Quantum Automata Theory – A Review*

Mika Hirvensalo

Department of Mathematics, University of Turku, FIN-20014, Turku, Finland and
TUCS – Turku Centre for Computer Science
mikhirve@utu.fi

Abstract. In the first part of this survey paper, the notions of finite automata and regular languages are reviewed from various points of view. The middle part contains an introduction to the Hilbert space formalism of finite-level quantum systems, and the final part is a presentation of the most notable quantum finite automata models introduced up to date.

1 Finite Automata

The theory of finite automata is one of the cornerstones of theoretical computer science. Finite automata were introduced in 1940's and 1950's via a series of papers: notable ones include those of McCulloch and Pitts [31], Kleene [28], Mealy [32], Moore [33], and Rabin and Scott [37]. The notion of finite automaton, as well as the other notions of this section, are formally defined in the next section, but to understand the meaning and importance of finite automata, it may be necessary to consider various points of view.

Finite automata are theoretical models for real-time computing with a finite memory. By real-time computing we mean here that the automaton reads its input once, and gives the answer immediately when the whole input is read. Such a model is evidently an important object of research by itself, but seeing finite automata merely as computing machines gives only a part of the picture. More points of view arise from the language theory: As a language accepted by a finite automaton we understand the set of inputs the automaton "permits" in a sense defined later. It turns out that the languages accepted by finite automata are exactly regular languages, which can be built from finite languages by using concatenation, union, and Kleene star. The third point of view arises from formal power series: The supports of power series of rational functions are exactly regular languages. The fourth point of view is connected to the monoids: Rational languages are exactly the languages having a finite syntactic monoid.

Listed as such, the aforementioned viewpoints are merely mathematically provable equivalences, but the most important point lies among them: finite automata and languages they accept have many faces. It is true that finite automata have various applications from compiling and parsing [2] to image compression [15], [16], but the applications only are hardly the propelling force which has made finite automata an interesting research object for researchers over the decades. Instead, the significance of finite automata most likely arises

* Dedicated to Symeon Bozapalidis on his very special occasion.

W. Kuich and G. Rahonis (Eds.): Bozapalidis Festschrift, LNCS 7020, pp. 146–167, 2011.

from the fact that they, and the languages they determine are mathematically extremely fascinating objects: Regular languages are closed under union, intersection, complementation, concatenation and Kleene star, and all the closure properties can be proven true constructively from the automata point of view. Multiple characterizations for a single object from different viewpoints almost always enriches mathematics, and here the theory of finite automata serves as an exemplar of elegance.

1.1 Formal Definitions

The literature on automata theory is very rich, and it is certainly possible to exhaust all pages (and far more) of a short article like this by only listing all notable work on the topic, hence there is no point in trying to do so. Instead, we just mention a classic work by Eilenberg [19], and another recommendable presentation by Sheng Yu [43] from language theory viewpoint. The same minimalistic line will be followed when introducing automata theory notions: only those of major importance to this article will be presented.

This presentation does not follow literally any particular source, but the definitions presented in this section are generally recognized anyway.

Definition 1. *An alphabet Σ is a finite set. Taking concatenation as the operation, Σ^* denotes the free monoid generated by Σ. Elements of Σ^* are called* words*. The neutral element of Σ^* is denoted by* 1 *and called* the empty word*. The* length *of a word w is defined as $|w| = 0$, if $w = 1$ is the empty word, and n, if $w = a_1 \dots a_n$, where $a_i \in \Sigma$. For any word $w = a_1 a_2 \dots a_n$, its* mirror image *is defined as $w^R = a_n \dots a_2 a_1$. Any subset of Σ^* is called a* language *over Σ.*

Definition 2 (Regular languages). *A language $L \subseteq \Sigma^*$ is* regular*, if 1) L is finite or 2) L is obtained from regular languages L_1 and L_2 by either union $L_1 \cup L_2$, concatenation $L_1 L_2 = \{w_1 w_2 \mid w_1 \in L_1, w_2 \in L_2\}$, or Kleene star: $L_1^* = \{w_1 w_2 \dots w_k \mid k \geq 0, w_i \in L_1\}$.*

Any language constructed by using the above rules can be proven regular simply by showing the derivation tree. Such an expression is called *regular expression.*

Example 1. The language over alphabet $\Sigma = \{a, b\}$ consisting of words which contain an even number of letters a or at least one b is regular, as it can be presented as a regular expression $(\Sigma^* a \Sigma^* a \Sigma^*)^* \cup \Sigma^* b \Sigma^*$.

Definition 3 (DFA). *A deterministic finite automaton (DFA) $\mathcal{F}$ is a quintuple*

$$\mathcal{F} = (Q, \Sigma, \delta, q_I, F),$$

where Q is a finite set of states*, Σ an alphabet, $\delta : Q \times \Sigma \to Q$ the* transition function[1]*, $q_I \in Q$ the* initial state*, and $F \subseteq Q$ the set of* final states*.*

[1] In this article, we always assume the automata *complete*, meaning that the transition function is total. This can always be achieved by adding an extra state.

The transition function δ can be extended into a function $\delta : Q \times \Sigma^* \to Q$ by $\delta(q,1) = q$ and if $w = aw_1$, where $a \in \Sigma$ and $w_1 \in \Sigma^*$, then $\delta(q,w) = \delta(\delta(a,q),w_1)$.

Definition 4. *The* Boolean semiring $\mathbb{B} = \{0,1\}$ *is equipped with obvious multiplication and addition, but* $1+1=1$.

Definition 5. *A DFA* $\mathcal{F}$ *computes a function* $f_{\mathcal{F}} : \Sigma^* \to \mathbb{B}$ *defined as*

$$f_{\mathcal{F}}(w) = \begin{cases} 1 \text{ if } \delta(q_I,w) \in F \\ 0 \text{ otherwise.} \end{cases}$$

Definition 6 (Recognizable Languages). *The language recognized (accepted) by finite automaton* $\mathcal{F}$ *is*

$$L(\mathcal{F}) = \{w \in \Sigma^* \mid f_{\mathcal{F}}(w) = 1\}.$$

The following characterization is well known (see [19], for instance).

Theorem 1. *A language* L *is regular if and only if it is recognized by a DFA.*

It turns out that the matrix formalism is very useful for introducing generalizations and variants of DFA. We fix an order $Q = \{q_1, \ldots, q_n\}$ on the state set and for each letter $a \in \Sigma$, define a matrix

$$(M_a)_{ij} = \begin{cases} 1 \text{ if } \delta(q_j,a) = q_i \\ 0 \text{ otherwise.} \end{cases}$$

over the Boolean semiring. From the definition it is clear that M_a has exactly one 1 in each column. The initial vector $\boldsymbol{x} \in \mathbb{B}^n$ is defined so that $\boldsymbol{x}_i = 1$, if q_i is the initial state, and $\boldsymbol{x}_i = 0$ otherwise. The final state vector $\boldsymbol{y} \in \mathbb{B}^n$ is then defined so that $\boldsymbol{y}_i = 1$ if and only if $q_i \in F$. Both $\boldsymbol{x}$ and $\boldsymbol{y}$ are regarded as row vectors.

Now if $\boldsymbol{z}$ is any vector with only one 1 at position k and $\delta(q_k,a) = q_l$, then $(M_a)_{lk} = 1$ is the only nonzero element in the kth column, and hence

$$(M\boldsymbol{z}^T)_m = \sum_{j=1}^{n} (M_a)_{mj}\boldsymbol{z}_j = (M_a)_{mk} = \begin{cases} 1 \text{ if } m = l \\ 0 \text{ otherwise.} \end{cases}$$

This simply means that M_a moves the nonzero element of $\boldsymbol{z}$ from position k to position l, as the automaton moves from state q_k to state q_l when reading letter a. If we now define recursively

$$M_w = \begin{cases} I \text{ if } w = 1, \\ M_a M_{w_1} \text{ if } w = aw_1, \text{ where } a \in \Sigma, \end{cases}$$

it is then clear that $(M\boldsymbol{x}^T)_m = 1$ if and only if $\delta(q_I,w) = q_m$, and we see that

$$f_{\mathcal{F}}(w) = \boldsymbol{y}M_{w^R}\boldsymbol{x}^T. \tag{1}$$

Equation (1) offers a good starting point for generalizations.

1.2 Classical Variants

Definition 7 (NFA). *A* nondeterministic finite automaton $\mathcal{N}$ *is defined exactly as DFA, but instead of a transition function, the dynamics is determined by a transition relation* $\delta \subseteq Q \times \Sigma \times Q$*, and there may be many initial states.*

The matrix representation of an NFA is as simple as that of DFA: $(M_a)_{ij} = 1$ if and only if $(q_j, a, q_i) \in \delta$, and the initial vector $\boldsymbol{x}$ has 1 exactly at the positions corresponding to the initial states. The function computed by an NFA $\mathcal{N}$ is

$$f_{\mathcal{N}}(w) = \boldsymbol{y} M_{w^R} \boldsymbol{x}^T,$$

and the languages recognized by NFA are defined as those recognized by DFA.

In principle, nondeterminism does not bring any advantage for the language recognition; the following fact is well-known [19], [43].

Theorem 2. *For each NFA* $\mathcal{N}$ *with* n *states there is a DFA* $\mathcal{F}$ *with at most* 2^n *states so that* $f_{\mathcal{N}} = f_{\mathcal{F}}$*.*

Even though NFAs does not bring any advantage for language recognition, the complexity (number of states needed) may change essentially: It is known that for any $n \in \mathbb{N}$, there are n-state NFAs recognizing languages which cannot be recognized by any DFA with less than 2^n states [19].

For a general treatment of probabilistic automata, the reader is advised to consult [36], here the introduction is short:

Definition 8 (PFA). *A probabilistic finite automaton* $\mathcal{P}$ *is defined as DFA, but instead of transition function, there is a transition probability function* $\delta : \Sigma \times Q \times \Sigma \to [0, 1]$*, and the initial state is replaced by an* initial distribution $\boldsymbol{x} \in \mathbb{R}^n$ *so that* $\boldsymbol{x}_j \geq 0$ *and* $\sum_{j=1}^{n} \boldsymbol{x}_j = 1$*.*

The matrix form consists now of matrices over $\mathbb{R}$ defined as

$$(M_a)_{ij} = \delta(q_j, a, q_i),$$

which stands for the transition probability: $(M_a)_{ij}$ is the probability that being in state q_j and reading a symbol a, the automaton will enter state q_j. The evident requirement is then that

$$\sum_{i=1}^{n} \delta(q_j, a, q_i) = \sum_{i=1}^{n} (M_a)_{ij} = 1,$$

meaning that each column is a probability distribution. Matrices satisfying these requirements are called *Markov matrices*.

A PFA computes a function $f_{\mathcal{P}} : \Sigma^* \to [0, 1]$ defined as

$$f_{\mathcal{P}}(w) = \boldsymbol{y} M_{w^R} \boldsymbol{x}^T,$$

but as this function is not anymore $\{0,1\}$-valued, it is not so straightforward how to define the language recognized (or accepted) by a PFA. The most evident approach begins with a *cut-point* $\lambda \in [0,1]$ and continues with a definition

$$L_{>\lambda}(\mathcal{P}) = \{w \in \Sigma^* \mid f_{\mathcal{P}}(w) > \lambda\}$$

or

$$L_{\geq\lambda}(\mathcal{P}) = \{w \in \Sigma^* \mid f_{\mathcal{P}}(w) \geq \lambda\}.$$

In other words, $L_{>\lambda}(\mathcal{P})$ (resp. $L_{\geq\lambda}(\mathcal{P})$) consists of words with acceptance probability greater (resp. equal or greater) than λ.

Such languages are no longer necessarily regular [38], [35], [42], but the regularity can be guaranteed by assuming that the cut-point is *isolated*.

Definition 9. *Let $\mathcal{P}$ be a probabilistic automaton and $\epsilon > 0$. A cut-point $\lambda \in (0,1)$ is* ϵ-isolated, *if $f_{\mathcal{P}}(w) \notin (\lambda - \epsilon, \lambda + \epsilon)$ whenever $w \in \Sigma^*$.*

The notion of isolated cut-point is very desirable for practical reasons: if values $f_{\mathcal{P}}(w)$ can get arbitrarily close to the cut-point, it is difficult in practice to decide whether the automaton accepts w. Indeed, the cases when a final state is reached with a probability of $\frac{1}{2} + \frac{1}{2^{|w|}}$ and $\frac{1}{2} - \frac{1}{2^{|w|}}$ cannot be separated reliably with less than $\Omega(2^{|w|})$ attempts. Unfortunately the isolation of the cut-point should usually emerge intentionally from the construction of the automaton, since, given an automaton $\mathcal{P}$ and cut-point λ, there is no way to determine whether the cut-point is isolated, but it is an undecidable problem [10].

On the other hand, if the cut-point is ϵ-isolated, then only a constant number (depends on ϵ) of runs is enough to determine the acceptance question with probability as close to one as desired.

The following theorem, due to Rabin [38], shows that PFA with isolated cut-point cannot recognize more languages than DFA.

Theorem 3. *Let $\mathcal{P}$ be a probabilistic automaton with n states and one final state.*[2] *Let also λ be an ϵ-isolated cut-point. Then there exists a DFA with at most $(1 + \frac{1}{\epsilon})^{n-1}$ states recognizing language $L_{\geq\lambda}(\mathcal{P}) = L_{>\lambda}(\mathcal{P})$.*

In [38] Rabin also demonstrated that the probabilistic automata can be more succinct than the deterministic ones. He indeed constructed a sequence L_n of languages and the corresponding cut-points λ_n so that each L_n is accepted by a 2-state PFA with isolated cut-point λ_n, but a DFA recognizing L_n requires at least n states.

Theorem 3 shows that if the cut-point is isolated, then PFAs can be at most exponentially more succinct than DFAs. In [21] R. Freivalds presented a subexponential gap: A sequence of languages L_n accepted by a PFA with n states and fixed cut-point isolation, whereas any DFA accepting L_n requires $\Omega(2^{\sqrt{n}})$ states. Eventually in [22] R. Freivalds proved the existence of a language sequence where the separation between PFA and DFA sizes is exponential.

[2] A probabilistic automaton $\mathcal{P}$ can be always translated into $\mathcal{P}_1$ with one more state and only one final state.

2 Syntactic Monoids

The notion of a syntactic monoid was presented by Rabin and Scott in [37] and it brings another aspect to the languages accepted by finite automata. In fact, the main idea connecting automata to monoids has been already presented: Any DFA can be represented as a set of matrices M_a over $\mathbb{B}$. As $\mathbb{B}$ is finite, it is clear that the matrices M_a generate a finite monoid. This can be represented in a bit more abstract form as follows.

Definition 10. *Let L be a language over Σ. We say that words v and u are (syntactically) congruent (with respect to L), denoted $u \sim_L v$, if for all $x, y \in \Sigma^*$ we have*

$$xuy \in L \iff xvy \in L.$$

It is straightforward to verify that the syntactic congruence is an equivalence relation, and that it is compatible with the concatenation, meaning that if $u_1 \sim_L u_2$ and $v_1 \sim_L v_2$, then also $u_1v_1 \sim_L u_2v_2$.[3] This implies that the multiplication on equivalence classes

$$[u] = \{v \in \Sigma^* \mid v \sim_L u\}$$

defined as $[u][v] = [uv]$ is a well-defined operation.

Definition 11. *The syntactic monoid of language L is the quotient*

$$M(L) = \Sigma^*/\sim = \{[u] \mid u \in \Sigma^*\}$$

equipped with operation $[u][v] = [uv]$.

Notice that the notion of syntactic monoid is defined for each language L independently of regularity or other assumptions.

Definition 12. *Let M_1 and M_2 be monoids. Mapping $\varphi : M_1 \to M_2$ is a* morphism*, if $\varphi(m_1m_2) = \varphi(m_1)\varphi(m_2)$ holds for all m_1, $m_2 \in M_1$ and $\varphi(1) = 1$.*

Definition 13. *Language L is* recognized *by a monoid M, if there exists a morphism $\varphi : \Sigma^* \to M$ and a subset $B \subseteq M$ so that $L = \varphi^{-1}(B)$.*

For each language $L \subseteq \Sigma^*$ there are some obvious choices for recognizing monoids. For instance taking $M = \Sigma^*$, φ the identity mapping and $B = L$ gives obviously a recognizing monoid. Another choice is $M = M(L)$ (the syntactic monoid), φ the projection $\varphi(w) = [w]$, and $B = \varphi(L)$.

In a very true sense, the syntactic monoid is the smallest one recognizing L: If N is another monoid recognizing L, then $M(L)$ is a quotient of a submonoid of N. To see this, let $\varphi : \Sigma^* \to N$ be a morphism and $L = \varphi^{-1}(B)$. Now for any words u and v for which $\varphi(u) = \varphi(v)$ we have also $\varphi(xuy) = \varphi(x)\varphi(u)\varphi(y) = \varphi(x)\varphi(v)\varphi(y) = \varphi(xvy)$, and hence $xuy \in L \iff xvy \in L$. This shows that in

[3] In general, a *congruence* on an algebraic structure is an equivalence relation $\sim$ compatible with the algebraic operations. For monoids, this means that $\sim$ must satisfy $u \sim v \Rightarrow xuy \sim xvy$.

the first place, relation $\varphi(u) = \varphi(v)$ is a congruence, and on the second hand, that $\varphi(u) = \varphi(v)$ implies $u \sim_L v$, meaning that relation $\sim_L$ is coarser than relation $\varphi(u) = \varphi(v)$. The remaining details are left to reader.

For any monoid M recognizing language L there is a straightforward way to construct an automaton (not necessarily with a finite state set) of recognizing L. In fact, we can define

$$\mathcal{F} = (M, \Sigma, \delta, 1, B),$$

where $\delta(m, a) = m\varphi(a)$ for each $m \in M$ and $a \in \Sigma$. It is then straightforward to see that $\delta(1, w) = \varphi(w)$, hence $\delta(1, w) \in B \iff \varphi(w) \in B$, so the language accepted by this automaton is indeed $L = \varphi^{-1}(B)$.

The aforementioned circumstances justify the following theorem, which was introduced in [37].

Theorem 4. *A language L is recognized by a finite automaton if and only if its syntactic monoid is finite.*

Definition 14. *A language L is called recognizable, if it is recognized by a finite monoid.*

Schützenberger was the first to characterize a highly nontrivial property of regular languages in terms of their syntactic monoids. He demonstrated that a language is *star-free* if and only if its syntactic monoid is aperiodic. For the notions and proofs, see [41]. Decades later, the properties of a certain subclass of quantum automata have been characterized by so-called *forbidden constructions*, which can be naturally interpreted as properties of the syntactic monoids [7].

3 Formal Power Series

We will shortly present the basics of formal power series here. For a detailed exposition, we refer to [30].

Definition 15. *Let Σ be an alphabet. A formal power series over a semiring R is a function $S : \Sigma^* \to R$. It is usual to write S as*

$$S = \sum_{w \in \Sigma^*} S(w)w,$$

and the elements of Σ are understood as (noncommutative) variables.

Example 2.

$$\frac{1}{1-ab} = 1 + ab + abab + ababab + abababab + \dots$$

is a formal power series over $\mathbb{R}$ so that $S(w) = 1$, if w is of form $(ab)^i$, and $S(w) = 0$ otherwise.

For a PFA $\mathcal{P}$ (DFA can be viewed as a subcase) we define

$$S = \sum_{w \in \Sigma^*} f_{\mathcal{P}}(w) w, \tag{2}$$

and recall that $f_{\mathcal{P}}$ can be represented as $f_{\mathcal{P}}(w) = \boldsymbol{y} M_{w^R} \boldsymbol{x}^T = (\boldsymbol{x} (M_{w^R})^T \boldsymbol{y}^T)^T$. This implies that

$$\begin{aligned} \sum_{|w|=n} f_{\mathcal{P}}(w) w &= (\boldsymbol{x} \sum_{|w|=n} w (M_{w^R})^T \boldsymbol{y}^T)^T = (\boldsymbol{x} (\sum_{a \in \Sigma} a M_a^T)^n \boldsymbol{y}^T)^T \\ &= (\boldsymbol{x} (M^T)^n \boldsymbol{y}^T)^T = \boldsymbol{y} M^n \boldsymbol{x}^T, \end{aligned}$$

where we have denoted $M^T = \sum_{a \in \Sigma} a M_a^T$. Hence

$$S = \sum_{w \in \sigma^*} f_{\mathcal{P}}(w) w = \sum_{n=0}^{\infty} \sum_{|w|=n} f_{\mathcal{P}}(w) w = \boldsymbol{y} \sum_{n=0}^{\infty} M^n \boldsymbol{x}^T = \boldsymbol{y} (1 - M)^{-1} \boldsymbol{x}^T,$$

which shows that S is a rational function in variables $a \in \Sigma$. Especially we see that if L is a recognizable language, then

$$S = \sum_{w \in L} w$$

is a rational function. This connection was initially presented by Kleene [28] and Schützenberger [40] (in fact, Schützenberger's theorem is a generalization of the below theorem):

Theorem 5. *A formal power series*

$$S = \sum_{w \in L} w$$

is rational if and only if L is a recognizable language.

4 Formalism of Finite Quantum Systems

Before introducing quantum automata, it is necessary to present the formalism shortly. For more details, see [23].

4.1 Hilbert Space Preliminaries

Mathematical description of finite-level quantum systems is built on *Hilbert spaces* of finite dimension. As an n-dimensional Hilbert space H_n we understand the complex vector space $\mathbb{C}^n$ equipped with *Hermitian inner product* $\langle \boldsymbol{x} \mid \boldsymbol{y} \rangle = x_1^* y_1 + \ldots + x_n^* y_n$. The inner product induces *norm* by $||\boldsymbol{x}|| = \sqrt{\langle \boldsymbol{x} \mid \boldsymbol{x} \rangle}$.

For each $\boldsymbol{x} = (x_1, \ldots, x_n)$ in H_n we define $|\,\boldsymbol{x}\rangle$ to be a column vector ($n \times 1$-matrix)

$$|\,\boldsymbol{x}\rangle = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}$$

and $\langle \boldsymbol{x}\,| = (x_1^*, \ldots, x_n^*)$ a row vector ($1 \times n$-matrix). $|\,\boldsymbol{x}\rangle$ is called a *ket*-vector and $\langle \boldsymbol{x}\,|$ a *bra*-vector. If necessary, we can identify H_n either with column vector space or row vector space. The set of all linear mappings $H_n \to H_n$ is denoted by $L(H_n)$. For $\boldsymbol{x}$, $\boldsymbol{y} \in H_n$ we define a mapping $|\,\boldsymbol{x}\rangle\langle\boldsymbol{y}\,|\colon H_n \to H_n$ by setting $|\,\boldsymbol{x}\rangle\langle\boldsymbol{y}\,||\,\boldsymbol{z}\rangle = \langle\boldsymbol{y}\mid\boldsymbol{z}\rangle\,|\,\boldsymbol{x}\rangle$. Clearly $|\,\boldsymbol{x}\rangle\langle\boldsymbol{y}\,|$ is a linear mapping, and in a special case $\boldsymbol{y} = \boldsymbol{x}$, $||\boldsymbol{x}|| = 1$ we see that $|\,\boldsymbol{x}\rangle\langle\boldsymbol{x}\,||\,\boldsymbol{z}\rangle = \langle\boldsymbol{x}\mid\boldsymbol{z}\rangle\,|\,\boldsymbol{x}\rangle$, meaning that $|\,\boldsymbol{x}\rangle\langle\boldsymbol{x}\,|$ is an orthogonal projection onto a one-dimensional subspace spanned by $|\,\boldsymbol{x}\rangle$. It is straightforward to interpret $|\,\boldsymbol{x}\rangle\langle\boldsymbol{y}\,|$ as a Kronecker product: If A is an $r \times s$-matrix and B an $t \times u$-matrix, then $A \otimes B$ is an $rt \times su$-matrix

$$A \otimes B = \begin{pmatrix} a_{11}B & a_{12}B & \ldots & a_{1s}B \\ a_{21}B & a_{22}B & \ldots & a_{2s}B \\ \vdots & \vdots & \ddots & \vdots \\ a_{r1}B & a_{r2}B & \ldots & a_{rs}B \end{pmatrix}.$$

Now

$$|\,\boldsymbol{x}\rangle \otimes \langle\boldsymbol{y}\,| = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} \otimes (y_1^*, y_2^*, \ldots, y_n^*) = \begin{pmatrix} x_1y_1^* & x_1y_2^* & \ldots & x_1y_n^* \\ x_2y_1^* & x_2y_2^* & \ldots & x_2y_n^* \\ \vdots & \vdots & \ddots & \vdots \\ x_ny_1^* & x_ny_2^* & \ldots & x_ny_n^* \end{pmatrix}$$

is the matrix of mapping $|\,\boldsymbol{x}\rangle\langle\boldsymbol{y}\,|$ in the natural basis.

It is worth noticing that if $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$ is an orthonormal basis, then the matrix representation of $|\,\boldsymbol{x}_i\rangle\langle\boldsymbol{x}_j\,|$ in this orthonormal basis consists only of zeros but a single one in the intersection of the ith row and the jth column. The *trace* of a mapping is the sum of diagonal elements of the matrix: $\mathrm{Tr}(A) = \sum_{j=1}^{n}\langle\boldsymbol{x}_j \mid A\boldsymbol{x}_j\rangle$. It can be shown that the trace is independent of the choice of the orthonormal basis $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$.

The mapping A is *positive*, if $\langle\boldsymbol{x} \mid A\boldsymbol{x}\rangle \geq 0$ for each $\boldsymbol{x} \in H_n$. The *adjoint* mapping A^* is defined by condition $\langle\boldsymbol{x} \mid A\boldsymbol{y}\rangle = \langle A^*\boldsymbol{x} \mid \boldsymbol{y}\rangle$, and it is easy to see that the matrix presentation for A^* is obtained from that of A by transposing and taking complex conjugates. Mapping A is *normal*, if $AA^* = A^*A$, *self-adjoint*, if $A^* = A$, and *unitary*, if $A^* = A^{-1}$. All normal mappings have a remarkable representation introduced in the following theorem, whose proof can be found in [27], for instance.

Theorem 6. *For each normal mapping A there is an orthonormal basis $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$ of H_n consisting of the eigenvectors of A so that*

$$A = \lambda_1\,|\,\boldsymbol{x}_1\rangle\langle\boldsymbol{x}_1\,| + \ldots + \lambda_n\,|\,\boldsymbol{x}_n\rangle\langle\boldsymbol{x}_n\,|\,. \tag{3}$$

The numbers λ_i are the eigenvalues of A, and representation (3) is called a spectral representation *of A.*

In terms of matrices, a spectral representation corresponds to a diagonal form. Thus the above theorem states that all normal matrices can be diagonalized unitarily, meaning that there is an orthonormal basis on which the matrix becomes diagonal. Self-adjointness then means that all the eigenvalues in (3) are real, positivity means that they are nonnegative, and unitarity that they lie in the unit circle.

4.2 States and Observables

In this article, we will not define the physical notions *state* or *observable* in a rigorous manner, but they are understood only intuitively.

Definition 16. *As an n-level quantum system we understand a physical system whose mechanics is depicted according to quantum physics and that has exactly n (but no more) perfectly distinguishable values for some observable.*

Definition 17 (States). *The states of an n-level quantum system are described as self-adjoint positive mappings of H_n with unit trace. A matrix representation of a state is called a* density matrix. *H_n is called the* state space *of the system.*

According to Theorem 6, any state S has a presentation

$$S = \lambda_1 \, |\boldsymbol{x}_1\rangle\langle\boldsymbol{x}_1| + \ldots + \lambda_n \, |\boldsymbol{x}_n\rangle\langle\boldsymbol{x}_n|,$$

where $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$ is an orthonormal basis of H_n, $1 = \mathrm{Tr}(S) = \lambda_1 + \ldots + \lambda_n$ (unit trace), and $\lambda_j \geq 0$ (positivity). It is evident that if S_1 and S_2 are states, so is also $\lambda S_1 + (1-\lambda)S_2$ for any $\lambda \in (0,1)$, meaning that the state set is *convex.*

Definition 18. *The state S is* pure*, if representation $S = \lambda S_1 + (1-\lambda)S_2$ with $S_1 \neq S_2$ implies $\lambda \in \{0,1\}$.*

According to the previous definition, pure states are the *extremals* of the state set, i.e., states that cannot be represented as a convex combination in a nontrivial way. The following theorem is well-known.

Theorem 7. *The state S is pure if and only if $S = |\boldsymbol{x}\rangle\langle\boldsymbol{x}|$ is a projection onto a one-dimensional subspace (recall that then $||\boldsymbol{x}|| = 1$ must hold).*

Pure states are also called *vector states*, since to describe $S = |\boldsymbol{x}\rangle\langle\boldsymbol{x}|$ it is enough to give $\boldsymbol{x} \in H_n$. It is required that $||\boldsymbol{x}|| = 1$, but unfortunately this does not fix $\boldsymbol{x}$ uniquely, as any $e^{i\theta}\boldsymbol{x}$ with $\theta \in \mathbb{R}$ also satisfies $\left|\left|e^{i\theta}\boldsymbol{x}\right|\right| = 1$ and belongs to the subspace generated by $\boldsymbol{x}$. Nevertheless, it is easy to see that all such vector states generate the same state, meaning that $|e^{i\theta}\boldsymbol{x}\rangle\langle e^{i\theta}\boldsymbol{x}| = |\boldsymbol{x}\rangle\langle\boldsymbol{x}|$.

Definition 19 (Observable). *A (sharp) observable of a quantum system is a self-adjoint mapping $H_n \to H_n$.*

As a self-adjoint mapping, any observable A has a spectral representation

$$A = \mu_1 \, |\boldsymbol{y}_1\rangle\langle\boldsymbol{y}_1| + \ldots + \mu_n \, |\boldsymbol{y}_n\rangle\langle\boldsymbol{y}_n|, \tag{4}$$

where $\mu_i \in \mathbb{R}$. The eigenvalues of A are the potential values of observable A. For any set X of real numbers we define

$$E_A(X) = \sum_{\{j|\mu_j \in X\}} |\boldsymbol{y}_j\rangle\langle\boldsymbol{y}_j|,$$

hence $E_A(X)$ is a projection onto the subspace spanned by those vectors $\boldsymbol{y}_j$ whose eigenvalues belong to X.

States and observables are the primary objects of quantum theory, but they have to be connected for a meaningful interpretation. This connection is presented as an axiom referred as to the *minimal interpretation* of quantum mechanics. Quantum mechanics is ultimately a probabilistic theory, meaning that the outcome, when measuring the value of an observable, is not necessarily determined even if the system is in a pure state.

Definition 20 (Minimal Interpretation). *Let notations be as before. If A is an observable and S a state of quantum system, then the probability that the observed value of A is in set X is given by*

$$\mathbb{P}_S(X) = \mathrm{Tr}(SE_A(X)).$$

Example 3. Let H_n be a state space of an n-level quantum system and $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$ an orthonormal basis. Then any unit-length vector $\boldsymbol{y} \in H_n$ can be represented as

$$\boldsymbol{y} = \alpha_1\boldsymbol{x}_1 + \ldots + \alpha_n\boldsymbol{x}_n,$$

where $|\alpha_1|^2 + \ldots + |\alpha_n|^2 = 1$. Numbers α_i are called *amplitudes* and we say that $\boldsymbol{y}$ is a *superposition* of $\boldsymbol{x_1}, \ldots, \boldsymbol{x}_n$. Let $S = |\boldsymbol{y}\rangle\langle\boldsymbol{y}|$ be a pure state and

$$A = 1\cdot \, |\boldsymbol{x}_1\rangle\langle\boldsymbol{x}_1| + 2\cdot \, |\boldsymbol{x}_2\rangle\langle\boldsymbol{x}_2| + \ldots + n\cdot \, |\boldsymbol{x}_n\rangle\langle\boldsymbol{x}_n| \tag{5}$$

be an observable. Then the probability of observing value k is given by

$$\mathbb{P}_S(j) = \mathrm{Tr}(S \, |\boldsymbol{x}_j\rangle\langle\boldsymbol{x}_j|) = |\alpha_j|^2 .$$

This can be interpreted so that if a pure state is expanded using the eigenvectors of observable A, then the coefficient α_j (which is called the amplitude of $\boldsymbol{x}_j$) induces the probability of measuring value j by $\mathbb{P}(j) = |\alpha_j|^2$ (this is known as Born probability rule). In fact, *observing a vector state*

$$\boldsymbol{y} = \alpha_1\boldsymbol{x}_1 + \ldots + \alpha_n\boldsymbol{x}_n$$

is a common term in quantum computing, and it refers to measuring observable (5). The probability of seeing j as the value of A is then given as $|\alpha_j|^2$, and the usual terminology speaks about "observing $\boldsymbol{x}_j$", which is a synonym for measuring value j.

It may be noted that the numerical values 1, 2, ..., n as potential values of observable (5) are not important, but can be replaced by an arbitrary set of distinct values.

4.3 Compound Systems

To form a description of two quantum systems with state spaces H_n and H_m, we use the tensor product construction. Hence the state space of the joint system is mn-dimensional space $H_n \otimes H_m$, and the principal objects (states, observables) can be constructed as tensor products.[4]

Especially, if S_1 and S_2 are states of subsystems, then $S_1 \otimes S_2$ is a state of the joint system. Analogously it is possible to construct an observable of the whole system from observables A_1 and A_2 of the subsystems. It is however worth noticing that space $H_n \otimes H_m$ contains much more states than those ones of form $S_1 \otimes S_2$. Indeed, a state S is called *decomposable*, if there is a representation

$$S = \sum p_i S_i^{(1)} \otimes S_i^{(2)},$$

otherwise S is *entangled*.

The subsystem states of a compound system state are defined via statistical basis: We say that S_1 and S_2 are obtained from S via *partial trace*, and are formally defined as

$$S_1 = \mathrm{Tr}_1(S) \iff \mathrm{Tr}(S_1 A) = \mathrm{Tr}(S(A \otimes I))$$

whenever $A \in L(H_n)$ is an observable. The state $S_2 = \mathrm{Tr}_2(S)$ of the second subsystem is defined analogously. For an explicit formula for S_1 and S_2, see [23].

4.4 State Transformations

States and observables are sufficient to give a description of a quantum system at a fixed time. For the purposes of quantum computing, we need to describe how quantum systems change in time. In so-called *Schrödinger picture*, the state depends on the time and the observables remain, whereas the *Heisenberg picture* is built on time-dependent observables. Both representations are mathematically equivalent, and here we choose the usual Schrödinger picture.

The task is then to describe how quantum systems change in time. Recall that a state of a quantum system is a positive, unit-trace mapping in $L(H_n)$. We should then find out the following: if S_1 and S_2 are states of a quantum system, which properties a mapping $V : L(H_n) \to L(H_n)$ taking S_1 to S_2 should satisfy? As both S_1 and S_2 are states (unit-trace positive mappings), V should preserve the unit trace. In the same spirit, V should preserve positivity. This serves a good basis when characterizing mappings V: state transformations should be trace-preserving, and positivity-preserving mappings on the state set.

Mapping $V : L(H_n) \to L(H_n)$ is called *positive*, if $V(S)$ is always a positive mapping when S is. Unfortunately it turns out that trace-preserving property and the positivity are not enough to characterize all acceptable state transformations. Instead, we need to take care of the *environment*, and say that

[4] In the matrix representations, tensor products are represented as Kronecker products.

$V : L(H_n) \to L(H_n)$ is *completely positive*, if $V \otimes I$ is a positive mapping in $H_n \otimes H_m \to H_n \otimes H_m$, where I is an identity mapping on any potential environment H_m of H_n.

The proof of the following theorem can be found in [23].

Theorem 8. *The following are equivalent:*

1. *The mapping $V : L(H_n) \to L(H_n)$ is completely positive, trace-preserving mapping.*
2. *$V(S) = \sum_{j=1}^{n^2} V_i S V_i^*$, where $V_i \in L(H_n)$ satisfy $\sum_{j=1}^{n^2} V_i^* V_i = I$.*
3. *$V(S) = \mathrm{Tr}_1(U(S \otimes E)U^*)$, where E is a pure state of the "environment" system and $U \in L(H_n \otimes H_m)$ a unitary mapping.*

Definition 21. *A quantum system is called* closed*, if its state transformations are of form $V(S) = USU^*$, where U is a unitary mapping, i.e., $U^*U = 1$.*

Notice that the "closedness" is a subcase of both conditions 2 and 3 of the previous theorem: When the system is closed, there is only one single mapping $V_1 = U$ in condition 2, and in condition 3, either the "environment system" is nonexistent or $U = U_1 \otimes I$ does not change the environment state space H_m at all.

For all mappings $A, B \in L(H_n)$ it is easy to see that $A\,|\boldsymbol{x}\rangle\langle\boldsymbol{y}|\,B = |A\boldsymbol{x}\rangle\langle B^*\boldsymbol{y}|$ holds. Hence the state transformation on a pure state $|\boldsymbol{x}\rangle\langle\boldsymbol{x}|$ of a closed system is described as follows: $V(|\boldsymbol{x}\rangle\langle\boldsymbol{x}| = U\,|\boldsymbol{x}\rangle\langle\boldsymbol{x}|\,U^* = |U\boldsymbol{x}\rangle\langle U\boldsymbol{x}|$. This means that a vector state $\boldsymbol{x}$ is transformed into $U\boldsymbol{x}$, if the system is closed. A frequently occurring phrase "quantum time evolution is unitary" simply refers to closed systems beginning at a pure state. It is also important to notice that state transformations in a closed quantum system are always reversible: from $U\boldsymbol{x}$ one can always recover $\boldsymbol{x}$ by $U\boldsymbol{x} \mapsto U^{-1}U\boldsymbol{x} = \boldsymbol{x}$.

4.5 Projection Postulate

The case of the measurement theory of quantum mechanics is far from being closed. In fact, the *measurement problem* of quantum mechanics is a profound and fundamental problem for which a satisfactory resolution is not in sight [13].

The *projection postulate* describes the state transformation in a measurement process in an simple way, but there is no easy way to embed the projection postulate into the theory of quantum mechanics consistently. To introduce the projection postulate, consider a vector state

$$\boldsymbol{x} = \alpha_1 \boldsymbol{x}_1 + \ldots + \alpha_n \boldsymbol{x}_n$$

and an observable $A = 1 \cdot |\boldsymbol{x}_1\rangle\langle\boldsymbol{x}_1| + \ldots + n \cdot |\boldsymbol{x}_n\rangle\langle\boldsymbol{x}_n|$. Now the probability of seeing value j is $|\alpha_j|^2$, and, according to the projection postulate, if value j is observed, *then the post-observation state* is $\boldsymbol{x}_j$. Projection postulate, as presented here can be naturally extended to non-pure states, too, but here we will not need such an extension. Notice that in the definition of observable A, numbers 1, 2,

..., n are not important, but can be replaced with any set of distinct numbers (which would then become the potential values of the observable).

In the theory of quantum computing, it is usually possible to avoid referring to the projection postulate, but sometimes using it makes the notations simpler.

5 Finite Quantum Automata

The theory of quantum computing was implicitly launched by Richard Feynman in 1982, when he suggested that it may be impossible to simulate quantum mechanical systems with a classical computer without an exponential slowdown [20]. Quantum computing attracted only little attention in the beginning, but nevertheless, important theoretical works were conducted by David Deutsch [17], [18]. In 1994 Peter Shor succeeded in raising the theory of quantum computing from the margin by introducing his famous quantum algorithms for factoring integers and extracting discrete logarithms in polynomial time [39]. However, the early research on quantum computing was mainly focused on quantum algorithms with unlimited computing space, and the first studies of finite memory quantum computing were presented as late as 1997.

5.1 Early Models

Quantum finite automata (QFA) were introduced in 1997 by A. Kondacs and J. Watrous [29], and independently by C. Moore and J. P. Crutchfield (although the journal version [34] we cite here appeared later). The model of Kondacs and Watrous is frequently referred as to "Measure-Many" model (MM), and that one by Moore and Crutchfield as "Measure-Once" (MO). The models are crucially different, and formally, MO-QFA seems to be the model more faithful to DFA.

More QFA variants have been introduced later, and in the sequel, we will present some of the most notable ones. When presenting the notion of QFA, the matrix formalism is evidently the most useful way to choose. The intuition behind MO-QFA is that the state set of the automaton is a closed quantum mechanical system, and all state transformation are determined by the input letters.

Let $Q = \{q_1, \ldots, q_n\}$ be the state set of the automaton. There is a "canonical" way to introduce a "quantum model" for any classical one, and here it works as follows: We introduce an n-dimensional Hilbert space with an orthonormal basis $\{|\,q_1\rangle, |\,q_2\rangle, \ldots, |\,q_n\rangle\}$. A general state of the automaton is hence a superposition of basis states $|\,q_i\rangle$:

$$|\,q\rangle = \alpha_1\,|\,q_1\rangle + \alpha_2\,|\,q_2\rangle + \ldots + \alpha_n\,|\,q_n\rangle, \tag{6}$$

where $|\alpha_1|^2 + \ldots + |\alpha_n|^2 = 1$. It is possible to select a model with an initial state $|\,q_I\rangle$, but also a superposition over all states $|\,q_i\rangle$ is equally acceptable for an initial state of the automaton.

The dynamics of an MO-automaton is that one of a closed quantum system, meaning that for any input letter a, there is a unitary mapping $U_a : H_n \to H_n$

describing how (6) changes under a read input letter. Finally, if states in F are specified to be final, then the probability of observing

$$|\,q\rangle = \alpha_1\,|\,q_1\rangle + \ldots + \alpha_n\,|\,q_n\rangle$$

in a final state is $\sum_{q\in F}|\alpha_q|^2$. By writing $P = \sum_{q\in F}|\,q\rangle\langle q\,|$ we notice that P is a projection onto the final states, and that

$$P\,|\,q\rangle = \sum_{q\in F}\alpha_q\,|\,q\rangle.$$

Hence the observation probability can be written in form

$$\sum_{q\in F}|\alpha_q|^2 = ||P\,|\,q\rangle||^2\,.$$

The definition of MO-QFA follows these outlines, but the generalization allows the initial state and the final projection to be chosen more generally.

Definition 22 (MO-QFA). *A measurement-once quantum finite automaton $\mathcal{Q}$ with n states over the alphabet Σ is a triplet $(\boldsymbol{x}, \{U_a \mid a\in\Sigma\}, P)$ where $\boldsymbol{x}\in H_n$ is the* initial vector*, $\{U_a \mid a\in\Sigma\}$ is the set of* unitary transition matrices*, and P is* the final projection*.*

Remark 1. It also is possible to define an MO-QFA as a fivetuple $(Q, \Sigma, \delta, q_I, F)$, where other components are as in the definition of DFA, but $\delta : \Sigma \to L(H_n)$ is a transition function so that each $\delta(a)$ is a unitary mapping in $L(H_n)$. The definition we used here is then obtained from this different one by specifying $\boldsymbol{x} = |\,q_I\rangle$, $U_a = \delta(a)$, and the final projection as $P = \sum_{f\in F}|\,f\rangle\langle f\,|$.

As in the case of probabilistic automata, the primary function of an MO-QFA is to compute a probability for every word $w\in\Sigma^*$, and in the MO-case it is done as follows:

$$f_{\mathcal{Q}}(w) = \left|\left|PU_{w^R}\boldsymbol{x}^T\right|\right|^2.$$

This is a quantum analogue of the probability computed by PFA. In the article presenting the Moore-Crutchfield model, the authors demonstrated that many properties of classical automata also hold for MO-QFA [34]. For instance, they prove that for any QFA $\mathcal{Q}$, the series

$$\sum_{w\in\Sigma^*} f_{\mathcal{Q}}(w)w$$

is rational, and that the following closure properties hold: Let f and $g : \Sigma^* \to [0,1]$ be functions computed by MO-QFA and $|\alpha|^2 + |\beta|^2 = 1$. Then also $\alpha f + \beta g$ is a function $\Sigma^* \to [0,1]$ computed by an MO-QFA (convexity). Moreover, fg (intersection) and $1-f$ (complement) are computed by an MO-QFA. The closure under inverse morphism was also demonstrated: if $h : \Sigma_1^* \to \Sigma^*$ is a morphism, then $fh : \Sigma_1 \to [0,1]$ is again a function computed by an MO-QFA. All the

aforementioned properties are well-known for regular languages. In [34], Moore and Crutchfield use the compactness of the unit sphere of H_n to prove also a version of the *pumping lemma*. On the other hand that version is very different from the classical pumping lemma: in the MO-QFA case, it is shown that for any $w \in \Sigma^*$ and each $\epsilon > 0$, there exists $k \in \mathbb{N}$ so that for all u, $v \in \Sigma^*$, $\left|f_{\mathcal{Q}}(uw^k v) - f_{\mathcal{Q}}(uv)\right| \leq \epsilon$.

The studies of Moore and Crutchfield can be understood within a generalized notion of a language. Whereas the basic definition simply means a subset of Σ^*, the generalized notion refers to a *fuzzy* subset of Σ^*, i.e., a function $f : \Sigma^* \to [0,1]$. The traditional notion of a (crisp) language then means that f is actually onto $\{0,1\}$. From any fuzzy language $f : \Sigma^* \to [0,1]$ it is then possible to obtain a traditional language by discretizing f, and that can be done exactly in the same way as for the probabilistic automata to obtain cut-point languages $L_{>\lambda}(\mathcal{Q})$ and $L_{\geq\lambda}(\mathcal{Q})$.

Now if the cut-point is not isolated, languages recognized by MO-QFA need not to be regular. For a concrete example, the reader is invited to design a two-state MO-QFA $\mathcal{Q}$ over the binary alphabet $\Sigma = \{a, b\}$ with the following property: $f_{\mathcal{Q}}(w) = 0$, if $|w|_a = |w|_b$ (the number of as and bs in w coincide), and $f_{\mathcal{Q}} > 0$ otherwise.

On the other hand, it was noted in [1] that the technique of Rabin can be used also for QFA (in fact, Rabin's technique applies to H_n more elegantly than to the classical probability polyhedra) to show that quantum cut-point languages with isolated cut-points are all regular (If not explicitly mentioned otherwise, the language recognition by automata will here and hereafter mean recognition of a cut-point language with an isolated cut-point). Most regularity proofs for isolated cut-point model, including the first one by Rabin [38], are based on the compactness of the state set. It is therefore interesting to notice that Symeon Bozapalidis has shown that the regularity of cut-point languages can be derived also in a different way, which applies for a large class of languages containing those recognized by MO-QFA [11].

For language recognition, the unitarity of the evolution matrices turns out to be an essential restriction. All unitary matrices are invertible, and that may lead to a sophisticated guess that the syntactic monoid of a cut-point language of QFAs should also contain the inverses of its elements. This guess turns out to be true: In [34] Moore and Crutchfield already point out that if the characteristic function of a regular language L equals to $f_{\mathcal{Q}}$ for some quantum automaton $\mathcal{Q}$, then L is a *group language*, meaning that its syntactic monoid is a group. This result was extended to all isolated cut-point languages recognized by MO-QFA in [12].

The aforementioned results show that MO-QFA and their related languages are mathematically very elegant objects: They satisfy a number of closure properties, and the model itself seems to be a satisfactory "quantum counterpart" of a probabilistic model. Unfortunately their language recognition power is very weak: isolated cut-point languages of MO-QFA are all group languages, which is a small subset of all regular languages. This is essentially different from PFA,

which are in fact genuine generalizations of DFA and can accept all regular languages (in the cut-point acceptance model). Therefore MO-QFA cannot be seen as generalizations of DFA, but rather as "variants".

The other model, MM-QFA has almost inverse properties: Language recognition power is greater, but the closure properties and mathematical elegancy are weak. In [29] Kondacs and Watrous introduced 1-way and 2-way MM-QFA, and studied mainly their language recognition power in the isolated cut-point model.

Definition 23 (MM-QFA). *A measure-many (1-way) quantum finite automaton $\mathcal{Q}$ consists of a state set Q, set of unitary transition mappings $\{U_a \mid a \in \Sigma\}$, and an initial vector $\boldsymbol{x} \in H_n$. The state set Q is divided into disjoint sets of* accepting*,* rejecting*, and* neutral *states: $Q = Q_a \cup Q_r \cup Q_n$.*

To describe the computation of, let first P_a, P_r and P_n be projections onto the subspaces H_a, H_r, and H_n spanned by the accepting, rejecting, and neutral states, respectively.

The computation of a MM-QFA goes as follows: The automaton begins at the state $\boldsymbol{x}$, and the input word w is scanned one letter at time. When letter a is read, transition U_a is applied to the state of the automaton, and then the state is observed to see whether the automaton is in an accepting, rejecting or neutral. This means measuring the value of three-valued observable $A = 1{\cdot}P_a{+}2{\cdot}P_r{+}3{\cdot}P_n$ (numbers 1, 2, and 3 are not important and can be replaced with any set of distinct numbers). If the state was observed to be accepting (resp. rejecting), then the input word is accepted (resp. rejected), and if the state seen was neutral, then the computation goes on, and the next input letter is read. The post-observation state after each observation is determined according to the projection postulate, meaning that the state

$$\sum_{q\in Q_a} \alpha_q \,|q\rangle + \sum_{q\in Q_r} \alpha_q \,|q\rangle + \sum_{q\in Q_n} \alpha_q \,|q\rangle \tag{7}$$

collapses into

$$\frac{1}{\sqrt{\mathbb{P}(a)}} \sum_{q\in Q_a} \alpha_q \,|q\rangle, \qquad \frac{1}{\sqrt{\mathbb{P}(r)}} \sum_{q\in Q_r} \alpha_q \,|q\rangle, \quad \text{or} \quad \frac{1}{\sqrt{\mathbb{P}(n)}} \sum_{q\in Q_n} \alpha_q \,|q\rangle,$$

according to which type of state was seen. Here $\mathbb{P}(a) = \sum_{q\in Q_a} |\alpha_q|^2$, $\mathbb{P}(r) = \sum_{q\in Q_r} |\alpha_q|^2$, and $\mathbb{P}(n) = \sum_{q\in Q_n} |\alpha_q|^2$ are the probabilities of seeing the automaton in state (7) in an accepting, rejecting, or neutral state, respectively. It is assumed that the input word is surrounded by special *endmarkers* which do not belong to the actual input alphabet, and that when reading the right endmarker, the automaton cannot any more stay in a neutral state, but must decide whether the word is accepted or rejected. Unlike the other models previously presented in this article, the MM-QFA can accept or reject a word without reading all letters of it.

The computational process is more complicated than for MO-QFA, but nevertheless, an MM-QFA $\mathcal{Q}$ also computes a function $f_{\mathcal{Q}} : \Sigma^* \to [0, 1]$ (the acceptance probability for each word $w \in \Sigma^*$). It was noted already in [29], that

in the isolated cut-point model, all languages recognized by MM-QFA are also regular. This follows by applying the technique of Rabin [37] or Bozapalidis [11]. However, even MM-QFAs with isolated cut-point are not powerful enough to recognize all regular languages. In fact, it was noted in [29] that even the language $L = \{a, b\}^* a$ cannot be recognized with a MM-QFA with isolated cut-point.

In this paper, we are not going to treat 2-way finite automata capable of scanning the input various times, but it may be worth emphasizing that in [29], the authors demonstrated that 2-way MM-QFA with isolated cut-point can indeed recognize all regular languages, and even more: The non-regular language $\{a^n b^n \mid n \in \mathbb{N}\}$ can be accepted with a 2-way MM-QFA with an isolated cut-point. This may appear somewhat surprising, as 2-way DFA are known equally as powerful as ordinary DFAs.

Subsequent studies have revealed rather strange behaviours of MM-QFA. It was shown in [3], that if MM-QFA are required to work it a high probability (at least $\frac{7}{9}$), then there is a classical *reversible automaton* doing the same job (see [3] for the definitions). But for weaker correctness probabilities, MM-QFA are more powerful than the classical automata. In [3] it was also demonstrated that in some cases, MM-QFA can be exponentially smaller than PFA recognizing the same language. The aforementioned probability $\frac{7}{9}$ was subsequently improved to $0.7726\ldots$ in [6].

A direction of [3] was followed in [4], where the authors constructed a hierarchy of languages recognizable by MM-QFA with isolated cut-point, but whose potential isolation tends to zero.

More troublesome news for MM-QFA were brought forth in [5], where it was shown that the class of languages recognized by MM-QFA is not closed under union, intersection, or under any genuinely binary Boolean operation. Very interestingly, the authors of [5] also launched the study of so-called *forbidden constructions*. Forbidden constructions are properties of the graph of the minimal DFA for the language in question which prohibit the language to be accepted by an MM-QFA (with isolated cut-point, of course).

The forbidden constructions can be translated into the properties of syntactic monoids, and this course of research was followed in [7]. However, any good description to syntactic monoids of languages recognized by MM-QFA is not known.

From the aforementioned description it is obvious that MM-QFA is not very elegant model mathematically, but more satisfactory models have been introduced subsequently.

5.2 Latvian QFA

In [7] the authors introduced another variant of quantum automata called *Latvian QFA*. A Latvian QFA $\mathcal{Q}$ is a sixtuple $\mathcal{Q} = (Q, \Sigma, \{U_a \mid a \in \Sigma\}, \{M_a \mid a \in \Sigma\}, q_I, F)$, Q and Σ are as for DFA, U_a is an unitary transition associated to letter a, and M_a is a measurement determined by an orthogonal decomposition $H_n = V_1^a \oplus \ldots \oplus V_k^a$. The automaton starts in state $|\, q_I\rangle$, then reads the input one letter at time. When reading the input letter a, transition U_a is applied,

and then the measurement M_a is performed. The procedure continues until the right endmarker is read. It is required that the measurement associated to the right endmarker projects either to the subspace generated by the final states or to its orthogonal complement, the subspace generated by non-final states. The probability function computed by the automaton is then the probability that a final state is seen.

In [7] it is demonstrated that Latvian QFA have far more elegant closure properties than MM-QFA. Indeed, languages recognized by Latvian QFA are closed under union, intersection, complement, and inverse morphishms. It was also shown that the languages recognized by Latvian automata are exactly those whose syntactic monoid is of wreath product form $J * G$, where J is a $\mathcal{J}$-trivial monoid and G group (for the definitions, see [7]). As form $J * G$ does not cover all finite monoids, this characterization shows also that even the Latvian QFA are not sufficient to recognize all regular languages.

To enrich the model, Bertoni & al. introduced a QFA model with a control language, where also the state is measured after each transition, and the acceptance depends on whether the sequence of the measurement results belong to the control language [9], thus obtaining an QFA model capable of recognizing a set of regular languages closed under Boolean operations. In [14] Ciamarra introduced a reversibility construction using extra space to provide a model capable of accepting all regular languages.

5.3 Open Quantum Automata

We have seen that sometimes QFAs can provide some advantage over DFA or PFA, when the efficiency is measured in the number of states needed to recognize a language. On the other hand, many models of QFA are unfortunately restricted very heavily, implying that they cannot recognize even all regular languages. The reason for the restrictions has been correctly located by the aforementioned authors: Unitary time evolution is always reversible, and the real-time computation of 1-way finite automaton does not allow any reversibility construction: It was shown by Charles Bennett that all computation can be made reversible [8], but the price to be paid for this is to introduce an extra memory to write the history of the computation. Such a construction is not applicable to finite automata.

On the other hand, the state transformations in quantum systems need not to be reversible, but reversibility is just a property of closed quantum systems. In the classical theory of computation irreversibility (and subsequent information loss) is perfectly acceptable, and hence there is no reason to assume quantum systems closed when studied in the context of computability. Therefore, the most general QFA model should be naturally based on open system state transitions (completely positive, trace-preserving mapping, see definitions in section 4.4).

Such a model was introduced in [24] and [25], and its properties were studied in [26].

Definition 24. *A QFA with open time evolution (or shortly open QFA) is a quintuple $\mathcal{Q} = (Q, \Sigma, \delta, q_I, F)$, where Q and Σ are as before, q_I is the initial state, and $F \subseteq Q$ is the set of final states, and $\delta : \Sigma \to L(L(H_n))$ is a transition function associating to each letter $a \in \Sigma$ a trace-preserving completely positive mapping $L(H_n) \to L(H_n)$, $\delta(a) = V_a$.*

The computation of an open QFA begins in (pure) state $|\, q_I\rangle\langle q_I \,|$, and each read input letter changes the state by $V_a : L(H_n) \to L(H_n)$. The acceptance probability of the word is then given by

$$f_{\mathcal{Q}}(w) = \mathrm{Tr}(PV_{w^R} \,|\, q_I\rangle\langle q_I \,|),$$

where $P = \sum_{f \in F} |\, f\rangle\langle f \,|$ is the projection onto the subspace generated by the final states.

In [24] it was shown that PFA (and subsequently DFA), is a subcase of open QFA, and in [26] it was shown that MM-QFA (and subsequently MO-QFA), and Latvian QFA can considered as subcases of QFA with open time evolution. Hence it is justified to say that QFAs with open time evolution are the genuine quantum extensions of DFAs.

In [26], it was also demonstrated the functions $f_{\mathcal{Q}} : \Sigma^* \to [0, 1]$ computed by QFA with open time evolution satisfy the same closure properties as those computed by MO-QFA, and that the formal power series

$$\sum_{w \in \Sigma^*} f_{\mathcal{Q}}(w) w$$

is rational. As open QFA is an extension of all other automata models mentioned in this paper, it follows that there are cut-point languages accepted by open QFA which are not regular. The question of regularity with isolated cut-point was not studied in [26]. But as the set of all unit-trace, positive linear mappings $H_n \to H_n$ is evidently compact, the technique of Rabin obviously applies and the isolated cut-point languages accepted by open QFA are evidently regular.

Another possibility to settle the regularity question is to notice that Bozapalidis' theorem [11] evidently covers also the dynamics of open QFAs.

References

1. Ablayev, F., Gainutdinova, A.: On the Lower Bounds for One-Way Quantum Automata. In: Nielsen, M., Rovan, B. (eds.) MFCS 2000. LNCS, vol. 1893, pp. 132–140. Springer, Heidelberg (2000)
2. Aho, A.V., Sethi, R., Ullman, J.D.: Compilers: Principles, Techniques, and Tools. Addison-Wesley, Reading (1986)
3. Ambainis, A., Freivalds, R.: 1-way quantum finite automata: strengths, weaknesses and generalizations. In: Proceedings of the 39th FOCS, pp. 376–383 (1998)
4. Ambainis, A., Bonner, R.F., Freivalds, R., Ķikusts, A.: Probabilities to Accept Languages by Quantum Finite Automata. In: Asano, T., Imai, H., Lee, D.T., Nakano, S.-i., Tokuyama, T. (eds.) COCOON 1999. LNCS, vol. 1627, pp. 174–185. Springer, Heidelberg (1999)

5. Ambainis, A., Ķikusts, A., Valdats, M.: On the class of languages recognizable by 1-way quantum finite automata. In: Ferreira, A., Reichel, H. (eds.) STACS 2001. LNCS, vol. 2010, pp. 75–86. Springer, Heidelberg (2001)
6. Ambainis, A., Ķikusts, A.: Exact results for accepting probabilities of quantum automata. Theoretical Computer Science 295(1), 3–25 (2003)
7. Ambainis, A., Beaudry, M., Golovkins, M., Ķikusts, A., Mercer, M., Thérien, D.: Algebraic Results on Quantum Automata. Theory of Computing Systems 39, 165–188 (2006)
8. Bennett, C.H.: Logical reversibility of computation. IBM Journal of Research and Development 17, 525–532 (1973)
9. Bertoni, A., Mereghetti, C., Palano, B.: Quantum computing: 1-way quantum automata. In: Ésik, Z., Fülöp, Z. (eds.) DLT 2003. LNCS, vol. 2710, pp. 1–20. Springer, Heidelberg (2003)
10. Blondel, V.D., Canterini, V.: Undecidable problems for probabilistic automata of fixed dimension. Theory of Computing systems 36, 231–245 (2003)
11. Bozapalidis, S.: Extending Stochastic and Quantum Functions. Theory of Computing Systems 2, 183–197 (2003)
12. Brodsky, A., Pippenger, N.: Characterizations of 1-Way Quantum Finite Automata. SIAM Journal on Computing 31(5), 1456–1478 (2002)
13. Busch, P., Lahti, P., Mittelstaedt, P.: The quantum theory of measurement. Springer, Heidelberg (1996)
14. Ciamarra, M.: Quantum reversibility and a new model of quantum automaton. Fundamentals of Computation Theory 13, 376–379 (2001)
15. Culik II, K., Kari, J.: Image compression using weighted finite automata. Computers and Graphics 17, 305–313 (1993)
16. Culik II, K., Kari, J.: Image-data compression using edge-optimizing algorithm for WFA inference. Journal of Information Processing and Management 30, 829–838 (1994)
17. Deutsch, D.: Quantum theory, the Church-Turing principle and the universal quantum computer. Proceedings of the Royal Society of London A 400, 97–117 (1985)
18. Deutsch, D.: Quantum computational networks. Proceedings of the Royal Society of London A 425, 73–90 (1989)
19. Eilenberg, S.: Automata, languages, and machines, vol. A. Academic Press, London (1974)
20. Feynman, R.P.: Simulating physics with computers. International Journal of Theoretical Physics 21(6/7), 467–488 (1982)
21. Freivalds, R.: On the growth of the number of states in result of the determinization of probabilistic finite automata. Avtomatika i Vichislitelnaya Tekhnika 3, 39–42 (1982) (Russian)
22. Freivalds, R.: Non-constructive Methods for Finite Probabilistic Automata. International Journal of Foundations of Computer Science 19(3), 565–580 (2008)
23. Hirvensalo, M.: Quantum Computing, 2nd edn. Springer, Heidelberg (2004)
24. Hirvensalo, M.: Some Open Problems Related to Quantum Computing. In: Paun, G., Rozenberg, G., Salomaa, A. (eds.) Current Trends in Theoretical Computer Science – The Challenge of the New Century, vol. 1. World Scientific, Singapore (2004)
25. Hirvensalo, M.: Various Aspects of Finite Quantum Automata. In: Ito, M., Toyama, M. (eds.) DLT 2008. LNCS, vol. 5257, pp. 21–33. Springer, Heidelberg (2008)
26. Hirvensalo, M.: Quantum Automata with Open Time Evolution. International Journal of Natural Computing Research 1, 70–85 (2010)

27. Horn, R.A., Johnson, C.R.: Matrix Analysis. Cambridge University Press, Cambridge (1985)
28. Kleene, S.: Representation of Events in Nerve Nets and Finite Automata. In: Shannon, C., McCarthy, J. (eds.) Automata Studies, pp. 3–41. Princeton University Press, Princeton (1956)
29. Kondacs, A., Watrous, J.: On the power of quantum finite state automata. In: Proceedings of the 38th IEEE Symposium on Foundations of Computer Science, pp. 66–75 (1997)
30. Kuich, W., Salomaa, A.: Semirings, Automata and Languages. EATCS Monographs on Theoretical Computer Science, vol. 5. Springer, Heidelberg (1986)
31. McCulloch, W., Pitts, W.: A logical calculus of the ideas immanent in nervous activity. Bulletin of Mathematical Biophysics 7, 115–133 (1943)
32. Mealy, G.: A Method for Synthesizing Sequential Circuits. Bell Systems Technical Journal 34, 1045–1079 (1955)
33. Moore, E.: Gedanken-experiments on Sequential Machines. In: Shannon, C., Ashby, W. (eds.) Automata Studies, pp. 129–153. Princeton University Press, Princeton (1956)
34. Moore, C., Crutchfield, J.P.: Quantum automata and quantum grammars. Theoretical Computer Science 237(1-2), 275–306 (2000)
35. Paz, A.: Some aspects of probabilistic automata. Information and Control 9, 26–60 (1966)
36. Paz, A.: Introduction to Probabilistic Automata. Academic Press, London (1971)
37. Rabin, M.O., Scott, D.: Finite Automata and Their Decision Problems. IBM Journal of Research and Development 3(2), 114–125 (1959)
38. Rabin, M.O.: Probabilistic Automata. Information and Control 6, 230–245 (1963)
39. Shor, P.W.: Algorithms for quantum computation: discrete log and factoring. In: Proceedings of the 35th Annual Symposium on the Foundations of Computer Science, pp. 20–22 (1994); Physical Review Letters 81(17), 3563–3566 (1998)
40. Schützenberger, M.-P.: On the Definition of a Family of Automata. Information and Control 4, 245–270 (1961)
41. Schützenberger, M.-P.: On finite monoids having only trivial subgroups. Information and Control 8, 190–194 (1965)
42. Turakainen, P.: On Stochastic Languages. Information and Control 12, 304–313 (1968)
43. Yu, S.: Regular Languages. In: Rozenberg, G., Salomaa, A. (eds.) Handbook of Formal Languages. Word, Language, Grammar, vol. 1. Springer, Heidelberg (1997)