A Measure-Once 1-way QFA Based on the Quantum Circuit Implementation of a QECC

Julia Katrina Dy[1]*, Alfonso Labao[1], and Henry Adorna[1]

[1]Department of Computer Science, College of Engineering, University of the Philippines - Diliman, Quezon City, Philippines;
jmdy1@alumni.up.edu.ph, ablabao@up.edu.ph, hnadorna@up.edu.ph

Abstract. One major problem faced by quantum computing is how vulnerable qubits are to noise. Given this, the importance of quantum error correction in the field of quantum computing cannot be understated. Much research has been done on this topic, and many models of quantum error correction codes (QECCs) have been created. However, quantum finite automata (QFAs) as a model is not widely researched in this field. Using QFAs as models for linear QECCs, often modeled using quantum circuits, may provide an alternative means of analysis. Quantum circuits and one-way QFAs have similar structures and elements that allow quantum circuits to be easily adapted into QFAs. This paper aims to discuss (a) basic circuit simulation using QFAs and (b) how it can be used to model simple linear QECCs.

Keywords: quantum automata, quantum error correction codes, quantum circuits

1 Introduction

Quantum computing is a field that has generated interest because of its potential to be more powerful than classical computing. On a quantum computer, problems that are difficult to compute classically, such as factoring and discrete logarithm, can be solved in polynomial time[2]. Today, there are many existing quantum algorithms that have significant speedups compared to their corresponding classical algorithms.

With the emergence of quantum computing, many researchers have proposed theoretical models of quantum systems. Among them are different quantum finite automata (QFAs). QFAs serve as theoretical models for quantum computers with finite memory, and they are usually used in problems involving language recognition. An example of this is a study that presented improved circuit-based implementations of QFA algorithms that recognize a certain language[5]. Other applications of QFAs have not been explored as widely.

Early QFA models such as measure-once one-way QFAs (MO-1QFAs) and measure-many one-way QFAs (MM-1QFAs) have limited recognition power, only recognizing a subset of regular languages. Other models such as the 2-way QFA on the other hand, have been proven to be more powerful than their classical counterparts[2][4]. However, even the relatively weaker early QFA models have been shown to be more efficient than

J. Caro et al. (eds.), *Proceedings of the Workshop on Computation: Theory and Practice (WCTP 2025)*, Atlantis Highlights in Computer Sciences 24,
https://doi.org/10.2991/978-94-6239-638-8_24

their classical counterparts when recognizing certain languages[1]. Considering the capabilities of weaker QFA models alone, the potential of quantum computing systems is evident.

Although the potential of quantum computing is great, there are some major barriers to its practical implementation. Quantum devices are vulnerable to noise, so the development of quantum error correction is particularly important. Shor's 1995 paper[16] provided an early major breakthrough in the field of quantum error correction. The paper discussed a reasonable model for a quantum error correction code (QECC) based on classical repetition codes, modified to adjust for quantum rules such as the no-cloning rule. Further research prompted by Shor's code lead to the development of more kinds of linear QECCs. Now, different kinds of non-linear QECCs have been developed, such as topological codes or the 2D planar surface code.

Current research on the applications of quantum automata continues. In a study by Chen et al. (2023), quantum tree automata were used as a framework for the verification of quantum circuits[8]. In line with this, it is possible that QFAs may serve as an alternative framework for circuit analysis. As QECCs also have circuit implementations, perhaps using QFAs as a framework for analysis may provide another avenue of analysis for QECCs, specifically linear QECCs.

The necessary prerequisite terms and knowledge for this paper are given in Section 2, followed by the main results in Section 3 and conclusion in Section 4 of the paper.

2 Preliminaries

Quantum states are vectors in the complex Hilbert space $\mathcal{H}$. They can be represented using the bra-ket notation $|\psi\rangle$. Qubits are the quantum equivalent of classical bits and are represented as quantum states, specifically as superpositions of classical bit states 0 and 1. A qubit can be written as $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle = \left[\begin{smallmatrix}\alpha\\ \beta\end{smallmatrix}\right]$, where $\alpha, \beta \in \mathbb{C}, |\alpha|^2 + |\beta|^2 = 1$. $|\alpha|^2$ and $|\beta|^2$ are the probabilities that the qubit is measured to 0 or 1 respectively. Note that $|0\rangle = \left[\begin{smallmatrix}1\\ 0\end{smallmatrix}\right]$ and $|1\rangle = \left[\begin{smallmatrix}0\\ 1\end{smallmatrix}\right]$ are the basis states of which the qubit is a linear combination of.

2.1 Classical Finite Automata

Finite automata (FAs) with classical mechanics, also known as finite state machines (FSMs), are some of the simplest models of computation. They recognize the set of regular languages. They are used to simulate computers with limited memory and can generally be separated into 2 categories: deterministic finite automata (DFAs) and nondeterministic finite automata (NFAs).

A DFA is defined as a 5-tuple $(Q, \Sigma, \delta, q_0, Q_F)$ where Q is the finite set of states, Σ is the finite set of input symbols called the alphabet, $\delta : Q \times \Sigma \to Q$ is the transition function, $q_0 \in Q$ is the initial or start state, and $Q_F \subseteq Q$ is the set of accepting states.

An NFA is defined the same way as a DFA except for its transition function, which is defined as $\delta : Q \times \Sigma_\varepsilon \to \wp(Q)$. $\Sigma_\varepsilon = \Sigma \cup \{\varepsilon\}$ and $\wp(Q)$ is the power set of Q. Given an

input string, it may have 0, 1, or more than 1 possible end states in an NFA. This is not the case for a DFA which only has 1 possible end state for any given input string.

2.2 Probabilistic Automata

Before discussing the principles behind QFAs, it may be prudent to first discuss Probabilistic Automata (PAs). PAs were first proposed by Rabin[15], and they recognize the set of stochastic languages, of which regular languages are a subset. PAs generally have the same definition as DFAs and NFAs except for the transition function and initial state. The initial state of a PA is a stochastic vector, which is a vector with non-negative entries that add up to 1[14]. PA transitions are probabilistic choices between multiple next states, relating states and inputs to a probability distribution over the set of states[17]. If the set of states is defined as $Q = \{q_0, \ldots, q_n\}$, the transition function can be defined as $\delta : Q \times \Sigma \to [0,1]^{n+1}$ such that for all $(q, \sigma) \in Q \times \Sigma$, $\delta(q, \sigma) = \{p_0(q, \sigma), \ldots, p_n(q, \sigma)\}, 0 \leq p_i(q, \sigma)$ and $\sum_{i=0}^{n} p_i(q, \sigma) = 1$. Note that $p_i(q, \sigma)$ indicates the probability of a transition from state q to state q_i occurring on input σ[15].

An alternative way to look at this is as a set of $|Q| \times |Q|$ transition matrices or transition tables $\{M_\sigma\}_{\sigma \in \Sigma}$ with the value of each entry M_{ij} being the probability of transitioning to state q_j from state q_i on input σ. Each transition is multiplied to the current state vector to get the next state vector.

The condition for acceptance is as follows: given a cut-point λ (normally taken to be $\in [0,1)$), if the PA were to be run on the same input a very large number of times, there would be a greater than λ probability of the PA ending on a state in Q_F[15].

2.3 Quantum Finite Automata

As described in [2], quantum systems utilize quantum mechanics most obviously in the use of superpositions as states. A superposition $|\psi\rangle$ can be seen as a linear combination of basis states with corresponding amplitudes. Given $n+1$ basis states, a superposition of these states can be written as $|\psi\rangle = \alpha_0 |q_0\rangle + \alpha_1 |q_1\rangle + \cdots + \alpha_n |q_n\rangle$.

Quantum systems and probabilistic systems share many similarities. Although the basic concepts are similar, quantum systems use amplitudes $\alpha_0, \alpha_1, \ldots, \alpha_n$ in place of probabilities $p_0, p_1, \ldots, p_n$. The major difference between the two is that probabilities are in the set of real numbers $\mathbb{R}$ while amplitudes are in the set of complex numbers $\mathbb{C}$. The amplitudes of a quantum system must fulfill the following condition: $\sum_{i=0}^{n} |\alpha_i|^2 = 1$.

Generally, QFAs have an initial superposition and, much like PAs, perform state transitions by matrix multiplication. Each input symbol has a transition matrix. Transition matrices normally have some kind of restriction, such as needing to be unitary. Unitary transitions can be defined as follows: given a state $|\psi\rangle$ and a transformation U, a transition is unitary iff when $\||\psi\rangle\| = 1$, $\|U|\psi\rangle\| = 1$[2]. After reading to the end of a string, a measurement is performed on the current state to decide acceptance. Measurement collapses the superposition.

One-way QFAs are QFAs that read input left to right. A general definition of a one-way QFA is as follows:

Definition 1. *A one-way QFA is generally defined as having a finite set of classical states Q, a finite set of input symbols Σ, a transition function δ, an initial superposition $|\psi_0\rangle \in \mathbb{C}$, and a way to accept strings, usually given by using a set of accepting states $Q_{acc} \subseteq Q$.*

Note that the quantum state of a one-way QFA is a superposition of classical basis states and can be defined $\{|q\rangle \mid q \in Q\} : |\psi\rangle = \sum_{q \in Q} \alpha_q |q\rangle$ as stated in [2].

There are many models of one-way QFAs. Below are some basic models that are relevant or mentioned in this paper.

Measure-once one-way QFA The measure-once one-way QFA (MO-1QFA), also known as the Moore-Crutchfield QFA in some literature, was first proposed in [13].

Definition 2. *An MO-1QFA is defined as a 5-tuple $(Q, \Sigma, \delta, |q_0\rangle, Q_{acc})$ where Q is the finite set of states, Σ is the finite set of input symbols, $\delta : Q \times \Sigma \times Q \to \mathbb{C}$ is the transition function, $|q_0\rangle$ is the initial superposition, and $Q_{acc} \subseteq Q$ is the set of accepting states[12].*

Remark 1. The transition function of an MO-1QFA can also be represented as a set of unitary transitions corresponding to each input symbol. Transitions can be represented as transition matrices.

The set of unitary transition matrices can be written as $\{U_\sigma\}_{\sigma \in \Sigma}$.

Theorem 1. *Transitions can be applied to a quantum state by multiplying the transition matrix to the state. Given an initial state $|q_0\rangle$, after reading n input symbols the current state of the QFA would be computed as $|\psi\rangle = U_{\sigma_{n-1}} ... U_{\sigma_1} U_{\sigma_0} |q_0\rangle$.*

This can be seen in the examples given in [1] and [13].

Given a string $x = \sigma_0 \sigma_1 \ldots \sigma_n$, An MO-1QFA would apply the transition matrices $U(\sigma_0)U(\sigma_1)\ldots U(\sigma_n)$ to the initial state $|\psi\rangle$. A projective measurement P_{acc} is performed on the resulting final superposition. If the projection leaves the automaton in an accepting state, the string is accepted.

Example: $\mathsf{MOD_p}$ **Problem QFA** A simple example of one such QFA is one that recognizes the language $\mathcal{L} = \{a^i | i \text{ is divisible by } p\}$, where p is a prime number[1][5]. A DFA that recognizes this language would have at least p states. Let $\mathcal{M}$ be the MO-1QFA that recognizes $\mathcal{L}$. $\mathcal{M}$ is defined as a 6-tuple $(Q, \Sigma, \delta, q_0, Q_{acc}, Q_{rej})$, where:

- Q is the set of states $\{q_0, q_1, q_{acc}, q_{rej}\}$,
- $\Sigma = \{a\} \cup '\$'$, where $\{a\}$ is the input alphabet and $'\$'$ is the end-marker for the input text,

– δ is the transition function such that when reading $'a'$, the transition matrix

$$U_a = \begin{bmatrix} \cos\left(\frac{2\pi}{p}\right) & -\sin\left(\frac{2\pi}{p}\right) \\ \sin\left(\frac{2\pi}{p}\right) & \cos\left(\frac{2\pi}{p}\right) \end{bmatrix}$$

is applied to the current state, and while reading the end-marker $'\$'$, the mappings

$$U_\$ = \begin{cases} |q_0\rangle \mapsto |q_{rej}\rangle \\ |q_1\rangle \mapsto |q_{acc}\rangle \end{cases}$$

are applied to the current state,

– q_0 is the initial state $|q_0\rangle$, and

– Q_{acc} is the set of accepting states $\{q_{acc}\}$.

– Q_{rej} is the set of rejecting states $\{q_{rej}\}$.

This simple automaton accepts all words in $\mathcal{L}$ with probability 1 and all words not in $\mathcal{L}$ with probability of at most $7/8$, which is quite high. In the same paper, [1] optimized the construction until they had a QFA with $O(\log p)$ states that recognized $\mathcal{L}$ with a probability of at least $1-\varepsilon$, but the simple construction above is the easiest to follow. Nevertheless, the final results show that QFAs have the potential to be exponentially more space-efficient compared to their classical counterparts.

Measure-many one-way QFA The measure-many one-way QFA (MM-1QFA) is another basic example of a one-way QFA. It is defined similarly to the MO-1QFA.

Definition 3. *An MM-1QFA is defined as a 6-tuple $(Q,\Sigma,\delta,|\psi_0\rangle,Q_{acc},Q_{rej})$, where Q is the finite set of states, Σ is the finite set of input symbols, $|q_0\rangle$ is the initial superposition, $Q_{acc}\subseteq Q$ is the set of accepting states, $Q_{rej}\subseteq Q$ is the set of rejecting states, and $\delta:\{U(\sigma)\}_{\sigma\in\Sigma\cup\{\#,\$\}}$, $\{\#,\$\}\notin\Sigma$ is the set of transition matrices over all symbols in the alphabet, including the start- and end-markers $'\#'$ and $'\$'$ respectively[4][12].*

The main difference between the MO-1QFA and the MM-1QFA is that the MO-1QFA undergoes only one measurement, the final projective measurement. An MM-1QFA, as the name implies, undergoes many intermediate partial measurements. After every transition, the current state of the MM-1QFA is partially measured with respect to the subspaces E_{non},E_{acc},E_{rej}, which are the span of all non-halting, accepting, and rejecting states respectively. The result of the partial measurement is a measured subspace[4].

Both MO-1QFAs and MM-1QFAs are very restricted models of QFAs with relatively weak language recognition powers. They only recognize a proper subset of regular languages, with MO-1QFAs only recognizing group languages and MM-1QFAs being more powerful than MO-1QFAs but still unable to recognize the set of regular languages[12].

Example: A simple example given in Section 2.2 [1] describes an MM-1QFA defined to have $Q = \{q_0, q_1, q_{acc}, q_{rej}\}$, $\Sigma = \{a\}$, $Q_{acc} = \{q_{acc}\}$, $Q_{rej} = \{q_{rej}\}$, and start state q_0. The transition function could be defined as the set of transitions for each input symbol $\{V_\sigma\}_{\sigma \in \Sigma}$. They defined V_a and $V_\$$ as follows:

- $V_a(|q_0\rangle) = \frac{1}{2}|q_0\rangle + \frac{1}{2}|q_1\rangle + \frac{1}{\sqrt{2}}|q_{rej}\rangle$,
- $V_a(|q_1\rangle) = \frac{1}{2}|q_0\rangle + \frac{1}{2}|q_1\rangle - \frac{1}{\sqrt{2}}|q_{rej}\rangle$,
- $V_\$(|q_0\rangle) = |q_{rej}\rangle$,
- $V_\$(|q_1\rangle) = |q_{acc}\rangle$.

The transitions not described can have arbitrary values, with the only rule being that the unitary property of the transitions is maintained.

On the input *aa*, the following steps are taken:

1. V_a is applied to the start state $|q_0\rangle$. We now have the current state $\frac{1}{2}|q_0\rangle + \frac{1}{2}|q_1\rangle + \frac{1}{\sqrt{2}}|q_{rej}\rangle$. The partial measurement is then taken, resulting in either:
   - $|q_{rej}\rangle$ being observed with probability $|\frac{1}{\sqrt{2}}|^2 = \frac{1}{2}$ resulting in the state collapsing to $|q_{rej}\rangle$ and rejecting the word, or
   - a non-halting state ($|q_0\rangle$ or $|q_1\rangle$) being observed with probability $|\frac{1}{2}|^2 + |\frac{1}{2}|^2 = \frac{1}{2}$ and the state collapses to $\frac{1}{2}|q_0\rangle + \frac{1}{2}|q_1\rangle$, continuing the computation.
2. If the computation continues, V_a will be applied to the current state, which causes the current state to be mapped to itself. A non-halting state is observed and the computation continues.
3. No further inputs can be read at this point, so $V_\$$ will be applied to the current state. The current state is now $|q_{rej}\rangle + |q_{acc}\rangle$. An accepting state is observed with probability $|\frac{1}{2}|^2 = \frac{1}{4}$, and a rejecting state is observed with the same probability.

The probability of acceptance is stated to be $\frac{1}{4}$ and the probability of rejection is $\frac{3}{4}$[1].

2.4 Quantum Error Correction

As classical error correction is a vital part of computer communications, quantum error correction is similarly important in quantum communications. The demand for quantum error correction is high due to the fragile nature of quantum systems. However, there are many complications surrounding quantum error correction. One such complication is the no-cloning theorem, which renders it impossible to 'copy' quantum states and directly reproduce classical error correction methods such as repetition codes. Another is the nature of quantum systems preventing direct observation of a qubit, as observation would collapse the system into a singular state. In order to work around these issues, one of the first big breakthroughs in quantum error correction was Shor's quantum error correction scheme[16], which leveraged entanglement to encode information redundantly. From here, extensions and new schemes were developed, leading to the well-researched and technically feasible methods of error correction of today.

Quantum Error Correction Codes (QECCs) Unlike classical bits, qubits are vulnerable to both bit-flip and phase-flip errors. Some basic codes, such as the three-qubit code further discussed below in Section 3.2, only correct one kind of error, but practical QECCs should be able to correct both kinds of errors.

Basic linear QECCs are based on classical repetition codes. They work around the no-cloning rule by encoding a single qubit as an n-qubit logical qubit, denoted as $|\psi\rangle_L$. The logical qubit can be encoded using controlled-not (CNOT) gates. After encoding, error detection is normally done by using stabilizer codes which anti-commute with the error and allow auxiliary ancilla qubits, called the syndrome, to detect errors. The error can then be corrected by applying the appropriate Pauli gate to the flipped qubit based on the measured syndrome.

$[\![n,k,d]\!]$ is the notation for a code that uses n qubits to encode k qubits with distance d. The lower bound for the existence of a code is

$$\left(\sum_{j=1}^{d-1} 3^j \binom{n}{j} = 1\right) 2^k \leq 2^n$$

as defined in [10]. The notation for a five-qubit code is $[\![5,1,3]\!]$ and the notation for a nine-qubit code is $[\![9,1,3]\!]$.

2.5 Quantum Gates

Definition 4. *Quantum gates are unitary operators that can be represented as square matrices and can be applied to one or more qubits.*

Single-qubit quantum gates are represented by 2×2 matrices while gates that operate on multiple qubits like the CNOT gate or the Toffoli gate are represented using larger matrices, such as 4×4 and 8×8 matrices for the CNOT and Toffoli gates respectively.

Some relevant examples of quantum gates are the Pauli gates (X, Y, Z), the identity gate, the controlled-not (CNOT or CX) gate, and the Hadamard gate. The matrix and circuit representations of the aforementioned gates are given in Table 1.

An important thing to note is that composite quantum states are created using the tensor product ($\otimes$). Similarly, the tensor product must be used in order to have a gate operate on a composite state. For example, there exists a state composed of 2 qubits that can be computed by getting their tensor product, $|\psi\rangle = |\psi_0\rangle \otimes |\psi_1\rangle$. To apply an X-gate on only the first qubit, the operator would be the tensor product of the X-gate and the identity matrix, $X \otimes I$.

Applying quantum gates is simple.

Theorem 2. *Quantum operators can be applied to quantum states using matrix-vector multiplication between the matrix representation of the operator and the vector representation of the quantum state.*

This is defined in Postulate 2 of [18].

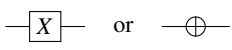
Gate Name	Matrix Representation	Circuit Representation
Pauli – X	$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$	
Pauli – Y	$\begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}$	
Pauli – Z	$\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$	
Identity	$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$	
Hadamard	$\frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$	
Controlled – Not (CNOT)	$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$	
Toffoli	$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$	

Table 1. Some quantum gate representations

2.6 Quantum Circuits

Quantum circuits are computational models that are similar in appearance to their classical counterparts. They differ from their classical counterparts in two main ways: (1) quantum circuits use qubits instead of classical bits, and (2) quantum gate operations are used in place of classical ones.

Definition 5. *A quantum circuit is defined as an ordered set of qubits on which an ordered sequence of gate operations is applied[3][18].*

Measurement gates, used to measure qubit states, are represented in the circuit diagram using the symbol  .

3 Main Results

Quantum circuits and one-way QFAs have similar definitions, which is why it is possible to simulate a quantum circuit using a QFA. For this paper, the MO-1QFA will be used to simulate all quantum circuit implementations. The circuits to be adapted are the encoding and error detection parts of n-qubit linear QECCs based on classical repetition codes described in Section 2.4.

Note that for the purpose of this paper, the scope is limited to circuits that only measure at the end of the circuit. Other circuit elements besides qubits and gates, such as quantum teleportation, are also not considered within the scope of this paper.

3.1 Circuit simulation

Given a quantum circuit, it is simple to adapt it to a QFA. From Definition 5, it can be seen that the main elements to adapt in a quantum circuit are the qubits and the gate operations. Let $\mathcal{A}$ be a quantum circuit with n input qubits $|\psi_c\rangle = |\psi_{n-1}\rangle \otimes \cdots \otimes |\psi_1\rangle \otimes |\psi_0\rangle$ and m ordered gate operations represented as unitary matrices $M_{m-1}, \ldots, M_1, M_0$. Now, let $\mathcal{B}$ be a MO-1QFA that adapts $\mathcal{A}$. B is defined as follows:

- $\mathcal{B}$ has initial superposition $|q_0\rangle = |\psi_c\rangle$.

  The definition of a qubit states that it is a vector in the complex Hilbert space. Using tensor products, multiple qubits can be composited into one quantum state vector. From Definition 2, the initial superposition of a MO-1QFA is also a quantum state that can be represented as a complex-valued vector. Thus, the composite input qubits of a quantum circuit can be taken as the initial superposition of a MO-1QFA.

- The input alphabet is a set of m unique symbols $\Sigma = \{\sigma_{m-1}, \ldots, \sigma_1, \sigma_0\}$.

- The transition function δ can be represented as a set of transition matrices $\{U_\sigma\}_{\sigma \in \Sigma}$ with $U_{\sigma_i} = M_i$ for all $i \in (0, m-1)$.

  Definition 4, taken with the use of a tensor product to create operators that can act on composite quantum states, shows that each gate can be represented as a unitary matrix. Remark 1 shows that the transition function of a MO-1QFA can also be represented as a set of unitary transition matrices. Gates and transition matrices also operate on qubits and quantum states the same way, via matrix multiplication. This is shown in Theorems 1 and 2. Therefore, to adapt quantum gate operations in the MO-1QFA model, it is simply a matter of making sure each unique gate operation is represented as a unique symbol in the input alphabet, with the corresponding transition matrix being the gate operation matrix itself.

- Q is the finite set of all combinations of n bits. Qubits are superpositions of classical bit states, so they are measured to 0 or 1. If n qubits are used in the circuit, when measured together they become n bits. The finite set of classical states is therefore all the possible combinations of n classical bits.

- $Q_{acc} \in Q$ is the set of accepting states. Q_{acc} is the set of all intended possible measured outputs of the quantum circuit.

The final state of $\mathcal{A}$ is $|\psi_{\mathcal{A}}\rangle = M_{m-1}\ldots M_1 M_0 |\psi_c\rangle$. The final state of $\mathcal{B}$ is $|\psi_{\mathcal{B}}\rangle = U_{\sigma_{m-1}}\ldots U_{\sigma_1} U_{\sigma_0} |q_0\rangle$. $|q_0\rangle = |\psi_c\rangle$, and $U_{\sigma_i} = M_i$ for all $i \in (0, m-1)$. Therefore, $|\psi_{\mathcal{A}}\rangle = |\psi_{\mathcal{B}}\rangle$. However, even if they have the same final state, measurements of the same state can result in different outcomes due to the probabilistic nature of quantum systems. $|\psi_{\mathcal{A}}\rangle$ and $|\psi_{\mathcal{B}}\rangle$ can then be said to have the same measurement probabilities but not necessarily the same outcomes once measured. If both $\mathcal{A}$ and $\mathcal{B}$ are run a very large amount of times, the set of possible outcomes they can produce are the same.

Q_{acc} is the set of all intended possible measured outputs of the quantum circuit. The set of possible outcomes of $\mathcal{A}$ and $\mathcal{B}$ are the same. Therefore, if the implementation of $\mathcal{A}$ is as intended, any measured outcome of $\mathcal{B}$ would be in Q_{acc}.

If there exists an outcome of $\mathcal{B}$ not in Q_{acc}, it would be because there exists an outcome of $\mathcal{A}$ with the same value. If the output of $\mathcal{B}$ is not accepted, the circuit is not working as intended and has an error.

In conclusion, we have Theorem 3.

Theorem 3. *Let $\mathcal{A}$ be a quantum circuit. Then we can construct a MO-1QFA $\mathcal{B}$ such that $\mathcal{B}$ simulates $\mathcal{A}$.*

3.2 Adapting QECCs

In the case of adapting linear QECCs, there are 2 categories: codes that use and measure a syndrome, and codes that do not have any intermediate measurements. In the case that the code has no intermediate measurement gate, Q_{acc} is $0^n, 1^n$ since the value of the logical qubit should be uniform with no errors after going through a QECC. In the case that the code measures a syndrome, the circuit is adapted only up to the error detection part. Q_{acc} would still have the set of bits with no errors, but it would also have the set of all possible errors. The set of all possible errors are the bit-strings where the first n bits represent the bits with errors and the remaining bits represent the syndrome. The syndrome must match the error. A simple example of this is given below.

Example: Three-qubit code circuit The three-qubit code is a basic introductory code that is described as not being a full QECC[9] because it can only detect and correct one kind of error (bit-flip or phase-flip).

Now, using a simple circuit implementation of the error detection portion of a three-qubit quantum error correction code, we can create a QFA. This code can be used to detect bit-flip errors on a single qubit $|\psi\rangle$. The qubit is first encoded as a logical qubit $|\psi\rangle_L$. To detect errors, ancilla qubits controlled on the logical qubit are used. They can detect errors on the first and second qubits and the second and third qubits respectively, which is sufficient to detect a single bit-flip error. The possible final states of the system as listed in tables 1 and 2 of [9] are as follows:

- No bit-flip errors: $\alpha|000\rangle|00\rangle + \beta|111\rangle|00\rangle$ before measuring the ancilla qubits, and $\alpha|000\rangle + \beta|111\rangle$ after measuring the ancilla qubits, with the value of the measurement being 00.

- Bit-flip error on first qubit: $\alpha|100\rangle|10\rangle+\beta|011\rangle|10\rangle$ before measuring the ancilla qubits, and $\alpha|100\rangle+\beta|011\rangle$ after measuring the ancilla qubits, with the value of the measurement being 10.
- Bit-flip error on second qubit: $\alpha|010\rangle|11\rangle+\beta|101\rangle|11\rangle$ before measuring the ancilla qubits, and $\alpha|010\rangle+\beta|101\rangle$ after measuring the ancilla qubits, with the value of the measurement being 11.
- Bit-flip error on third qubit: $\alpha|001\rangle|01\rangle+\beta|110\rangle|01\rangle$ before measuring the ancilla qubits, and $\alpha|001\rangle+\beta|110\rangle$ after measuring the ancilla qubits, with the value of the measurement being 01.

Note that the first three qubits of each state represent the logical qubit while the last two qubits represent the ancilla qubits.

The circuit implementation of the code is shown in Figure 1.

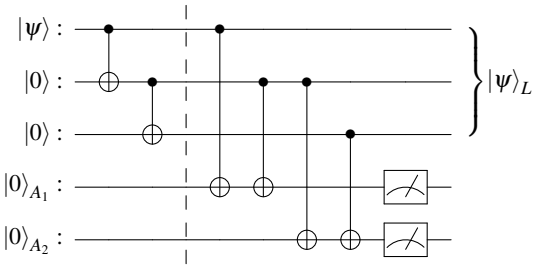


Fig. 1. Three-qubit code circuit diagram

The circuit has gates $C_{0,1}, C_{1,2}, C_{0,3}, C_{1,3}, C_{1,4}, C_{2,4}$. The inputs of the circuit are the logical qubit $|\psi\rangle_L = |\psi\rangle \otimes |0\rangle \otimes |0\rangle$ and ancilla qubits $|00\rangle_A$. The circuit has a measurement gate for the ancilla qubits at the end of the circuit diagram, but for this example, the measurement will be applied to all the qubits.

Let $\mathcal{M} = (Q, \Sigma, \delta, q_0, Q_{acc})$ be a MO-1QFA that simulates the above circuit, where:

- Q is the finite set of all combinations of 5 bits,
- $\Sigma = \{C_{0,1}, C_{1,2}, C_{0,3}, C_{1,3}, C_{1,4}, C_{2,4}\}$ is the finite set of input symbols taken from the gates of the circuit,
- $q_0 = |\psi\rangle_L \otimes |00\rangle_A$ is the initial state and is equal to the inputs of the circuit, which are the logical qubit $|\psi\rangle_L$ and the ancilla qubits $|00\rangle_A$,
- δ is the transition function defined as such:
  - The transition matrices of $C_{0,1}, C_{1,2}, C_{0,3}, C_{1,3}, C_{1,4}, C_{2,4}$ are the matrix representations of the CNOT gates corresponding to each input $C_{x,y}$, where x is the control qubit and y is the target qubit.

- Q_{acc} is the set of accepting states $\{00000, 11100, 10010, 01110, 01011, 10111, 00101, 11001\}$.

The order of application of the transition matrices is $U_{C_{2,4}}U_{C_{1,4}}U_{C_{1,3}}U_{C_{0,3}}U_{C_{1,2}}U_{C_{0,1}}|q_0\rangle$. This is equal to the final state of the automaton and is also the order of application of the gates in the circuit. The MO-1QFA is measured at this final state.

The set of accepting states is the set of states that can be produced by the circuit. In this example, they are the set of states where at most one of the first three qubits has a bit-flip error and the syndrome reflects which bit has the error.

Simulating error on the three-qubit code In the error-detecting circuit, bit-flip errors can occur on any qubit in the logical qubit at the dotted line in Figure 1. In order to incorporate this in $\mathcal{M}$, an intermediate error gate E_x that simulates bit-flip errors on the logical qubit can be added to the MO-1QFA. Thus, $\Sigma = \Sigma \cup \{E_x\}$ and the order of application of the transition matrices becomes $U_{C_{2,4}}U_{C_{1,4}}U_{C_{1,3}}U_{C_{0,3}}U_{E_x}U_{C_{1,2}}U_{C_{0,1}}|q_0\rangle$. Assuming that the error can occur on any of the qubits in the logical qubit and on multiple qubits, $U_{E_x} = (H \times X)^{\otimes 3} \otimes I^{\otimes 2}$. The final state of $\mathcal{M}$, when measured, has a probability of producing every outcome in Q_{acc}. This simulates general bit-flip errors on qubits in the logical qubit.

For bit-flip errors on specific qubits in the logical qubit, the error must be known in advance and the behavior of the matrix U_{E_x} must be adjusted to apply X-gates on the qubits with error. Suppose that an error occurs on the second qubit and the original qubit is initialized to the value $|1\rangle$. Then, $U_{C_{1,2}}U_{C_{0,1}}|q_0\rangle = C_{1,2}C_{0,1}(|\psi\rangle_L \otimes |00\rangle) = |11100\rangle$. At the current states of the circuit and $\mathcal{M}$, their states have the same value $|11100\rangle$. If a bit-flip error occurs at the second qubit, the final state of the circuit will be $|10111\rangle$. To perfectly simulate the circuit's behavior, $U_{E_x} = I \otimes X \otimes I^{\otimes 3}$ to simulate a bit-flip error on the second qubit by applying an X-gate to it. The final state of $\mathcal{M}$ will be the same as the circuit, $|10111\rangle$.

Simulating error correction on the three-qubit code In order to simulate error correction on the three-qubit code, syndrome measurement was removed. It uses CNOT and Toffoli gates to correct bit-flip errors. The CNOT and Toffoli gates $C_{3,0}$ and $T_{(3,4),0}$ are applied consecutively to correct bit-flip errors on the first qubit. The Toffoli gate $T_{(3,4),1}$ is applied to correct bit-flip errors on the second qubit. Lastly, the CNOT and Toffoli gates $C_{4,2}$ and $T_{(3,4),2}$ are applied consecutively to correct bit-flip errors on the third qubit. The notation of the Toffoli gate used above is $T_{(x,y),z}$, where (x,y) are the control qubits and z is the target qubit. The error-correcting circuit is illustrated in Figure 2. Note that in this circuit, error can occur at the first dotted line.

Below is an explanation of the error-correcting circuit.

- $C_{3,0}$ will flip qubit 0 if qubit 3, the first ancilla qubit, is $|1\rangle$. $T_{(3,4),0}$ will flip qubit 0 if both ancilla qubits are $|1\rangle$. If the value of the syndrome is $|00\rangle$ (no error) or $|01\rangle$ (error on third qubit), none of the gates will operate on the first qubit as there is no error on it. If the value of the syndrome is $|11\rangle$ (error in the second qubit),

$C_{3,0}$ will flip qubit 0 and $T_{(3,4),0}$ will flip qubit 0, resulting in qubit 0 retaining its original value because Pauli matrices are involutory. However, if the value of the syndrome is $|10\rangle$ (error on first qubit), $C_{3,0}$ will flip qubit 0 and $T_{(3,4),0}$ will not operate, ultimately flipping qubit 0 and correcting the bit-flip error on it.

- $T_{(3,4),1}$ will flip qubit 1 if both ancilla qubits are $|1\rangle$. If there is no error on qubit 1, the value of the syndrome will be $|00\rangle$, $|10\rangle$, or $|01\rangle$. In those cases, qubit 1 will remain as is. However, if there is an error on qubit 1 the syndrome will have the value $|11\rangle$, triggering the Toffoli gate and flipping qubit 1 to correct the bit-flip error.
- Much like with qubit 0, qubit 2 will be flipped by $C_{4,2}$ or $T_{(3,4),2}$ if qubit 4 or both qubits 3 and 4 are $|1\rangle$. If there is no bit-flip error on qubit 2, the syndrome will be $|00\rangle$, $|10\rangle$, or $|11\rangle$ and qubit 2 will retain its original value. Otherwise, the syndrome will be $|01\rangle$, and qubit 2 will be flipped to correct the bit-flip error on it.

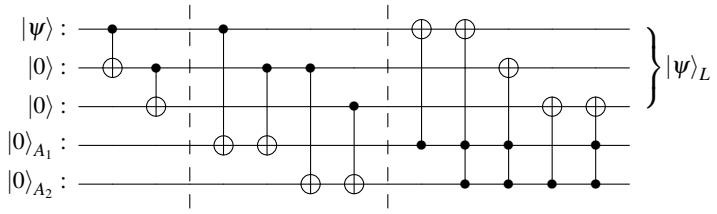


Fig. 2. Three-qubit code circuit with error correction

The MO-1QFA $\mathcal{M}$ (from Section 3.2) can be modified to simulate this new circuit by appending the symbols $\{C_{3,0}, T_{(3,4),0}, T_{(3,4),1}, C_{4,2}, T_{(3,4),2}\}$ to Σ. Their corresponding matrix representations will also be appended to δ and will be applied in the same order. Q_{acc} is redefined the set of all bit-strings of length 5 that start with '000' or '111'. This is because at the end of the error-correcting sequence, the bit-flip error should be corrected, and all the qubits that compose the logical qubit should have uniform values.

4 Conclusion

Recent notable advances in quantum error correction are in the implementation of surface codes[6]. Surface codes, which are types of topological codes, are complex codes that operate on two-dimensional lattices of qubits. In line with this, current research trends on the application of automata theory in quantum error correction mainly leverage the properties of quantum cellular automata (QCAs) to build circuits and systems[7] [11]. The approach used in this paper is markedly different. Rather than building QECCs based on automata, it capitalizes on the structural similarities of simple QFAs and QECCs to build QFAs based on existing QECCs. Basic introductory linear QECCs are used in building these QFAs because their structure is very similar

to that of a basic QFA, so it is relatively straightforward to adapt QFAs as models for linear QECCs. It is possible to simply use the easiest method of a 1-to-1 translation from quantum gate operations to QFA transition matrices and qubits to quantum state vectors.

The main benefit of using QFA models for linear QECCs is the increased flexibility of representation. Using the QFA representation of a quantum circuit, the gates can be broken down or put together. Stabilizer-controlled operations can be broken down into individual gate operations when represented as state transitions, and multiple operators can be condensed into one transition. Previous research noted that the number of steps in the QFA could also serve as an algorithm performance metric[18], so different transition functions may be able to serve as different benchmarks for algorithms.

Future research can explore alternative transition functions that are not 1-to-1 copies of circuit gate operations. Outside of QECC research, QFA-based algorithm performance metrics and language recognition performance can also be explored. Using different kinds of QFA as models for quantum circuits could also be another avenue for further studies. If a MM-1QFA, for example, could be modified to selectively measure only the ancilla qubits, perhaps some circuits with intermediate measurements could be adapted. Other models may be used to adapt some kinds of quantum circuits that are out of the scope of this paper, such as those that use quantum teleportation and those that have the aforementioned intermediate measurement gates.

References

1. A. Ambainis and R. Freivalds. 1-way quantum finite automata: strengths, weaknesses and generalizations, 1998.
2. Andris Ambainis and Abuzer Yakaryilmaz. Automata and quantum computing. CoRR, abs/1507.01988, 2015.
3. Koustubh Phalak Avimita Chatterjee and Swaroop Ghosh. Quantum error correction for dummies. In 2023 IEEE International Conference on Quantum Computing and Engineering (QCE), page 1433–1440. IEEE, September 2023.
4. Amandeep Singh Bhatia and Ajay Kumar. Quantum finite automata: survey, status and research directions. CoRR, abs/1901.07992, 2019.
5. Utku Birkan, Özlem Salehi, Viktor Olejar, Cem Nurlu, and Abuzer Yakaryılmaz. Implementing quantum finite automata algorithms on noisy devices, 2021.
6. Earl Campbell. A series of fast-paced advances in quantum error correction. Nature Reviews Physics, 6(3):160–161, 2024.
7. Thiago F. Cesar, Luiz F.M. Vieira, Marcos A.M. Vieira, and Omar P. Vilela Neto. Cellular automata-based byte error correction in qca. Nano Communication Networks, 23:100278, 2020.
8. Yu-Fang Chen, Kai-Min Chung, Ondřej Lengál, Jyun-Ao Lin, Wei-Lun Tsai, and Di-De Yen. An automata-based framework for verification and bug hunting in quantum circuits. Proc. ACM Program. Lang., 7(PLDI), June 2023.
9. Simon J Devitt, William J Munro, and Kae Nemoto. Quantum error correction for beginners. Reports on Progress in Physics, 76(7):076001, June 2013.
10. Daniel Gottesman. An introduction to quantum error correction and fault-tolerant quantum computation, 2009.

11. T. L. M. Guedes, D. Winter, and M. Müller. Quantum cellular automata for quantum error correction and density classification. Phys. Rev. Lett., 133:150601, Oct 2024.
12. Lvzhou Li, Daowen Qiu, Xiangfu Zou, Lvjun Li, Lihua Wu, and Paulo Mateus. Characterizations of one-way general quantum finite automata. Theoretical Computer Science, 419:73–91, 2012.
13. Cristopher Moore and James P. Crutchfield. Quantum automata and quantum grammars. Theoretical Computer Science, 237(1):275–306, 2000.
14. Azaria Paz. Introduction to Probabilistic Automata. Academic Press, 1971.
15. Michael O. Rabin. Probabilistic automata. Information and Control, 6(3):230–245, 1963.
16. Peter W. Shor. Scheme for reducing decoherence in quantum computer memory. Phys. Rev. A, 52:R2493–R2496, Oct 1995.
17. Marielle Stoelinga. An introduction to probabilistic automata. Bull. EATCS, 78:176–198, 2002.
18. G.F. Viamontes, I.L. Markov, and J.P. Hayes. Quantum Circuit Simulation. Springer Netherlands, 2009.