Embedding measure-once quantum finite automata in the Feynman quantum computer model

Carlo Mereghetti[1,4,*,†], Beatrice Palano[1,4,*,†], Beatrice Simoncioni[2,†] and Dario Tamascelli[2,3,*,†]

[1] *Dipartimento di Informatica "Giovanni Degli Antoni", Università degli Studi di Milano, via Celoria 18, 20133 Milano, Italy*
[2] *Dipartimento di Fisica "Aldo Pontremoli", Università degli Studi di Milano, via Celoria 16, 20133 Milano, Italy*
[3] *Institute of Theoretical Physics and IQST, Albert-Einstein Allee 11, Ulm University, 89081 Ulm, Germany*
[4] *Gruppo Nazionale Calcolo Scientifico-Istituto Nazionale di Alta Matematica (GNCS-INdAM), Italy*

Abstract

We present an explicit embedding of measure-once quantum finite automata (MO-QFAs) into the Feynman quantum computer model. Given an MO-QFA, we construct the corresponding Feynman machine whose register encodes both the input word and the automaton state, while a cursor evolving under a Hamiltonian implements the sequential application of the automaton transition operators. The construction relies on nested SWITCH mechanisms that select transitions according to the input symbols.

We prove that, for every MO-QFA and every input word, measuring the cursor in its final position yields exactly the quantum state reached by the automaton after processing the same input. Consequently, acceptance probabilities are preserved exactly, and every language recognized by an MO-QFA with cutpoint can be recognized within the Feynman framework with the same acceptance behavior.

This establishes an explicit connection between discrete-time quantum computation and continuous-time, physically motivated Hamiltonian models.

Keywords

quantum finite automata, Feynman quantum computer, Hamiltonian quantum computation, embedding

1. Introduction

Quantum computation can be studied from both theoretical computer science and physics perspectives. In the former, computation is typically described through discrete-time models, including quantum circuits, quantum Turing machines, and quantum finite automata (see, e.g., [1, 2, 3, 4, 5]), which provide abstract frameworks for the specification and analysis of quantum computations. In the latter, computation is viewed as the continuous-time evolution of a quantum system governed by the Schrödinger equation and is naturally described through Hamiltonian models [6, 7, 8, 9]. Establishing precise connections between these two perspectives is an important goal in quantum computing, as it clarifies how abstract computational processes can be realized within physically motivated dynamics.

Among Hamiltonian models, the quantum computer proposed by Feynman [8] occupies a distinguished position. In this model, computation emerges from the interaction between a quantum register and a cursor evolving under a time-independent Hamiltonian. The propagation of the cursor controls the sequential application of computational steps, yielding a natural description in terms of continuous-time quantum walks. This separation between the computational register and the cursor makes the model particularly well suited for embedding computational processes whose evolution is already specified as a sequence of elementary unitary transformations, as is the case for MO-QFAs.

ICTCS 2026: Italian Conference on Theoretical Computer Science, September 7–9, 2026, Udine, Italy
[*] Corresponding author.
[†] These authors contributed equally.
carlo.mereghetti@unimi.it (C. Mereghetti); beatrice.palano@unimi.it (B. Palano); beatrice.simoncioni@studenti.unimi.it (B. Simoncioni); dario.tamascelli@unimi.it (D. Tamascelli)
https://mereghetti.di.unimi.it (C. Mereghetti); https://palano.di.unimi.it (B. Palano); https://www2.fisica.unimi.it/~dario.tamascelli/ (D. Tamascelli)
0000-0002-7778-7257 (C. Mereghetti); 0000-0003-3948-4658 (B. Palano); 0000-0001-6575-4469 (D. Tamascelli)

Variants of this construction have played a central role in several foundational results in quantum complexity theory, including Hamiltonian formulations of quantum computation and the study of the Local Hamiltonian problem [9, 10]. More generally, the Feynman model provides a natural connection between computation and physical evolution, since the computational process is encoded into the dynamics of the system.

Among discrete-time models, quantum finite automata (QFAs) provide a resource-constrained model of quantum computation. Introduced as quantum counterparts of classical finite automata, QFAs describe finite-state quantum devices and have been extensively studied from algorithmic, algebraic, and language-theoretic viewpoints. Of the various models, measure-once quantum finite automata (MO-QFAs) [11, 12, 13, 14, 15] are among the simplest and most thoroughly investigated. An MO-QFA processes an input word through a sequence of unitary transformations determined by the input symbols and performs a single projective measurement at the end of the computation. Despite their restricted computational resources, MO-QFAs exhibit several interesting properties, including exponential state-complexity advantages for specific language-recognition tasks and deep connections with algebraic and spectral methods [3, 16].

While the two models share important conceptual similarities, previous applications of the Feynman construction have mainly focused on universal quantum computation and complexity-theoretic encodings, whereas here we establish an exact structural embedding of a finite-state quantum computational model. Establishing a precise correspondence between these settings contributes to a better understanding of the relationship between discrete-time and continuous-time quantum computation, while also showing how finite-state quantum dynamics can be naturally represented within the Feynman framework. Furthermore, recent experimental advances in quantum walks and photonic quantum information processing [17, 18, 19, 20, 21, 22, 23] make such connections increasingly relevant from an experimental perspective.

In this paper, we establish an explicit embedding of MO-QFAs into the Feynman quantum computer. Given an MO-QFA, we construct a corresponding Feynman machine whose register encodes both the input word and the internal state of the automaton, while the cursor dynamics implements the ordered application of the automaton transition operators through Hamiltonian evolution. A key ingredient of the construction is a hierarchy of nested SWITCH mechanisms that enables the cursor to select the appropriate transition according to the currently processed input symbol.

Our main result is an explicit embedding theorem showing that the resulting Hamiltonian evolution exactly simulates the computation of the embedded MO-QFA. More precisely, we prove that, for every MO-QFA and every input word, measuring the cursor in its final position yields a register state identical to the quantum state reached by the automaton after processing the same input. Consequently, acceptance probabilities are preserved exactly, and every language recognized by an MO-QFA with cutpoint can be recognized within the Feynman framework with the same acceptance behavior.

This embedding establishes a precise correspondence between discrete-time quantum computation and continuous-time Hamiltonian computation. Beyond clarifying the relationship between discrete-time and continuous-time models, it provides a concrete realization of MO-QFA computations within a physically motivated framework and further highlights the versatility of the Feynman model.

2. Measure-once quantum finite automata

We briefly review the model of a measure-once quantum finite automaton (MO-QFA), fixing the notation and conventions used throughout the paper. Comprehensive presentations and analyses of this and other relevant QFA models can be found, for instance, in [5, 24, 25, 26].

We assume familiarity with basic notions of formal language theory (see, e.g., [27]). Let Σ be a finite alphabet. The set of all finite words over Σ is denoted by Σ^*. For a word $w \in \Sigma^*$, we denote by $|w|$ its length and by $|w|_a$ the number of occurrences of the symbol $a \in \Sigma$ in w. For every $k \geq 0$, we denote by Σ^k the set of all words of length k. In particular, $\Sigma^0 = \{\varepsilon\}$, where ε denotes the empty word. We also define $\Sigma^{\leq k} = \bigcup_{j=0}^{k} \Sigma^j$. A *language* over Σ is any subset $L \subseteq \Sigma^*$.

Let Q be a finite set of internal states, and let $\mathcal{H}_Q = \mathrm{span}\{|q\rangle : q \in Q\}$ be the associated complex Hilbert space with orthonormal basis indexed by the elements of Q. A *measure-once quantum finite automaton* (MO-QFA) [11, 12, 13, 14, 15] is a tuple

$$\mathcal{A} = (Q, \Sigma, \{M_a\}_{a\in\Sigma}, q_0, Q_F),$$

where Σ is a finite input alphabet, $q_0 \in Q$ is the initial state, $Q_F \subseteq Q$ is the set of accepting states, and, for each $a \in \Sigma$, $M_a : \mathcal{H}_Q \to \mathcal{H}_Q$ is the corresponding *unitary* operator. At any time, the state of the automaton is a unit vector

$$|\psi\rangle = \sum_{q\in Q} \alpha_q |q\rangle, \qquad \| |\psi\rangle \|^2 = \sum_{q\in Q} |\alpha_q|^2 = 1.$$

Given an input word $w = a_1 a_2 \cdots a_m \in \Sigma^*$, the automaton evolves by sequentially applying the unitary operators associated with the symbols of the input:

$$|\psi_w\rangle = M_{a_m} \cdots M_{a_1} |q_0\rangle .$$

A single projective measurement is performed at the end of the computation. Let

$$\Pi_F = \sum_{q\in Q_F} |q\rangle\langle q|$$

denote the projector onto the accepting subspace. The probability that $\mathcal{A}$ accepts the input word w is

$$P^{\mathcal{A}}_{\mathrm{accept}}(w) = \langle \psi_w| \Pi_F |\psi_w\rangle = \|\Pi_F |\psi_w\rangle \|^2.$$

It is worth remarking that the computation of a MO-QFA consists of a discrete sequence of unitary transformations followed by a single final measurement. In Section 4, we will construct a Feynman quantum computer whose continuous-time Hamiltonian evolution reproduces exactly this discrete dynamics.

Computational capabilities. Classical and quantum finite automata are primarily studied through their language-recognition capabilities. In particular, deterministic finite automata recognize exactly the class of regular languages [27].

For probabilistic and quantum models, language recognition is often defined with respect to a cutpoint. A language $L \subseteq \Sigma^*$ is recognized by an MO-QFA or a probabilistic finite automaton (PFA) [28, 29][1] with cutpoint $\lambda \in [0, 1]$ if

$$w \in L \Rightarrow P_{\mathrm{accept}}(w) \geq \lambda, \qquad w \notin L \Rightarrow P_{\mathrm{accept}}(w) < \lambda.$$

The cutpoint is said to be *isolated* if there exists $\epsilon > 0$ such that

$$|P_{\mathrm{accept}}(w) - \lambda| \geq \epsilon \qquad \text{for all } w \in \Sigma^*,$$

and *non-isolated* otherwise. It is well known that PFAs and MO-QFAs with non-isolated cutpoint recognize exactly the same class of languages, namely the class of stochastic languages [11, 12, 28, 29, 30], which strictly contains the regular languages. With isolated cutpoint, however, PFAs recognize exactly the regular languages, whereas MO-QFAs recognize precisely the class of group languages, a strict subclass of the regular languages [12, 13, 14, 15].

Despite their restricted computational resources, MO-QFAs can exhibit significant advantages in state complexity. In particular, several language families admit MO-QFA representations that are exponentially more succinct than their classical counterparts [30, 31, 32, 33, 34, 35]. This combination of mathematical simplicity and state efficiency makes MO-QFAs an attractive model both from a theoretical and from an implementation-oriented perspective [20, 21, 23, 36].

[1] Informally, a PFA can be viewed as an MO-QFA in which unitary operators are replaced by stochastic transformations.

3. The Feynman quantum computer

3.1. Architecture

The Feynman quantum computer [8] consists of two interacting subsystems, namely the register and the cursor, whose joint evolution drives the computation. Each subsystem is composed of a sufficiently large collection of two-level systems, modeled here as spin-$\frac{1}{2}$ particles, with operator algebra generated by the Pauli matrices σ_x, σ_y, and σ_z. The two components play distinct roles:

1. The *register* is the computational subsystem and consists of N_r spins, with associated Hilbert space $\mathcal{H}_r$. It is initialized in the state $|\eta(0)\rangle_r$, which encodes the input together with any ancillary information required by the computation.
2. The *cursor* controls the flow of the computation and consists of N_c spins, with associated Hilbert space $\mathcal{H}_c$. It is initialized in the state

$$|\phi(0)\rangle_c = |\uparrow \underbrace{\downarrow \cdots \downarrow}_{N_c-1}\rangle \,, \tag{3.1}$$

where $|\uparrow\rangle$ and $|\downarrow\rangle$ denote the eigenstates of σ_z with eigenvalues $+1$ and -1, respectively.

The Hilbert space of the Feynman machine and its initial state are

$$\mathcal{H} = \mathcal{H}_r \otimes \mathcal{H}_c, \qquad\qquad |\psi(0)\rangle = |\eta(0)\rangle_r \otimes |\phi(0)\rangle_c \,.$$

3.2. Linear dynamics

Consider a quantum computation on the register specified by a sequence of unitary transformations

$$A_K A_{K-1} \cdots A_1.$$

The ordered application of the operators A_j is implemented by introducing a cursor of $N_c = K + 1$ sites, initially prepared as in (3.1), and by defining the Feynman Hamiltonian

$$H = \sum_{j=1}^{K} \left(A_j \otimes \sigma_j^+ \sigma_{j-1}^- + A_j^\dagger \otimes \sigma_{j-1}^+ \sigma_j^- \right) , \tag{3.2}$$

where $\sigma_j^\pm = (\sigma_j^x \pm i\sigma_j^y)/2$ are the raising and lowering operators acting on the j-th cursor spin. Each interaction term $\sigma_j^+ \sigma_{j-1}^- + \sigma_{j-1}^+ \sigma_j^-$ in (3.2) acts nontrivially only in the presence of a spin-up excitation at either the $(j-1)$-th or the j-th cursor site, transferring the excitation between neighboring sites. When the excitation moves from site $j-1$ to site j, the operator A_j is applied to the register; the reverse transition applies $A_j^\dagger$. Since the cursor is initialized as in (3.1), with a single excitation localized at the first site, its dynamics remains confined to the one-excitation sector

$$\mathcal{H}_{|N_z=1} = \mathrm{span}\{|j\rangle_c\}_{j=0}^{K},$$

where

$$N_z = \mathbb{I}_r \otimes \sum_{j=0}^{K} \frac{1 + \sigma_j^z}{2}.$$

Here $|j\rangle_c$ denotes the cursor state with a single excitation at site j. Consequently, the Hamiltonian (3.2) can be written equivalently as

$$H = \sum_{j=1}^{K} \left(A_j \otimes |j\rangle\langle j-1| + A_j^\dagger \otimes |j-1\rangle\langle j| \right) . \tag{3.3}$$

Decomposing (3.3) as $H = H_F + H_B$, where

$$H_F = \sum_{j=1}^{K} A_j \otimes |j\rangle \langle j-1| \qquad \text{(forward Hamiltonian)}$$

$$H_B = \sum_{j=1}^{K} A_j^\dagger \otimes |j-1\rangle \langle j| \qquad \text{(backward Hamiltonian)},$$

and following Peres' cursor-based formulation of Hamiltonian computation [37], the repeated action of H_F generates the sequence of logical successors of the initial state $|\psi(0)\rangle$, illustrated in Fig. 1.

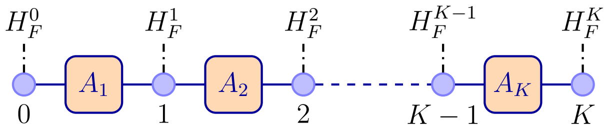

Figure 1: The linear Feynman machine. Cursor sites form a one-dimensional chain, and the operators A_j acting on the register are associated with transitions between neighboring sites. The propagation of the cursor generates the sequence of logical successors $|\phi_k\rangle$.

More precisely, the states

$$|\phi_j\rangle = H_F^j |\psi(0)\rangle = \left(\prod_{m=1}^{j} A_m\right) |\eta(0)\rangle_r \otimes |j\rangle_c, \qquad j = 0, 1, \ldots, K,$$

form an orthonormal basis of an invariant subspace of the Hilbert space $\mathcal{H} = \mathcal{H}_r \otimes \mathcal{H}_c$, where the empty product (corresponding to $j = 0$) is understood as the identity operator. The state $|\phi_j\rangle$ represents the configuration reached after j computational steps. In particular,

$$|\phi_K\rangle = A_K A_{K-1} \cdots A_1 |\eta(0)\rangle_r \otimes |K\rangle_c = A\, |\eta(0)\rangle_r \otimes |K\rangle_c,$$

corresponds to the completion of the computation.

The Hamiltonian H generates the unitary evolution

$$|\psi(t)\rangle = U(t)\, |\psi(0)\rangle, \qquad U(t) = e^{-iHt},$$

where t denotes time and, throughout this paper, we set $\hbar = 1$. Expanding the state in the logical-successor basis,

$$|\psi(t)\rangle = \sum_{j=0}^{K} c_j(t) |\phi_j\rangle, \qquad c_j(t) = \langle \phi_j | U(t) | \psi(0)\rangle.$$

The quantity $P_j(t) = |c_j(t)|^2$ is the probability of observing the machine in the logical successor $|\phi_j\rangle$ at time t. In the basis $\{|\phi_j\rangle\}_{j=0}^{K}$, the Hamiltonian reduces to the isotropic tight-binding Hamiltonian on a chain of $K+1$ sites. Its spectral decomposition is well known (see, e.g., Section 13.5 of [38]). The eigenvectors form the discrete sine basis, while the corresponding eigenvalues are $2\cos\left(\frac{n\pi}{K+2}\right)$, $1 \le n \le K+1$. Expressing the evolution operator in this eigenbasis yields

$$c_j(t) = \frac{2}{K+2} \sum_{n=1}^{K+1} e^{-i2t\cos\left(\frac{n\pi}{K+2}\right)} \sin\left(\frac{n\pi}{K+2}\right) \sin\left(\frac{n\pi(j+1)}{K+2}\right).$$

In particular,

$$P_K(t) = |c_K(t)|^2$$

is the probability of observing the final logical successor. This behavior reflects the quantum-walk nature of the cursor dynamics along the chain. Most importantly, if a measurement of the cursor position yields $|K\rangle_c$, then the register collapses to the state

$$A_K A_{K-1} \cdots A_1 |\eta(0)\rangle_r.$$

Hence, measuring the cursor position determines whether the computation has been completed and, in that case, allows one to retrieve the output by measuring the register.

However, the probability of observing the final logical successor is generally less than one. In particular, the maximum value of $P_K(t)$ scales as $K^{-2/3}$ and is attained at times of order K [39].

3.3. Conditional dynamics

The linear Feynman machine can be extended to implement conditional computational paths controlled by auxiliary qubits. This extension is based on the SWITCH construction introduced by Feynman [8].

A SWITCH is controlled by a qubit C, whose state determines which branch is followed by the cursor. A SWITCH starting at cursor site j, which applies the transformation A (respectively, B) along the upper (respectively, lower) branch, is shown schematically in Fig. 2. As in the linear case, we decompose its Hamiltonian H_{SW} into a forward part $H_{\mathrm{SW},F}$ and a backward part $H_{\mathrm{SW},B}$, corresponding respectively to the forward and backward propagation of the cursor through the SWITCH.

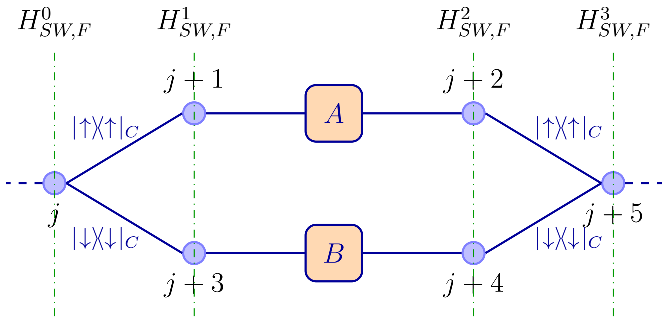

Figure 2: The SWITCH construction at cursor site j. If the control qubit C is in the state $|\uparrow\rangle_C$ (respectively, $|\downarrow\rangle_C$), the cursor follows the upper (respectively, lower) branch and the transformation A (respectively, B) is applied. The forward Hamiltonian $H_{\mathrm{SW},F}$ generates the logical successors associated with the selected branch.

$$
\begin{aligned}
H_{\mathrm{SW}}(C;A,B) &= H_{\mathrm{SW},F}(C;A,B) + H_{\mathrm{SW},B}(C;A,B),\\
H_{\mathrm{SW},F}(C;A,B) &= |\uparrow\rangle_C \langle\uparrow|_C \otimes |j+1\rangle\langle j| + A\otimes |j+2\rangle\langle j+1| + |\uparrow\rangle_C \langle\uparrow|_C \otimes |j+5\rangle\langle j+2|\\
&\quad + |\downarrow\rangle_C \langle\downarrow|_C \otimes |j+3\rangle\langle j| + B\otimes |j+4\rangle\langle j+3| + |\downarrow\rangle_C \langle\downarrow|_C \otimes |j+5\rangle\langle j+4|\,,\\
H_{\mathrm{SW},B}(C;A,B) &= (H_{\mathrm{SW},F}(C;A,B))^\dagger\,.
\end{aligned}
$$

If the control qubit is prepared in one of the computational basis states, the cursor follows the corresponding branch and applies the associated transformation to the register. More generally, if the control qubit is prepared in a superposition $\alpha\,|\uparrow\rangle_C + \beta\,|\downarrow\rangle_C$, the cursor coherently propagates along both branches, producing the corresponding superposition of computational paths.

Since the two branches have equal length, the cursor accumulates the same dynamical phase along both paths. Consequently, no relative phase is introduced by the cursor dynamics itself [40].

The notion of logical successors introduced for the linear Feynman machine extends naturally to architectures containing SWITCHes. The corresponding states remain mutually orthogonal, and the dynamics can still be described in terms of an effective one-dimensional quantum walk.

A noteworthy difference is that a SWITCH applies a transformation to the register only after a branch has been selected. Consequently, the register is not modified during the branch-selection and branch-joining transitions. The operators governing these transitions are projectors onto the state of the control qubit. Consequently, they are Hermitian but not unitary. This poses no difficulty, however, since only the overall Hamiltonian is required to be Hermitian.

The SWITCH construction is intrinsically composable. By suitably nesting SWITCHes [8], one can implement increasingly sophisticated conditional selection mechanisms.

Conceptually, a nested SWITCH network performs two complementary tasks. The first is a binary branching process that selects the computational path according to the bits of the encoded input symbol, much like the branching stage of a classical binary decision tree. Once the appropriate branch has been reached and the selected transformation has been applied to the register, the computational branches are coherently merged back into a single cursor path. This coherent recombination, which has no classical analogue, preserves the unitary evolution while restoring a unique cursor path.

An example is shown in Fig. 3, where the joint state of two control qubits determines which one of four possible transformations is applied to the register.

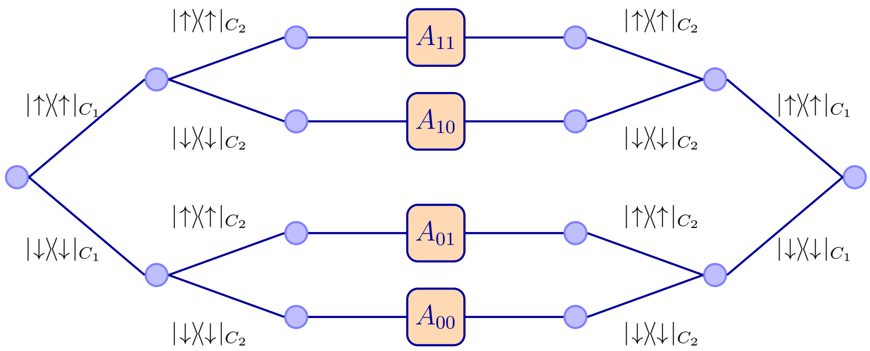

Figure 3: A nested SWITCH. The control qubits C_1 and C_2 determine the branches selected in the outer and inner SWITCHes, respectively. The overall transformation applied to the register depends on the joint state $|C_1, C_2\rangle$.

This compositional property will play a central role in the embedding construction developed in Section 4, where nested SWITCHes are used to select the transition operator associated with the input symbol currently encoded in the register.

4. Embedding MO-QFAs in the Feynman machine

Before proceeding, we clarify an important modeling aspect of the construction. While MO-QFAs are defined over input strings of arbitrary length, any realization within the Feynman model necessarily involves a finite number of degrees of freedom and must therefore be tailored to inputs of bounded size.

Accordingly, the embedding presented here does not consist of a single Feynman machine, but rather of a family of Feynman machines $\{\mathcal{F}_m\}_{m\geq 0}$, indexed by the input-length bound m. For each fixed m, the machine $\mathcal{F}_m$ simulates the behavior of the given MO-QFA on inputs in $\Sigma^{\leq m}$.

This perspective is standard when relating abstract computational models to finite physical implementations and allows us to preserve the computational behavior of the automaton while respecting the intrinsic finiteness of the underlying Hamiltonian system.

We now describe the embedding of an MO-QFA $\mathcal{A} = (Q, \Sigma, \{M_a\}_{a\in\Sigma}, q_0, Q_F)$ into the corresponding member of this family. We first explain how the input words and internal states of $\mathcal{A}$ are encoded in the register of the Feynman machine and how the transition operators M_a are implemented through cursor dynamics. We then prove that the resulting Hamiltonian evolution exactly simulates the computation of the original MO-QFA. Finally, we illustrate the construction with an explicit example.

4.1. Register and clock construction

4.1.1. Register

Input word encoding. Let Σ be the input alphabet of $\mathcal{A}$. For a fixed integer $m \geq 0$, every word $w \in \Sigma^{\leq m}$ is padded on the right with a blank symbol $\# \notin \Sigma$, yielding the word $w\#^{m-|w|} \in (\Sigma')^m$, where $\Sigma' = \Sigma \cup \{\#\}$. Each symbol of Σ' is encoded using $d = \lceil \log_2 |\Sigma'| \rceil$ bits. Thus, the padded input is represented by the binary string $\bar{w}$ of length dm:

$$\bar{w} = b_{1,1} \cdots b_{1,d}\, b_{2,1} \cdots b_{2,d} \;\cdots\; b_{m,1} \cdots b_{m,d} = b_1\, b_2 \;\cdots\; b_m,$$

where each block $b_j \in \{0,1\}^d$ encodes the j-th symbol of the padded word.

The corresponding *input register*, with Hilbert space $\mathcal{H}_{r,w}$, consists of dm qubits initialized in the computational-basis state $|\bar{w}\rangle_{r,w}$, where the j-th qubit is in the state $|\uparrow\rangle$ if the j-th bit of $\bar{w}$ is 1, and in the state $|\downarrow\rangle$ otherwise.

MO-QFA state encoding. Let $Q = \{q_0, q_1, \ldots, q_N\}$ be the set of internal states of $\mathcal{A}$, with q_0 the initial state. These states are encoded in a separate *state register*, with Hilbert space $\mathcal{H}_{r,s}$, using a unary representation: the state q_j is represented by the basis vector having a single excitation at position j. For example, if $Q = \{q_0, q_1, q_2\}$, then

$$|q_0\rangle_{r,s} = |\uparrow\downarrow\downarrow\rangle_{r,s}, \qquad |q_1\rangle_{r,s} = |\downarrow\uparrow\downarrow\rangle_{r,s}, \qquad |q_2\rangle_{r,s} = |\downarrow\downarrow\uparrow\rangle_{r,s}.$$

Register composition. The full register of the Feynman machine consists of the input and state registers. It contains $dm + |Q|$ spins, has Hilbert space $\mathcal{H}_r = \mathcal{H}_{r,w} \otimes \mathcal{H}_{r,s}$, and is initialized in the state

$$|\eta(0)\rangle_r = |\bar{w}\rangle_{r,w} \otimes |q_0\rangle_{r,s}.$$

4.1.2. Clock and cursor dynamics

MO-QFA dynamics encoding via SWITCHes. We extend the transition operators of the MO-QFA $\mathcal{A}$ to the padded alphabet $\Sigma' = \Sigma \cup \{\#\}$ by setting $M_\# = I$. Thus, for each $a \in \Sigma'$, the operator M_a acts on the state register, and $M_\#$ leaves it unchanged.

For any symbol $a \in \Sigma'$, let $\bar{a} \in \{0,1\}^d$ denote its binary encoding. The cursor must determine which symbol of the input word is being processed and apply the corresponding transition operator M_a to the state register. This task is implemented through a nested SWITCH construction of depth d, corresponding to the length of the binary encoding of symbols in Σ'. The outermost SWITCH resolves the first bit of $\bar{a}$, while the inner SWITCHes recursively resolve the remaining bits. Once the branch corresponding to the symbol $a \in \Sigma'$ has been selected, the operator M_a is applied to the state register. The subsequent cursor transitions merge the branches and prepare the next symbol-selection stage.

In this way, the nested SWITCH structure implements a conditional application of the transition operators M_a, controlled by the input symbol encoded in the input register. The example presented at the end of this section provides a concrete illustration of the resulting cursor structure and of the corresponding transition-selection mechanism.

Cursor structure. The cursor structure consists of a quasi-linear arrangement of m nested SWITCHes. The j-th nested SWITCH selects the transition operator associated with the j-th symbol of the padded input word. If each nested SWITCH contributes $2d + 2$ logical successors and consecutive SWITCHes share one cursor position, the resulting cursor path contains $L + 1 = (2d + 2)m - (m - 1)$ logical successors and therefore has length

$$L = (2d + 1)m.$$

This quasi-linear structure defines the effective sequence of transformations applied along the computational path.

4.2. Effective logical successors

As discussed in Section 3.3, within a SWITCH the register is modified only at the transition associated with the selected branch. The branch-selection and branch-joining transitions affect only the cursor and the control structure. Therefore, for the purpose of tracking the simulated MO-QFA computation, it is sufficient to record the state of the system after the completion of each nested SWITCH.

Let $f(j)$, for $j = 0, 1, \ldots, m$, denote the cursor position corresponding to the end of the j-th nested SWITCH, with $f(0) = 0$. Let S_k be the operator applied to the state register by the k-th nested SWITCH. We define the *effective logical successors* by

$$|\widetilde{\phi}_j\rangle = |\bar{w}\rangle_{r,w} \otimes \left(\prod_{k=1}^{j} S_k\right) |q_0\rangle_{r,s} \otimes |f(j)\rangle_c, \qquad j = 0, 1, \ldots, m, \tag{4.1}$$

where the empty product, corresponding to $j = 0$, is understood as the identity operator. These states describe the computationally relevant stages of the Feynman evolution, namely the configurations reached after the successive applications of the transition operators selected by the nested SWITCHes.

4.3. Correctness of the embedded dynamics

We now show that the effective logical successors reproduce exactly the step-by-step evolution of the original MO-QFA.

Lemma 4.1 (Correctness of the embedded dynamics). *Let $\mathcal{A} = (Q, \Sigma, \{M_a\}_{a\in\Sigma}, q_0, Q_F)$ be an* MO-QFA*, and consider its embedding into the Feynman machine described above. For every input word $w = a_1 a_2 \cdots a_\ell \in \Sigma^{\leq m}$, let $a'_1 a'_2 \cdots a'_m = w\#^{m-\ell}$ be its padded version. Then, for every $j = 0, 1, \ldots, m$,*

$$|\widetilde{\phi}_j\rangle = |\bar{w}\rangle_{r,w} \otimes \left(M_{a'_j} \cdots M_{a'_1} |q_0\rangle_{r,s} \right) \otimes |f(j)\rangle_c ,$$

where the product is understood as the identity for $j = 0$. In particular, the Feynman machine reproduces exactly the sequential application of the transition operators of the MO-QFA, with $M_\# = I$.

Proof. The proof follows from the construction of the nested SWITCHes.

The input register is initialized in the basis state $|\bar{w}\rangle_{r,w}$ and is not modified during the computation. At the j-th nested SWITCH, the cursor reads the binary block encoding the symbol a'_j. By construction, the nested SWITCH selects precisely the branch associated with a'_j and applies the corresponding operator $M_{a'_j}$ to the state register. If $a'_j = \#$, this operator is $M_\# = I$.

For $j = 0$, the claim gives the initial configuration. After the first nested SWITCH, the state register is $M_{a'_1} |q_0\rangle_{r,s}$. Assume that, after $j - 1$ nested SWITCHes, the state register is $M_{a'_{j-1}} \cdots M_{a'_1} |q_0\rangle_{r,s}$. The j-th nested SWITCH applies $M_{a'_j}$, yielding $M_{a'_j} M_{a'_{j-1}} \cdots M_{a'_1} |q_0\rangle_{r,s}$. This proves the claim by induction. □

4.4. Completion probability

The previous lemma establishes that the embedded dynamics reproduces the discrete evolution of the MO-QFA at the effective logical-successor positions. We now focus on the probability of observing the final configuration of the Feynman machine.

Theorem 4.1 (Completion probability). *Consider the Feynman machine implementing the embedded* MO-QFA *on inputs of length at most m, where $m \geq 1$. Let $d = \lceil \log_2 |\Sigma'| \rceil$, and let $L = (2d + 1)m$ be the length of the quasi-linear cursor path. Then the maximum probability of observing the final logical successor scales as*

$$\max_t P_L(t) = \Theta(L^{-2/3}) = \Theta((dm)^{-2/3}).$$

Proof. By Lemma 4.1, the effective logical successors $\{|\widetilde{\phi}_j\rangle\}_{j=0}^m$ encode the successive computational stages of the embedded MO-QFA. In particular, after the j-th nested SWITCH has been completed, the state register contains the result of the sequential application of the first j transition operators of the automaton.

The states $|\widetilde{\phi}_j\rangle$ form a distinguished subsequence of the logical successors $\{|\phi_j\rangle\}_{j=0}^L$ of the Feynman machine, corresponding to the cursor positions $f(j)$ at the end of the nested SWITCHes. In particular, the final effective logical successor $|\widetilde{\phi}_m\rangle$ coincides with the final logical successor $|\phi_L\rangle$, where $L = f(m)$.

As discussed in Section 3.3, in the logical-successor basis the Hamiltonian of the Feynman machine with SWITCHes reduces to the tight-binding Hamiltonian on a one-dimensional chain:

$$H = \sum_{j=0}^{L-1} \left(|\phi_{j+1}\rangle \langle\phi_j| + |\phi_j\rangle \langle\phi_{j+1}| \right).$$

The nested SWITCHes determine which unitary operators are applied to the state register, but they do not alter the effective one-dimensional quantum-walk dynamics of the cursor. Therefore, the evolution from the initial logical successor can be written as

$$|\psi(t)\rangle = \sum_{j=0}^{L} c_j(t)\,|\phi_j\rangle\,,$$

where the amplitudes $c_j(t)$ are those of a continuous-time quantum walk on a finite line starting from $|\phi_0\rangle$. As recalled in Section 3, for a continuous-time quantum walk on a finite line, the maximum probability of reaching the last site scales as $L^{-2/3}$. Hence,

$$\max_t P_L(t) = \max_t |\langle\phi_L|\,\psi(t)\rangle|^2 = \max_t |\langle\widetilde{\phi}_m|\,\psi(t)\rangle|^2 = \Theta(L^{-2/3}).$$

Since in the present construction $L = (2d+1)m = \Theta(dm)$, the claim follows. ☐

4.5. Preservation of acceptance probabilities

We now state the main consequence of the embedding. Let

$$\Pi_F = \sum_{q\in Q_F} |q\rangle\langle q|$$

be the accepting projector of the MO-QFA, acting on the state register. The corresponding accepting projector for the Feynman machine is set to be

$$\Pi_F^{\mathrm{Feyn}} = I_{r,w} \otimes \Pi_F \otimes |f(m)\rangle\langle f(m)|_c\,.$$

Theorem 4.2 (Preservation of acceptance probabilities)**.** *Let $\mathcal{A}$ be an* MO-QFA *and let $w \in \Sigma^{\le m}$. Consider the Feynman machine obtained from the embedding described above. Conditioned on observing the cursor in the final effective position $f(m)$, the state register coincides with the final state reached by $\mathcal{A}$ on input w. Consequently, the conditional acceptance probability of the Feynman machine equals the acceptance probability of the* MO-QFA*:*

$$P_{\mathrm{accept}}^{\mathrm{Feyn}}(w \mid f(m)) = P_{\mathrm{accept}}^{\mathcal{A}}(w).$$

Proof. Let $w = a_1a_2\cdots a_\ell$ and let $a_1'a_2'\cdots a_m' = w\#^{m-\ell}$ be its padded version. By Lemma 4.1, the final effective logical successor is

$$|\widetilde{\phi}_m\rangle = |\bar{w}\rangle_{r,w} \otimes \Big(M_{a_m'}\cdots M_{a_1'}\,|q_0\rangle_{r,s}\Big) \otimes |f(m)\rangle_c\,.$$

Since $M_\# = I$, this state can be rewritten as

$$|\widetilde{\phi}_m\rangle = |\bar{w}\rangle_{r,w} \otimes \Big(M_{a_\ell}\cdots M_{a_1}\,|q_0\rangle_{r,s}\Big) \otimes |f(m)\rangle_c\,.$$

Hence, conditioning on the cursor being in the final effective position $f(m)$ and disregarding the unchanged input register, the state register is left in the state

$$|\psi_w\rangle = M_{a_\ell}\cdots M_{a_1}\,|q_0\rangle_{r,s}\,.$$

This is precisely the final state reached by the MO-QFA on input w. Therefore,

$$\langle\psi_w|\,\Pi_F\,|\psi_w\rangle = \|\Pi_F\,|\psi_w\rangle\,\|^2 = P_{\mathrm{accept}}^{\mathcal{A}}(w).$$

Since the acceptance probability of the Feynman machine is computed through the projector Π_F^{Feyn}, it follows that

$$P_{\mathrm{accept}}^{\mathrm{Feyn}}(w \mid f(m)) = \langle\widetilde{\phi}_m|\,\Pi_F^{\mathrm{Feyn}}\,|\widetilde{\phi}_m\rangle = \langle\psi_w|\,\Pi_F\,|\psi_w\rangle = P_{\mathrm{accept}}^{\mathcal{A}}(w).$$

This establishes the preservation of acceptance probabilities. ☐

Corollary 4.1. *Every language recognized by an* MO-QFA *with cutpoint λ is recognized by the corresponding embedded Feynman machine with the same conditional acceptance probability. In particular, the cutpoint behavior is preserved.*

Proof. The statement follows immediately from Theorem 4.2. □

Remark. Theorem 4.1 concerns the probability of observing the cursor in the final effective position $f(m)$, whereas Theorem 4.2 concerns the acceptance probability conditioned on this event. Thus, the embedding preserves exactly the computational behavior of the MO-QFA. The scaling of the completion probability is a property of the underlying Feynman dynamics and does not affect the exact preservation of acceptance probabilities established in Theorem 4.2.

Effective logical resources. For an input-length bound m, the input register consists of dm spins, where $d = \lceil \log_2 |\Sigma'| \rceil$, while the unary encoding of the automaton state requires one spin for each state of the MO-QFA. The cursor evolves along a quasi-linear sequence of m nested SWITCHes, whose effective logical path has length $L = (2d+1)m$. The completion probability established in Theorem 4.1 depends only on the length of this effective logical path. A more detailed analysis of the physical resources required by the nested SWITCH network is left for future work.

4.6. An illustrative example

To illustrate the embedding construction, let $m \geq 0$ be fixed and consider the language

$$L_m = \{\, w \in \{a,b\}^{\leq m} : |w|_a = |w|_b \,\}.$$

Inspired by a construction of [11], we define an MO-QFA $\mathcal{A}$ recognizing L_m with non-isolated cutpoint. Since L_m is not a group language, recognition with isolated cutpoint is impossible for MO-QFAs, as recalled in Section 2. Therefore, non-isolated cutpoint acceptance is the strongest form of recognition achievable for this language within the MO-QFA model.

Specifically, let $\mathcal{A} = (\{q_0,q_1\},\{a,b\},\{M_a,M_b\},q_0,\{q_0\})$, with $\Pi_F = |q_0\rangle\langle q_0|$. The transition operators are

$$M_a = \begin{pmatrix} \cos(\pi\theta) & -\sin(\pi\theta) \\ \sin(\pi\theta) & \cos(\pi\theta) \end{pmatrix}, \qquad M_b = M_a^\dagger.$$

Given an input word $w = a_1 a_2 \cdots a_\ell \in \{a,b\}^{\leq m}$, the final state of the automaton is $M_{a_\ell} \cdots M_{a_1} |q_0\rangle$. Since $M_b = M_a^\dagger = M_a^{-1}$, both operators are powers of the same rotation and therefore commute, yielding

$$P^{\mathcal{A}}_{\text{accept}}(w) = \left\| \Pi_F \left(M_a^{|w|_a} M_b^{|w|_b} |q_0\rangle \right) \right\|^2 = \cos^2(d(w)\pi\theta), \qquad d(w) = |w|_a - |w|_b. \tag{4.2}$$

By definition, $w \in L_m$ if and only if $d(w) = 0$. Thus, according to (4.2), every word in L_m is accepted with probability 1.

To characterize the asymptotic acceptance behavior of the family $\{L_m\}_{m\geq 0}$, we distinguish between the cases in which θ is rational or irrational. In the former case, the sequence of acceptance probabilities given by (4.2) is periodic, and there exist words outside L_m that are accepted with probability 1. In the latter case, every word outside L_m is accepted with probability strictly smaller than 1, although acceptance probabilities arbitrarily close to 1 can occur. Indeed, when θ is irrational, the set $\{\cos^2(k\pi\theta) : k \in \mathbb{Z}\}$ is dense in the interval $[0,1]$, a consequence of the classical Kronecker–Weyl theorem (see, e.g., [41]). Consequently, the cutpoint associated with the recognition of L_m is non-isolated.

Embedding: the SWITCH. Given the MO-QFA $\mathcal{A}$ for the language L_m, we now provide the equivalent (in the sense of Theorem 4.2) Feynman machine. In particular, we focus on the nested SWITCH structure responsible for selecting the transition operator associated with the current input symbol.

First, we extend the alphabet to $\Sigma' = \{a,b,\#\}$, and set $M_\# = I$. Next, we fix the binary encoding of the symbols of Σ' on $\lceil \log_2(3) \rceil$ bits as follows: $a \mapsto |\uparrow\uparrow\rangle$, $b \mapsto |\uparrow\downarrow\rangle$, $\# \mapsto |\downarrow\downarrow\rangle$. The resulting

Feynman machine consists of a sequence of m SWITCHes, each having the structure shown in Fig. 4.

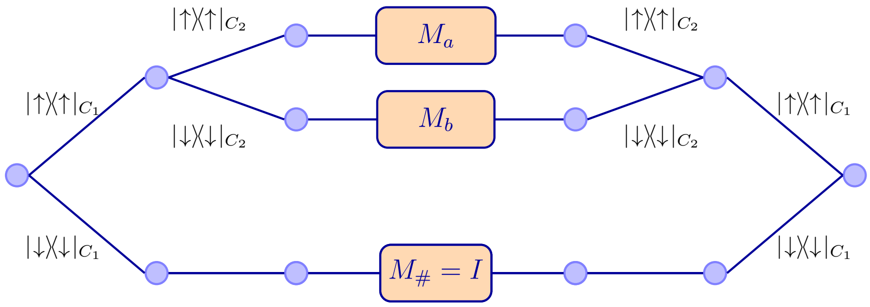

Figure 4: The SWITCH used to select the transition operator associated with a single input symbol. With the symbol encoding $a \mapsto |{\uparrow\uparrow}\rangle$, $b \mapsto |{\uparrow\downarrow}\rangle$, and $\# \mapsto |{\downarrow\downarrow}\rangle$, the blank symbol is identified already by the first qubit, and therefore the branch associated with $|{\downarrow}\rangle\langle{\downarrow}|_{C_1}$ does not require further nested branching.

5. Conclusions and future work

In this work, we presented an explicit embedding of measure-once quantum finite automata (MO-QFAs) into the Feynman quantum computer model. By encoding both the input word and the internal state of the automaton into the register, and exploiting the cursor dynamics to implement the sequential application of transition operators, we constructed a Hamiltonian system that exactly simulates the computation of the embedded automaton.

A central ingredient of the construction is the use of nested SWITCHes, which enable the conditional selection of transition operators according to the symbols encoded in the input register. This yields a flexible and composable framework for implementing symbol-dependent quantum evolutions while preserving the effective one-dimensional quantum-walk structure of the Feynman model.

Our main result shows that, conditioned on the cursor reaching the final effective position, the state register coincides with the final state of the MO-QFA. Consequently, acceptance probabilities and cutpoints are preserved exactly. This establishes a concrete connection between finite-state quantum computation and continuous-time Hamiltonian computation.

The formulation naturally takes the form of a family of finite-dimensional Feynman machines, reflecting the correspondence between abstract computational models defined over unbounded inputs and their finite physical realizations.

Several directions for future research naturally arise from this work. One interesting direction is the extension of the construction to more general models of quantum finite automata [5, 24, 25, 26]. More generally, the proposed Hamiltonian simulation may provide a different perspective and new tools for tackling several problems related to QFAs, including decision problems (see, e.g., [42, 43, 44, 45]).

Another promising direction concerns the analysis of resource requirements, such as the scaling of the cursor size and the locality of interactions, with the aim of identifying more efficient implementations. Finally, it would be interesting to investigate similar embeddings for other computational models, further clarifying relationships between different paradigms of quantum computation.

Acknowledgments

The authors wish to acknowledge the anonymous referees for helpful comments and remarks.

Declaration on generative AI

The authors have not employed any Generative AI tools.

References

[1] D. Deutsch, Quantum theory, the Church–Turing principle and the universal quantum computer, Proceedings of the Royal Society of London A 400 (1985) 97–117. URL: https://doi.org/10.1098/rspa.1985.0070. doi:10.1098/rspa.1985.0070.

[2] N. D. Mermin, Quantum Computer Science: An Introduction, Cambridge University Press, 2007. URL: https://doi.org/10.1017/CBO9780511813870.

[3] M. A. Nielsen, I. L. Chuang, Quantum Computation and Quantum Information: 10th Anniversary Edition, Cambridge University Press, Cambridge, UK, 2010.

[4] A. C.-C. Yao, Quantum circuit complexity, in: Proceedings of the 34th Annual Symposium on Foundations of Computer Science (FOCS), IEEE, 1993, pp. 352–361. doi:10.1109/SFCS.1993.366852.

[5] D. Qiu, L. Li, P. Mateus, J. Gruska, Quantum finite automata, in: J. Wang (Ed.), Handbook of Finite State Based Models and Applications, CRC Press, Boca Raton, FL, USA, 2016, pp. 113–144.

[6] D. Aharonov, W. van Dam, J. Kempe, Z. Landau, S. Lloyd, O. Regev, Adiabatic quantum computation is equivalent to standard quantum computation, SIAM Journal on Computing 37 (2007) 166–194. doi:10.1137/S0097539705447323.

[7] A. M. Childs, Universal computation by quantum walk, Physical Review Letters 102 (2009) 180501. doi:10.1103/PhysRevLett.102.180501.

[8] R. P. Feynman, Quantum mechanical computers, Optics News 11 (1985) 11–20. doi:10.1364/ON.11.2.000011, based on a plenary talk at CLEO/IQEC, 1984.

[9] A. Y. Kitaev, A. H. Shen, M. N. Vyalyi, Classical and Quantum Computation, volume 47 of *Graduate Studies in Mathematics*, American Mathematical Society, 2002. doi:10.1090/gsm/047, english translation of the original Russian edition published by MCCME, Moscow, 1999.

[10] D. Aharonov, T. Naveh, Quantum NP—A survey, arXiv preprint quant-ph/0210077 (2002).

[11] A. Bertoni, M. Carpentieri, Analogies and differences between quantum and stochastic automata, Theoretical Computer Science 262 (2001) 69–81. doi:10.1016/S0304-3975(00)00154-7.

[12] A. Bertoni, M. Carpentieri, Regular languages accepted by quantum automata, Information and Computation 165 (2001) 174–182. doi:10.1006/inco.2000.2911.

[13] A. Brodsky, N. Pippenger, Characterizations of 1-way quantum finite automata, SIAM Journal on Computing 31 (2002) 1456–1478. doi:10.1137/S0097539700370736.

[14] A. Kondacs, J. Watrous, On the power of quantum finite state automata, in: Proceedings of the 38th Annual Symposium on Foundations of Computer Science (FOCS), IEEE, 1997, pp. 66–75. doi:10.1109/SFCS.1997.646094.

[15] C. Moore, J. P. Crutchfield, Quantum automata and quantum grammars, Theoretical Computer Science 237 (2000) 275–306. doi:10.1016/S0304-3975(99)00176-1.

[16] A. Ambainis, D. Thérien, M. Mercer, Algebraic results on quantum automata, Theory of Computing Systems 39 (2006) 165–188. doi:10.1007/s00224-005-1231-x.

[17] A. Peruzzo, M. Lobino, J. C. F. Matthews, N. Matsuda, A. Politi, K. Poulios, X.-Q. Zhou, Y. Lahini, N. Isailovic, K. Worhoff, Y. Bromberg, Y. Silberberg, M. G. Thompson, J. L. O'Brien, Quantum walks of correlated photons, Science 329 (2010) 1500–1503. doi:10.1126/science.1182730.

[18] A. Schreiber, K. N. Cassemiro, V. Potoček, A. Gábris, P. J. Mosley, E. Andersson, I. Jex, C. Silberhorn, Photons walking the line: A quantum walk with adjustable coin operations, Physical Review Letters 104 (2010) 050502. doi:10.1103/PhysRevLett.104.050502.

[19] L. Sansoni, F. Sciarrino, G. Vallone, P. Mataloni, A. Crespi, R. Ramponi, R. Osellame, Two-particle bosonic–fermionic quantum walk via integrated photonics, Physical Review Letters 108 (2012) 010502. doi:10.1103/PhysRevLett.108.010502.

[20] B. Bartlett, A. Dutt, S. Fan, Deterministic photonic quantum computation in a synthetic time dimension, Optica 8 (2021) 1515–1523. doi:10.1364/OPTICA.443353.

[21] C. Mereghetti, B. Palano, S. Cialdi, V. Vento, M. G. A. Paris, S. Olivares, Photonic realization of a quantum finite automaton, Physical Review Research 2 (2020) 013089. doi:10.1103/PhysRevResearch.2.013089.

[22] Y. Tian, T. Feng, M. Luo, S. Zheng, X. Zhou, Experimental demonstration of quantum finite automaton, npj Quantum Information 5 (2019) 56. doi:10.1038/s41534-019-0163-x.

[23] A. Candeloro, C. Mereghetti, B. Palano, S. Cialdi, M. G. A. Paris, S. Olivares, An enhanced photonic quantum finite automaton, Applied Sciences 11 (2021) 8768. doi:10.3390/app11188768.

[24] M. P. Bianchi, C. Mereghetti, B. Palano, Quantum finite automata: Advances on Bertoni's ideas, Theoretical Computer Science 669 (2017) 38–55. doi:10.1016/j.tcs.2016.01.045.

[25] C. Mereghetti, B. Palano, P. Raucci, Latvian quantum finite state automata for unary languages, International Journal of Foundations of Computer Science 36 (2025) 419–455. doi:10.1142/S0129054124430032.

[26] C. Piazza, R. Romanello, Mirrors and memory in quantum automata, in: Proceedings of the International Conference on Quantitative Evaluation of Systems (QEST), volume 13479 of *Lecture Notes in Computer Science*, Springer, 2022, pp. 359–380. doi:10.1007/978-3-031-16336-4_18.

[27] J. E. Hopcroft, R. Motwani, J. D. Ullman, Introduction to Automata Theory, Languages, and Computation, 3rd ed., Pearson, 2006.

[28] A. Paz, Introduction to Probabilistic Automata, Academic Press, 1971.

[29] M. O. Rabin, Probabilistic automata, Information and Control 6 (1963) 230–245. doi:10.1016/S0019-9958(63)90290-0.

[30] A. Ambainis, R. Freivalds, 1-way quantum finite automata: Strengths, weaknesses and generalizations, in: Proceedings of the 39th Annual Symp. on Foundations of Computer Science (FOCS), IEEE, 1998, pp. 332–341. doi:10.1109/SFCS.1998.743485.

[31] A. Bertoni, C. Mereghetti, B. Palano, Some formal tools for analyzing quantum automata, Theoretical Computer Science 356 (2006) 14–25. doi:10.1016/j.tcs.2006.01.042.

[32] M. P. Bianchi, C. Mereghetti, B. Palano, Complexity of promise problems on classical and quantum automata, in: Computing with New Resources, volume 8808 of *Lecture Notes in Computer Science*, Springer, 2014, pp. 161–175. doi:10.1007/978-3-319-13350-8_12.

[33] M. P. Bianchi, C. Mereghetti, B. Palano, Size lower bounds for quantum automata, Theoretical Computer Science 551 (2014) 81–91. doi:10.1016/j.tcs.2014.07.004.

[34] C. Mereghetti, B. Palano, Quantum automata for some multiperiodic languages, Theoretical Computer Science 387 (2007) 177–186. doi:10.1016/j.tcs.2007.07.037.

[35] L. Xiao, D. Qiu, State complexity of one-way quantum finite automata together with classical states, Journal of Computer and System Sciences 154 (2025) 103659. doi:10.1016/j.jcss.2025.103659.

[36] Y. Tian, T. Feng, M. Luo, S. Zheng, X. Zhou, Experimental demonstration of quantum finite automaton, npj Quantum Information 5 (2019) 56. doi:10.1038/s41534-019-0163-x.

[37] A. Peres, Reversible logic and quantum computers, Physical Review A 32 (1985) 3266–3276. doi:10.1103/PhysRevA.32.3266.

[38] R. P. Feynman, R. B. Leighton, M. Sands, The Feynman Lectures on Physics, Volume III, Addison-Wesley, Reading, MA, 1965.

[39] B. Apolloni, D. D. Falco, The clock of a quantum computer, Journal of Physics A: Mathematical and General 35 (2002) 10033–10051. doi:10.1088/0305-4470/35/47/305.

[40] D. D. Falco, D. Tamaselli, Quantum timing and synchronization problems, International Journal of Modern Physics B 18 (2004) 623–631. doi:10.1142/S0217979204024240.

[41] L. Kuipers, H. Niederreiter, Uniform Distribution of Sequences, Wiley, 1974.

[42] M. P. Bianchi, B. Palano, Behaviours of unary quantum automata, Fundamenta Informaticae 104 (2010) 1–15. doi:10.3233/FI-2010-345.

[43] F. D'Alessandro, C. Mereghetti, B. Palano, P. Papi, On some decision problems on quantum automata, in: Developments in Language Theory, volume 16578 of *Lecture Notes in Computer Science*, Springer, 2026, pp. 3–17. doi:10.1007/978-3-032-28404-4_1.

[44] L. Li, D. Qiu, Determining the equivalence for one-way quantum finite automata, Theoretical Computer Science 403 (2008) 42–51. doi:10.1016/j.tcs.2008.04.039.

[45] C. Mereghetti, B. Palano, P. Raucci, Unary quantum finite state automata with control language, Applied Sciences 14 (2024) 1490. doi:10.3390/app14041490.