Languages Recognizable by Quantum Finite Automata*

Rūsiņš Freivalds

Institute of Mathematics and Computer Science,
University of Latvia, Raiņa bulv. 29, Rīga, Latvia
Rusins.Freivalds@mii.lu.lv

Abstract. There are several nonequivalent definitions of quantum finite automata. Nearly all of them recognize only regular languages but not all regular languages. On the other hand, for all these definitions there is a result showing that there is a language l such that the size of the quantum automaton recognizing L is essentially smaller than the size of the minimal deterministic automaton recognizing L.

For most of the definitions of quantum finite automata the problem to describe the class of the languages recognizable by the quantum automata is still open. The partial results are surveyed in this paper. Moreover, for the most popular definition of the QFA, the class of languages recognizable by a QFA is not closed under union or any other binary Boolean operation where both arguments are significant.

The end of the paper is devoted to unpublished results of the description of the class of the recognizable languages in terms of the second order predicate logics. This research is influenced by the results of Büchi [1, 2], Elgot [3], Trakhtenbrot [4] (description of regular languages in terms of MSO), R.Fagin [5, 6] (description of NP in terms of ESO), von Neumann [7] (quantum logics), Barenco, Bennett et al. [8] (universal quantum gates).

1 Introduction

A quantum finite automaton (QFA) is a theoretical model for a quantum computer with a finite memory.

If we compare them with their classical (non-quantum) counterparts, QFAs have both strengths and weaknesses. The strength of QFAs is shown by the fact that quantum automata can be exponentially more space efficient than deterministic or probabilistic automata [9]. The weakness of QFAs is caused by the fact that any quantum process has to be reversible (unitary). This makes quantum automata unable to recognize some regular languages.

2 Definitions

Quantum finite automata (QFA) were introduced independently by Moore and Crutchfield [10] and Kondacs and Watrous [11]. They differ in a seemingly small

* Research supported by Grant No.05.1528 from the Latvian Council of Science and European Commission, contract IST-1999-11234.

detail. The first definition allows the measurement only at the very end of the computation process. Hence the computation is performed on the quantum information only. The second definition allows the measurement at every step of the computation. In the process of the measurement the quantum information (or rather, a part of it) is transformed into the classical information. The classical information is not processed in the subsequent steps of the computation. However, we add the classical probabilities obtained during these many measurements. There is something not 100 percent natural in this definition. We will see below that this leads to unusual properties of the quantum automata and the languages recognized by these automata.

To distinguish these quantum automata, we call them, correspondingly, MO-QFA (measure-once) and MM-QFA (measure-many).

Definition 1. *An MM-QFA is a tuple $M = (Q; \Sigma; V; q_0; Q_{acc}; Q_{rej})$ where Q is a finite set of states, Σ is an input alphabet, V is a transition function, $q_0 \in Q$ is a starting state, and $Q_{acc} \subseteq Q$ 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 called* halting states *and the states in $Q_{non} = Q - (Q_{acc} \cup Q_{rej})$ are called* non halting states. *κ and \$ are symbols that do not belong to Σ. We use κ and \$ as the left and the right endmarker, respectively. The* working alphabet *of M is $\Gamma = \Sigma \cup \{\kappa; \$\}$.*

The state of M can be any superposition of states in Q (i. e., any linear combination of them with complex coefficients). We use $|q\rangle$ to denote the superposition consisting of state q only. $l_2(Q)$ denotes the linear space consisting of all superpositions, with l_2-distance on this linear space.

The transition function V is a mapping from $\Gamma \times l_2(Q)$ to $l_2(Q)$ such that, for every $a \in \Gamma$, the function $V_a : l_2(Q) \to l_2(Q)$ defined by $V_a(x) = V(a, x)$ is a unitary transformation (a linear transformation on $l_2(Q)$ that preserves l_2 norm).

The computation of a MM-QFA starts in the superposition $|q_0\rangle$. Then transformations corresponding to the left endmarker κ, the letters of the input word x and the right endmarker \$ are applied. The transformation corresponding to $a \in \Gamma$ consists of two steps.

1. First, V_a is applied. The new superposition ψ' is $V_a(\psi)$ where ψ is the superposition before this step.
2. Then, ψ' is observed with respect to $E_{acc}, E_{rej}, E_{non}$ where $E_{acc} = span\{|q\rangle : q \in Q_{acc}\}$, $E_{rej} = span\{|q\rangle : q \in Q_{rej}\}$, $E_{non} = span\{|q\rangle : q \in Q_{non}\}$. It means that if the system's state before the measurement was

$$\psi' = \sum_{q_i \in Q_{acc}} \alpha_i |q_i\rangle + \sum_{q_j \in Q_{rej}} \beta_j |q_j\rangle + \sum_{q_k \in Q_{non}} \gamma_k |q_k\rangle$$

then the measurement accepts ψ' with probability $\Sigma \alpha_i^2$, rejects with probability $\Sigma \beta_j^2$ and continues the computation (applies transformations corresponding to next letters) with probability $\Sigma \gamma_k^2$ with the system having state $\psi = \Sigma \gamma_k |q_k\rangle$.

We regard these two transformations as reading a letter a. We use V'_a to denote the transformation consisting of V_a followed by projection to E_{non}. This is the transformation mapping ψ to the non-halting part of $V_a(\psi)$. We use V'_w to denote the product of transformations $V'_w = V'_{a_n} V'_{a_{n-1}} \dots V'_{a_2} V'_{a_1}$, where a_i is the i-th letter of the word w. We also use ψ_y to denote the non-halting part of QFA's state after reading the left endmarker κ and the word $y \in \Sigma^*$. From the notation it follows that $\psi_w = V'_{\kappa w}(|q_0\rangle)$.

We will say that an automaton recognizes a language L with probability p ($p > \frac{1}{2}$) if it accepts any word $x \in L$ with probability $\geq p$ and rejects any word $x \notin L$ with probability $\geq p$.

The MO-QFA differ from MM-QFA only in the additional requirement demanding that non-zero amplitudes can be obtained by the accepting and rejecting states no earlier than on reading the end-marker of the input word.

A probability distribution $\{(p_i, \phi_i) | 1 \leq i \leq k\}$ on pure states $\{\phi_i\}_{i=1}$ with probabilities $0 \leq p_i \leq 1$ ($\sum_{i=1}^{k}(p_i) = 1$), is called a mixed state or mixture.

A quantum finite automaton with mixed states is a tuple

$$(Q, \Sigma, \phi_{init}, \{T_\delta\}, Q_a, Q_r, Q_{non}),$$

where Q is finite a set of states, Σ is an input alphabet, ϕ_{init} is a initial mixed state, $\{T_\delta\}$ is a set of quantum transformations, which consists of defined sequence of measurements and unitary transformations, $Q_a \sqsubseteq Q$, $Q_r \sqsubseteq Q$ and $Q_{non} \sqsubseteq Q$ are sets of accepting, rejecting and non-halting states.

3 MO-Quantum Finite Automata

Sometimes even MO-QFA can be size-efficient compared with the classical FA.

Theorem 1. *[9]*

1. *For every prime p the language $L_p = \{$ the length of the input word is a multiple of p $\}$ can be recognized by a MO-QFA with no more than const $\log p$ states.*
2. *For every p a deterministic FA recognizing L_p needs at least p states.*
3. *For every p a probabilistic FA with a bounded error recognizing L_p needs at least p states.*

4 MM-Quantum Finite Automata

4.1 First Results

The previous work on 1-way quantum finite automata (QFAs) has mainly considered 3 questions:

1. What is the class of languages recognized by QFAs?
2. What accepting probabilities can be achieved?
3. How does the size of QFAs (the number of states) compare to the size of deterministic (probabilistic) automata?

In this paper, we consider the first question. The first results in this direction were obtained by Kondacs and Watrous [11].

Theorem 2. *[11]*

1. *All languages recognized by 1-way MM-QFAs are regular.*
2. *There is a regular language that cannot be recognized by a 1-way MM-QFA with probability* $\frac{1}{2} + \epsilon$ *for any* $\epsilon > 0$.

Brodsky and Pippenger [12] generalized the second part of Theorem 2 by showing that any language satisfying a certain property is not recognizable by an MM-QFA.

Theorem 3. *[12] Let L be a language and M be its minimal automaton (the smallest DFA recognizing L). Assume that there is a word x such that M contains states q_1, q_2 satisfying:*

1. $q_1 \neq q_2$,
2. *If M starts in the state q_1 and reads x, it passes to q_2,*
3. *If M starts in the state q_2 and reads x, it passes to q_2, and*
4. *There is a word y such that if M starts in q_2 and reads y, it passes to q_1,*

then L cannot be recognized by any 1-way quantum finite automaton (Fig.1).

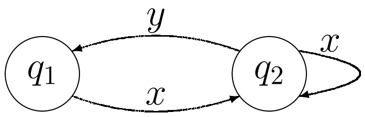
$$\begin{tikzcd} q_1 \arrow[r, bend right, "x"'] & q_2 \arrow[l, bend right, "y"'] \arrow[loop right, "x"] \end{tikzcd}$$

Fig. 1. Conditions of theorem 3

A language L with the minimal automaton not containing a fragment of Theorem 3 is called *satisfying the partial order condition* [13]. [12] conjectured that any language satisfying the partial order condition is recognizable by a 1-way QFA. In this paper, we disprove this conjecture.

Another direction of research is studying the accepting probabilities of QFAs.

Theorem 4. *[9] The language a^*b^* is recognizable by an MM-QFA with probability 0.68... but not with probability $7/9 + \epsilon$ for any $\epsilon > 0$.*

This shows that the classes of languages recognizable with different probabilities are different. Next results in this direction were obtained by [14] where the probabilities with which the languages $a_1^* \ldots a_n^*$ can be recognized are studied.

There is also a lot of results about the number of states needed for QFA to recognize different languages. In some cases, it can be exponentially less than for deterministic or even for probabilistic automata [9, 15]. In other cases, it can be exponentially bigger than for deterministic automata [16, 17].

A good survey on early results on quantum automata is Gruska [18].

4.2 Necessary Condition

First, we give the new condition which implies that the language is not recognizable by an MM-QFA. Similarly to the previous condition (Theorems 3), it can be formulated as a condition about the minimal deterministic automaton of a language. This condition is visualized in Figure 2.

Theorem 5. *[19] Let L be a language. Assume that there are words x, y, z_1, z_2 such that its minimal automaton M contains states q_1, q_2, q_3 satisfying:*

1. *$q_2 \neq q_3$,*
2. *if M starts in the state q_1 and reads x, it passes to q_2,*
3. *if M starts in the state q_2 and reads x, it passes to q_2,*
4. *if M starts in the state q_1 and reads y, it passes to q_3,*
5. *if M starts in the state q_3 and reads y, it passes to q_3,*
6. *for all words $t \in (x|y)^*$ there exists a word $t_1 \in (x|y)^*$ such that if M starts in the state q_2 and reads tt_1, it passes to q_2,*
7. *for all words $t \in (x|y)^*$ there exists a word $t_1 \in (x|y)^*$ such that if M starts in the state q_3 and reads tt_1, it passes to q_3,*
8. *if M starts in the state q_2 and reads z_1, it passes to an accepting state,*
9. *if M starts in the state q_2 and reads z_2, it passes to a rejecting state,*
10. *if M starts in the state q_3 and reads z_1, it passes to a rejecting state,*
11. *if M starts in the state q_3 and reads z_2, it passes to an accepting state.*

Then L cannot be recognized by a 1-way MM-QFA.

For languages whose minimal automaton does not contain the construction of Figure 3, this condition (together with Theorem 3) is necessary and sufficient.

Theorem 6. *[19] Let U be the class of languages whose minimal automaton does not contain "two cycles in a row" (Fig. 3). A language that belongs to U can be recognized by a 1-way MM-QFA if and only if its minimal deterministic automaton does not contain the "forbidden construction" from Theorem 3 and the "forbidden construction" from Theorem 5.*

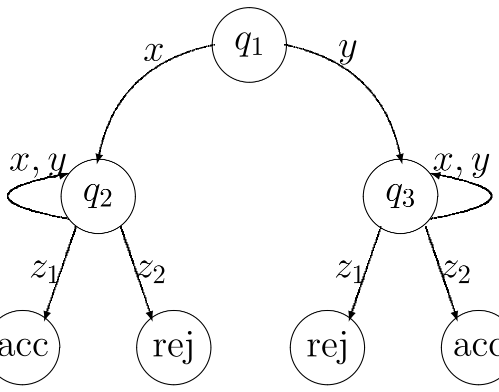

Fig. 2. Conditions of theorem 5, conditions 6 and 7 are shown symbolically

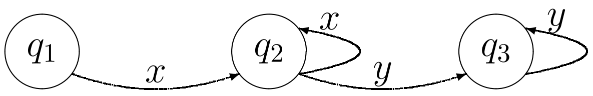

Fig. 3. Conditions of theorem 6

4.3 Non-closure Under Union

Let L_1 be the language consisting of all words that start with <u>any</u> number of letters a and after first letter b (if there is one) there is an odd number of letters a.

This language satisfies the conditions of Theorem 5. (q_1, q_2 and q_3 of Theorem 5 are just q_1, q_2 and q_3 of G_1. x, y, z_1 and z_2 are b, aba, ab and b.) Hence, it cannot be recognized by a QFA.

Consider 2 other languages L_2 and L_3 defined as follows.

L_2 consists of all words which start with an <u>even</u> number of letters a and after first letter b (if there is one) there is an odd number of letters a.

L_3 consists of all words which start with an <u>odd</u> number of letters a and after first letter b (if there is one) there is an odd number of letters a.

It is easy to see that $L_1 = L_2 \bigcup L_3$.

These languages (or rather their minimal automata) do not contain any of the "forbidden constructions" of Theorem 6. Therefore, L_2 and L_3 can be recognized by a MM-QFA and we get

Theorem 7. *[20] There are two languages L_2 and L_3 which are recognizable by a MM-QFA but the union of them $L_1 = L_2 \bigcup L_3$ is not recognizable by a MM-QFA.*

Corollary 1. *[20] The class of languages recognizable by a MM-QFA is not closed under union.*

As $L_2 \bigcap L_3 = \emptyset$ then also $L_1 = L_2 \Delta L_3$. So the class of languages recognizable by MM-QFA is not closed under symmetric difference. From this and from the fact that this class is closed under complement, it easily follows:

Corollary 2. *[20] The class of languages recognizable by a MM-QFA is not closed under any binary boolean operation where both arguments are significant.*

Theorem 8. *[19] If 2 languages L_1 and L_2 are recognizable by a MM-QFA with probabilities p_1 and p_2 and $\frac{1}{p_1} + \frac{1}{p_2} < 3$ then $L = L_1 \bigcup L_2$ is also recognizable by QFA with probability $\frac{2p_1p_2}{p_1+p_2+p_1p_2}$.*

Theorem 9. *[19] If 2 languages L_1 and L_2 are recognizable by a MM-QFA with probabilities p_1 and p_2 and $p_1 > 2/3$ and $p_2 > 2/3$, then $L = L_1 \bigcup L_2$ is recognizable by QFA with probability $p_3 > 1/2$.*

4.4 More "Forbidden" Constructions

If we allow the "two cycles in a row" construction, Theorem 6 is not longer true. More and more complicated "forbidden fragments" that imply non-recognizability by an MM-QFA are possible.

Theorem 10. *[19] Let L be a language and M be its minimal automaton. If M contains a fragment of the form shown in Figure 4 where $a, b, c, d, e, f, g, h, i \in \Sigma^*$ are words and q_0, q_a, q_b, q_c, q_{ad}, q_{ae}, q_{bd}, q_{bf}, q_{ce}, q_{cf} are states of M and*

1. *If M reads $x \in \{a, b, c\}$ in the state q_0, its state changes to q_x.*
2. *If M reads $x \in \{a, b, c\}$ in the state q_x, its state again becomes q_x.*
3. *If M reads any string consisting of a, b and c in a state q_x ($x \in \{a, b, c\}$), it moves to a state from which it can return to the same state q_x by reading some (possibly, different) string consisting of a, b and c.*
4. *If M reads $y \in \{d, e, f\}$ in the state q_x ($x \in \{a, b, c\}$), it moves to the state q_{xy}.*[1]
5. *If M reads $y \in \{a, b, c\}$ in a state q_{xy}, its state again becomes q_{xy}.*
6. *If M reads any string consisting of d, e and f in the state q_{xy} it moves to a state from which it can return to the same state q_{xy} by reading some (possibly, different) string consisting of d, e and f.*
7. *Reading g in the state q_{ad}, h in the state q_{bf} and i in the state q_{ce} leads to accepting states. Reading h in the state q_{ae}, i in the state q_{bd}, g in the state q_{cf} leads to rejecting states.*

then L is not recognizable by an MM-QFA.

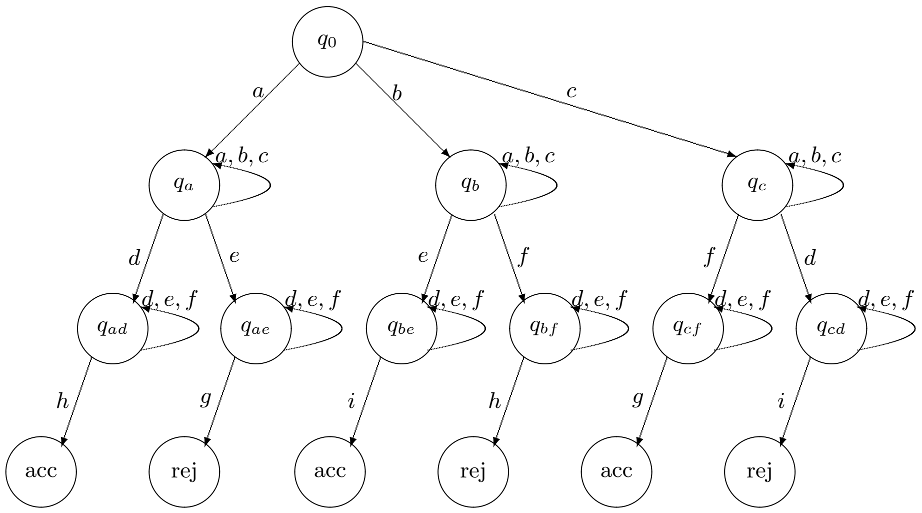

Fig. 4. Conditions of theorem 10

[1] Note: We do not have this constraint (and the next two constraints) for pairs $x = a, y = f$, $x = b$, $y = e$ and $x = c$, $y = d$ for which the state q_{xy} is not defined.

5 Descriptive Complexity

Deterministic finite automata can be regarded as a special type of Turing machines working real-time. Deterministic finite automata can also be regarded as a special type of Turing machines working in small space. Hence theory of finite automata is a part of computational coplexity theory. However, computational complexity theory was soon followed by descriptive complexity theory. The origins and the first impressive results of the decsriptive complexity theory is described by N.Immerman [21, 22].

Computational complexity began with the natural physical notions of time and space. Given a property, S, an important issue is the computational complexity of checking whether or not an input satisfies S. For a long time, the notion of complexity referred to the time or space used in the computation. A mathematician might ask, "What is the complexity of *expressing* the property S?" It should not be surprising that these two questions - that of cheching and that of expressing - are related. However, it is startling how tied they are when the second question refers to expressing the property in first-order logic. Many complexity classes originally defined in terms of time or space resources have precise definitions in first-order or second-order logic. At first, this was discovered for finite automata.

In early sixties Büchi [1, 2], Elgot [3] and Trakhtenbrot [4] showed how a logical formula may effectively be transformed into a finite state automaton accepting the language specified by the formula, and *vice versa*. It demonstrates how to relate the specification of a system behaviour (the formula) to a possible implementation (the behaviour of an automaton) - which underlies modern checking tools.

The monadic second-order (MSO) logic of one successor is a logical framework that allows one to specify string prperties using quantification over sets of positions in the string.

Now we consider an example how an automaton can be described by a formula. Let the input word have the length n in the alphabet $\{a, b\}$. Then the considered sets are subsets of the set $\{1, 2, \ldots, n\}$. $P_a(x)$ and $P_b(x)$ are, respectively, predicates

$$P_a(x) = \{\text{the symbol number } x \text{ in the input word equals } a\}$$

$$P_b(x) = \{\text{the symbol number } x \text{ in the input word equals } b\}$$

Now we wish to show how the regular language

$$\{\text{the length of the input word is a multiple of } 3\}$$

can be described. We use also individual predicates

$$S(x, y) = \{y = x + 1\}$$

$$first(x) = \{x = 1\}$$

$$last(x) = \{x = n\}$$

We use in our example three set-variables having the following meaning:

$$X_1 = \{\text{all the positions } i \text{ such that } i \equiv 1(\text{mod}3)\}$$

$$X_2 = \{\text{all the positions } i \text{ such that } i \equiv 2(\text{mod}3)\}$$

$$X_0 = \{\text{all the positions } i \text{ such that } i \equiv 0(\text{mod}3)\}$$

The MSO formula is as follows.

$$\exists X_1 X_2 X_0((X_1 \cap X_2 = \phi) \wedge (X_1 \cap X_2 = \phi) \wedge (X_1 \cap X_2 = \phi) \wedge$$

$$\wedge \forall (first(x) \Rightarrow X_1(x)) \wedge$$

$$\wedge \forall xy(S(x,y) \wedge ((X_1 \wedge X_2(y)) \vee (X_2(x) \wedge X_0(y)) \vee (X_0(x) \wedge X_1(y)) \wedge$$

$$\wedge \forall x(last(x) \Rightarrow X_0(x)))$$

It needs to be reminded that Büchi [1] considers description of automata on infinite strings. On the other hand, up to now quantum automata have been considered as processing finite words only. Perhaps there is some quantum machanics based motivation behind this restriction.

As for classical Büchi automata, in the 1970's there was relatively little interest in these automata. There was some theoretical work on automata with infinite state spaces such as pushdown tree automata. However, the decision problems usually became undecidable. Thus, while of some theoretical interest, it did not appear to have major impact on Computing Science. The situation changed on 1977 when Pnueli's paper [23] appeared. Pnueli proposed the use of Temporal Logic for reasoning about continuously operating concurrent programs. Temporal Logic is atype of modal logic that provides a formalism for describing how the truth values of assertions vary over time. While there are a variety of different systems of Temporal Logic, typical temporal operators or modalities include Fp ("sometimes p") which is true now provided there is a future moment where p holds, and Gp ("always p) which is true now provided that p holds at all future moments. As Pnueli argued, Temporal Logic seems particularly well-suited to describing correct behaviour of continouosly operating concurrent programs.

Automata provide strictly more expressive power than (ordinary) Temporal Logic. The property G_2p, meaning that at all even moments p holds, is easily described by an automaton, but not in Temporal Logic. Today Büchi and related automata are studied both from theoretical and practical viewpoint. The central technical use of automata by Büchi - to provide a decision procedure for a logical theory by reduction to the emptiness problem for the automata - remains today the main use of such automata in connection with logical theories, such as Temporal Logic, for reasoning about program correctness.

Many automata classes are now described in terms of logics. For instance, Engelfriet and Hoogeboom [24] equated 2DGSM, the family of string transductions realized by deterministic two-way finite state transducers (i.e. finite state automata equiped with a two-way input tape and a one-way output tape) and MSOS, the family obtained by restricting monadic second-order definable graph transductions to strings. Thus, string transductions that are specified in

MSO logic can be implemented on two-way finite state transducers, and vice versa.

In 1974 Fagin gave a characterization of nondeterministic polynomial time (NP) as the set of properties expressible in second-order existential logic. Some the results arising from this approach include characterizing polynomial time (P) as the set of properties expressible in first-order logic plus a least fixed point operator, and showing that the set of first-order inductive definitions for finite structures is closed under complementation.

It is well known that second-order formulas may be transformed into prenex form, with all second-order quantifiers in front. Let SO be the set of second-order expressible properties, and let ESO be the set of second-order properties that may be written in prenex form with no universal second-order quantifiers.

We consider the following example. Let the structure $G = (\{1, 2, \cdots, n\}, E)$ represent a graph of n vertices, and E be a single binary relation representing the edges of the graph. We say that the graph G is 3-colourable (in colors Red, Yellow, Blue) iff its vertices may be coloured with one of three colours such that no two adjacent vertices are the same colour. Three colourability is an NP-complete property. Consider the following ESO-formula α where R, Y, B are set-variables expressing the set of the vertices coloured correspondingly.

$$\alpha = (\exists R)(\exists Y)(\exists B)((R(x) \vee Y(x) \vee B(x)) \wedge (\forall y)(E(x, y) \Rightarrow$$

$$\neg(R(x) \wedge R(y)) \wedge \neg(Y(x) \wedge Y(y)) \wedge \neg(B(x) \wedge B(y))))$$

Observe that a graph G satisfies α iff G is 3-colourable. Fagin [5] proves that all NP-properties and only these properties are ESO-expressible.

Theorem 11. *[5] (ESO) = NP.*

Stockmeyer [25] followed this theorem by a nice characterization of the polynomial-time hierarchy.

Theorem 12. *[25] (SO) = PH.*

Definition 2. *We now define (FO + LFP) to be the set of first-order inductive definitions. We do this by adding a least fixed point operator (LFP) to first-order logic. If $\phi(R^k, x_1, \cdots, x_k)$ is an R^k -positive formula (i.e. R does not occure within any negation signs) in (FO + LFP) then $(LFP_{R^k_{x_1,\cdots,x_k}} \phi)$ is a formula in (FO + LFP) denoting the least fixed point of ϕ. We also define $IND[f(n)]$ to be the sublanguage of (FO + LFP) in which we only include least fixed points of first-order formulas ϕ for which $|\phi|$ is $O[f(n)]$. For example, the reflexive, transitive closure of E is expressible as $(LFP_{Rxy}\beta)$ and is thus in $IND[log\ n]$. Note also that,*

$$(FO + LFP) = \cup_{k=1}^{\infty} IND[n^k].$$

Immerman [26] and Vardi [27] characterized the complexity of (FO + LFP) as follows,

Theorem 13. *[26, 27] (FO + LFP) = P.*

Immerman [28] characterized the complexity of PSPACE similar way,

Theorem 14. *[28]* $PSPACE = \cup_{k=1}^{\infty} FO[2^{n^k}]$.

These theorems are most exciting. Indeed, Theorem 13 says that if we add to first-order logic the power to define new relations by induction, then we can express exactly the properties that are checkabale in polynomial time. Polynomial time is characterized using only basic logical notions and no mention of computation. The famous open problems in Theory of Computation turn out to be equivalent to purely logical problems. For instance, $P = ?NP$ is equivalent to whether or not every second-order expressible property over finite ordered structures is already expressible in first-order logic using inductive definitions.

6 Description of Languages Recognized by QFA

We noted in Section 4 (Theorem 2) that QFA recognize only regular languages but not all regular languages. Hence the logical description of these languages should be weaker that MSO considered by Büchi. The first intension is to consider "natural" subclasses of MSO. However, Theorem 7 shows that even the most popular logical operations conjunctions and disjunctions cannot be present in the logics we are seeking for. The only way out is to consider less standard logics, like Quantum Logics introduced by von Neumann[7].

Instead of $\vee, \wedge, \neg$ we use only unitary operations. However, von Neumann's quantum logics turns out to be very far removed from qubits, discrete unitary transformations and all the usual machinery of quantum finite automata. Next ideas come from fuzzy logics by L.Zadeh [29]. Predicates $P_a(x)$ were replaced by distributions of probabilities in [29]. Following this line, I constructed a logic allowing distributions of amplitudes with amplitudes being complex numbers [where *distribution* means that the total of square moduli of these values equals 1]. Finally, I use a result by D. P. DiVincenzo [30] who proved that two-bit quantum gates are universal for quantum computation. I use also a result by A.Barenco et al. [8] where the authors prove that all one-bit quantum gates (U(2)) and the two-bit exclusive-or gate (that maps Boolean values (x, y) to $(x, x \oplus y)$) is universal in the sense that all unitary operations on arbitrarily many bits n (U(2^n)) can be expressed as compositions of these gates. This allows to use in this new logics only a logical operation *exclusive-or* and arbitrary *rotations* of one qbit.

However, I do not describe here the details of this logics because in June 2005 Ilze Dzelme defended in the University of Latvia her Master thesis [31] containing the theorem 15 (below).

Generalized quantifiers were introduced by Mostowski [32]. This theorem uses the notion of *Lindström quantifiers.* [We take the definition of these quantifiers from [33]].

Consider the classical first-order existential quantifier applied to some quantifier-free formula ψ with free variable x, i.e. consider the formula $\Psi = \exists x \psi(x)$. Given an ordered structure A, we can associate a binary (i.e., 0-1) sequence a_ψ with ψ by evaluating ψ for every possible value of x from U^A and then adding 0 for false and 1 for true to a_ψ. To be more formal: If n is assigned to x then $a_{\psi(n)=1}$ iff $\psi(x)$ evaluates to true. The formula Ψ evaluates to true in A if the above defined sequence a_ψ is such that it has at least one position with the value 1. It is immediate to give a condition for sequences corresponding to a universal quantifier (all positions must be 1), or for the $\exists!$ quantifier (exactly one position must be 1), or for modular quantifiers $\exists_{\equiv k}$ (the number of 1 positions must be equivalent to 0 mod k).

Thus, it is very natural to define generalized quantifiers by considering arbitrary conditions on binary sequences (which we will call logical acceptance types). Let us give a formal definition.

Let τ be a set of s-tuples of binary sequences, i.e., τ consists of tuples $(a_1, \cdots, a_s)$ where for every $i (1 \leq i \leq s), a_i$ is a mapping from $\{1, \cdots k\}$ to $\{0, 1\}$ for some k. We call such a τ a *logical acceptance type.* The set of all s-tuples of finite binary sequences will in the following be denoted by $\tau(s)$.

Then we denote the Lindström quantifier given by τ by Q_τ. By $Q_\tau \Sigma_k - FO$ we denote the set of formulae built as follows: If $\psi_1, \cdots, \psi_s$ are $\Sigma_k - FO$ formulae, each over r free variables.

Let A be a finite structure over the corresponding signature.. Then $A \models Q_\tau \overrightarrow{x}[\psi_1(\overrightarrow{x}), \cdots, \psi_s(\overrightarrow{x})]$ if the tuple $(a_1, \cdots, a_s)$ is in τ, where the sequences a_i are defined as follows: For $1 \leq n \leq |U^A|^r, a_i(n) = 1$ if and only if $A \models \psi_i(\overrightarrow{x})$ where n is the rank of $\overrightarrow{x}$ on the order of r-tuples over $A(1 \leq i \leq s)$. $Q_\tau \Sigma_k - FO$ is defined analogously. We write $Q^0_\tau \Sigma^0_k$ for $Mod(Q^0_\tau \Sigma^0_k)$ and $Q^0_\tau \Pi^0_k$ for $Mod(Q^0_\tau \Pi^0_k)$.

Given a Lindström quantifier Q_τ, define Q^+_τ to be the set of first-order formulae in prenex normal form that starts with one quantifier Q_τ followed by arbitrary first-order formulae.

Second-order Lindström quantifiers are defined as follows. Let $\tau \in \tau(s)$ be a logical acceptance type as above. By $Q_\tau \Sigma_k - SO$ we denote the set of formulae built as follows: If $\psi_1, \cdots, \psi_s$ are $\Sigma_k - SO$ formulae, each over q free predicates $\overrightarrow{R} = R_1, R_2, \cdots, R_r$, then $Q_\tau \overrightarrow{R}[\psi_1(\overrightarrow{R}), \cdots, \psi_s(\overrightarrow{R})]$ is a $Q_\tau \Sigma_k - SO$ formula. The semantics of such a formula is defined as follows: Let r the sum of arities of all predicate symbols in $\overrightarrow{R}$. Then we can identify one possible assigment of $\overrightarrow{R}$ over a set U_A with its characteristic sequence $c_{\overrightarrow{R}}$ which is a binary string of length $|U^A|^r$.

Based on the lexicographical ordering of these strings, we define an ordering on assignments of $\overrightarrow{R}$. Let now A be a finite structure over the corresponding signature. Then $A \models Q_\tau \overrightarrow{R}[\psi_1(\overrightarrow{R}), \cdots, \psi_s(\overrightarrow{R})]$ if the tuple $(a_1, \cdots, a_s)$ is in τ, where the sequences a_i are defined as follows: For $1 \leq n \leq |U^A|^r, a_i(n) = 1$ if and only if $A \models \psi_i(\overrightarrow{x})$ where n is the rank of $\overrightarrow{x}$ in the above-sketched order of assignements of $\overrightarrow{R}(1 \leq i \leq s)$. Analogously to the first-order case, we can also define $Q_\tau \Pi_k - SO$. We use $Q_\tau - FO$ and $Q_\tau - SO$ as abbreviations for $Q_\tau \Sigma_0 - SO$ and $Q_\tau \Sigma_0 - SO$, resp.

Theorem 15. *[31] A language can be recognized by a MO-QFA if and only if this language can be described by a second-order Lindström quantifier formula corresponding to group languages.*

References

1. Büchi, J.R.: Weak second-order arithmetic and finite automata. Zeitschrift für Mathematische Logik und Grundlagen der Mathematik **6** (1960) 66–92
2. Büchi, J.R.: On a decision method in restricted second order arithmetic. In Nagel, E., ed.: Proceeding of the International Congress on Logic, Methodology and Philosophy of Science, Stanford, CA, Stanford University Press (1960) 1–11
3. Elgot, C.C.: Decision problems of finite automata design and related arithmetics. Trans. Amer. Math. Soc. **98** (1961) 21–51
4. Trakhtenbrot, B.A.: Finite automata and the logic of one-place predicates. Siberian Mathematical Journal **3** (1962) 103–131 (in Russian), English translation: American Mathematical Society Translations **59** (1966) 23–55.
5. Fagin, R.: Generalized first-order spectra and polynomial-time recognizable sets. In Karp, R.M., ed.: Complexity of Computation. Volume 7 of SIAM-AMS Proceedings. (1974) 43–73
6. Fagin, R.: Monadic generalized spectra. Zeitschrift für Mathematische Logik und Grundlagen der Mathematik **21** (1975) 89–96
7. von Neumann, J.: Mathematical Foundations of Quantum Mechanics. Princeton University Press, Princeton, NJ (1932)
8. Barenco, A., Bennett, C.H., Cleve, R., DiVincenzo, D.P., Margolus, N.H., Shor, P.W., Sleator, T., Smolin, J.A., Weinfurter, H.: Elementary gates for quantum computation. Physical Review A **52** (1995) 3457–3467
9. Ambainis, A., Freivalds, R.: 1-way quantum finite automata: Strengths, weaknesses and generalizations. In: Proc. FOCS'98. (1998) 332–341 also quant-ph/9802062[2].
10. Moore, C., Crutchfield, J.P.: Quantum automata and quantum grammars. Theor. Comput. Sci. **237** (2000) 275–306 also quant-ph/9707031.
11. Kondacs, A., Watrous, J.: On the power of quantum finite state automata. In: Proc. FOCS'97. (1997) 66–75
12. Brodsky, A., Pippenger, N.: Characterizations of 1-way quantum finite automata. SIAM J. Comput. **31** (2002) 1456–1478 also quant-ph/9903014.
13. Meyer, A.R., Thompson, C.: Remarks on algebraic decomposition of automata. Mathematical Systems Theory **3** (1969) 110–118
14. Ambainis, A., Bonner, R.F., Freivalds, R., Kikusts, A.: Probabilities to accept languages by quantum finite automata. In: Proc. COCOON'99. (1999) 174–183 also quant-ph/9904066.
15. Kikusts, A.: A small 1-way quantum finite automaton (1998) quant-ph/9810065.
16. Ambainis, A., Nayak, A., Ta-Shma, A., Vazirani, U.: Dense quantum coding and quantum finite automata. J. ACM **49** (2002) 496–511 also quant-ph/9804043.
17. Nayak, A.: Optimal lower bounds for quantum automata and random access codes. In: Proc. FOCS'99. (1999) 369–377 also quant-ph/9904093.
18. Gruska, J.: Descriptional complexity issues in quantum computing. Journal of Automata, Languages and Combinatorics **5** (2000) 191–218

[2] Quant-ph preprints are available at http://www.arxiv.org/abs/quant-ph/preprint-number.

19. Ambainis, A., Kikusts, A., Valdats, M.: On the class of languages recognizable by 1-way quantum finite automata. In Ferreira, A., Reichel, H., eds.: Proc. STACS'01. Volume 2010 of Lecture Notes in Computer Science., Springer (2001) 75–86
20. Valdats, M.: The class of languages recognizable by 1-way quantum finite automata is not closed under union. In: Proc. Int. Workshop Quantum Computation and Learning, Sundbyholm Slott, Sweden (2000) 52–64
21. Immerman, N.: Descriptive and computational complexity. In Csirik, J., Demetrovics, J., Gécseg, F., eds.: Proc. FCT'89. Volume 380 of Lecture Notes in Computer Science., Springer (1989) 244–245
22. Immerman, N.: Descriptive complexity: A logician's approach to computation. Notices of the AMS **42** (1995) 1127–1133
23. Pnueli, A.: The temporal logic of programs. In: Proc. FOCS'77. (1977) 1–14
24. Engelfriet, J., Hoogeboom, H.J.: MSO definable string transductions and two-way finite-state transducers. ACM Trans. Comput. Logic **2** (2001) 216–254
25. Stockmeyer, L.: The polynomial-time hierarchy. Theoretical Computer Science **3** (1977) 1–22
26. Immerman, N.: Relational queries computable in polynomial time (extended abstract). In: Proc. STOC '82, New York, NY, USA, ACM Press (1982) 147–152
27. Vardi, M.Y.: Complexity of relational query languages. In: Proc. STOC'82. (1982) 137–146
28. Immerman, N.: Upper and lower bounds for first order expressibility. J. Comput. Syst. Sci. **25** (1982) 76–98
29. Zadeh, L.A.: Fuzzy sets. Information and Control **8** (1965) 338–353
30. Vincenzo, D.P.D.: Two-bit gates are universal for quantum computation. Physical Review A **51** (1995) 1015–1022
31. Dzelme, I.: Quantum finite automata and logics. Master's thesis, University of Latvia (2005) Advisor: Freivalds, R.
32. Mostowski, A.: On a generalization of quantifiers. Fundamenta Mathematicae **44** (1957) 12–36
33. Burtschik, H.J., Vollmer, H.: Lindström quantifiers and leaf language definability. In: Electronic Colloquium on Computational Complexity. (1996) TR96–005