Software Pauli Tracking for Quantum Computation

Alexandru Paler[1]   Simon Devitt[2]   Kae Nemoto[2]   Ilia Polian[1]

[1]Faculty of Informatics and Mathematics
University of Passau
Innstr. 43, D-94032 Passau, Germany
{alexandru.paler|ilia.polian}@uni-passau.de

[2]National Institute of Informatics
2-1-2 Hitotsubashi, Chiyoda-ku
Tokyo, Japan
{devitt|nemoto}@nii.ac.jp

Abstract—**The realisation of large-scale quantum computing is no longer simply a hardware question. The rapid development of quantum technology has resulted in dozens of control and programming problems that should be directed towards the classical computer science and engineering community. One such problem is known as Pauli tracking. Methods for implementing quantum algorithms that are compatible with crucial error correction technology utilise extensive quantum teleportation protocols. These protocols are intrinsically probabilistic and result in correction operators that occur as byproducts of teleportation. These byproduct operators do not need to be corrected in the quantum hardware itself. Instead, byproduct operators are tracked through the circuit and output results *reinterpreted*. This tracking is routinely ignored in quantum information as it is assumed that tracking algorithms will eventually be developed. In this work we help fill this gap and present an algorithm for tracking byproduct operators through a quantum computation. We formulate this work based on quantum gate sets that are compatible with all major forms of quantum error correction and demonstrate the completeness of the algorithm.**

I. INTRODUCTION

Quantum computing promises exponential speed-up for a number of relevant computational problems. Building a scalable and reliable quantum computer is one of the grand challenges of modern science. While small-scale quantum computers are routinely being fabricated and operated in the laboratory [1], [2], [3], [4], [5], [6], they can only serve as feasibility studies, and fundamental breakthroughs will be required before a truly practical quantum computer can be built. As the size of computers within the reach of state-of-the-art technologies increases, the focus of interest shifts from their basic physical principles to structured design methodologies that will allow to realise large-scale systems [7], [8], [9], [10].

A given technology is suited for construction of general-purpose quantum computers if it supports a direct realisation of a *universal quantum gate set* which can implement or approximate arbitrary functions [11]. Moreover, today's quantum systems exhibit high error rates and require effective *quantum error-correcting codes* (QECC) [12]. Consequently, building a practical quantum computer requires an universal gate set which can be implemented in an error-corrected manner.

In this paper, we consider a class of quantum circuits based on an universal gate set that consists of just two types of operations: injection of specific quantum states into the circuit and the controlled-not (CNOT) operation. Using the technique of *quantum teleportation*, state injections are mapped to *rotational gates* that together with the CNOT operation provide universality [20]. The advantage of this gate set is that it can be seamlessly integrated into very advanced QECC schemes, allowing for scalable, large-scale information processing [12], [13]. However, as quantum teleportation is inherently probabilistic the direction of qubit rotations is random. This randomness can be corrected via a technique known as *Pauli tracking*. Pauli tracking operates by constructing a classical record of each teleportation result and reinterprets later results during the computation. This tracking means that we do not need to perform active quantum corrections because of the probabilistic nature of teleportation operations. This technique is well known in the quantum information community and routinely ignored (referred to as working in the *Pauli frame*). However, to our knowledge, no details on the algorithm necessary to perform this tracking have been presented.

The contributions of this paper are as follows. First, we present teleportation-based quantum computing in a generic and algorithmic way accessible to the design community. There are many types of corrections that are combined to define the Pauli frame of a quantum computation. The two most important arise from teleportation operations when performing arbitrary rotations and the second arises from QEC decoding operations. Without loss of generality we will focus on the first. By doing this, we completely detach our description from a particular type of QECC; in fact, an arbitrary QECC can be applied on top of the basic scheme with a minor adjustment to the algorithm.

Second, we introduce a new algorithm for *Pauli tracking*. This algorithm allows us to postpone corrections until the end of computation where we adjust the output of the computation based on the current state of the Pauli frame. The algorithm is completely classical and can be implemented in software and run on the control computer rather than on the quantum hardware. We formalise the algorithm and prove its correctness. Experimental results show that Pauli tracking is efficient.

The paper is organised as follows: In section II we introduce the basics of quantum computation. Section III details the compatible gate sets for fault-tolerant, error-corrected quantum computation and section IV introduces teleportation-based quantum gates. Finally, section V illustrates the Pauli tracking algorithm and section VI presents several simulation results.

II. QUANTUM COMPUTING

Quantum circuits represent and manipulate information in *qubits* (quantum bits). While classical bits assume either logic value 0 or 1, qubits may be in *superposition* of these two values. A single qubit has a *quantum state* $|\psi\rangle = (\alpha_0, \alpha_1)^T = \alpha_0|0\rangle + \alpha_1|1\rangle$. Here, $|0\rangle = (1, 0)^T$ and $|1\rangle = (0, 1)^T$ are quantum analogons of classical logic values 0 and 1,

respectively. α_0 and α_1 are complex number called *amplitudes* with $|\alpha_0|^2 + |\alpha_1|^2 = 1$. $|0\rangle$ and $|1\rangle$ are orthonormal vectors and form a basis of $\mathbb{C}^2$.

A state $(\alpha_0, \alpha_1)^T$ may be modified by applying single-qubit *quantum gates*. Each quantum gate corresponds to a complex unitary matrix, and gate function is given by multiplying that matrix with the quantum state. Two single-qubit gates that are highly relevant in the context of this paper are the X and the Z gate with the following matrices:

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \qquad Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

The application of X to a state results in a *bit flip*: $X(\alpha_0, \alpha_1)^T = (\alpha_1, \alpha_0)^T$; $|0\rangle$ is mapped to $|1\rangle$, and vice versa. The application of the Z gate results in a *phase flip*: $Z(\alpha_0, \alpha_1)^T = (\alpha_0, -\alpha_1)^T$. Bit and phase flips are used for modelling the effects of errors on the quantum state as qubit errors can be decomposed into combination of bit and/or phase flips. Further important single-qubit quantum gates, in the context of a fully error-corrected system, are

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \quad P = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix} \quad T = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\frac{\pi}{4}} \end{pmatrix},$$

where $T^2 = P$ and $P^2 = Z$.

It is not possible to directly read out the amplitudes of a qubits state. A *measurement* has to be performed instead. Quantum measurement is defined with respect to a basis and yields one of the basis vectors with a probability related to the amplitudes of the quantum state. Of importance in this work are Z- and X-measurements. Z-measurement is defined with respect to basis $(|0\rangle, |1\rangle)$. Applying a Z-measurement to a qubit in state $|\psi\rangle = \alpha_0|0\rangle + \alpha_1|1\rangle$ yields $|0\rangle$ with probability $|\alpha_0|^2$ and $|1\rangle$ with probability $|\alpha_1|^2$. Moreover, the state $|\psi\rangle$ *collapses* into the measured state, that is, becomes either $|0\rangle$ or $|1\rangle$. X-measurement is defined with respect to the basis $(|+\rangle, |-\rangle)$, where $|+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ and $|-\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$.

A multi-qubit circuit processes states represented by an exponential number of amplitudes. The state of a circuit with n qubits has 2^n amplitudes α_y with $y \in \{0,1\}^n$ and $\sum_y |\alpha_y|^2 = 1$. For example, the state of a 2-qubit circuit is $|\psi\rangle = (\alpha_{00}, \alpha_{01}, \alpha_{10}, \alpha_{11})^T = \alpha_{00}|00\rangle + \alpha_{01}|01\rangle + \alpha_{10}|10\rangle + \alpha_{11}|11\rangle$. Here, $|00\rangle = (1,0,0,0)^T$, $|01\rangle = (0,1,0,0)^T$, $|10\rangle = (0,0,1,0)^T$ and $|11\rangle = (0,0,0,1)^T$ form a basis for $\mathbb{C}^4$. Measuring multiple qubits of a circuit again results in one basis vector with the probability given by the corresponding amplitude, $|\alpha_y|^2$.

Quantum gates may act on several qubits simultaneously. A gate that acts on n qubits is represented by a $2^n \times 2^n$ complex unitary matrix. One important two-qubit gate is the *controlled-not* $CNOT(c, t)$ gate, where the c qubit conditionally flips the state of the t qubits when set to $|1\rangle$. Moreover, it is possible to represent a single-qubit gate (or, more generally, a gate acting on less qubits than n) by using tensor product. For example, the $2^2 \times 2^2$ matrix $H \otimes I$ (where I is the identity matrix) applies the Hadamard gate H to the first qubit of a two-qubit circuit while leaving the second qubit unchanged. The first qubit of the CNOT gate is called *control qubit* c and the second is the *target*

qubit t. Below are the matrices of the $CNOT(c = 1, t = 2)$ and the $H \otimes I$ operations.

$$CNOT = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \quad H \otimes I = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & -1 \end{pmatrix}$$

A quantum circuit with n qubits and m gates $g_1, \ldots, g_m$ takes an *input state* $\phi_0 \in \mathbb{C}^{2^n}$ and successively applies the transformations corresponding to each gate: $\phi_1 = M_{g_1}\phi_0$; $\phi_2 = M_{g_2}\phi_1$; and so forth, where M_{g_i} is the $2^n \times 2^n$ matrix of gate g_i. Selective qubit measurement may also be interspersed with these gates and at the end of computation the *output state* ϕ_m of the circuit is completely measured.

III. FAULT-TOLERANT GATE SET

It is important to distinguish between the generic mathematical model of quantum computation and subsets of it that are suited for an actual physical realisation. A number of technologies have been suggested for implementing quantum computation [14], [15], [16], [17]. While every unitary complex matrix qualifies as a quantum gate in the mathematical formalism, most implementation technologies only allow a direct physical realisation of relatively few gates. Therefore, *universal gate sets*, that is, collections of quantum gates that can represent or approximate an arbitrary quantum circuit, are of interest This concept is similar to universal gate libraries in digital circuit design, where each Boolean function can be mapped to a circuit composed of, for instance, AND2 gates and inverters. One instance of a universal quantum gate set is $\{CNOT, H, P, T\}$. A technology is suitable for realisation of arbitrary quantum algorithms if it has a direct implementation for at least one universal gate set.

A further key requirement for successful realisation of quantum circuits is the ability to perform *error correction* during computation. States of actual quantum systems are inherently fragile and are affected even by the slightest interaction with their environment. Therefore, *quantum error-correcting codes* (QECC) introduce substantial redundancy to compensate for impact on the quantum state. For instance, Shor's code [18] uses nine qubits to represent one *encoded* qubit: the qubit is first triplicated in order to detect and correct phase flips, and the resulting qubit triplet is again triplicated to protect them against bit flips. It can be shown that this construction is sufficient to correct all errors affecting any one of the nine physical qubits. We call the nine qubits used for encoding *physical qubits* and one error-corrected qubit *logical qubit*. The strength of a QECC can be quantified by how many physical qubit errors that have to occur before the logical qubit is corrupted. For Shor's code, a single error on any of the nine physical qubit is tolerated; the probability of failure for the Shor's code is bounded by the probability of two or more errors occurring on separate qubits. More advanced codes can tolerate multiple errors and have lower probabilities of failure at the expense of more physical qubits to encode a single logical qubit.

A *fault-tolerant gate* for a given QECC acts directly on encoded qubits and produces legal encodings with respect to that QECC at its outputs. For example, a fault-tolerant

implementation of a CNOT gate for the Shor's code would take 18 physical qubits as inputs, interpret nine of the qubits as the logical control qubit and the other nine qubits as the logical target qubit, and produce 18 physical qubits that encode the two logical qubits as its outputs. In self-checking design of classical circuits, error-correcting codes with this property are called *closed* with respect to the gate's operation; for example, bi-residue codes are closed under both addition and multiplication [19]. The closure property is advantageous because decoding and re-encoding of codewords before and after operation are avoided. This advantage is even more pronounced for quantum circuits with their extremely high expected error rates. As a consequence, a practical universal gate set should consist of fault-tolerant gates that allow circuit operation with errors continuously taking place.

IV. TELEPORATION-BASED QUANTUM COMPUTING

In this paper, we focus on a set of operations that can be implemented in a fault-tolerant manner with respect to several state-of-the-art QECC [20]. This set consists of the CNOT gate and two *state injection* operations. State injection refers to initializing a qubit in one of the two following states

$$|A\rangle = \frac{1}{\sqrt{2}}(|0\rangle + e^{i\pi/4}|1\rangle) \qquad |Y\rangle = \frac{1}{\sqrt{2}}(|0\rangle + i|1\rangle).$$

Using these states and a technique called *quantum teleportation*, it is possible to obtain the following three quantum gates:

$$R_x\left(\frac{\pi}{4}\right) = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & -i \\ -i & 1 \end{pmatrix}$$

$$R_z\left(\frac{\pi}{4}\right) = P = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix} \qquad R_z\left(\frac{\pi}{8}\right) = T = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\frac{\pi}{8}} \end{pmatrix}$$

In the *Bloch sphere representation* of a quantum state, $R_x(\theta)$ and $R_z(\theta)$ stand for a rotation around the X- and the Z-axis by angle θ; see [11] for details. We use the following abbreviations for brevity: $R_x^4 := R_x(\pi/4)$; $R_z^4 := R_z(\pi/4) \equiv P$; $R_z^8 := R_z(\pi/8) \equiv T$. Using the relationship $H = R_z^4 R_x^4 R_z^4$, the complete universal gate set $\{CNOT, H, P, T\}$ can be obtained based on CNOT, state injection and quantum teleportation. All these gates are compatible with error-corrected, fault-tolerant computation [20], [21]. However, quantum teleportation is probabilistic itself and may require (classical) correction that will be tracked. This is described in detail below.

The rotational gates R_x^4, R_z^4 and R_z^8 are constructed by combining state injection with quantum teleportation. Applying the three employed rotational gates to an arbitrary state $|\phi\rangle$ by quantum teleportation is shown in Fig. 1. An auxilliary qubit is initialised in state $|Y\rangle$ or $|A\rangle$ (depending on the desired rotation), and a CNOT gate is applied at the qubit that holds $|\phi\rangle$ and the auxilliary qubit (the control and target qubits are denoted by ● and ⊕, respectively). Finally, a measurement (either X or Z) is performed at the control output of the CNOT gate, indicated in Fig. 1 by an encircled X or Z. The effect at the target output is shown in Fig. 1

Gate R_x^4 : The X-measurement in circuit of Fig. 1a yields either $|+\rangle$ or $|-\rangle$. If the measurement result is $|+\rangle$, then the desired rotation $R_x(\pi/4)$ was executed and the new state at the output of the circuit is correct. If the measurement result
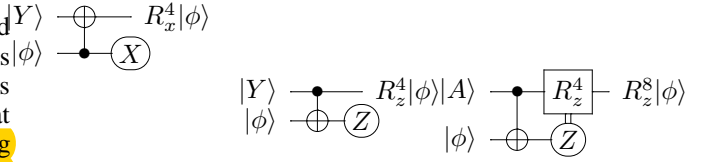

Fig. 1: Teleportation circuits used for (a) R_x^4, (b) R_z^4, (c) R_z^8

is $|-\rangle$, the applied rotation was $R_x(-\pi/4)$, i.e., the direction of the rotation was wrong. This is easily compensated by performing another rotation by angle $\pi/2$, namely applying the gate $R_x(\pi/2) = X$. It is easily checked that $XR_x(-\pi/4) = R_x(\pi/4)$. Consequently, quantum teleportation must be followed by executing the X gate at the obtained state if the measurement result is $|-\rangle$. We call this X-*correction*. Note that the decision whether X-correction is required is based on classical information (a measurement result) and can be taken by a classical computer.

Gate R_z^4 : The two possible Z-measurement results from the circuit in circuit of Fig. 1b implementing R_z^4 are $|0\rangle$ and $|1\rangle$. The state $|0\rangle$ indicates a correct teleportation where the resulting state is $|\psi\rangle = R_z(\pi/4)|\phi\rangle$. A measured state $|1\rangle$ is an indicator for the state $|\psi_f\rangle = \alpha_1|0\rangle - i\alpha_0|1\rangle$ where the input state was $|\phi\rangle = \alpha_0|0\rangle + \alpha_1|1\rangle$. In order to obtain the correct state, a Z operation followed by the X operation is applied, as it is easily verified that $|\psi\rangle = XZ|\psi_f\rangle = \alpha_0|0\rangle + i\alpha_1|1\rangle = R_z^4|\phi\rangle$. This operation is called XZ correction.

Gate R_z^8 : This gate is implemented in two stages (see Fig. 1c). The first teleporation maps state $|A\rangle$ to an intermediate state, which is then given to the R_z^4 gate from Fig. 1b that also incorporates a teleportation. The following three measurement outcomes have to be distinguished:

1) If the first measurement results in $|0\rangle$, the intermediate state is the correct result already. No correction is required, and the second rotation (including the corresponding measurement) does not have to be applied.

2) If the first measurement results in $|1\rangle$ and the input state was $|\phi\rangle = \alpha_0|0\rangle + \alpha_1|1\rangle$, the calculated state is $|\psi_{f1}\rangle = XR_z^{4\dagger}|\phi\rangle = \alpha_0|1\rangle + e^{-i\pi/4}\alpha_1|0\rangle$. This state will be used as an input for the $\pi/2$ correctional rotation (R_z^4 from Fig. 1b). If the second measurement returns $|1\rangle$, then the correction succeeded, and no further corrections are necessary: $|\psi\rangle = \alpha_0|0\rangle + e^{i\pi/4}\alpha_1|1\rangle = R_z^8|\phi\rangle$.

3) If the first measurement returns $|1\rangle$, and the second measurement yields $|0\rangle$, then the R_z^4 correction will produce state $|\psi_{f2}\rangle = i\alpha_0|0\rangle + e^{-i\pi/4}\alpha_1|0\rangle$. Then, as seen above, the XZ correction leads to $|\psi\rangle = XZ|\psi_{f2}\rangle = \alpha_0|0\rangle + e^{i\pi/4}\alpha_1|1\rangle = R_x^8|\phi\rangle$.

In summary, teleporations are probabilistic and either X or XZ corrections may be required depending on the outcomes of the measurement. It is important to understand that this non-determinism is *not* due to errors but is inherent to teleportation-based quantum computing. Algorithm 1 summarises the complete computation procedure incorporating all the required corrections in detail. The algorithm assumes a circuit that has already been mapped to the universal gates set consisting of

Algorithm 1 Teleportation-based quantum computation

Input: n-qubit quantum circuit with m gates $g_1, \ldots, g_m \in \{CNOT, R_x^4, R_z^4, R_x^8\}$, input state ϕ_0
Output: Output state ϕ_m
1: **for** $i := 1$ **to** m **do**
2:     **if** g_i is a CNOT gate **then**
3:         // Apply CNOT to current state
4:         $\phi_i := M_g \phi_{i+1}$;
5:     **else if** g_i is a R_x^4 gate on qubit k **then**
6:         Introduce new qubit l; inject state $|Y\rangle$ on l;
7:         Perform CNOT(k, l); X-measurement on qubit k;
8:         **if** measurement result is $|+\rangle$ **then**
9:             apply X-correction on qubit l;
10:         **end if**
11:         Replace qubit k in ϕ_{i-1} by qubit l to obtain ϕ_i;
12:     **else if** g_i is a R_z^4 gate **then**
13:         Introduce new qubit l; inject state $|Y\rangle$ on l;
14:         Perform CNOT (l, k); Z-measurement on k;
15:         **if** measurement result is $|0\rangle$ **then**
16:             Apply XZ-correction on qubit l;
17:         **end if**
18:         Replace qubit k in ϕ_{i-1} by qubit l to obtain ϕ_i;
19:     **else if** g_i is a R_z^8 gate **then**
20:         Introduce new qubit l; inject state $|A\rangle$ on l;
21:         Perform CNOT(l, k); Z-measurement on k;
22:         **if** measurement result is $|0\rangle$ **then**
23:             Introduce new qubit l'; inject state $|Y\rangle$ on l;
24:             Perform CNOT(l',l); Z-measurement on l;
25:             **if** measurement result is $|0\rangle$ **then**
26:                 Apply XZ-correction on qubit l';
27:             **end if**
28:             Replace qubit k in ϕ_{i-1} by qubit l' to obtain ϕ_i;
29:         **else**
30:             Replace qubit k in ϕ_{i-1} by qubit l to obtain ϕ_i;
31:         **end if**
32:     **end if**
33: **end for**
34: **return** ϕ_m;

---

the CNOT gate and the three considered rotational gates. Note that the only operations applied are state injections and CNOT gates, and that these operations are compatible with standard fault-tolerant error correction.

The computation contiuuously requests new qubits and abandons the old ones, such that the total number of used logical qubits is n or $n+1$ at any given time. In many relevant implementation technologies, hardware for each abandoned qubit can be reused for the newly requested ones. For example, if a new qubit is introduced for injecting the $|Y\rangle$ state in order to implement the R_x^4 gate, the old qubit is no longer required after the X-measurement, and it can be used for implementing further rotational gates.

V. PAULI TRACKING ALGORITHM

Teleportation-based quantum computing suffers from the necessity to conditionally perform corrections based on the measurement results. Applying correction in real time, immediately after the rotational operation, such as in Algorithm 1, results in significant interaction between quantum hardware

and the classical control computer. This may have a detrimental impact on the speed of computation and is ultimately unnecessary. In this section, we demonstrate how applying corrections can be postponed to the end of calculation without losing accuracy. For this purpose, teleportation-based quantum computation method is modified as follows.

The considered circuits still consist of CNOT gates and the three types of rotational gates implemented by state injection and quantum teleportation. Measurements are still performed during quantum teleportation, however their outcomes are stored in a variable rather than used for immediate correction. For each rotational gate g_i, variable b_i holds the result of the measurement. Note that $b_i \in \{|+\rangle, |-\rangle\}$ if g_i is a R_x^4 gate, $b_i \in \{|0\rangle, |1\rangle\}$ if g_i is a R_z^4 gate, $b_i \in \{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}$ if g_i is a R_z^8 gate, where pairs of values refer to the outcomes of two consecutive measurements.

We derive an algorithm that calculates, for a given combination of b_i values, the vector of *equivalent output correction statuses* $S = (s_1, \ldots, s_n)$. For qubit k, s_k assumes one of four values that indicate the required corrections: I (no correction), X (X-correction, i.e., a bit flip), Z (Z-correction, or a phase flip), and XZ (both X- and Z-correction). The values in S are calculated such that running the teleportation-based quantum computing (Algorithm 1) *without applying corrections in Lines 8–10, 15–17 and 22–31 and applying the correction in S to the obtained output state is equivalent to teleportation-based quantum computing with immediate correction.*

S is calculated by propagating (*tracking*) the correction status $(s_1, \ldots, s_n)$ through the circuit. The calculation is applied *after* the teleportation-based quantum computation took place and the measurement results b_i associated with all rotational gates g_i are available. Each s_k is initialised to I (no correction). Then, the gates are considered in their regular order $g_1, \ldots, g_m$. If g_i is a rotational gate on qubit k, its b_i is consulted to decide whether a correction is needed and the s_k is updated (the correction status is propagated). The propagated correction status shows up at the inputs of subsequent rotational and CNOT gates and must be taken into account when calculating the correction status at the output of that gates. We introduce the *correction status tracking function* τ that formalises the propagation.

There are two versions of τ: one for CNOT gates and for rotational gates. CNOT gates do not employ teleportation and therefore require no corrections; however, corrections that originated from rotational gates may show up at the inputs of the CNOT gate and have to be propagated to its outputs. Let c and t be the control and the target qubit of the CNOT gate, and let s_c^{in} and s_t^{in} be the correction statuses at the inputs of these qubits, respectively. Then, $\tau(s_c^{\text{in}}, s_t^{\text{in}})$ produces a pair of correction statuses $(s_c^{\text{out}}, s_t^{\text{out}})$ at the outputs of the CNOT gates by the following calculation:

$$s_c^{\text{out}} = \begin{cases} s_c^{\text{in}} & \text{if} \quad s_t^{\text{in}} \in \{I, X\} \\ s_c^{\text{in}} \oplus Z & \text{if} \quad s_t^{\text{in}} \in \{Z, XZ\} \end{cases} \quad (1)$$

$$s_t^{\text{out}} = \begin{cases} s_t^{\text{in}} & \text{if} \quad s_c^{\text{in}} \in \{I, Z\} \\ s_t^{\text{in}} \oplus X & \text{if} \quad s_c^{\text{in}} \in \{X, XZ\} \end{cases} \quad (2)$$

Here, $s \oplus Z$ and $s \oplus X$ are flipping the status of the respective

TABLE I: Correction status tracking τ for rotational gates

g_i	s_k^{in}	b_i	s_k^{out}
R_x^4	I	$|+\rangle$	I
		$|-\rangle$	X
	Z	$|+\rangle$	X
		$|-\rangle$	I
	X	$|+\rangle$	X
		$|-\rangle$	Z
	XZ	$|+\rangle$	I
		$|-\rangle$	Z
R_z^4	I	$|0\rangle$	I
		$|1\rangle$	XZ
	Z	$|0\rangle$	Z
		$|1\rangle$	X
	X	$|0\rangle$	XZ
		$|1\rangle$	I
	XZ	$|0\rangle$	X
		$|1\rangle$	X

g_i	s_k^{in}	b_i	s_k^{out}
R_z^8	I	$|0*\rangle$	I
		$|10\rangle$	XZ
		$|11\rangle$	I
	Z	$|0*\rangle$	Z
		$|10\rangle$	X
		$|11\rangle$	Z
	X	$|00\rangle$	XZ
		$|01\rangle$	I
		$|1*\rangle$	I
	XZ	$|00\rangle$	X
		$|01\rangle$	Z
		$|10\rangle$	X
		$|11\rangle$	Z

Algorithm 2 Pauli tracking

Input: n-qubit quantum circuit with m gates $g_1, \ldots, g_m \in \{CNOT, R_x^4, R_z^4, R_x^8\}$, measurement results b_i for every rotational gate g_i

Output: Equivalent output correction status $S = (s_1, \ldots, s_n)$

```
1: s₁ := s₂ := · · · := sₙ := I;
2: for i := 1 to m do
3:     if gᵢ is a CNOT gate with control/target qubits c/t then
4:         (s_c, s_t) := τ(s_c, s_t);        // Use Eqs. 1, 2
5:     else if gᵢ is a rotational gate on qubit k then
6:         s_k := τ(s_k, b_i);               // Use Table I
7:     end if
8: end for
9: return  S = (s₁, . . . , sₙ);
```

correction in s:

s	$s \oplus Z$	$s \oplus X$	s	$s \oplus Z$	$s \oplus X$
I	Z	X	X	XZ	I
Z	I	XZ	XZ	X	Z

For a rotational gate g_i on qubit k, the τ function takes the pre-stored measurement result b_i and the correction status s_k^{in} at its input and calculates the correction status s_k^{out} at its output. The values calculated by τ for the three types of rotational gates considered are given in Table I.

Algorithm 2 summarises the Pauli tracking procedure.

Example: Consider the two-qubit circuit in Fig. 2. Assume that teleportation-based quantum computing has been done without applying corrections. The recorded measurement results b_i and the calculated correction statuses $S = (s_1, s_2)$ are shown in the following table.
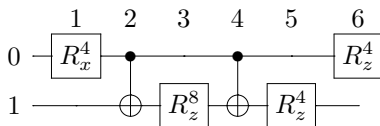
Fig. 2: Quantum circuit implemented using fault-tolerant gates

i		1	2	3	4	5	6
g_i		R_x^4	CNOT	R_z^8	CNOT	R_z^4	R_z^4
b_i		$|+\rangle$	n/a	$|10\rangle$	n/a	$|0\rangle$	$|1\rangle$
s_1	I	I	I	I	Z	Z	X
s_2	I	I	I	XZ	XZ	X	X

Initially, s_1 and s_2 are set to I (no correction required). Since $b_1 = |+\rangle$ is measured for gate g_1, no correction is required and s_1 remains I. The CNOT gate g_2 does not introduce new corrections. The measurement outcome $b_3 = |10\rangle$ of gate g_3 necessitates the XZ-correction, as can be seen in Table I. Since the target input (qubit 2) of the CNOT gate g_4 includes Z, the correction status at the control qubit is determined, using Eq. 2, as $s_1 = s_1 \oplus Z = I \oplus Z = Z$, while s_2 remains unchanged. $b_5 = |0\rangle$ leads to $s_2 = X$ and $b_6 = |1\rangle$ leads to $s_1 = X$. As a result, X-corrections must be applied at both outputs of the circuit. $\square$

The correctness of the tracking algorithm is now formally proven. The following two lemmas formulate the validity of the τ function for the individual gates. They can be verified for every combination of inputs for τ. Instead of providing the complete proof, we quote the calculation for one specific input combination, whereas the derivations for other combinations are similar.

Lemma 1 Let g_i be a CNOT gate with correction status s_c^{in} at its control and s_t^{in} at its target input and $(s_c^{\text{out}}, s_t^{\text{out}}) = \tau(s_c^{\text{in}}, s_t^{\text{in}})$. Then, performing the s_c^{in}-correction at the control input and the s_t^{in}-correction at the target input followed by application of CNOT is the same function as applying the CNOT gate first and performing s_c^{out}-correction at the control output and the s_t^{out}-correction at the target output.

Proof for $s_c^{\text{in}} = X, s_t^{\text{in}} = I$: According to Eqs. 1, 2, $(s_c^{\text{out}}, s_t^{\text{out}}) = \tau(X, I) = (X, X)$. Without loss of generality, assume that the control and target qubit of the CNOT gate are qubit 1 and 2 respectively. Then, the X-correction at the control qubit is described by matrix $X_1 = X \otimes I$ and the X-correction at the target qubit is described by matrix $X_2 = I \otimes X$. Performing the corrections first followed by the CNOT operation corresponds to the matrix $CNOT \cdot X_1 \cdot I$ while the CNOT operation followed by the two corrections are described by the matrix $X_1 \cdot X_2 \cdot CNOT$.

The equivalence of these matrices is shown below:

$$CNOT \cdot X_1$$

$$= \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix}$$

$$= \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix} \cdot \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

$$= X_1 \cdot X_2 \cdot CNOT$$

All other combinations can be calculated similarly. $\square$

Lemma 2 Let g be a rotational gate, s^{in} the correction status at its input, b the outcome of the associated measurement and $s^{out} = \tau(s^{\text{in}}, b)$ the tracked correction status at its output according to Table I. Then, performing the s^{in} correction,

TABLE II: Run-times RT (in seconds) of the Pauli tracking algorithm for circuits with n qubits and m quantum gates

n	m	RT [s]	n	m	RT [s]	n	m	RT [s]
100	1000	0	1100	1000	0.011	5100	1000	0.052
100	5000	0.002	1100	5000	0.069	5100	5000	0.309
100	10000	0.004	1100	10000	0.128	5100	10000	0.709
100	20000	0.011	1100	20000	0.300	5100	20000	1.930
100	50000	0.030	1100	50000	0.680	5100	50000	5.435

applying g and performing, if needed, correction according to b, yields the equivalent state as applying g first and performing s^{out}-correction.

Proof for gate R_z^4, $b = |1\rangle$ und $s^{\text{in}} = Z$: Since $b = |1\rangle$, regular teleportation-based computing requires an XZ-correction, such that the following four operations are applied to the state: Z for s^{in}-correction; R_z^4 for gate functionality, and XZ for the correction of the wrong rotation. From Table I, $s^{\text{out}} = \tau(Z, |1\rangle) = X$ for the gate in question. The equivalence stated in the lemma is verified by

$$XZR_z^4 Z \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

$$= \begin{pmatrix} 0 & i \\ 1 & 0 \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix} = XR_z^2$$

Other cases are checked similarly. □

Inductively applying the two lemmas to all gates in the circuit leads to the validity of the following theorem.

Theorem 1 Applying corrections calculated by Algorithm 2 on the state obtained by Algorithm 1 without performing immediate corrections results in the same state as the state obtained by Algorithm 1 when all corrections are performed immediately. □

VI. SIMULATION RESULTS

We implemented the Pauli tracking algorithm and applied it to a number of randomly generated quantum circuits with n logical qubits and m gates from the considered gate set. The results for 100, 1100 and 5100 qubits are shown in Table II. $n = 100$ is indicative of largest quantum circuits within reach of today's state-of-the-art technology and $n = 1100$ and $n = 5100$ are the expected sizes of quantum computers within a decade. It can be seen that Pauli tracking is fast and all calculations can be performed within a few seconds for all cases.

The expected number of corrections without Pauli tracking is $0.5 \cdot m_4 + 0.75 m_8 \approx m$, where m_4 is the number of gates R_x^4 and R_z^4 which require a correction with a probability 0.5 and m_8 is the number of gates R_z^8 which may require one or two corrections with the expected number of corrections equal to 0.75. The expected number of corrections with Pauli tracking is bounded by n, because corrections have to be performed on each output k with $s_k \neq I$. As most relevant circuits have far more gates than qubits, Pauli tracking substantially reduces the overall effort for corrections.

VII. CONCLUSIONS

We have presented an algorithm that can be used to perform Pauli tracking on quantum circuits compatible with all major classes of QECC. This result helps fill an important gap in the classical control software needed for large-scale quantum computation. Pauli tracking is instrumental for both error correction and for teleportation based protocols and this algorithm is easily adjustable to incorporate the required tracking for a specific implementation of quantum error correction. Future work will be focused on adapting this algorithm to popular error correction techniques such as Topological codes [13], [22] which requires even more intensive Pauli tracking due to the enormous number of teleportation gates necessary to perform a fully fault-tolerant, error corrected computation.

REFERENCES

[1] Hanson, R. & Awschalom, D. Coherent manipulation of single spins in semiconductors. *Nature (London)* **453**, 1043–1049 (2008).

[2] Press, D., Ladd, T. D., Zhang, B. & Yamamoto, Y. Complete quantum control of a single quantum dot spin using ultrafast optical pulses. *Nature (London)* **456**, 218–221 (2008).

[3] Politi, A., Matthews, J. & O'Brien, J. Shor's quantum factoring algorithm on a photonic chip. *Science* **325**, 1221 (2009).

[4] Blatt, R. & Wineland, D. Entangled states of trapped atomic ions. *Nature (London)* **453**, 1008–1015 (2008).

[5] Pla, J. *et al.* A single-atom electron spin qubit in Silicon. *Nature (London)* **489**, 541–545 (2012).

[6] Lucero, E. *et al.* Computing prime factors with a Josephson phase qubit quantum processor. *Nature Physics* **8**, 719–723 (2012).

[7] Cheung, D., Maslov, D. & Severini, S. Translationt echniques between quantum circuit architectures. *Workshop on Quantum Information* (2013).

[8] Paler, A. *et al.* Synthesis of topological quantum circuits. *NANOARCH*, (2012).

[9] Saeedi, M. & Markov, I. Synthesis and optimization of reversible circuits—a survey. *ACM Computing Surveys* **45**, 21–34 (2013).

[10] Saeedi, M., Wille, R. & Drechsler, R. Synthesis of quantum circuits for linear nearest neighbor architectures. *Quantum Information Processing* **10**, 355–377 (2011).

[11] M.A. Nielsen and I.L. Chuang. *Quantum Computation and Information*. Cambridge University Press, second edition, 2000.

[12] Lucero, S.J., Munro, W.J. & Nemoto, K. Quantum Error Correction for Beginners. *Rep. Prog. Phys* **76**, 076001 (2013).

[13] Fowler, A., Mariantoni, M., Martinis, J. & Cleland, A. Surface Codes, Towards practical large-scale quantum computation. *Phys. Rev. A.* **86**, 032324 (2012).

[14] Devitt, S. *et al.* Architectural design for a topological cluster state quantum computer. *New. J. Phys.* **11**, 083032 (2009).

[15] Yao, N. *et al.* Scalable Architecture for a Room Temperature Solid-State Quantum Information Processor. *Nature Communications* **3**, 800 (2012).

[16] Meter, R. V., Ladd, T., Fowler, A. & Yamamoto, Y. Distributed Quantum Computation Architecture Using Semiconductor Nonophotonics. *Int. J. Quant. Inf.* **8**, 295–323 (2010).

[17] Jones, N. C. *et al.* A Layered Architecture for Quantum Computing Using Quantum Dots. *Phys. Rev. X.* **2**, 031007 (2012a).

[18] Shor, P.W. A Scheme for reducing decoherence in quantum computer memory. *Phys. Rev. A.* **52**, R2493 (1995).

[19] Koren, I. & Krishna, C. M. *Fault-Tolerant Systems*. Morgan-Kaufman Publishers, San Francisco, 2007.

[20] Bravyi, S. & Kitaev, A. Universal quantum computation with ideal Clifford gates and noisy ancillas. *Phys. Rev. A.* **71**, 022316 (2005).

[21] Devitt, S.J., Stephens, A.M. Munro, W.J., & Nemoto, K. Requirements for fault-tolerant factoring on an atom-optics quantum computer. *Nature (communications)* In Press, (2013).

[22] Raussendorf, R., Harrington, J. & Goyal, K. Topological fault-tolerance in cluster state quantum computation. *New J. Phys.* **9**, 199 (2007).