Fast equivalence checking of quantum circuits of Clifford gates

Dimitrios Thanos, Tim Coopmans, and Alfons Laarman

Leiden Institute of Advanced Computer Science (LIACS), Leiden University, Leiden 2333 CA, The Netherlands

Abstract. Checking whether two quantum circuits are equivalent is important for the design and optimization of quantum-computer applications with real-world devices. We consider quantum circuits consisting of Clifford gates, a practically-relevant subset of all quantum operations which is large enough to exhibit quantum features such as entanglement and forms the basis of, for example, quantum-error correction and many quantum-network applications. We present a deterministic algorithm that is based on a folklore mathematical result and demonstrate that it is capable of surpassing previously considered state-of-the-art method in terms of performance. In particular, given two Clifford circuits as sequences of single- and two-qubit Clifford gates, checks their equivalence in $O(n^2 + n \cdot m)$ time in the number of qubits n and number of elementary Clifford gates m. Using the performant Stim simulator, we check equivalence of quantum circuits up to 1000 qubits (with a circuit depth of 10.000 gates) in ∼22 seconds and 100.000 qubits (depth 10) in ∼15 minutes, outperforming the existing SAT-based approach by an order of magnitude. This approach shows that the correctness of application-relevant subsets of quantum operations can be verified up to large circuits in practice.

1 Introduction

Quantum computing promises to perform classically intractable tasks for a large range of applications [35,33]. While we are entering the era of Noisy Intermediate-Scale Quantum computing [38], the high noise levels require us to very accurately compile textbook quantum circuits onto real-world devices, which can only handle shallow-depth circuits and have various constraints (connectivity, topology, native gate sets, etc.) [20,18]. A crucial part of the design and optimization over quantum circuits which satisfy the specification is *verifying* whether two quantum circuits, each presented by a classical description, implement the same quantum operation, i.e. checking equivalence of quantum circuits.

Correctness verification is a well-studied field in the classical domain [22] [31] [30] but unfortunately not all methods directly carry over to quantum computing because the state of n quantum bits is generally represented as 2^n complex values [35]. Equivalence checking of quantum circuits, phrased as either exact 'non-identity checking'[1] or approximate non-identity checking whether the circuit is close or far away from the identity circuit, falls in quantum complexity classes which are analogs of NP [10,4] (NQP [39] and QMA [27,28], respectively). Thus we should not hope for efficient algorithms in general.

Existing deterministic methods analyzing circuits only consisting of only quantum gates as quantum operations (no quantum measurements) are based on encoding as Boolean satisfiability instances [9] (also [46,47] for restricted circuits), satisfiability modulo theories [8], path-sums [5,6], rewrite rules [37] [19][45], and on various flavors of decision diagrams, including QMDD [16,41,36,15], LIMDD [42], Tensor-DD [26], BDD [44,17] and others [43,48]. In addition, some probabilistic methods are known [14,32].

In this paper, we focus on exact equivalence checking of two (classical descriptions of) circuits with Clifford gates only, a subset of all quantum gates which is ubiquitous to quantum computing and is highly relevant for quantum error correction [23,40] and quantum networking applications [25]. For exact non-identity check of Clifford circuits, a reduction to satisfiability was presented by [9], in a tool called QuSAT. For approximate non-identity check, a polynomial-time algorithm exists [7] whose runtime scales with the accuracy of the approximation (a polynomial in the number of qubits).

Here, we demonstrate that a folklore result (10.5.2 in [35]) translates into a deterministic algorithm of polynomial-time $O(n^2 + m \cdot n)$ for exact equivalence checking of Clifford circuits, with n the number of qubits and m the sum of elementary Clifford gates in the two circuits. We will state that algorithm explicitly in a self-contained manner. The main idea holds for arbitrary quantum circuits, and thus we will state the theorem in its full generality. In the particular case of equivalence checking of Clifford circuits, the algorithm is reduced to simulating the Clifford circuit, which can be done efficiently [24,3].

We empirically evaluate the algorithm by using the performant Clifford-circuit simulator Stim [21], reaching circuit depths of 1000 qubits and 10.000 elementary Clifford gates in less than a minute, and 100.000 qubits for depth-10 circuits in approximately 15 minutes, outperforming the SAT-based approach by an order of magnitude. Our open-source implementation can be found on [2].

We emphasize that the task in this work is equivalence checking given a white-box *classical* descriptions of the quantum circuit, as opposed to the different task where one is given access to a *quantum* computer which performs the quantum circuit as black box [34]. For Cliffords, specifically see [11] and [32].

In Section 2, we provide the necessary background to quantum computing and a simple example of applying the algorithm for comparing two equivalent circuits. We state the theorem explicitly and the resulting algorithm in Section 3. In Section 4, we use the Stim simulator to run the algorithm, which outperforms the SAT-based approach by an order of magnitude. We conclude in Section 5.

[1] Verifying equivalence of circuits C_1, C_2 is reducible to checking if the circuit C_1, followed by the inverse of C_2, is equivalent to the identity circuit, i.e. it does not change the inputs

2 Preliminaries

We briefly introduce relevant quantum computing concepts and refer to [35] for a more elaborate introduction.

2.1 Quantum circuits and fundamental concepts

Classical circuits are limited to bits, which take values 0 and 1. In contrast, the state of a quantum bit or qubit can be expressed as a complex-valued 2-vector of unit norm. Examples of single-qubit states are $\begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix}$ and $\begin{bmatrix} \frac{1}{\sqrt{5}} \\ \frac{2i}{\sqrt{5}} \end{bmatrix}$ where i is the imaginary unit ($i^2 = -1$). Two possible quantum states are the computational-basis states $\begin{bmatrix} 1 \\ 0 \end{bmatrix}$ and $\begin{bmatrix} 0 \\ 1 \end{bmatrix}$, usually denoted in Dirac notation as $|0\rangle$ and $|1\rangle$. Thus, we can write an arbitrary single-qubit state $|\phi\rangle = \begin{bmatrix} \alpha_0 \\ \alpha_1 \end{bmatrix} = \alpha_0 |0\rangle + \alpha_1 |1\rangle$ where the complex numbers α_i satisfy $|\alpha_0|^2 + |\alpha_1|^2 = 1$. Here, $|z|$ denotes the modulus of the complex number z: when writing $z = a + b \cdot i$ for real numbers a, b and i the imaginary unit satisfying $i^2 = -1$, the modulus equals $|z| = \sqrt{a^2 + b^2}$ and can also be defined through the complex conjugate $z^* = a - b \cdot i$ as $|z| = \sqrt{z \cdot z^*}$. With this notation, our examples become $\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ and $\frac{1}{\sqrt{5}}(|0\rangle + 2i |1\rangle)$.

Two single-qubit quantum states $|\phi\rangle, |\psi\rangle$ are combined into a two-qubit state $|\phi\rangle \otimes |\psi\rangle$, where $\otimes$ denotes the tensor product (Kronecker product) from linear algebra. In general, an n-qubit state is a complex vector of 2^n entries and can be written in Dirac notation as $\sum_{x \in \{0,1\}^n} \alpha_x |x\rangle$, where $|x\rangle$ are defined as e.g. $|0010\rangle = |0\rangle \otimes |0\rangle \otimes |1\rangle \otimes |0\rangle$. Here, the complex values α_x should satisfy $\sum_{x \in \{0,1\}^n} |\alpha_x|^2 = 1$, i.e. the norm of the vector representing the quantum state equals 1. Examples of two-qubit states are $|00\rangle$ and $\frac{1}{\sqrt{6}}(|00\rangle + i |01\rangle - 2 |11\rangle)$. Any $(n_A + n_B)$-qubit quantum state $|\phi\rangle$ that cannot be written as $|\phi_A\rangle \otimes |\phi_B\rangle$, with $|\phi_A\rangle$ ($|\phi_B\rangle$) a state of n_A (n_B) qubits, is called entangled, e.g. $\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$.

There are two main operations on quantum states in the usual circuit model: quantum gates and quantum measurements. We will only use gates here. A quantum gate on n qubits is represented by a 2^n by 2^n unitary matrix U. A quantum state $|\phi\rangle$ is updated by a unitary matrix as $U \cdot |\phi\rangle$ where $\cdot$ denotes matrix-vector multiplication. As example, consider the following single-qubit gates:

$$\text{Hadamard } H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \qquad \text{Phase gate } S = \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix}.$$

Applying this to the state $|0\rangle$ for example, we obtain

$$\mathrm{H} \cdot |0\rangle = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \frac{|0\rangle + |1\rangle}{\sqrt{2}}$$

and similarly one can compute $S |0\rangle = |0\rangle$.

A quantum gate U is a unitary matrix, which means $U \cdot U^\dagger = U^\dagger \cdot U = \mathbb{1}_{2^n}$, where $\mathbb{1}_{2^n}$ is the identity matrix on vectors of 2^n entries (i.e. the matrix that has the property $\mathbb{1}_{2^n} \cdot \vec{v} = \vec{v}$ for each vector $\vec{v}$ of 2^n complex numbers) and the adjoint operator $(.)^\dagger$ means transposing the matrix and replacing each matrix entry by its complex conjugate. For example, the Hadamard gate and phase gate have adjoint operators

$$H^\dagger = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} = H \qquad S^\dagger = \begin{bmatrix} 1 & 0 \\ 0 & -i \end{bmatrix}.$$

It is not hard to check that indeed $H^\dagger \cdot H = S^\dagger \cdot S = \mathbb{1}_2 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$. Applying an n-qubit gate A to the first part of an $(n+m)$-qubit quantum state $|\phi\rangle$ is done by tensoring with the identity, i.e. $A \otimes \mathbb{1}_{2^m}$ is applied to the entire state $|\phi\rangle$.

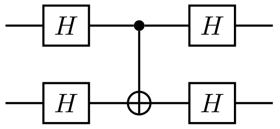
A notion we will use later on is the bra $\langle\phi| = (|\phi\rangle)^\dagger$ and the inner product $\langle\phi|\psi\rangle = \sum_{x\in\{0,1\}^n} a_x^* \cdot b_x$ for $|\phi\rangle = \sum_{x\in\{0,1\}^n} a_x |x\rangle$ and $|\psi\rangle = \sum_{x\in\{0,1\}^n} b_x |x\rangle$.

A quantum circuit is a sequence of gates, applied to an initial quantum state (typically $|0\rangle^{\otimes n}$). A quantum circuit can be better explained by an elementary example (see Example 1 in Section 2.1).

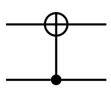
A special but important class of quantum circuits are the Clifford circuits. Any Clifford gate can be written as Clifford circuit[2] consisting only of three elementary Clifford gates: H, S and CNOT (defined above). One of the significant features of Clifford circuits is that they can be simulated in polynomial time in the number of qubits and number of elementary Clifford gates by classical computers, as shown by the Gottesman-Knill theorem [24] (their methods are discussed in section 2.2). In addition, Clifford circuits can generate many interesting entangled states (as demonstrated in the example of section 2.1), and become universal – meaning they can approximate any quantum circuit – if non-Clifford gates are added to the gate set [13].

Example 1. We provide an example for calculating the output states of two simple circuits using standard linear algebra methods.

A)



B)



For both circuits, we will assume that the initial state on each qubit is $|0\rangle$.

– We start by making the calculations for the **A** circuit. For a Hadamard gate applied on the first qubit we get:

$$H|0\rangle = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \cdot 1 + 1 \cdot 0 \\ 1 \cdot 1 + (-1) \cdot 0 \end{bmatrix} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

[2] When we say 'circuit', we mean the sequence of quantum gates, i.e. without the input states $|0\rangle^{\otimes n}$.

The result of applying a Hadamard gate on the second qubit will be the same. Since the two qubits are independent (i.e. we don't use an entangling gate such as CNOT) we can calculate the state of the (whole) system by tensoring the two states:

$$\begin{aligned}&\left(\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)\right) \otimes \left(\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)\right)\\ &= \frac{1}{\sqrt{2}}\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \otimes (|0\rangle + |1\rangle)\\ &= \frac{1}{2}(|0\rangle \otimes |0\rangle + |0\rangle \otimes |1\rangle + |1\rangle \otimes |0\rangle + |1\rangle \otimes |1\rangle).\end{aligned}$$

In matrix notation this would be:

$$\frac{1}{2}\begin{bmatrix}1\\0\\0\\0\end{bmatrix} + \frac{1}{2}\begin{bmatrix}0\\1\\0\\0\end{bmatrix} + \frac{1}{2}\begin{bmatrix}0\\0\\1\\0\end{bmatrix} + \frac{1}{2}\begin{bmatrix}0\\0\\0\\1\end{bmatrix} = \frac{1}{2}\begin{bmatrix}1\\1\\1\\1\end{bmatrix}.$$

In other words the state is $\frac{1}{2}(|00\rangle + |01\rangle + |10\rangle + |11\rangle)$.
Now we will calculate how this state is transformed when we apply a CNOT gate where the first qubit is the control qubit and the second one is the target qubit:

$$CNOT \cdot \begin{bmatrix}1\\1\\1\\1\end{bmatrix} = \begin{bmatrix}1\,0\,0\,0\\0\,1\,0\,0\\0\,0\,0\,1\\0\,0\,1\,0\end{bmatrix} \cdot \frac{1}{2}\begin{bmatrix}1\\1\\1\\1\end{bmatrix} = \frac{1}{2}\begin{bmatrix}1\\1\\1\\1\end{bmatrix}$$

So the state remains the same. A more intuitive way to apply the CNOT is to flip the second input of $|{*}{*}\rangle$ whenever the first input is 1. I.e. $\frac{1}{2}(|00\rangle + |01\rangle + |10\rangle + |11\rangle)$ becomes $\frac{1}{2}(|00\rangle + |01\rangle + |11\rangle + |10\rangle)$ which is identical and therefore the state is not affected. Finally, we apply the two remaining Hadamard gates to the state $\frac{1}{2}(|00\rangle + |01\rangle + |10\rangle + |11\rangle)$.

$$(H \otimes H)\left(\frac{1}{2}\begin{bmatrix}1\\1\\1\\1\end{bmatrix}\right) = \frac{1}{4}\begin{bmatrix}1 & 1 & 1 & 1\\1 & -1 & 1 & -1\\1 & 1 & -1 & -1\\1 & -1 & -1 & 1\end{bmatrix}\begin{bmatrix}1\\1\\1\\1\end{bmatrix} = \begin{bmatrix}1\\0\\0\\0\end{bmatrix} = |00\rangle$$

Therefore, applying Hadamard gates to each qubit of the circuit starting with the initial state $\frac{1}{2}(|00\rangle + |01\rangle + |10\rangle + |11\rangle)$ results to the state $|00\rangle$.
- Now for the much simpler circuit **B**, we start again by both qubits in the state $|0\rangle$. Following the same rules we applied for the CNOT of **B**, the resulting state will be $|00\rangle$.

Both circuits return the same state when we start by $|00\rangle$. This is not unexpected, in fact the circuits were chosen such that they are equivalent up to global phase. Therefore, as long as both circuits start with the same initial states, they will output states that can only differ by a phase (at most).

2.2 Stabilizer states

The Pauli gates are the following set of gates:

$$I = \mathbb{1}_2 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \quad Y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}, \quad Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}.$$

The n-qubit Pauli group $\mathcal{P}_n$ is the set $\{\alpha P \mid \alpha \in \{\pm 1, \pm i\}, P \in \textsc{Pauli}_n\}$ where $\textsc{Pauli}_n$ is the tensor product of n Pauli operators (a "Pauli string"). For example $X \otimes Z \otimes Y \otimes Y \in \textsc{Pauli}_4$. The Pauli group forms a group under matrix multiplication. Any two elements $P_k, P_l \in \mathcal{P}_n$ either commute or anti-commute: Meaning that either $P_k \cdot P_l = P_l \cdot P_k$ or $P_k \cdot P_l = -P_l \cdot P_k$. Finally, we can give an alternative, equivalent definition of the Clifford group in terms of Pauli matrices: the Clifford group is the set of unitary operators that *stabilize* the Pauli group when acting on it by conjugation i.e. all the $2^n \times 2^n$ unitary matrices V such that $VPV^\dagger \in \mathcal{P}_n$ for all $P \in \mathcal{P}_n$.

We will now lay out the stabilizer formalism for efficient classical simulation of Clifford circuits with $|0\rangle^{\otimes n}$ as input state [24,3]. A unitary operator U *stabilizes* a quantum state if $U|\phi\rangle = |\phi\rangle$. The so-called stabilizer states form a strict subset of all quantum states which can be described by maximal commutative subgroups of the Pauli group using n elements of $\mathcal{P}_n$. For example, the state $|0\rangle$ is stabilized by the Pauli group $\{Z, I\}$ while the state $|+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ is stabilized by $\{X, I\}$. If $|\phi\rangle$ and $|\psi\rangle$ are stabilizer states with stabilizer groups G, H, respectively, then $|\phi\rangle \otimes |\psi\rangle$ is also a stabilizer state with stabilizer group $\{g \otimes h \mid g \in G, h \in G\}$. For example, the state $|00\rangle = |0\rangle \otimes |0\rangle$ is stabilized by the group $\{I \otimes I, I \otimes Z, Z \otimes I, Z \otimes Z\}$.

Maximal commutative subgroups of the Pauli group only have a single quantum state they stabilize; thus, we can *represent* any stabilizer state by its stabilizer group, instead of by providing its description as a vector of 2^n complex numbers. The stabilizer group of an n-qubit stabilizer state has 2^n elements, so storing all of those would not yield a succinct description of the state. However, the stabilizer group can be succinctly represented by the generator set of the stabilizer group, which only has n elements $\in \mathcal{P}_n$. Since there are four Pauli gates, we can represent each of these using $\log_2(4) = 2$ bits. Furthermore, one can show that each element of a stabilizer group is of the form $\pm P_1 \otimes \cdots \otimes P_n$ with P_j a Pauli gate; thus, $2n+1$ bits are needed to represent an element of an n-qubit stabilizer group ($2n$ for the Pauli gates in the tensor product and the 1 bit for the prefactor $\pm$). Therefore, by this method, only $n \cdot (2n+1) = 2n^2 + n$ bits are required for the description of a quantum state that can be generated by Clifford circuits while a naive description would require 2^n complex numbers.

Updating the generators of a stabilizer state after an elementary Clifford gate is applied to the corresponding stabilizer state can be done in time $O(n)$, as follows. Suppose that $P = \pm P_1 \otimes \cdots \otimes P_n$ stabilizes an n-qubit state $|\phi\rangle$. Then given an n-qubit gate U, $UPU^\dagger$ stabilizes $U|\phi\rangle$. This is because $UPU^\dagger U|\phi\rangle = UP|\phi\rangle = U|\phi\rangle$. Now if U is a single-qubit operation, we can write $U = U_j$, where we denote $U_j = I \otimes I \cdots \otimes I \otimes U \otimes I \ldots I$ with U at the j-th position in the tensor product. Therefore the application of U to the j-th qubits of $|\phi\rangle$ updates each element of the stabilizer group to $U_j P U_j^\dagger = \pm I P_1 I \otimes \cdots \otimes U_j P_j U_j^\dagger \otimes \ldots I P_n I = \pm P_1 \otimes \cdots \otimes U_j P_j U_j^\dagger \otimes \ldots P_n$. That is, only the j-th entry in the tensor product of P should be updated. This can be done in constant time by a lookup table for each of H, S and each Pauli gate. It can be shown that updating only the n generators of the stabilizer group suffices, so that the update of H or S takes $O(n)$ time. A similar procedure works for the two-qubit gate CNOT, also requiring $O(n)$ time to update the stabilizer generators.

2.3 Circuit Equivalence-Check Problem

We proceed to formally state the main problem. We are presented with two n-qubit Clifford quantum circuits U and V, each represented by a (classical description of) a circuit of only elementary Clifford gates (for example, H, S and CNOT). The aim of the method is to determine whether or not U and V are equivalent.

Definition 1. *Fix the number of qubits $n \geq 1$. Given two n-qubit unitaries U, V, we say that U is* equivalent *to V, denoted $U \simeq V$, if $U = cV$ for some complex number c.*

The factor c is often called 'global phase' and is irrelevant to any observable properties of the two unitaries (for details, see [35]). We remark that if $U = cV$, then c satisfies $|c|^2 = 1$. This follows from the fact that U and V are unitaries: $\mathbb{1} = UU^\dagger = (cV) \cdot (cV)^\dagger = cV \cdot c^* V^\dagger = |c|^2 \cdot VV^\dagger = |c|^2 \cdot \mathbb{1}$, hence $|c|^2 = 1$.

3 Reducing Clifford Circuit Equivalence to Classical Simulation

We explicitly formulate the theorem (the statement appears in [35] 10.5.2) that gives the necessary and sufficient conditions for two gates to be equal. An explicit formulation of the statement together with a formal proof will provide a better understanding of the algorithm.

Theorem 1. *Let U, V be two unitaries on $n \geq 1$ qubits. Then U is equivalent to V if and only if the following conditions hold:*

1. *for all $j \in \{1, 2, \ldots, n\}$, we have $UZ_jU^\dagger = VZ_jV^\dagger$; and*
2. *for all $j \in \{1, 2, \ldots, n\}$, we have $UX_jU^\dagger = VX_jV^\dagger$.*

Proof. If $U \simeq V$, then $U = cV$ for some $c \in \mathbb{C}, |c| = 1$, so $UZ_jU^\dagger = cVZ_j(cV)^\dagger = cVZ_j(V)^\dagger \cdot c^* = |c|^2 VZ_jV^\dagger = VZ_jV^\dagger$ and similarly for X_j where c^* is the complex conjugate of c.

For the converse direction, we first note that if U and V coincide on X_j and Z_j by conjugation, then they must coincide by conjugation on *all Pauli strings.* The reason for this is that any Pauli string can be written as a product of $\{X_j, Z_j\}_{j=1}^n$ modulo a complex number from $\{\pm 1, \pm i\}$. Given such a product

$P = \prod_{k=1}^{n} X_k^{x_k} Z_k^{z_k}$ where $x_k, z_k \in \{0,1\}$ determine if X_k or Z_k is included in the product, we see that $UPU^\dagger = U\left(\prod_{k=1}^{n} X_k^{x_k} Z_k^{z_k}\right)U^\dagger = \prod_{k=1}^{n} UX_k^{x_k}U^\dagger UZ_k^{z_k}U^\dagger$ because $U^\dagger U = \mathbb{1}^{\otimes n}$ (as U is unitary). This shows that $UPU^\dagger = VPV^\dagger$ if $UX_kU^\dagger = VX_kV^\dagger$ and $UZ_kU^\dagger = VZ_kV^\dagger$ for all $k = 1, 2, \ldots, n$.

Given an n-qubit quantum state $|\phi\rangle$, we write $|\phi\rangle\langle\phi|$ in the Pauli basis:

$$|\phi\rangle\langle\phi| = \sum_{P \in \{\mathbb{1}, X, Y, Z\}^{\otimes n}} \alpha_P P$$

where $\alpha_P \in \mathbb{C}$ are unique. Note that the state $|\phi\rangle\langle\phi|$ can be seen as a density operator and any density operator can be generated by a Pauli basis [35,29]. Hence, using the observation that U and V coincide on all Pauli strings by conjugation, we find that $U|\phi\rangle\langle\phi|U^\dagger = \sum_{P\in\{\mathbb{1},X,Y,Z\}^{\otimes n}} \alpha_P UPU^\dagger$, which by the observation above equals $\sum_{P\in\{\mathbb{1},X,Y,Z\}^{\otimes n}} \alpha_P VPV^\dagger = V|\phi\rangle\langle\phi|V^\dagger$, hence $A|\phi\rangle\langle\phi|A^\dagger = |\phi\rangle\langle\phi|$ for $A = V^\dagger U$. By conjugating both sides with $|\phi\rangle$, we obtain $|\langle\phi|A|\phi\rangle|^2 = |\langle\phi|\phi\rangle|^2 = 1$. Thus, the modulus of the inner product between $A|\phi\rangle$ and $|\phi\rangle$ equals the product of their norms (which both equal 1), hence the tightness condition of the Cauchy-Schwarz inequality implies that $A|\phi\rangle$ and $|\phi\rangle$ are linearly dependent. That is, $|\phi\rangle$ is an eigenvector of A.

Since this holds for arbitrary n-qubit states $|\phi\rangle$, each vector is an eigenvector of A. By standard linear algebra, we know that this implies that A is a multiple of the identity operator. Thus $A = c\mathbb{1}$, hence $U = cV$. $\square$

For equivalence checking of Clifford circuits, the theorem induces an algorithm which can be reduced to simulating the Clifford circuit. This is well known to efficient [24,3].

The algorithm:

From Section 2, we know that $S_0 = \{Z_j \mid j = 1, 2, ..., n\}$ generate the stabilizer group of the state $|0\rangle^{\otimes n}$, and thus S_0 "represents" $|0\rangle^{\otimes n}$ in the stabilizer formalism. Similarly for $\{X_j \mid j = 1, 2, \ldots, n\}$ for the state $|+\rangle^{\otimes n}$. Furthermore, in Section 2 we explained that updating a stabilizer state representation $\{g_1, g_2, \ldots, g_n\}$ (i.e. the g_j are generators of the state's stabilizer group), after a Clifford gate U is found as $\{Ug_1U^\dagger, \ldots, Ug_nU^\dagger\}$. Combining these facts with Theorem 1, we obtain the following algorithm for equivalence checking of Clifford circuits by a reduction to Clifford-circuit simulation:

1. simulate U gate-by-gate in the stabilizer formalism, where the stabilizer group generators of the input state are $\{Z_1, Z_2, \ldots, Z_n\}$, i.e. the input state is $|0\rangle^{\otimes n}$. This yields the output stabilizer generator set $\{UZ_1U^\dagger, UZ_2U^\dagger, \ldots, UZ_nU^\dagger\}$
2. do the same for V, yielding $\{VZ_1V^\dagger, VZ_2V^\dagger, \ldots, VZ_nV^\dagger\}$
3. check for each $j = 1, 2, \ldots, n$, whether the Pauli elements $UZ_jU^\dagger$ and $VZ_jV^\dagger$ are equal. If there is some j for which they are non-equal, return "Unequivalent".

4. Repeat steps (1-3) for the input stabilizer generator set $\{X_1, X_2, \dots, X_n\}$, which is produced by starting with the generator set of $|0\rangle^{\otimes n}$, followed by applying the Hadamard gate H on each qubit (since $HZH^\dagger = X$).
5. If the algorithm reaches this point, U and V agree by conjugation on all X_j and Z_j. Return "Equivalent".

Since storing the n stabilizer generators for an n-qubit state requires $O(n^2)$ space and updating them for a single-qubit gate or two-qubit gate takes time $O(n)$, the runtime of the algorithm is $O(n^2 + m \cdot n)$ with n the number of qubits and m the sum of the number of elementary Clifford gates in U and V.

We note that requiring that U and V agree on all X_j and Z_j by conjugation is a stronger statement than requiring that U and V output the same state on input $|0\rangle^{\otimes n}$ and $|+\rangle^{\otimes n}$. As counterexample for $n = 2$, note that $U = \mathbb{1}_2 \otimes \mathbb{1}_2$ and $V = CNOT$ map both states to themselves, while they are not equivalent, as witnessed by $U(\mathbb{1}_2 \otimes Z)U^\dagger = \mathbb{1}_2 \otimes Z \neq Z \otimes Z = V \cdot (\mathbb{1}_2 \otimes Z) \cdot V^\dagger$.

Example 2. Here we provide an example of applying the algorithm for comparing two simple circuits **A** and **B**, the same circuits as in Example 1 of Section 2.1. The algorithm will evaluate both circuits for two distinct initializations of states. In particular, in order to decide if two circuits are equivalent we simulate them using the stabilizer formalism for both the initial states $|0\rangle$ and $\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$. More precisely, we need to simulate the circuits gate-by-gate using the update rules of the stabilizers formalism.

If M_i stands for the operator that corresponds to the i^{th} qubit, the algorithm (Section 3) decides that two circuits are equivalent when:

1. Applying the sequence of gates of the first circuit initializing by Z_i on every qubit i returns the same stabilizers as for the gates of the second circuit, and
2. Applying the sequence of gates of the first circuit initializing by X_i on every qubit i returns the same stabilizers as for the gates of the second circuit.

If the stabilizers agree then we deduce that the circuits are equivalent, otherwise they are different.

In the calculations below, the "$\mapsto$" indicates that some transformation is taking place (i.e. applying some gate), while we use equality when we rewrite by the appropriate stabilizers states. Applying a single qubit gate U to the state P_i of the i^{th} qubit corresponds to the operation $U_i P_i U_i^\dagger$, where the indices indicate the qubit to which each operator acts. For the case of $CNOT$ gates, which are 2 qubit operators, we will simplify by omitting the indices. That is because we can keep track of the qubits they act to by consulting the circuit diagrams **A** and **B**.

– For the case of $|0\rangle^{\otimes n}$ we update the states **A** as follows:

$$\begin{aligned}
\{Z_1, Z_2\} &\mapsto \{H_1\ Z_1\ H_1^\dagger, H_2\ Z_2\ H_2^\dagger\} \\
&= \{X_1, X_2\} \\
&\mapsto \{CNOT\ X_1\ CNOT^\dagger, CNOT^\dagger\ X_2\ CNOT^\dagger\} \\
&= \{X_1\ X_2, I_1\ X_2\} \\
&\mapsto \{H_1\ X_1\ H_1^\dagger\ H_2\ X_2\ H_2^\dagger, H_1\ I_1\ H_1^\dagger H_2\ X_2\ H_2^\dagger\} \\
&= \{Z_1\ Z_2, I_1\ Z_2\} \\
&= \{I_1\ Z_2, Z_1\ Z_2\}
\end{aligned}$$

– For the case of $|0\rangle^{\otimes n}$ we update the states of $\mathbf{B}$ as follows:

$$\{Z_1, Z_2\} \mapsto \{CNOT\ Z_1\ CNOT^\dagger, CNOT\ Z_2\ CNOT^\dagger\} = \{I_1\ Z_2, Z_1\ Z_2\}$$

Therefore the two circuits agree on the $|0\rangle^{\otimes n}$ case. Analogously, by performing the updates of the states but starting with states X_i (instead of Z_i) we can observe that the resulting generators agree there too. Therefore we come to the conclusion that the two circuits are equivalent.

4 Experiments

We implemented the algorithm from Section 3 in Python using the open-source Stim Clifford-circuit simulator [21] as a simulator backend. See [2] for our open-source implementation.

We empirically evaluated the implementation and compared the runtime to QuSAT, a recent SAT-based Clifford equivalence checker [9]. Across all instances in which QuSat successfully concluded its computations, the outcomes aligned with those obtained by the method we use. Thereby signifying a consistent classification of the tested circuit pairs as either equivalent or non-equivalent by both approaches. We have used a laptop with a 3.2 GHz M1 processor with 8Gb RAM. For making a fair comparison we generate random circuits using QuSAT [1], which consists of generating random sequences of elementary Clifford gates $H, S,$ CNOT. QuSAT generates circuits which are completely 'filled' in the sense that if the depth is d, the number of gates applied to each qubit is also d; thus, since only $H, S,$ CNOT are used, the number of gates in a depth-d circuit is between $d \cdot \frac{n}{2}$ (only two-qubit gates) and $d \cdot n$ (only single-qubit gates). We emphasize that the runtime of this work's method is deterministic and a function of the number of qubits and the number of gates only, and is hence independent of the internal structure of the two input circuits.

Next, we ran both QuSAT and our implementation on both equivalent and non-equivalent random Clifford circuits which were thus produced by QuSAT.

The results are shown in Figure 1, for varying number of qubits (Figure 1a, Figure 1b and Figure 1c) and varying quantum-circuit depth (Figure 1d). Our experiments for equivalence checking of non-equivalent random circuits show very similar running times as for equivalent random circuits (not displayed). Finally, we have tested the method for circuits of 1000 qubits and depth 10000 for identical and non-equivalent circuits, yielding runtimes of 21.72 and 22.80 seconds, respectively (not displayed).

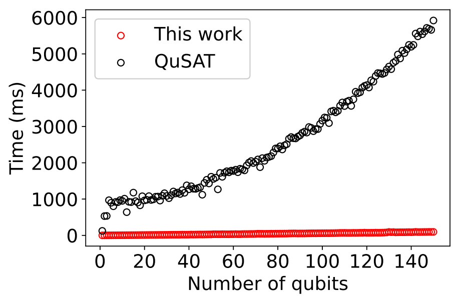
We observe that the implementation is very fast and can handle large circuits: up to 1000 qubits with depth of 10.000 gates in ∼22 seconds, and 100.000 qubits with 10 gates in ∼15 minutes (Figure 1b). The tested regime of the method consistently outperforms QuSAT by one to two orders of magnitude (10× to 100×) or even more. We also see that the runtime of QuSAT, whose runtime is probabilistic, scales exponentially in the number of qubits whereas the runtime of our approach is deterministic and we know it has polynomial scaling in both number of qubits and number of gates (Section 3).

Finally, we would like to note that there is yet another tool with polynomial scaling to the number of qubits and gates, this tool is named "feynman" [5]. In particular, the submodule that performs equivalence checking of circuits is named "feynver". Feynver is based on Feynman sums and it can verify the equivalence of general quantum circuits. We will not make an detailed comparison to this tool as it is not equivalent to ours: It can reliably verify that two circuits are equivalent but it can only conclude that two circuits are different for special classes of circuits. Our experiments showed that Feynver is slower than both QuSAT and our method. For example, the next table shows the running time of all three tools for a random circuit of 50 qubits and depth=50. In particular this circuit had 11092 gates:

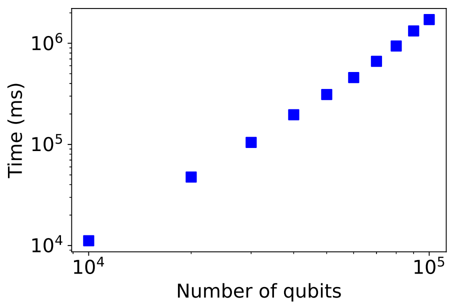
Circuit	**This work**	**QuSAT**	**Feynver**
$\boldsymbol{q = 50, d = 50}$	2 sec	120 sec	3856 sec

5 Conclusions

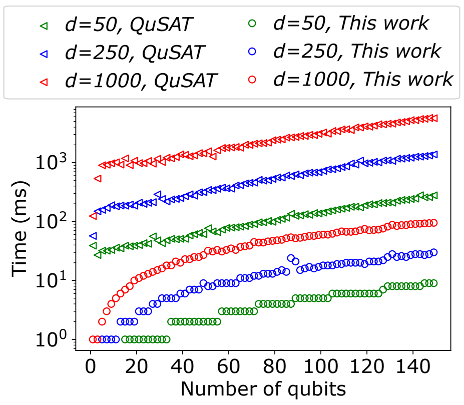
In this paper, we demonstrate that a deterministic algorithm, which is based on a folklore mathematical result, can surpass the efficiency of current methods for exact equivalence checking of quantum circuits consisting of Clifford gates. The algorithm reduces equivalence checking to classical simulation of Clifford circuits and runs in time $O(n^2 + n \cdot m)$, with n the number of qubits and m the total number of elementary Clifford gates of the two input circuits. We have implemented our algorithm using the Stim simulator. We tested our method on a variety of benchmark circuits with different sizes and depths, and compared it to QuSAT. Our results, reaching up to 1000 qubits (with depth 10.000) in less than a minute and 100.000 qubits (depth 10) in ∼15 minutes, demonstrate that this approach consistently outperforms the existing SAT-based-approach QuSAT. Furthermore, since the method is deterministic, its scaling behavior is known. Possible future work includes extending this method to arbitrary circuits



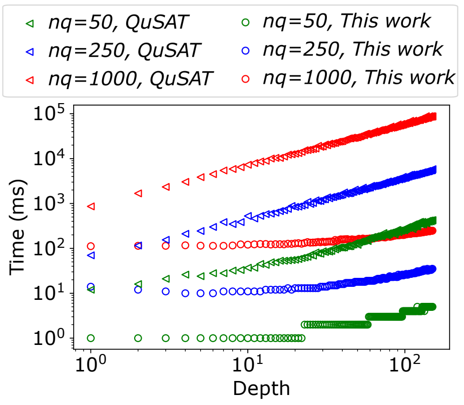
(a) Fixed depth of 1000. This data-set reappears in Figure 1c.



(b) Our approach reaching beyond what was previously feasible: Fixed depth 10 and number of qubits up to 100.000. Both axes are in logarithmic scale, so that a polynomial scaling shows up as a straight line.



(c) Fixed depths (“d”) and increasing number of qubits. The vertical axis is on logarithmic scale.



(d) Fixed number of qubits (“nq”) and increasing depth. Both axes are on logarithmic scale, so that a polynomial scaling shows up as a straight line .

Fig. 1: Runtime of circuit-equivalence checking between identical randomly-generated Clifford circuits for various circuit depths and number of qubits. The step pattern observed for lower values is a result of the limitations of the time-measuring function which operates in milliseconds. The runtimes for non-equivalent circuits (not displayed) are very similar.

using non-Clifford gates [7,5], following existing classical simulation formalisms of such circuits [12].

6 Acknowledgments

We thank Prof. M. Bonsangue for helpful discussions. This work was supported by the Dutch National Growth Fund, as part of the Quantum Delta NL program.

References

1. MQT QuSAT - a tool for utilizing sat in quantum computing. `https://github.com/cda-tum/qusat` (2022)
2. CCEC (Clifford-circuit equivalence checking). `https://github.com/System-Verification-Lab/CCEC` (2023)
3. Aaronson, S., Gottesman, D.: Improved simulation of stabilizer circuits. Physical Review A 70(5) (nov 2004), `https://doi.org/10.1103%2Fphysreva.70.052328`
4. Adleman, L.M., Demarrais, J., Huang, M.D.A.: Quantum computability. SIAM Journal on Computing 26(5), 1524–1540 (1997)
5. Amy, M.: Towards large-scale functional verification of universal quantum circuits. arXiv:1805.06908 (2018)
6. Amy, M.: Formal methods in quantum circuit design (PhD thesis) (2019)
7. Arunachalam, S., Bravyi, S., Nirkhe, C., O'Gorman, B.: The parameterized complexity of quantum verification. arXiv:2202.08119 (2022)
8. Bauer-Marquart, F., Leue, S., Schilling, C.: symQV: Automated symbolic verification of quantum programs. In: Formal Methods: 25th International Symposium, FM 2023, Lübeck, Germany, March 6–10, 2023, Proceedings. pp. 181–198. Springer (2023)
9. Berent, L., Burgholzer, L., Wille, R.: Towards a SAT encoding for quantum circuits: A journey from classical circuits to Clifford circuits and beyond. arXiv:2203.00698 (2022)
10. Bookatz, A.D.: QMA-complete problems. arXiv:1212.6312 (2012)
11. Brakerski, Z., Sharma, D., Weissenberg, G.: Unitary subgroup testing. arXiv:2104.03591 (2021)
12. Bravyi, S., Browne, D., Calpin, P., Campbell, E., Gosset, D., Howard, M.: Simulation of quantum circuits by low-rank stabilizer decompositions. Quantum 3, 181 (Sep 2019), `https://doi.org/10.22331/q-2019-09-02-181`
13. Bravyi, S., Kitaev, A.: Universal quantum computation with ideal Clifford gates and noisy ancillas. Phys. Rev. A 71, 022316 (Feb 2005), `https://link.aps.org/doi/10.1103/PhysRevA.71.022316`
14. Burgholzer, L., Kueng, R., Wille, R.: Random stimuli generation for the verification of quantum circuits. In: Proceedings of the 26th Asia and South Pacific Design Automation Conference. pp. 767–772 (2021)
15. Burgholzer, L., Wille, R.: Advanced equivalence checking for quantum circuits. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 40(9), 1810–1824 (2020)
16. Burgholzer, L., Wille, R.: Improved DD-based equivalence checking of quantum circuits. In: 2020 25th Asia and South Pacific Design Automation Conference (ASP-DAC). pp. 127–132 (2020)

17. Chen, T.F., Jiang, J.H.R., Hsieh, M.H.: Partial equivalence checking of quantum circuits. In: 2022 IEEE International Conference on Quantum Computing and Engineering (QCE). pp. 594–604. IEEE (2022)
18. Córcoles, A.D., Kandala, A., Javadi-Abhari, A., McClure, D.T., Cross, A.W., Temme, K., Nation, P.D., Steffen, M., Gambetta, J.M.: Challenges and opportunities of near-term quantum computing systems. arXiv:1910.02894 (2019)
19. Duncan, R., Kissinger, A., Perdrix, S., van de Wetering, J.: Graph-theoretic Simplification of Quantum Circuits with the ZX-calculus. Quantum 4, 279 (Jun 2020), `https://doi.org/10.22331/q-2020-06-04-279`
20. Finigan, W., Cubeddu, M., Lively, T., Flick, J., Narang, P.: Qubit allocation for noisy intermediate-scale quantum computers. arXiv:1810.08291 (2018)
21. Gidney, C.: Stim: a fast stabilizer circuit simulator. Quantum 5, 497 (Jul 2021), `https://doi.org/10.22331/q-2021-07-06-497`
22. Goldberg, E., Novikov, Y.: How good can a resolution based SAT-solver be? In: Giunchiglia, E., Tacchella, A. (eds.) Theory and Applications of Satisfiability Testing. pp. 37–52. Springer Berlin Heidelberg, Berlin, Heidelberg (2004)
23. Gottesman, D.: Stabilizer codes and quantum error correction. arXiv:quant-ph/9705052 (1997)
24. Gottesman, D.: The Heisenberg representation of quantum computers. arXiv:quant-ph/9807006v1 (1998)
25. Hein, M., Dür, W., Eisert, J., Raussendorf, R., Nest, M., Briegel, H.J.: Entanglement in graph states and its applications. arXiv:0602096 (2006)
26. Hong, X., Ying, M., Feng, Y., Zhou, X., Li, S.: Approximate equivalence checking of noisy quantum circuits. In: 2021 58th ACM/IEEE Design Automation Conference (DAC). pp. 637–642 (2021)
27. Janzing, D., Wocjan, P., Beth, T.: " non-identity-check" is QMA-complete. International Journal of Quantum Information 3(03), 463–473 (2005)
28. Ji, Z., Wu, X.: Non-identity check remains QMA-complete for short circuits. arXiv:0906.5416 (2009)
29. Kimura, G.: The bloch vector for n-level systems. Physics Letters A 314(5), 339–349 (2003)
30. Kuehlmann, A.: Dynamic transition relation simplification for bounded property checking. In: IEEE/ACM International Conference on Computer Aided Design, 2004. ICCAD-2004. pp. 50–57 (2004)
31. Kuehlmann, A., Paruthi, V., Krohm, F., Ganai, M.: Robust boolean reasoning for equivalence checking and functional property verification. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 21(12), 1377–1394 (2002)
32. Linden, N., de Wolf, R.: Lightweight detection of a small number of large errors in a quantum circuit. Quantum 5, 436 (2021)
33. Montanaro, A.: Quantum algorithms: an overview. npj Quantum Information 2(1), 15023 (Jan 2016), `https://doi.org/10.1038/npjqi.2015.23`
34. Montanaro, A., de Wolf, R.: A survey of quantum property testing. arXiv:1310.2035 (2013)
35. Nielsen, M.A., Chuang, I.L.: Quantum information and quantum computation. Cambridge: Cambridge University Press 2(8), 23 (2000)
36. Niemann, P., Wille, R., Drechsler, R.: Equivalence checking in multi-level quantum systems. In: Reversible Computation: 6th International Conference, RC 2014, Kyoto, Japan, July 10-11, 2014. Proceedings 6. pp. 201–215. Springer (2014)

37. Peham, T., Burgholzer, L., Wille, R.: Equivalence checking of quantum circuits with the ZX-calculus. IEEE Journal on Emerging and Selected Topics in Circuits and Systems 12(3), 662–675 (2022)
38. Preskill, J.: Quantum Computing in the NISQ era and beyond. Quantum 2, 79 (Aug 2018), https://doi.org/10.22331/q-2018-08-06-79
39. Tanaka, Y.: Exact non-identity check is nqp-complete. International Journal of Quantum Information 8(05), 807–819 (2010)
40. Terhal, B.M.: Quantum error correction for quantum memories. Rev. Mod. Phys. 87, 307–346 (Apr 2015), https://link.aps.org/doi/10.1103/RevModPhys.87.307
41. Viamontes, G.F., Markov, I.L., Hayes, J.P.: Checking equivalence of quantum circuits and states. In: 2007 IEEE/ACM International Conference on Computer-Aided Design. pp. 69–74 (2007)
42. Vinkhuijzen, L., Grurl, T., Hillmich, S., Brand, S., Wille, R., Laarman, A.: Efficient implementation of LIMDDs for quantum circuit simulation. In: International Symposium on Model Checking of Software (SPIN) (2023)
43. Wang, S.A., Lu, C.Y., Tsai, I.M., Kuo, S.Y.: An XQDD-based verification method for quantum circuits. IEICE transactions on fundamentals of electronics, communications and computer sciences 91(2), 584–594 (2008)
44. Wei, C.Y., Tsai, Y.H., Jhang, C.S., Jiang, J.H.R.: Accurate BDD-based unitary operator manipulation for scalable and robust quantum circuit verification. In: Proceedings of the 59th ACM/IEEE Design Automation Conference. pp. 523–528 (2022)
45. van de Wetering, J.: ZX-calculus for the working quantum computer scientist (2020)
46. Wille, R., Przigoda, N., Drechsler, R.: A compact and efficient sat encoding for quantum circuits. In: 2013 Africon. pp. 1–6. IEEE (2013)
47. Yamashita, S., Markov, I.L.: Fast equivalence-checking for quantum circuits. In: 2010 IEEE/ACM International Symposium on Nanoscale Architectures. pp. 23–28. IEEE (2010)
48. Yamashita, S., Minato, S.i., Miller, D.M.: DDMF: An efficient decision diagram structure for design verification of quantum circuits under a practical restriction. IEICE transactions on fundamentals of electronics, communications and computer sciences 91(12), 3793–3802 (2008)