M=j*\phi(\sigma)^{-1}=i_2*1_M, i_1=j*\phi(\sigma)^{-1}=i_2.$$

We obtained that i_1 is equal to i_2, which is a contradiction. If a language is in the language class $MOD[<]$ then it can be recognized with a measure-many quantum finite-state automaton with probability 1. □

It has been proved that languages in the language class $MOD[<]$ can be recognized by MM-QFA with probability 1, but are there languages which can be recognized by MM-QFA with probability 1 and which do not belong to the language class $MOD[<]$?

Lemma 4.2. *$MOD[<]$ does not contain all languages recognizable by measure-many quantum finite-state automata with probability 1.*

Proof. Let us consider the language class L_2, which contains all languages defined by the following rules:

(1) $a_i\in\Sigma,\{a_1a_2\dots a_k\}\in L_2, k\in N$
(2) if $L_i\in L_2$ and $L_j\in L_2$, then $L_i\cup L_j\in L_2$ and $L_i\cap L_j\in L_2$.

The language class $MOD[<]$ does not contain languages in L_2. It can be easily proved that such languages can be recognized by measure-many quantum finite-state automata with probability 1 and syntactic monoids of these languages do not form a group. It means that the language class $MOD[<]$ does not contain these languages. □

Now we extend the language class $MOD[<]$ by $FO[<]$ formula $S(x,y)$ ($MOD[S,<]$).

Lemma 4.3. *Languages in the language class $MOD[S,<]$ defined by the formula of the form $\exists^{(n,m)}x(Q_a(x)\wedge S(x,y)\wedge Q_b(y))$ cannot be recognized by a measure-many quantum finite-state automaton.*

Proof. The minimal deterministic finite state automaton (DFA) recognizing the language is displayed in the Fig. 1. The automaton is minimal DFA as we need to remember n fragments of ab, that means we have $2n$ states. As the minimal DFA contains forbidden constructions [5] (an automaton has the following transitions $q_i\xrightarrow{x}q_j$, $q_j\xrightarrow{x}q_j$, and $q_j\xrightarrow{y}q_i$) the language cannot be recognized by a measure-many quantum finite-state automaton. □

It is still an open question if a language $L\in MOD[S,<]-MOD[<]$ can be recognized by a measure-many quantum finite-state automaton. If a language in $L\in MOD[S,<]-MOD[<]$ can be described by the formula of the form $\exists^{(n,m)}x(Q_a(x)\wedge S(x,y)\wedge Q_b(y))$ or it can be expressed using Boolean operations between these formulas then the language L cannot be recognized by a measure-many quantum finite-state automaton.

5. Future research

The obtained results in the connection between quantum finite-state automata and logic is just a small step in the initiated research. There are different kinds of logic which could be considered, for example, temporal logic or logic with more complex quantifiers. Besides we have observed the two most popular notations of the quantum finite-state automata, but there are also others.

Currently, there is ongoing research where quantum finite-state automata are considered from the temporal logic view point and the connection between logic and the quantum finite-state automata with mixed states is studied.

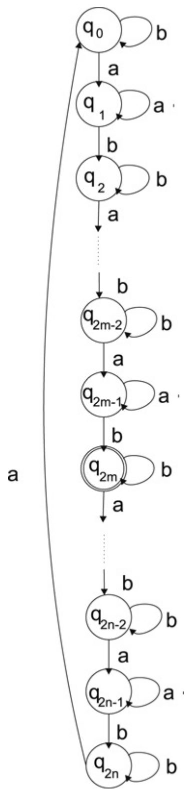


Fig. 1. The deterministic finite state automaton recognizing language defined by $\exists^{(n,m)}x(Q_a(x) \wedge S(x, y) \wedge Q_b(y))$.

Acknowledgement

The author's research has been supported by the European Social Fund.

References

[1] D. Aharonov, A. Kitaev, N. Nisan, Quantum circuits with mixed states, STOC (1998) 20–30.
[2] A. Ambainis, M. Beaudry, M. Golovkins, A. Kikusts, M. Mercer, D. Thrien, Algebraic results on quantum automata, STACS 2004 (2004) 93–104.
[3] Andris Ambainis, R.F. Bonner, R. Freivalds, A. Kikusts, Probabilities to accept languages by quantum finite automata, COCOON 1999 (1999) 174–183.
[4] A. Ambainis, R. Freivalds, 1-way quantum finite automata: Strengths, weaknesses and generalizations, FOCS (1998) 332–341.
[5] A. Ambainis, A. Kikusts, Exact results for accepting probabilities of quantum automata, Theoret. Comput. Sci. 295 (1) (2003) 5–23.
[6] A. Ambainis, J. Watrous, Two-way finite automata with quantum and classical states, Theoret. Comput. Sci. 287 (1) (2002) 299–311.
[7] A. Bertoni, C. Mereghetti, B. Palano, Quantum computing: 1-way quantum automata, Developments Language Theory 2003 (2003) 1–20.
[8] J.R. Büchi, Weak second-order arithmetic and finite automata, Z. Math. Logik Grundl. Math. 6 (1960) 66–92.
[9] J.R. Büchi, On decision method in restricted second-order arithmetic, in: Int. Congr. for Logic, Methodology and Philosophy of Science, Stanford Univ. Press, Stanford, 1962, pp. 1–11.
[10] I. Dzelme, Quantum finite automata and logics, SOFSEM 2006 (2006) 246–253.
[11] I. Dzelme-Bērziņa, Modular logic and quantum finite state automata, AQIS 2006 (2006) 145–146.
[12] I. Dzelme-Bērziņa, Formulas of first order logic and quantum finite automata, QCMC (2006).
[13] C.C. Elgot, Decision problems of finite automata design and related arithmetics, Trans. Amer. Math. Soc. 98 (1961) 21–52.
[14] R. Fagin, Generalized first order spectra and polynomial-time recognizable sets, Complexity Comput. SIAM-AMS Proc. 7 (1974) 43–73.
[15] N. Immerman, Relational queries computable in polynomial time, Inform. Control 68 (1986) 86–104.
[16] N. Immerman, Languages that capture complexity classes, SIAM J. Comput. 16 (4) (1987) 760–778.
[17] A. Kondacs, J. Watrous, On the power of quantum finite state automata, FOCS (1997) 66–75.
[18] R. McNaughton, Testing and generating infinite sequences by finite automaton, Inform. Control 9 (1966) 521–530.
[19] C. Moore, J. Crutchfield, Quantum automata and quantum grammars, Theoret. Comput. Sci. 237 (2000) 275–306.
[20] A. Mostowski, On a generalization of quantifiers, Fund. Math. 44 (1957) 12–36.
[21] A. Nayak, Optimal lower bounds for quantum automata and random access codes, FOCS (1999) 369–377. http://arxiv.org/abs/quant-ph/9904093.
[22] A. Pnueli, The temporal logic of programs, FOCS (1977) 1–14.
[23] M.O. Rabin, Decidability of second-order theories and automata on infinite trees, Trans. Amer. Math. Soc. 141 (1969) 1–35.
[24] M.P. Schüzenberger, On finite monoids having only trivial subgroups, Inform. Control 8 (1965) 283–305.
[25] H. Straubing, D. Therien, W. Thomas, Regular languages defined with generalized quantifiers, in: ICALP, in: Springer Lecture Notes in Computer Science, vol. 317, 1988, pp. 561–575.
[26] M. Vardi, Complexity of relational query languages, STOC (1982) 137–146.