Towards a SAT Encoding for Quantum Circuits

A Journey From Classical Circuits to Clifford Circuits and Beyond

Lucas Berent
Technical University of Munich, Germany

Lukas Burgholzer
Johannes Kepler University Linz, Austria

Robert Wille
Technical University of Munich, Germany
Software Competence Center Hagenberg GmbH (SCCH), Austria

Abstract

Boolean Satisfiability (SAT) techniques are well-established in classical computing where they are used to solve a broad variety of problems, e.g., in the design of classical circuits and systems. Analogous to the classical realm, quantum algorithms are usually modelled as circuits and similar design tasks need to be tackled. Thus, it is natural to pose the question whether these design tasks in the quantum realm can also be approached using SAT techniques. To the best of our knowledge, no SAT formulation for arbitrary quantum circuits exists and it is unknown whether such an approach is feasible at all. In this work, we define a propositional SAT encoding that, in principle, can be applied to arbitrary quantum circuits. However, we show that, due to the inherent complexity of representing quantum states, constructing such an encoding is not feasible in general. Therefore, we establish general criteria for determining the feasibility of the proposed encoding and identify classes of quantum circuits fulfilling these criteria. We explicitly demonstrate how the proposed encoding can be applied to the class of Clifford circuits as a representative. Finally, we empirically demonstrate the applicability and efficiency of the proposed encoding for Clifford circuits. With these results, we lay the foundation for continuing the ongoing success of SAT in classical circuit and systems design for quantum circuits.

2012 ACM Subject Classification Hardware → Electronic design automation; Hardware → Quantum computation

Keywords and phrases Satisfiability, Quantum Computing, Design Automation, Clifford Circuits

Digital Object Identifier 10.4230/LIPIcs.SAT.2022.18

Supplementary Material *Source Code:* https://github.com/cda-tum/qusat

Funding This work received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation program (grant agreement No. 101001318), was part of the Munich Quantum Valley, which is supported by the Bavarian state government with funds from the Hightech Agenda Bayern Plus, and has been supported by the BMK, BMDW, and the State of Upper Austria in the frame of the COMET program (managed by the FFG).

1 Introduction

Quantum computing [32] has recently gained far-reaching interest in academia and industry due to the potential advantage in efficiency compared to classical computers for many practically relevant problems. There are several important problems in computer science and related fields such as physics, chemistry, and mathematics, for which it is known that quantum algorithms offer significant improvements over classical algorithms [9, 22, 24, 35, 37]. In order to efficiently and correctly realize such quantum algorithms on actual quantum computers, a multitude of circuit design tasks needs to be addressed accordingly. Central

25th International Conference on Theory and Applications of Satisfiability Testing (SAT 2022).
Editors: Kuldeep S. Meel and Ofer Strichman; Article No. 18; pp. 18:1–18:17
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

problems in circuit design are, amongst others, compilation [2, 23, 38], synthesis [16, 20, 34], technology mapping [29, 40, 42], simulation [10, 26, 50], and verification [11, 41, 48]. Many of these problems have high worst-case complexity—some have even been proven to be NP-complete [6] or QMA-complete[1] [25]. Hence, efficient methods to tackle practically relevant instances are needed.

In the classical realm, solvers for *Boolean Satisfiability* (SAT, [3]) are one of the key means to efficiently solve design tasks for the realization of classical circuits [4, 7, 17, 19, 27, 28, 43]. All of these SAT-based approaches rely on symbolically encoding the functionality of a given (classical) circuit into a propositional formula which (enriched by further constraints encoding the design objective) is passed to a SAT solver. Because of the remarkable improvements of SAT solvers in the past decades, modern solvers are able to efficiently reason over large formulas and therefore compute the desired design solutions.

Having this power of efficient logical reasoning at hand, it is natural to wonder whether the prospects of SAT-based solutions can also be materialized for the complex design tasks outlined above for quantum circuits. However, while encoding a classical circuit is rather straight-forward (each signal is represented by a propositional variable; gates are symbolically encoded through corresponding propositional formulas), quantum circuits rely on so-called qubits that do not only assume discrete values 0 and 1, but also *superpositions* (i.e., complex-valued linear combinations) of both—creating an infinitely large state space even for a single qubit. Quantum-mechanical phenomena such as entanglement [32] further complicate the representation of states. This raises the question whether a generalization of SAT encodings for quantum circuits is possible and, furthermore, if there exist SAT-based approaches similar to those that have become one of the most established techniques in the classical realm[2].

In this work, we tackle these questions. More precisely, we

- provide a problem analysis and discussion on the limitations of (straight-forward) adaptations of classical SAT techniques to encode the functionality of quantum circuits,
- propose a new *generalized encoding* that can be used to encode arbitrary quantum circuits and show that this increased capability comes at a certain price due to the sheer complexity of representing quantum states,
- identify classes of quantum circuits for which the proposed generalized encoding can be constructed efficiently, and
- provide an empirical analysis on the scalability of the proposed satisfiability encoding and demonstrate its feasibility through experimental evaluations based on a proof-of-concept implementation.

With this work, we lay the foundation for further research towards leveraging powerful classical SAT techniques for quantum computing. In contrast to the classical realm and due to the inherent complexity of quantum states and operations, our work indicates that dedicated classes of quantum circuits will need to be considered in order to formulate efficient and scalable SAT encodings for design tasks involving quantum circuits.

The remainder of this work is organized as follows: Section 2 introduces the necessary background to keep this work self-contained. Then, Section 3 reviews how a SAT encoding

[1] The complexity class QMA is the quantum analogue to the classical complexity class NP [5].

[2] In the domain of quantum computing, first SAT-based approaches tackling a particular combinatorial problem have successfully been proposed (e.g., in [40, 42]). To the best of our knowledge, no complete SAT encoding for the functionality of quantum circuits exists. Current techniques are either limited to reversible circuits [46, 48] (an important subclass of quantum circuits) or quantum circuits prohibiting entanglement [45] (one of the core traits of quantum computing).

for classical circuits is derived and how it can be adapted to (certain) quantum circuits. Based on that, Section 4 describes a generalized SAT encoding and discusses its limitations. Afterwards, Section 5 shows how to overcome these limitations for certain classes of quantum circuits. Section 6 summarizes our empirical analysis, before Section 7 concludes the paper.

2 Background

In this section, we briefly review the main concepts of quantum computing needed throughout the rest of this work. While the individual descriptions are kept brief, we refer the interested reader to [32] for a detailed introduction.

In classical computing, the fundamental unit of information is a bit, which can assume any of the Boolean values 0 or 1. The analogue in the quantum realm is called quantum bit (or *qubit*), which cannot only assume any of the computational basis states $|0\rangle$ or $|1\rangle$ but also arbitrary complex-valued linear combinations (*superposition*) of these states. More specifically, the state $|\varphi\rangle$ of a single qubit is described as $|\varphi\rangle = \alpha_0 |0\rangle + \alpha_1 |1\rangle$ with $\alpha_0, \alpha_1 \in \mathbb{C}$ and $|\alpha_0|^2 + |\alpha_1|^2 = 1$. The complex-valued factors α_i are called *amplitudes* and it is convenient to represent the state of a quantum system by a vector of amplitudes (the *state vector*), i.e., $|\varphi\rangle \equiv \begin{bmatrix}\alpha_0 & \alpha_1\end{bmatrix}^\top$. The postulates of quantum mechanics state that the state vector cannot be observed directly. Instead, *measuring* a qubit collapses its state to one of the (classical) basis states $|i\rangle$—each with probability $|\alpha_i|^2$.

▶ **Example 1.** *Consider the plus and minus states $|+\rangle$ and $|-\rangle$ which are represented by the state vectors $1/\sqrt{2}\begin{bmatrix}1 & 1\end{bmatrix}^\top$ and $1/\sqrt{2}\begin{bmatrix}1 & -1\end{bmatrix}^\top$, respectively. Both states describe an equal superposition of the computational basis states. Measuring them yields either $|0\rangle$ or $|1\rangle$—each with a probability of $|\pm 1/\sqrt{2}|^2 = 0.5$.*

The basis states of an n-qubit system are formed by the tensor product (denoted $\otimes$ in the following) of single-qubit states, i.e., $|i_{n-1}\rangle \otimes \cdots \otimes |i_0\rangle \equiv |i_{n-1} \ldots i_0\rangle \equiv |\sum_{j=0}^{n-1} 2^j i_j\rangle \equiv |i\rangle$ with $i_{n-1}, \ldots, i_0 \in \{0,1\}$ and $i \in \{0, \ldots, 2^n - 1\}$. Any n-qubit state $|\varphi\rangle$ is then described as an arbitrary superposition of these basis states, i.e., $|\varphi\rangle = \sum_{i=0}^{2^n-1} \alpha_i |i\rangle$ with $\alpha_i \in \mathbb{C}$ and $\sum_{i=0}^{2^n-1} |\alpha_i|^2 = 1$. Again, this is conveniently represented by the state vector, i.e., $|\varphi\rangle \equiv \begin{bmatrix}\alpha_{0\ldots 0} & \ldots & \alpha_{1\ldots 1}\end{bmatrix}^\top$. One of the most fundamental differences of quantum states to classical states is that the individual qubits of a system can be *entangled*, i.e., their state can no longer be considered separately (as e.g., for computational basis states), but has to be considered as a whole.

▶ **Example 2.** *The four Bell states $|\Phi^\pm\rangle = 1/\sqrt{2}(|00\rangle \pm |11\rangle)$ and $|\Psi^\pm\rangle = 1/\sqrt{2}(|01\rangle \pm |10\rangle)$ are one of the most prominent examples of entangled quantum states. Consider, for example, the $|\Phi^+\rangle$ state and assume that the first of its qubits is measured. The measurement collapses the state and leaves the system either in the state $|00\rangle$ or $|11\rangle$—each with a probability of $|\pm 1/\sqrt{2}|^2 = 0.5$. Consequently, the state of the second qubit is completely determined by the measurement result of the first qubit—without ever being "touched".*

Similarly to how classical operations and logic gates are applied to the bits of a classical system, *quantum operations* or *quantum gates* can be used to change the state of a quantum system. To this end, a quantum operation acting on k qubits is described by a complex-valued unitary[3] matrix $U \in \mathbb{C}^{2^k \times 2^k}$.

[3] A matrix $U \in \mathbb{C}^{2^k \times 2^k}$ is *unitary* if $U^\dagger U = UU^\dagger = I$, where $U^\dagger$ is the complex-conjugate of U and I denotes the identity matrix.

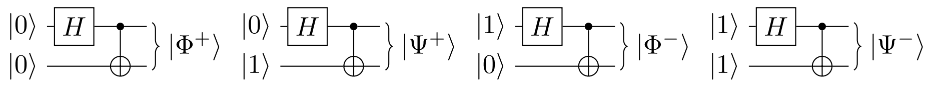


Figure 1 Generation of Bell states

▶ **Example 3.** *One of the most fundamental single-qubit quantum operations are the Pauli gates X, Y, and Z, the phase gate S, as well as the Hadamard gate H, which are described by*

$$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix},\ Y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix},\ Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix},\ S = \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix},\ H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}.$$

Some useful identities are $Z = SS$, $X = HZH$, and $Y = iXZ$. An important example of a two-qubit operation is the controlled-NOT or CNOT operation, which flips the state of a designated target qubit if the designated control qubit is in state $|1\rangle$. We write $CNOT_{c,t}$ for a CNOT gate controlled on qubit q_c and targeted at qubit q_t. The corresponding 4×4 matrix is given by $diag(I, X)$[4]*.*

Applying a quantum operation to the quantum state $|\varphi\rangle$ corresponds to the matrix-vector product of the respective matrix U with the state vector representing $|\varphi\rangle$—yielding a new quantum state $|\varphi'\rangle$[5]. A *quantum circuit* corresponds to a sequence of quantum gates or operations that are applied to a certain state.

▶ **Example 4.** *Similarly to the classical case, quantum circuits can be illustrated as diagrams where horizontal lines correspond to qubits and gates on the lines correspond to operations acting on the qubits. Figure 1 illustrates how the four Bell states from Example 2 can be generated by starting with any two-qubit computational basis state $|ij\rangle \in \{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}$ and applying a single-qubit Hadamard operation (illustrated as a box labelled H) to one of the qubits, followed by a CNOT operation controlled on the same qubit (with $\bullet$ and $\oplus$ indicating the control and the target qubit, respectively).*

3 From Encoding Classical Circuits to Quantum Circuits

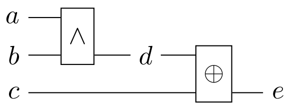
In this section, we briefly review how classical circuits and corresponding problems are usually encoded as SAT instances and how these techniques can directly be translated to quantum circuits. Based on that, we discuss the resulting implications and limitations that form the motivation for this work.

3.1 Classical Circuits

In order to obtain a SAT encoding symbolically representing the functionality of a classical circuit, each signal s of the circuit (representing inputs, intermediate signals, and outputs of the circuit) is represented by a corresponding Boolean variable x^s. Then, for each gate g (representing a Boolean operator) of the circuit, a corresponding set of functional constraints relating the input and the output signals of g is introduced. By further adding

[4] We use $diag(I, X)$ to denote the block matrix $\begin{bmatrix} I & 0 \\ 0 & X \end{bmatrix}$

[5] Technically, for the multiplication to make sense, any operation acting only on $k < n$ qubits must be extended to the full system size by forming appropriate tensor products with identity matrices before performing the multiplication.



(a) Classical Circuit

Signal Representation	Constraints
x^s for $s \in \{a, b, c, d, e\}$	$x^d = x^a \wedge x^b$ $x^e = x^d \oplus x^c$

(b) SAT encoding

Figure 2 Classical circuit and corresponding SAT encoding

constraints to the original SAT formulation of the circuit, e.g., miter structures [7] for verification or formulations for justifying and propagating faults in *Automatic Test Pattern Generation* (ATPG, [17, 19, 28, 43]), a multitude of circuit design tasks can be formulated as SAT instances. Moreover, SAT instances of circuits are frequently used to compute counterexamples, i.e., to show that certain unwanted assignments to the signal variables may occur during the computation of the circuit.

▶ **Example 5.** *Consider the classical circuit shown in Figure 2a. It consists of two gates, three primary inputs, and a single primary output. In order to encode this circuit, the signals a, b, c, d, e are represented as Boolean variables x^a, x^b, x^c, x^d, x^e, respectively. For each logical gate, a corresponding functional constraint representing its functionality is added, as shown in Figure 2b.*

The main question now is: How can this encoding technique be adapted to the domain of quantum computing?

3.2 Quantum Circuits

While each signal in a classical circuit can only assume the value 0 or 1, the state of a qubit is generally described as an arbitrary, complex-valued superposition of states, i.e., $|\varphi\rangle = \alpha_0 |0\rangle + \alpha_1 |1\rangle$. As such, one faces the rather grim perspective of an infinitely-valued logic for even encoding the state of a single qubit. Consequently, encoding the state vector $|\varphi\rangle$ of a qubit in a quantum circuit (which is the analogue to the signal s in a classical circuit) in a straight-forward fashion is infeasible in general.

However, in practice the number of unique quantum states throughout a quantum computation can be upper bounded since in most cases, the input of a quantum circuit is chosen from a fixed set of states, e.g., computational basis states $|i\rangle \in \{|0\dots0\rangle, \dots, |1\dots1\rangle\}$. In fact, most quantum algorithms assume the initial state to be the all-zero state $|0\dots0\rangle$[6]. Assuming that a computation may start out in any of v values, the number of different states that are produced by the gates of a quantum circuit $G = g_0, \dots, g_{|G|-1}$ is upper bounded by $2^{|G|} \cdot v$. This fact follows from the simple observation that every gate of the circuit can at most double the number of states (by transforming a state $|\varphi\rangle$ to a new state $|\varphi'\rangle$).

Thus, a structural analysis of the complete circuit allows to determine the set of unique quantum states that might occur in a quantum circuit and, accordingly, encode them using a multi-valued logic (in contrast to the binary encoding for classical circuits, as shown in Example 5). This has already been recognized in [45], showing that, whenever the individual qubits can be considered separately, quantum circuits can be treated similarly to classical circuits. An example illustrates the idea:

[6] This is a reasonable assumption, since any initial quantum state can be generated from the all-zero state $|0\dots0\rangle$ by prepending the actual quantum circuit with a state preparation circuit, e.g., the $|+\rangle$ state can be generated from the $|0\rangle$ state by applying a Hadamard gate.

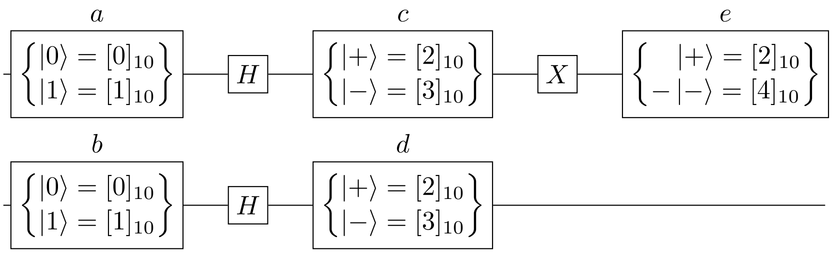

a
$\{|0\rangle = [0]_{10}, |1\rangle = [1]_{10}\}$
H
c
$\{|+\rangle = [2]_{10}, |-\rangle = [3]_{10}\}$
X
e
$\{|+\rangle = [2]_{10}, -|-\rangle = [4]_{10}\}$
b
$\{|0\rangle = [0]_{10}, |1\rangle = [1]_{10}\}$
H
d
$\{|+\rangle = [2]_{10}, |-\rangle = [3]_{10}\}$

Figure 3 Structural analysis of a simple quantum circuit

▶ **Example 6.** *Consider the simple two-qubit quantum circuit composed of three gates shown in Figure 3 and assume that both qubits may either assume the input value $|0\rangle$ or $|1\rangle$. Since both qubits do not interact over the course of the circuit, the circuit can be split into five signals—labelled a to e—in analogy to the classical circuit from Example 5. It is easy to compute that*

$$H\,|0\rangle = |+\rangle\,, \quad H\,|1\rangle = |-\rangle\,, \quad X\,|+\rangle = |+\rangle\,, \textit{ and } X\,|-\rangle = -\,|-\rangle\,. \tag{1}$$

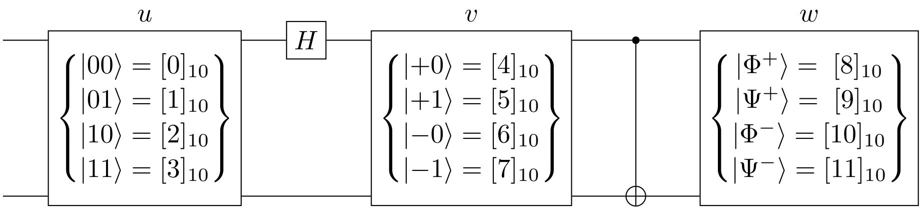
This simple analysis reveals that there are only five unique states for this particular arrangement: $|0\rangle$, $|1\rangle$, $|+\rangle$, $|-\rangle$, and $-\,|-\rangle$. As a consequence, each of the quantum circuit's signals s can be encoded using $\lceil log_2(5)\rceil = 3$ Boolean variables $x_2^s x_1^s x_0^s$. In a similar fashion as in Example 5, the circuit's signals can be related via functional constraints based on Equation 1.

3.3 Discussion

In general, the feasibility of such an encoding for quantum circuits stands and falls with the feasibility of a structural analysis, i.e., the viability of representing and evolving the state of the quantum system. In Example 6, this is feasible since the circuit only involves individual qubits that do not interact. This kind of analysis extends to the case, where the state of the entire system can always be described as a product state. A *product state* is a quantum state where the state of the whole system is described as the product of the states of the individual qubits, i.e., $|\varphi\rangle = |\varphi_{n-1}\rangle \otimes \cdots \otimes |\varphi_0\rangle$ with each $|\varphi_i\rangle$ being an arbitrary single qubit state for i from 0 to $n-1$. An important subclass of quantum circuits possessing this property are *reversible circuits*[7] [18], which effectively represent bijective Boolean functions and form a dedicated research area on their own.

However, while restricting the potential quantum states to product states allows to efficiently analyze the respective circuits, this restriction prohibits a fundamental quantum mechanical phenomenon to be employed: *entanglement*. As reviewed in Section 2, two qubits being entangled means that their state *cannot* be considered separately anymore but has to be considered as a whole. Entanglement is one of the core traits that allows quantum algorithms to surpass classical algorithms for certain applications, e.g., Shor's algorithm for factoring integers is exponentially faster on a quantum computer than on a classical computer [37]. Hence, different strategies are needed to replicate the ongoing success of SAT in classical circuit and system design also for quantum circuits.

[7] This assumes that the initial state of the circuit is a classical state, i.e., chosen from the computational basis.


u
$\left\{\begin{array}{l}|00\rangle = [0]_{10}\\ |01\rangle = [1]_{10}\\ |10\rangle = [2]_{10}\\ |11\rangle = [3]_{10}\end{array}\right\}$
H
v
$\left\{\begin{array}{l}|{+}0\rangle = [4]_{10}\\ |{+}1\rangle = [5]_{10}\\ |{-}0\rangle = [6]_{10}\\ |{-}1\rangle = [7]_{10}\end{array}\right\}$
w
$\left\{\begin{array}{l}|\Phi^+\rangle = [8]_{10}\\ |\Psi^+\rangle = [9]_{10}\\ |\Phi^-\rangle = [10]_{10}\\ |\Psi^-\rangle = [11]_{10}\end{array}\right\}$

Blocking Constraints	Functional Constraints	
$[\mathbf{x}^\mathbf{u}]_2 \leq 3$	$([\mathbf{x}^\mathbf{u}]_2 = 0) \Leftrightarrow ([\mathbf{x}^\mathbf{v}]_2 = 4)$	$([\mathbf{x}^\mathbf{v}]_2 = 4) \Leftrightarrow ([\mathbf{x}^\mathbf{w}]_2 = 8)$
$[\mathbf{x}^\mathbf{v}]_2 \leq 7$	$([\mathbf{x}^\mathbf{u}]_2 = 1) \Leftrightarrow ([\mathbf{x}^\mathbf{v}]_2 = 5)$	$([\mathbf{x}^\mathbf{v}]_2 = 5) \Leftrightarrow ([\mathbf{x}^\mathbf{w}]_2 = 9)$
$[\mathbf{x}^\mathbf{w}]_2 \leq 11$	$([\mathbf{x}^\mathbf{u}]_2 = 2) \Leftrightarrow ([\mathbf{x}^\mathbf{v}]_2 = 6)$	$([\mathbf{x}^\mathbf{v}]_2 = 6) \Leftrightarrow ([\mathbf{x}^\mathbf{w}]_2 = 10)$
	$([\mathbf{x}^\mathbf{u}]_2 = 3) \Leftrightarrow ([\mathbf{x}^\mathbf{v}]_2 = 7)$	$([\mathbf{x}^\mathbf{v}]_2 = 7) \Leftrightarrow ([\mathbf{x}^\mathbf{w}]_2 = 11)$

Figure 4 Structural analysis and resulting encoding for the Bell circuit from Figure 1.

4 Generalized Encoding

In order for the encoding of a quantum circuit to support entanglement, qubits can no longer be considered individually. As a result, the qubits in a quantum circuit no longer represent individual signals (as in the case of classical circuits or in Example 6) but are bundled together. This creates a structure where signals are interleaved with the gates of the circuit and serves to *generalize* the encoding from the previous section. Again, an example illustrates the idea:

▶ **Example 7.** *Consider again the scenario from Example 4 and assume we want to perform a structural analysis on the Bell circuit.*

Instead of the two different initial states for each qubit in Example 6 ($|0\rangle$ and $|1\rangle$), the analysis starts off considering four different states of both qubits ($|00\rangle$ $|01\rangle$, $|10\rangle$, and $|11\rangle$). Then, the state transitions can be computed in a similar fashion as in Equation 1:

$$(H \otimes I)\,|\varphi\rangle = \begin{cases} |+\rangle \otimes |0\rangle & \text{if } |\varphi\rangle = |00\rangle \\ |+\rangle \otimes |1\rangle & \text{if } |\varphi\rangle = |01\rangle \\ |-\rangle \otimes |0\rangle & \text{if } |\varphi\rangle = |10\rangle \\ |-\rangle \otimes |1\rangle & \text{if } |\varphi\rangle = |11\rangle \end{cases} \quad CNOT_{1,0}\,|\varphi\rangle = \begin{cases} |\Phi^+\rangle & \text{if } |\varphi\rangle = |+\rangle \otimes |0\rangle \\ |\Psi^+\rangle & \text{if } |\varphi\rangle = |+\rangle \otimes |1\rangle \\ |\Phi^-\rangle & \text{if } |\varphi\rangle = |-\rangle \otimes |0\rangle \\ |\Psi^-\rangle & \text{if } |\varphi\rangle = |-\rangle \otimes |1\rangle \end{cases} \tag{2}$$

Overall, twelve states have to be distinguished in this case. Consequently, for each signal s, $\lceil \log_2(12) \rceil = 4$ Boolean variables $x_3^s x_2^s x_1^s x_0^s$ are necessary to properly encode the circuit. Let $[\mathbf{x}^s]_2$ denote the interpretation of the variables of the signal s as a binary number, i.e., $[\mathbf{x}^s]_2 = \sum_{i=0}^{3} 2^i x_i^s = [k]_{10}$ for some $k \in \{0, \dots, 2^4 - 1\}$ in decimal representation. Then, the valid assignments to each of the signal variables and the functional constraints based on Equation 2 can be described by the equations shown on the bottom of Figure 4.

As the previous example shows, a generalization of the encoding from Section 3 indeed allows to incorporate entanglement. In particular, the proposed encoding can be viewed as a two-step procedure:

First, a structural analysis of the circuit is performed to determine the number of unique states that need to be encoded. This analysis starts off with a set of v possible input states $\{|\varphi_0\rangle, \dots, |\varphi_{v-1}\rangle\}$ which are given as an input. Then, the first gate of the circuit g_0 (with corresponding matrix U_0) is applied to all states in the initial set of states—yielding a new set of states $\{|\varphi'_0\rangle, \ldots, |\varphi'_{v-1}\rangle\}$, where $|\varphi'_i\rangle = U_0 |\varphi_i\rangle$ for i from 0 to $v-1$. This procedure is then continued for every gate, until the whole circuit has been processed. Over the course of this process, a separate set S is used to track the unique states. As already discussed in Subsection 3.2, a maximum of $2^{|G|} \cdot v$ unique states can result from such an analysis. However, as we will see later in Section 6, the number of unique states is typically much lower in practice.

If the first step revealed that there are $|S|$ unique states, $m = \lceil \log_2(|S|) \rceil$ Boolean variables are needed to encode every signal s in the circuit. Thus, the state space of a quantum circuit G with $|G|$ gates is encoded using a total of $m \cdot (|G| + 1)$ bits, i.e.,

$$\mathbf{x}^s = x^s_{m-1} \ldots x^s_0 \text{ for } s = s_0, \ldots, s_{|G|}.$$

Since $|S|$ might not be a power of two, some of the assignments of the signal variables $\mathbf{x}^s$ do not correspond to valid values. Let $[\mathbf{x}^s]_2$ again denote the interpretation of the variables of the signal s as a binary number, i.e.,

$$[\mathbf{x}^s]_2 = \sum_{i=0}^{m-1} 2^i x^s_i = [k]_{10}$$

for some $k \in \{0, \ldots, 2^m - 1\}$ in decimal representation. Then, so-called *blocking constraints* are introduced to limit the assignments of the signal variables, i.e.,

$$\forall s \in G\colon\ [\mathbf{x}^s]_2 < |S|.$$

In addition, the value of the first signal s_0 is restricted to one of the v (unique) input states, i.e.,

$$[\mathbf{x}^{s_0}]_2 < v.$$

In order to complete the encoding, the circuit's signals have to be related to each other. Assume that gate $g_i \in G$ with input signal s_i and output signal s_{i+1} maps the k^{th} unique state $|\varphi_k\rangle$ to the l^{th} unique state $|\varphi_l\rangle$. Then, the following *functional constraint* is added:

$$[\mathbf{x}^{s_i}]_2 = [k]_{10} \iff [\mathbf{x}^{s_{i+1}}]_2 = [l]_{10}.$$

Eventually, the final encoding is obtained by taking the conjunction over all constraints.

In contrast to the approach described in Subsection 3.2—which directly translates encoding techniques of classical circuits to quantum circuits and, in the process, is limited to only a small fraction of quantum circuits—the generalized encoding proposed above indeed allows to encode the functionality of arbitrary quantum circuits. However, this comes at a price (*"there is no free lunch"*[8]):

- Consider an n-qubit quantum circuit and assume that each of the qubits can either be in the $|0\rangle$ or $|1\rangle$ state. Then, incorporating entanglement into the encoding, i.e., no longer treating qubits in an isolated fashion, leads to an exponential growth of the potential state space—instead of two states per qubit (as in Example 6), suddenly 2^n states (all of the n-qubit computational basis states) need to be considered and encoded.
- As reviewed in Section 2, the state of an n-qubit quantum system is generally characterized by 2^n complex-valued amplitudes. Hence, the complexity of representing and manipulating each of the states (i.e., its amplitudes) grows exponentially with respect to the number of qubits—one of the core phenomena that makes simulating quantum circuits on classical computers incredibly hard.

[8] Adapted from a common version in complexity theory and optimization [47].

This seems like a very grim situation—not only does the encoding result in an exponentially large state space, but also in an exponentially large size of the representation of each individual state. Since, as stated earlier, most quantum algorithms actually assume the initial state to be the all-zero state $|0\dots0\rangle$, only a single initial state needs to be considered in these cases. However, even then, the exponential size of the states' representation remains a roadblock. In the following, we show that this dark picture considerably lightens up when not dealing with arbitrary quantum circuits but rather focusing on particular classes of quantum circuits.

5 Overcoming the Limitations for Certain Classes of Quantum Circuits

As argued in the previous section, an encoding for arbitrary circuits is not feasible in the quantum case due to the sheer complexity of representing quantum states—rendering the structural analysis the dominant source of complexity in the encoding. In the following, we demonstrate that an efficient encoding is possible for certain (restricted, yet still important) classes of quantum circuits. One of the most natural examples, *Clifford* circuits, is discussed first. Afterwards, we elaborate on other classes of quantum circuits the generalized encoding can be adapted to and briefly discuss merits and drawbacks for each class.

5.1 Clifford Circuits and the Stabilizer Formalism

Clifford circuits [32], i.e., quantum circuits generated by the set of gates $\{H, S, CNOT\}$, form one of the most important classes of quantum circuits. This is because

1. they describe interesting quantum mechanical phenomena such as entanglement, teleportation, and superdense coding [32],
2. they are heavily used in quantum error correcting codes [13, 36, 39], and
3. according to the Gottesman-Knill Theorem [21], they can be simulated in polynomial time and space on a classical computer using the *stabilizer formalism*.

The main idea of the stabilizer formalism is to represent a quantum state by a set of operators that identify the state uniquely, instead of representing the state by a complex-valued vector of amplitudes. A unitary operator U is said to *stabilize* a state $|\varphi\rangle$ if $U|\varphi\rangle = |\varphi\rangle$, i.e., $|\varphi\rangle$ is an eigenvector of U with eigenvalue 1.

▶ **Example 8.** *Recall the definition of the Pauli matrices* $X = \left[\begin{smallmatrix}0&1\\1&0\end{smallmatrix}\right]$ *and* $Z = \left[\begin{smallmatrix}1&0\\0&-1\end{smallmatrix}\right]$. *It holds that* $Z|0\rangle = |0\rangle$*, i.e., the zero state is stabilized by the Pauli* Z *operator. Furthermore, it holds that* $X|+\rangle = |+\rangle$*, i.e., the plus state is stabilized by the Pauli* X *operator.*

Quantum states that can be produced from the all-zero state $|0\dots0\rangle$ by applying Clifford gates are frequently called *stabilizer states*. It can be shown that any n-qubit stabilizer state $|\varphi\rangle$ can be uniquely described by n Pauli tensor products of the form $\pm P_{i,0}P_{i,1}\dots P_{i,n-1}$ with $P_{i,j} \in \{I, X, Y, Z\}$ for $i, j = 0, \dots, n-1$. These n-qubit Pauli operators are the generators of the group of stabilizers that stabilize a particular state, i.e., any operator U stabilizing $|\varphi\rangle$ can be generated from these Pauli operators. Since two bits are needed for each generator to specify each of the n Pauli matrices and one bit for the sign, $n(2n + 1)$ bits are needed in total to uniquely describe the state $|\varphi\rangle$.

▶ **Example 9.** *This idea is most conveniently represented via the so-called* stabilizer tableau *[1, 21]. For any n-qubit state, the tableau consists of n rows for the generators and $2n+1$ columns identifying each generator—which can be grouped as*

$$\left[\begin{array}{ccc|ccc|c} x_{0,0} & \cdots & x_{0,n-1} & z_{0,0} & \cdots & z_{0,n-1} & r_0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots \\ x_{n-1,0} & \cdots & x_{n-1,n-1} & z_{n-1,0} & \cdots & z_{n-1,n-1} & r_{n-1} \end{array}\right], \textit{ where } \begin{array}{l} x_{i,j} = \begin{cases} 1 & P_{i,j} = X \textit{ or } Y \\ 0 & \textit{otherwise} \end{cases} \\ z_{i,j} = \begin{cases} 1 & P_{i,j} = Z \textit{ or } Y \\ 0 & \textit{otherwise} \end{cases} \end{array}$$

and $r_i = 1$ if the i^{th} generator has negative phase for $i, j = 0, \ldots, n-1$.

The Gottesman-Knill theorem states that the generators of a stabilizer state can be updated in polynomial time after the application of Clifford gates. To this end, the following rules for updating the corresponding tableau are used where $\oplus$ denotes a bitwise XOR operation:

- H on q_j: $\forall i = 0, \ldots, n-1$: $r_i = r_i \oplus x_{i,j} z_{i,j}$ and swap $x_{i,j}$ and $z_{i,j}$,
- S on q_j: $\forall i = 0, \ldots, n-1$: $r_i = r_i \oplus x_{i,j} z_{i,j}$ and $z_{i,j} = z_{i,j} \oplus x_{i,j}$, and
- $CNOT$ with control q_c and target q_t: $\forall i = 0, \ldots, n-1$: $r_i = r_i \oplus x_{i,c} z_{i,t} (x_{i,t} \oplus z_{i,c} \oplus 1)$, $x_{i,t} = x_{i,t} \oplus x_{i,c}$, and $z_{i,c} = z_{i,c} \oplus z_{i,t}$.

▶ **Example 10.** *Consider the situation from Example 7. Then, the initial tableau is given by*

$$\left[\begin{array}{cc|cc|c} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{array}\right], \textit{ corresponding to generators } \{ ZI, IZ \} \mathrel{\hat{=}} |00\rangle \,.$$

Applying an H gate to q_1 leads to the updated tableau

$$\begin{array}{c} \begin{array}{ccccc} & x_1 & & z_1 & \end{array} \\ \left[\begin{array}{cc|cc|c} 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{array}\right] \end{array}, \textit{ corresponding to generators } \{ ZI, IX \} \mathrel{\hat{=}} |{+}0\rangle \,.$$

Finally, applying a $CNOT$ with control q_1 and target q_0 results in the final tableau

$$\begin{array}{c} \begin{array}{ccccc} x_t & x_c & z_t & z_c & \end{array} \\ \left[\begin{array}{cc|cc|c} 0 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 \end{array}\right] \end{array}, \textit{ corresponding to generators } \{ ZZ, XX \} \mathrel{\hat{=}} |\Phi^+\rangle \,.$$

Using this stabilizer tableau (rather than exponentially large complex-valued vectors of amplitudes), the individual state representations remain polynomial—since, as demonstrated above, $\mathcal{O}(n^2)$ bits suffice to uniquely describe a stabilizer state. Hence, instead of directly formalizing the quantum states in the generalized encoding described in Section 4 (as shown in Example 7), the respective stabilizer generators can be encoded (in the signal variables $\mathbf{x}^s$) and related to each other by functional constraints. That is, each of the k unique generators computed in the pre-processing step is encoded using $m = \lceil log_2(k) \rceil$ variables $\mathbf{x}^s = x^s_{m-1} \ldots x^s_0$ such that different assignments to the variables symbolically encode the different generators. Furthermore, respective functional constraints relating the generators to each other are added as described in Section 4.

As a result, SAT-based solutions (e.g., for equivalence checking) for this central class of quantum circuits with an undoubtedly wide range of applications can be efficiently realized.

5.2 Beyond Clifford Circuits

While Clifford circuits constitute an important class of quantum circuits, it is well known that they are not *universal* for quantum computing [32], i.e., not every quantum computation can be realized using only H, S, and $CNOT$ (it has even been proclaimed that Clifford circuits are not universal for *classical* computation [1]). Hence, to formulate efficient and scalable SAT encodings for design tasks involving non-Clifford gates, further dedicated classes of quantum circuits need to be considered. As discussed earlier, the feasibility of the generalized encoding proposed in this work depends on the viability of representing and evolving the state of the quantum system. To this end, several data-structures, and methods with the capability of efficiently simulating certain classes of quantum circuits have been proposed. Some of the most important examples are:

- *Dense Arrays*: As witnessed in Section 3, product states, i.e., multi-qubit states where each qubit can be considered in an isolated fashion, can be effectively described by linearly-sized vectors (with respect to the number of qubits).

- *Hash Maps* [26]: Many state vectors that occur during the simulation of a quantum circuit are sparse due to some structure in the algorithm or even the underlying problem. Thus, such states can be efficiently handled using hash maps instead of storing the dense state vector explicitly.

- *Decision Diagrams* [14, 30, 33, 49]: They take the idea of exploiting sparsity one step further by also taking advantage of any redundancies present in the underlying representation. To this end, the representation of a quantum state is recursively decomposed and sub-parts only differing by a constant factor are identified with each other—eventually forming a directed, acyclic graph with complex-valued edge weights. Whenever the number of nodes in the decision diagram can be kept low, an efficient representation is obtained.

- *Tensor Networks* [15]: Intuitively, tensors can be seen as generalization of matrices and tensor networks are a generalization of quantum circuits that have a similar diagrammatic representation. In quantum many-body physics (where, e.g., the collective behavior of interacting particles is studied), it generally holds that particles close to another interact strongly, while particles at a distance hardly interact. This induces a notion of topological locality that naturally motivates modeling such systems as tensor networks. Whenever, the dimensions of the tensors and their connections can be kept in check, again, an efficient representation results.

- *Stabilizer Rank Methods* [8]: The idea of these methods is to decompose a generic quantum circuit into (multiple) stabilizer/Clifford circuits that, as shown in Subsection 5.1, can be efficiently simulated, and adequately combining the respectively obtained results. Their performance scales with the number of non-Clifford gates. Hence, as long as this number is low, circuits can be handled efficiently.

Overall, if there is an efficient method to conduct a structural analysis of the quantum circuit in question, the generalized encoding proposed in this work can be used to yield a corresponding SAT formulation. In this fashion, design tasks for a large variety of quantum circuits can be tackled with SAT.

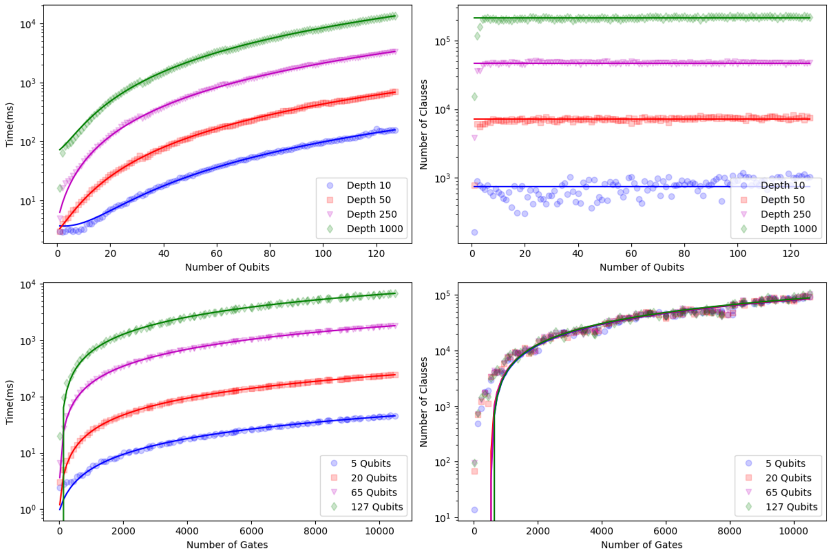


Figure 5 Scaling of the proposed SAT encoding algorithm applied to Clifford circuits. Runtime scales as $\mathcal{O}(n^2)$ and $\mathcal{O}(|G|)$ with n and $|G|$ denoting the number of qubits and gates, respectively. The number of clauses is constant in the number of qubits and otherwise scales as $\mathcal{O}(|G|)$.

6 Experimental Evaluation

In the following, we provide an empirical analysis on the scalability of the proposed generalized encoding applied to Clifford circuits and demonstrate the feasibility of the pre-processing and construction algorithms for a precise design task (namely *equivalence checking*). To this end, the generalized encoding proposed in Section 4 has been implemented in C++ on top of the JKQ quantum toolkit [44] using the SAT (SMT) solver Z3 [31]. We used Clifford circuits as a representative class of quantum circuits, since, as described in Subsection 5.1, this class of circuits is central in quantum computation and can be used to apply the proposed generalized encoding efficiently. The resulting implementation and evaluation is publicly available at `github.com/cda-tum/qusat`. All experiments were conducted on a machine with an 2.8 GHz Intel i7-1165G7 CPU and 32 GiB RAM running Ubuntu 20.04.

6.1 Scalability With Respect to Number of Qubits and Gates

In a first series of evaluations, we investigated how fast the encoding for Clifford circuits can be constructed and how many SAT clauses (as reported by Z3) are required with respect to the number of qubits as well as the number of gates. To this end, we randomly generated Clifford circuits with growing numbers of qubits and gates. Since the circuits are generated randomly, we sample ten circuits with the same parameters and then plot the mean of the computed values to obtain a representative set of results. Without loss of generality, we consider the all-zero state as the single input state. The obtained results are depicted in Figure 5.

The top-left graph shows the time needed (in ms) to construct a SAT instance using the proposed encoding, given a Clifford circuit. This includes the time needed to pre-process the circuit (i.e., to conduct the structural analysis) and to construct the corresponding Z3 instance. The graph indicates that the runtime of the proposed algorithm scales quadratically in the number of qubits (which can be attributed to the quadratic size of the stabilizer tableau).

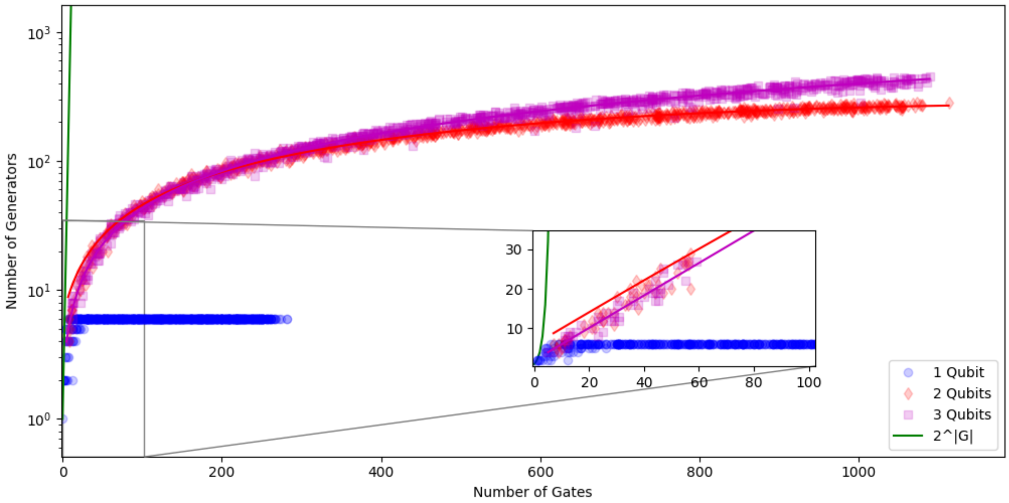


Figure 6 Number of unique generators (states) in Clifford circuits.

The top-right graph depicts the number of SAT clauses constructed by Z3 with respect to the number of qubits. As expected, these functions have constant behaviour in the number of qubits, since, in general, the number of clauses does not depend on that number. However, the number of qubits remains a dominant factor for the feasibility of the structural analysis.

The bottom two graphs show the runtime and the number of clauses with respect to the number of gates. These results indicate that both, the time needed to construct the encoding and the number of clauses, grow linearly with respect to the number of gates (which can be attributed to the tableau update rules being linear). Even circuits with 10 000 gates and 127 qubits only roughly take 10 s to encode.

6.2 Scalability With Respect to Number of Generators

As discussed in Subsection 3.2, a maximum of $2^{|G|} \cdot v$ unique states needs to be distinguished in order to encode a particular quantum circuit with v initial input states in the worst-case. Then, assuming that there are m unique states, a total of $\lceil \log_2(m) \rceil$ bits are needed to encode each signal of the circuit. In a second series of evaluations, we investigated how the number of unique states (or, in this case, the number of unique generators) evolves in practice with respect to the number of qubits and the number of gates in Clifford circuits. As in the previous experiments and without loss of generality, we assume that $v = 1$ and that all computations start out in the all-zero state $|0 \dots 0\rangle$.

For the experiments, we randomly generated Clifford circuits for one, two, and three qubits with a growing number of gates. Again, we sample ten circuits with the same parameter set and consider the means of the computed values to obtain a representative sample. The respectively obtained results are shown in Figure 6, which also includes a graph for the worst-case complexity $2^{|G|}$.

It can be seen that the actual number of unique generators in Clifford circuits is very small compared to the worst-case behavior, e.g., there are at most 6 different single-qubit stabilizer states and 360 two-qubit stabilizer states. This can be explained by the fact that the size of the stabilizer tableau used to describe particular states inherently limits the number of different states that can occur for a given number of qubits. As a consequence, at most $\lceil \log_2(6) \rceil = 3$ bits per signal are needed in the single-qubit case, while a maximum of $\lceil \log_2(360) \rceil = 9$ bits is needed for the two-qubit case—independent of the number of gates.

6.3 Application: Equivalence Checking

In a final series of evaluations, we demonstrate the applicability of the proposed encoding to the task of verifying the equivalence of two quantum circuits (in particular, Clifford circuits). To check whether two circuits are equivalent, we construct a *miter* structure [7], i.e., the same input is applied to both circuits and the outputs are fed into XOR gates, which are then ORed to produce the single output of the miter. The task for the SAT solver is to find a variable assignment for which the miter's output evaluates to one. Whenever this is the case, i.e., the solver returns *satisfiable*, both circuits have been shown to be non-equivalent and the variable assignment provides the counterexample.

For the experiments we again generated random Clifford circuits with a growing number of qubits. In a first batch of experiments, we use the same circuit twice and pass the resulting miter structure to the SAT solver (*Equivalent Instances*). To also test non-equivalent instances, a random single-gate error has been introduced in the second realization of the circuit, i.e., a random gate is removed (*Non-Equivalent Instances*). In order to detect this non-equivalence, sixteen random input states have been considered for all instances. Based on [12], this amount of input states is almost guaranteed to detect these kinds of errors.

The results are shown in Table 1. To this end, the first two columns list the number of qubits n and the number of gates $|G|$. Then, the runtime for the pre-processing and SAT instance construction t_{prep}, the runtime for the Z3 solver t_{sol}, and the number of conflicts (as reported by Z3) for both, equivalent as well as non-equivalent instances. These results indicate that even for instance with millions of gates, the respective instances can be constructed in the matter of around 600 s and solved (i.e., verified) within less than 20 s.

7 Conclusions and Outlook

In this work, we investigated the problem of constructing satisfiability encodings for quantum circuits. With the goal of establishing efficient SAT formulations for important classes of quantum circuits, we proposed a generalized encoding that can, in principle, be applied to any quantum circuit. Since quantum states and circuits require exponentially large representations in general, we identified classes of quantum circuits to which the proposed encoding can be applied in order to obtain efficient satisfiability formulations.

To highlight the practical relevance of the proposed encoding, we considered Clifford circuits, a central class of quantum circuits. Our experimental evaluations showed that, for this class of circuits, the proposed encoding can be constructed in an efficient and scalable manner. The resulting satisfiability formulation can even be generated for large circuit instances with more than one hundred qubits and more than one million quantum gates. Modern SAT solvers can then easily solve the instances produced by the proposed encoding technique in a matter of seconds even for numbers of qubits which are currently near the maximum number of qubits in actually available quantum computers. Therefore, we demonstrated that the proposed formulation can be used to solve highly relevant circuit design tasks for large quantum circuits—as exemplarily showcased by the design task of equivalence checking.

In the future, it will be interesting to apply the generalized encoding to further classes of circuits for which the proposed encoding can, in principle, be constructed efficiently, e.g., those described in Subsection 5.2, and analyze the expressiveness of these circuit classes and the respective scalability of the proposed encoding. Furthermore, it will be interesting to start exploring the myriad of design tasks that can now be tackled using SAT techniques. For all these endeavors, we believe the work summarized in this paper provides a very strong foundation.

Table 1 Equivalence checking for random Clifford circuits with growing number of qubits.

Circuit		Equivalent Instances			Non-Equivalent Instances		
n	$\|G\|$	t_{prep} [s]	t_{sol} [s]	Confl.	t_{prep} [s]	t_{sol} [s]	Confl.
4	31 730	3.32	4.46	4	3.28	4.22	12
8	73 682	6.15	6.44	29	5.71	7.94	248
12	116 712	10.07	6.58	28	9.74	7.61	7
16	159 094	15.88	6.78	32	14.13	5.59	0
20	201 092	21.88	7.23	77	19.64	5.84	7
24	242 964	30.04	19.55	294	26.08	12.86	409
28	283 878	36.94	6.74	388	36.39	6.78	6
32	326 492	43.28	4.68	353	38.40	6.42	7
36	370 466	57.46	6.60	27	50.02	6.16	6
40	411 944	69.09	6.85	27	61.87	6.06	8
44	454 908	84.46	7.45	37	75.20	5.70	0
48	497 702	98.99	16.46	458	88.49	5.65	0
52	537 808	110.03	6.37	26	90.47	7.10	72
56	581 668	127.31	7.87	210	127.96	6.11	8
60	624 326	154.88	7.46	38	127.09	5.98	7
64	666 372	168.69	8.19	248	146.94	5.39	7
68	708 072	188.80	6.64	31	148.91	8.28	173
72	751 056	211.98	18.82	423	171.36	8.76	346
76	792 770	221.63	5.78	29	211.90	6.08	9
80	834 440	259.55	6.78	41	232.48	6.24	10
84	877 294	263.19	10.95	332	275.78	6.00	0
88	918 338	297.66	5.21	26	275.32	5.79	8
92	960 228	347.95	7.62	105	305.41	5.33	8
96	1 004 426	355.85	7.70	184	287.92	5.72	0
100	1 045 466	405.66	7.74	270	325.89	6.95	0
104	1 089 230	417.63	6.38	30	409.73	6.26	0
108	1 129 920	447.50	7.26	400	442.26	6.49	8
112	1 171 906	487.94	6.85	40	453.64	5.64	6
116	1 214 050	516.56	6.33	26	490.41	6.35	8
120	1 254 944	573.97	7.22	40	507.70	6.41	7
124	1 300 580	579.71	5.68	25	586.58	5.99	0

n: Number of qubits $|G|$: Number of gates
t_{prep}: Pre-processing and SAT instance construction time t_{sol}: Z3 solving time

Confl.: Number of conflicts in Z3 DPLL solving

References

1 Scott Aaronson and Daniel Gottesman. Improved simulation of stabilizer circuits. *Physical Review A*, 70(5):052328, 2004.

2 Matthew Amy and Vlad Gheorghiu. staq—A full-stack quantum processing toolkit. *Quantum Sci. Technol.*, 5(3):034016, 2020.

3 A. Biere, A. Biere, M. Heule, H. van Maaren, and T. Walsh. *Handbook of Satisfiability.* IOS Press, 2009.

4 Armin Biere and Daniel Kröning. SAT-based Model Checking. In *Handbook of Model Checking*, pages 277–303. Springer, 2018.

5 Adam D. Bookatz. QMA-complete problems. *arXiv:1212.6312*, 2013.

6 A. Botea, A. Kishimoto, and Radu Marinescu. On the complexity of quantum circuit compilation. In *Int'l Symp. on Combinatorial Search*, 2018.

7 Daniel Brand. Verification of large synthesized designs. In *Int'l Conf. on CAD*, 1993.

8 Sergey Bravyi, Dan Browne, Padraic Calpin, Earl Campbell, David Gosset, and Mark Howard. Simulation of quantum circuits by low-rank stabilizer decompositions. *Quantum*, 3:181, 2019.

9 Sergey Bravyi, David Gosset, Robert Koenig, and Marco Tomamichel. Quantum advantage with noisy shallow circuits. *Nature Physics*, 16(10):1040–1045, 2020.

10 John Brennan et al. Tensor Network Circuit Simulation at Exascale. *arXiv:2110.09894*, 2021.

11 Lukas Burgholzer and Robert Wille. Advanced equivalence checking for quantum circuits. *IEEE Trans. on CAD of Integrated Circuits and Systems*, 2021.

12 Lukas Burgholzer, Robert Wille, and Richard Kueng. Characteristics of reversible circuits for error detection. *ArXiv201202037 Cs*, 2020.

13 A Robert Calderbank and Peter W Shor. Good quantum error-correcting codes exist. *Physical Review A*, 54(2):1098, 1996.

14 Lu Chin-Yung, Wang Shiou-An, and Kuo Sy-Yen. An extended XQDD representation for multiple-valued quantum logic. *IEEE Trans. Comput.*, pages 1377–1389, 2011.

15 Ignacio Cirac, David Perez-Garcia, Norbert Schuch, and Frank Verstraete. Matrix Product States and Projected Entangled Pair States: Concepts, Symmetries, and Theorems. *arXiv:2011.12127*, 2021.

16 Arianne Meijer-van de Griend and Ross Duncan. Architecture-aware synthesis of phase polynomials for NISQ devices. *arXiv:2004.06052*, 2020.

17 Stephan Eggersglüß, Robert Wille, and Rolf Drechsler. Improved SAT-based ATPG: More constraints, better compaction. In *Int'l Conf. on CAD*, 2013.

18 Michael P. Frank. The Future of Computing Depends on Making It Reversible. *IEEE Spectrum*, 2017.

19 Anteneh Gebregiorgis and Mehdi Baradaran Tahoori. Test pattern generation for approximate circuits based on Boolean satisfiability. In *Design, Automation and Test in Europe*, 2019.

20 Brett Giles and Peter Selinger. Exact synthesis of multiqubit Clifford+T circuits. *Phys. Rev. A*, 87(3):032332, 2013.

21 Daniel Gottesman. The Heisenberg representation of quantum computers. *arXiv:quant-ph/9807006*, 1998.

22 Lov K. Grover. A fast quantum mechanical algorithm for database search. *Proc. of the ACM*, pages 212–219, 1996.

23 Thomas Häner, Damian S. Steiger, Krysta Svore, and Matthias Troyer. A software methodology for compiling quantum programs. *Quantum Sci. Technol.*, 3(2):020501, 2018.

24 Hsin-Yuan Huang, Richard Kueng, and John Preskill. Information-theoretic bounds on quantum advantage in machine learning. *Physical Review Letters*, 126(19):190505, 2021.

25 Dominik Janzing, Pawel Wocjan, and Thomas Beth. "Non-identity check" is QMA-complete. *Int. J. Quantum Inform.*, 03(03):463–473, 2005.

26 Samuel Jaques and Thomas Häner. Leveraging state sparsity for more efficient quantum simulations. *arXiv:2105.01533*, 2021.

27 Daniela Kaufmann, Armin Biere, and Manuel Kauers. Verifying large multipliers by combining SAT and computer algebra. In *Int'l Conf. on Formal Methods in CAD*, pages 28–36, 2019.

28 T. Larrabee. Test pattern generation using Boolean satisfiability. *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, 11(1):4–15, 1992.

29 Gushu Li, Yufei Ding, and Yuan Xie. Tackling the qubit mapping problem for NISQ-era quantum devices. In *Int'l Conf. on Architectural Support for Programming Languages and Operating Systems*, 2019.

30 D.M. Miller and M.A. Thornton. QMDD: A decision diagram structure for reversible and quantum circuits. In *Int'l Symp. on Multi-Valued Logic*, 2006.

31 Leonardo de Moura and Nikolaj Bjørner. Z3: An efficient SMT solver. In *Int'l Conf. on Tools and Algorithms for the Construction and Analysis of Systems*, 2008. github.com/Z3Prover/z3.

32 Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, 2010.

33 Philipp Niemann et al. QMDDs: Efficient quantum function representation and manipulation. *IEEE Trans. on CAD of Integrated Circuits and Systems*, 2016.

34 Philipp Niemann, Robert Wille, and Rolf Drechsler. Efficient synthesis of quantum circuits implementing Clifford group operations. In *Asia and South Pacific Design Automation Conf.*, pages 483–488, 2014.

35 Diego Riste et al. Demonstration of quantum advantage in machine learning. *npj Quantum Information*, 3(1):1–5, 2017.

36 Peter W Shor. Scheme for reducing decoherence in quantum computer memory. *Physical review A*, 52(4):R2493, 1995.

37 Peter W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM J. Comput.*, 1997.

38 Kaitlin N. Smith and Mitchell A. Thornton. A quantum computational compiler and design tool for technology-specific targets. In *Int'l Symp. on Computer Architecture*, 2019.

39 Andrew M Steane. Error correcting codes in quantum theory. *Physical Review Letters*, 77(5):793, 1996.

40 Bochen Tan and Jason Cong. Optimal layout synthesis for quantum computing. In *Int'l Conf. on CAD*, 2020.

41 S.-A. Wang, C.-Y. Lu, I-M. Tsai, and S.-Y. Kuo. An XQDD-based verification method for quantum circuits. In *IEICE Trans. Fundamentals*, pages 584–594, 2008.

42 Robert Wille, Lukas Burgholzer, and Alwin Zulehner. Mapping quantum circuits to IBM QX architectures using the minimal number of SWAP and H operations. In *Design Automation Conf.*, 2019.

43 Robert Wille, Daniel Große, Finn Haedicke, and Rolf Drechsler. SMT-based stimuli generation in the SystemC Verification library. In *Forum on Specification and Design Languages*, 2009.

44 Robert Wille, Stefan Hillmich, and Lukas Burgholzer. JKQ: JKU tools for quantum computing. In *Int'l Conf. on CAD*, 2020.

45 Robert Wille, Nils Przigoda, and Rolf Drechsler. A compact and efficient SAT encoding for quantum circuits. In *Africon*, 2013.

46 Robert Wille, Hongyan Zhang, and Rolf Drechsler. ATPG for reversible circuits using simulation, boolean satisfiability, and pseudo boolean optimization. In *IEEE Annual Symp. on VLSI*, 2011.

47 David H. Wolpert and William G. Macready. No free lunch theorems for optimization. *IEEE Trans. Evol. Comput.*, 1(1):67–82, 1997.

48 S. Yamashita and I. L. Markov. Fast equivalence-checking for quantum circuits. In *Int'l Symp. on Nanoscale Architectures*, 2010.

49 Alwin Zulehner, Stefan Hillmich, and Robert Wille. How to efficiently handle complex values? Implementing decision diagrams for quantum computing. In *Int'l Conf. on CAD*, 2019.

50 Alwin Zulehner and Robert Wille. Advanced simulation of quantum computations. *IEEE Trans. on CAD of Integrated Circuits and Systems*, 2019.