Handling Non-Unitaries in Quantum Circuit Equivalence Checking

Lukas Burgholzer
Institute for Integrated Circuits, Johannes Kepler University Linz, Austria
lukas.burgolzer@jku.at

Robert Wille
Chair for Design Automation, Technical University of Munich, Germany
Software Competence Center Hagenberg GmbH, Austria
robert.wille@tum.de

ABSTRACT

Quantum computers are reaching a level where interactions between classical and quantum computations can happen in real-time. This marks the advent of a new, broader class of quantum circuits: *dynamic quantum circuits*. They offer a broader range of available computing primitives that lead to new challenges for design tasks such as simulation, compilation, and verification. Due to the non-unitary nature of dynamic circuit primitives, most existing techniques and tools for these tasks are no longer applicable in an out-of-the-box fashion. In this work, we discuss the resulting consequences for quantum circuit verification, specifically *equivalence checking*, and propose two different schemes that eventually allow to treat the involved circuits as if they did not contain non-unitaries at all. As a result, we demonstrate methodically, as well as, experimentally that existing techniques for verifying the equivalence of quantum circuits can be kept applicable for this broader class of circuits.

1 INTRODUCTION

Capabilities of quantum computers built today are steadily growing. New devices do not only feature more and more qubits which are less prone to errors, but also allow for a much tighter classical control loop. This is witnessed by the OpenQASM 3.0 specification recently published by IBM [1] and the ability to perform conditional resets on IBM's quantum computers [2]. Through the interaction of classical computation with the gates and measurements of a quantum circuit, new computing primitives such as mid-circuit measurements and resets as well as classically-controlled operations become possible within the coherence time for a single circuit execution. We adopt the naming established by IBM and call this new, broader class of circuits *dynamic quantum circuits*.

With these rapid advances in physical realizations comes the need for quantum software that aids developers and users to keep up with this pace. Otherwise, we might end up in a situation where we have powerful quantum computers available, but no efficient means to use them. Besides challenges, e.g., for classical/quantum design and compilation in general, this also poses new challenges for quantum circuit verification.

Verification of quantum circuits (more specifically, *equivalence checking*) is an essential part in the modern quantum design flow. To this end, the goal is to check whether two supposedly equivalent quantum circuits G and G' indeed realize the same functionality. Important use cases include (1) ensuring that the originally intended functionality of a quantum algorithm is preserved throughout the whole compilation process that the algorithm's circuit representation undergoes in order to be executable on an actual device, or (2) ensuring that alternative (e.g., optimized) realizations of certain building blocks in quantum circuits are functionally equivalent to their original implementation.

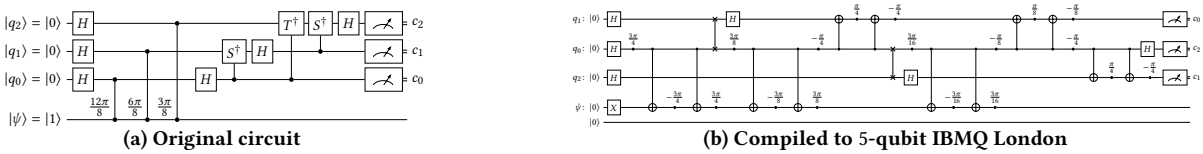
In the past, several complementary approaches have been proposed for tackling this problem [3]–[11]. However, practically all of these approaches expect the underlying functionality to be unitary—which circuits containing dynamic circuit primitives no longer are. As such, existing techniques for verifying conventional quantum circuits are not directly applicable in an out-of-the-box fashion. In this work, we discuss the resulting consequences for quantum circuit equivalence checking and show that reinventing the wheel is not necessary in order to use existing tools for verifying this broader class of circuits. To this end, we propose two different schemes targeted at two slightly different verification scenarios.

First, we consider the question whether two circuits G and G' which might contain dynamic circuit primitives are functionally equivalent as a whole. We show that, by combining well known results from quantum information, any such circuit can be *transformed* to a circuit only containing unitary operations. By transforming the dynamic circuit primitives in this fashion, existing techniques for checking the equivalence of quantum circuits can be employed for the broader class of dynamic circuits.

Second, we consider the question whether two circuits G and G' produce the same distribution of measurement outcomes given a fixed input state, i.e., whether they behave the same when executed on a quantum computer. We show how to extract the complete measurement probabilities of a dynamic circuit, as if it did not contain non-unitaries, by cleverly applying classical quantum circuit simulation.

Experimental evaluations confirm that the proposed schemes indeed allow to handle the non-unitaries introduced by dynamic circuit primitives in an efficient fashion. Overall, these schemes form a generic solution for handling non-unitaries in verifying the equivalence of quantum circuits that is applicable to any existing verification framework.

The rest of this work is structured as follows. Section 2 provides the necessary background and motivation. Then, Section 3 introduces dynamic circuits and explains the resulting problem for equivalence checking in detail, along with the general idea for solving this problem. Section 4 and Section 5 elaborate on the proposed schemes and provide some discussion, while Section 6 summarizes our experimental evaluations. Finally, we conclude in Section 7.



(a) Original circuit

(b) Compiled to 5-qubit IBMQ London

Figure 1: 3-bit precision QPE circuit for $U = p(\frac{3\pi}{8})$ and $|\psi\rangle = |1\rangle$, resulting in estimate $\tilde{\theta} = 0.c_2c_1c_0$

2 BACKGROUND AND MOTIVATION

This section establishes the notation used in the remainder of this work and provides the necessary background information on quantum circuits. We also review the *Quantum Phase Estimation* algorithm (which is used as a running example) and motivate the importance of verifying quantum circuits. While the descriptions are kept brief, we refer the unacquainted reader to the provided references for further details.

2.1 Quantum Circuits

In the traditional quantum circuit model [12], [13], a quantum circuit G, acting on n qubits, is specified by a sequence of $|G|$ quantum gates $g_0, \ldots, g_{|G|-1}$. Each quantum gate g_i, acting on $k \leq n$ qubits (most frequently $k = 1$ or $k = 2$), can be described by a $2^k \times 2^k$-dimensional unitary matrix U_i.

Given an initial state $|\varphi\rangle$ (represented as a 2^n-dimensional state vector), the evolution of this initial state under the quantum circuit can be described by successively multiplying the individual gate matrices with the current state vector. Eventually, performing all multiplications results in a final state vector that encodes the probabilities of measuring the individual computational basis states. When conducted on a classical computer, this is typically called *(classical) quantum circuit simulation.*

2.2 Quantum Phase Estimation

The key ideas of this work will be illustrated by means of a particular quantum algorithm, namely *Quantum Phase Estimation* (QPE, [12]), which represents one of the key subroutines in important quantum algorithms such as Shor's algorithm [14] for factoring numbers, the HHL algorithm [15] for solving linear systems, or quantum principal component analysis [16] for machine learning. It solves the problem of determining the phase of a unitary operator U given an eigenstate $|\psi\rangle$, i.e., determining $\theta \in [0, 1)$ such that $U|\psi\rangle = e^{2\pi i\theta}|\psi\rangle$.

To this end, the QPE algorithm determines an m-bit estimate $\tilde{\theta} = 0.c_{m-1}\ldots c_0$ of θ. First, controlled-U^{2^k} operations ($0 \leq k < m$) are used to write the m-bit Fourier basis representation of U's phase to an m-qubit register. Afterwards, the inverse *Quantum Fourier Transform* ($\text{QFT}^\dagger$, [12]) is applied to transform the result to the computational basis. Whenever θ is representable using m fractional bits, the algorithm succeeds with certainty, while otherwise, it yields a suitably high chance for success (with a probability larger than $\frac{4}{\pi^2} \approx 0.405$).

Example 1. *Assume U is given by $p(\frac{3\pi}{8}) = diag(1, e^{2\pi i\frac{3}{16}})$ and $|\psi\rangle = |1\rangle$. Then, Fig. 1a shows the quantum circuit realizing the 3-bit precision QPE algorithm. It applies three rounds of controlled-phase rotations and then uses the three-qubit inverse Fourier transform to obtain the desired estimate $\tilde{\theta} = 0.c_2c_1c_0$ from the measurement results. Since $\theta = \frac{3}{16} = 0.0011_2$ cannot be exactly represented using three fractional bits, running the algorithm yields $|001\rangle$ and $|010\rangle$ as the most probable output states.*

2.3 Verification of Compilation Results

Executing a quantum algorithm on an actual quantum computer requires *compiling* the algorithm's description G to a representation G' that adheres to all constraints imposed by the targeted device. This typically involves several steps such as synthesis [17]–[19], mapping [20]–[24], and optimizations [25]–[27].

Example 2. *Quantum computers manufactured by IBM natively support arbitrary single-qubit operations and the two-qubit controlled-NOT (or CNOT) operation. A possible realization of the QPE circuit from Fig. 1a on the five-qubit, T-shaped IBMQ London architecture is shown in Fig. 1b.*

Verifying that the original circuit's functionality is preserved throughout the individual stages of the compilation process is a vital task in the quantum computing design flow. In general, the functionality of a quantum circuit $G = g_0, \ldots, g_{|G|-1}$ is represented by the $2^n \times 2^n$ system matrix $U = U_{|G|-1} \cdots U_0$. Thus, comparing the functionality of two quantum circuits G and G' reduces to the comparison of the respective system matrices U and U'. While conceptually simple, this quickly amounts to a non-trivial task due to the fact that the involved matrices grow exponentially with respect to the number of qubits. Equivalence checking of quantum circuits has even been shown to be QMA-complete [28]. Nevertheless, several methods for this problem have been proposed [3]–[11].

3 DYNAMIC CIRCUITS AND RESULTING PROBLEM

The circuit model of quantum computing, as discussed in the previous section, has been the de-facto standard for designing quantum circuits to be executed on current generation quantum computers. However, this describes quantum circuits in a *static* fashion—with no opportunity to steer the computation in a direction based on outcomes of intermediate results. Recently, IBM announced that their quantum computers now allow for interactions with classical computing instructions within the runtime of a quantum circuit—enabling what IBM refers to as *dynamic quantum circuits* [2]. In the following, we describe what constitutes these new kind of circuits and discuss the resulting challenges for checking the equivalence of quantum circuits that might contain dynamic circuit primitives.

3.1 Dynamic Quantum Circuits and Their Benefits

By allowing the interaction of real-time classical computations with the gates and measurements of traditional quantum circuits, the quantum circuit model reviewed in Section 2.1 is extended by non-unitary primitives such as mid-circuit measurements and resets as well as classically-controlled quantum operations. As a consequence, circuits are no longer static, but rather *dynamic*.

Eventually, these primitives will be necessary for quantum computers to achieve fault-tolerance by realizing quantum error correction schemes. However, already in the near term, interesting

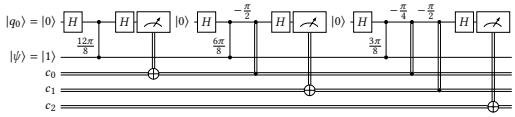


Figure 2: Dynamic version of the QPE circuit from Fig. 1a

use cases for teleportation [29] and algorithms like *Iterative QPE* (IQPE, [30]) arise that employ dynamic circuit primitives in order to, e.g., reduce the required number of qubits—a limited resource thus far.

For example, our running example, i.e., the QPE algorithm reviewed in Section 2.2, may exploit non-unitaries to reduce the number of qubits: Instead of an m-qubit register for computing the Fourier base representation of the unitary's phase, a single qubit is used and repeatedly measured. Starting from the least significant bit of the resulting estimate $\tilde{\theta} = 0.c_{m-1} \dots c_0$, each measurement adds one bit of information to the estimated phase. The result of each measurement then influences the rotation angles applied to the working qubit in the next iteration. This requires the availability of the measurement results and application of quantum operations based on them within the coherence time of the quantum computer's qubits. One of the first realizations of the IQPE algorithm on an actual system has recently been demonstrated by researchers from IBM Quantum on one of their devices [31].

Example 3. *Assume again that, as in Example 1, we want to iteratively estimate the phase θ of the unitary operator $U = p(\frac{3\pi}{8})$ corresponding to the eigenvector state $|\psi\rangle = |1\rangle$ up to a precision of three bits. Fig. 2 shows an alternative quantum circuit utilizing dynamic circuit primitives. Instead of the 3-qubit register considered before in Fig. 1a, a single working qubit in combination with mid-circuit measurements, resets, and classically-controlled single-qubit rotations is used to iteratively compute individual bits of the phase estimate. Compiling this circuit to an actual device requires no mapping at all, since only two qubits interact with each other. As a consequence, the quantum cost of the resulting circuit is considerably reduced—significantly improving the expected fidelity when executing the circuit on an actual device.*

3.2 Resulting Problem

Existing frameworks for verifying quantum circuits such as [3]–[11] generally assume the circuit to only contain unitary operations. Ultimately, only then it is possible to characterize the functionality of a quantum circuit as a unitary matrix. With the availability of dynamic circuit primitives for conducting quantum computations, the question arises how circuits using these primitives can be verified. After all, resets, measurements, and classically-controlled operations are all non-unitary operations. As such, existing techniques cannot be applied in an out-of-the-box fashion.

Several theoretical works on quantum program and protocol verification exist that deal with dynamic quantum circuits, e.g., [32], [33]. However, their goal is to prove the correctness of an algorithm, i.e., proving that it "works", rather than to check the equivalence of two circuits. Recent works on the equivalence of dynamic quantum circuits based on quantum Mealy machines [34] and ensembles of linear operators [35] show promise, but have only been evaluated on toy examples ($\approx$ 10 qubits) and have not led to available software packages for equivalence checking yet. In this work, we show that reinventing the wheel is not necessary to allow the usage of existing techniques and tools in combination with dynamic circuits. To this end, we propose two different schemes targeted at two slightly different verification scenarios.

First, we consider the question whether two circuits G and G' which might contain non-unitaries are functionally equivalent as a whole—an important question when, e.g., evaluating alternative realizations of certain building blocks in large quantum algorithms. Here, it has to be ensured that the alternative realization has the exact same functionality given *any* input. As already shown in Section 2.3, given two circuits G and G' which only contain unitary operations, this reduces to the comparison between the corresponding unitary matrices U and U'. We will show in Section 4 that any circuit containing non-unitary operations can be transformed to a circuit only containing unitary operations and no intermediate measurements by combining well known results from quantum information theory. This way, all existing techniques for verifying the equivalence of two (static) quantum circuits are kept applicable for the broader class of dynamic circuits.

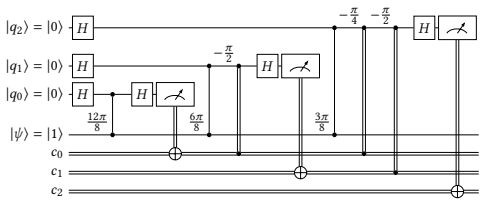
While the above technique conceptually allows to verify circuits containing non-unitaries, it requires to extend a circuits description by as many qubits as it contains mid-circuit resets. Due to the exponential scaling of the resulting unitary functionality, the complexity of verifying such instances may prove too much to handle for existing tools. The following observation helps to derive an alternative for these cases: In most quantum algorithms, the initial state of the computation can be assumed to be a fixed state (e.g., $|0 \dots 0\rangle$). Hence, it might not be necessary at all to ensure that two circuits are *fully* functionally-equivalent, but rather that they produce the same distribution of measurement outcomes for the fixed input state, i.e., that they behave the same when executed on a quantum computer. In Section 5, we show that the probability distribution of a circuit containing non-unitaries can be iteratively extracted from classically simulating the circuit using any available classical quantum circuit simulator.

4 UNITARY RECONSTRUCTION THROUGH CIRCUIT TRANSFORMATION

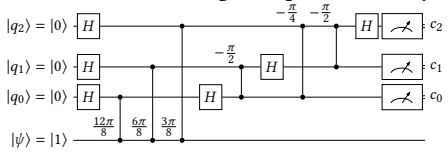
Dynamic circuit primitives allow to re-use qubits over the course of a quantum computation and to influence the execution based on classical measurement outcomes. In order to employ existing verification tools for verifying circuits using these primitives, the circuit descriptions G and G' have to be transformed to facilitate comparisons of the form $U \stackrel{?}{=} U'$. This is accomplished by transforming the dynamic circuit primitives to unveil the underlying unitary functionality.

Reset operations pose the first hurdle to overcome in this endeavour. Algorithmically, a reset can be interpreted as measuring a qubit, applying an X operation conditioned on the measurement result being $|1\rangle$ and, then, discarding the measurement result. Theoretically, any reset operation can be replaced by introducing a new qubit and applying all subsequent operations involving the qubit to be reset to the new qubit. In this fashion, any n-qubit circuit containing r reset instructions can be transformed to a circuit acting on $n + r$ qubits containing no reset primitives.

Example 4. *Consider again the circuit for the 3-bit precision IQPE algorithm from Example 3 shown in Fig. 2. By iteratively replacing each of the reset operations with a new qubit and translating all subsequent gates to the newly introduced qubits, a circuit acting on four qubits results, as shown in Fig. 3a.*



(a) Circuit after substituting new qubits for every reset



(b) Circuit after applying deferred measurement principle

Figure 3: Unitary reconstruction for IQPE circuit from Fig. 2

Once qubit re-use is eliminated from a dynamic circuit, the only potentially non-unitary primitives remaining are mid-circuit measurements and classically-controlled operations conditioned on their result. In order to get rid of these operations, we resort to one of the most fundamental results in quantum computing: the *deferred measurement principle* [12]. This principle states that delaying measurements until the end of a quantum computation does not affect the probability distribution of outcomes. As a consequence, it follows that measurement and classical-conditioning on its result commute. Thus, any mid-circuit measurement can be delayed until the very end of the quantum circuit—replacing any classically-controlled operations along the way by proper quantum operations controlled by the respective qubit.

Example 5. *Assume that all reset operations of the IQPE circuit from Example 3 have been eliminated, e.g., by transforming the circuit as described in Example 4. Then, applying the deferred measurement principle in order to delay all measurements to the end of the circuit and replacing the phase rotations controlled by the measurement outcomes with phase gates controlled on the respective qubits, results in a circuit as shown in Fig. 3b—free of dynamic circuit primitives.*

By combining both aforementioned steps, i.e., substituting reset operations with "fresh" qubits and applying the deferred measurement principle, any dynamic quantum circuit (including non-unitaries) can be transformed to a representation composed of unitary descriptions only. For one, this allows to verify that a dynamic circuit actually realizes the intended functionality of its static counterpart.

Example 6. *Compare the transformed circuit obtained in Example 5 (shown in Fig. 3b) to the original QPE algorithm shown in Fig. 1a. Due to them actually being the same, it is easy to conclude that both circuits are indeed equivalent.*

Note that it might seem that the proposed approach merely reverses the circuit construction or compilation process. As witnessed in Example 6, there is a one-to-one relation between the transformed version of the IQPE circuit shown in Fig. 3b and the original QPE circuit shown in Fig. 1a. As such, it could be argued that there is nothing to be gained from using the technique. However, this is not the case, as almost no assumptions are made about the relation between G and G' in general. Indeed, the only requirement is that the transformed versions of both circuits have the same number of primary inputs and outputs. The proposed transformation scheme "touches" nothing but reset, measurement, and classically-controlled operations—which are "reversed".

Conceptually, this approach allows to verify circuits containing non-unitaries, at the cost of extending a circuit's description by as many qubits as it contains mid-circuit resets. Since the resulting unitary functionality scales exponentially with the number of qubits, the complexity of verifying such instances increases quickly. However, this is an inevitable increase whenever verifying whether a dynamic implementation (acting on n_{dyn} qubits and using r resets) still realizes the same functionally as a static counterpart (acting on n_{static} qubits). Since in that case, $n_{dyn} + r = n_{static}$, the proposed scheme augments the dynamic circuit just enough to facilitate comparisons of the form $U \stackrel{?}{=} U'$.

5 EXTRACTING THE MEASUREMENT OUTCOME DISTRIBUTION BY SIMULATION

Although verification methodologies such as [3]–[5] frequently allow to reduce the complexity of the verification by exploiting the reversibility of quantum operations, they might not be able to handle this immense complexity in the worst case. Motivated by the fact that most high-level quantum algorithms assume a fixed input state, we argue that it might be sufficient to show that two realizations of such an algorithm produce the same distribution of measurement probabilities given the fixed input state. Verifying that two circuits G and G', which only contain unitary operations, produce equivalent probability distributions given a particular input state $|\psi\rangle$ amounts to classically simulating both computations with $|\psi\rangle$ as input and computing the overlap between the measurement probabilities described by the resulting state vectors.

However, in the presence of dynamic circuit primitives, the concept of a state vector responsible for producing the circuit's measurement outcome distribution (e.g., the probabilities of the individual bitstrings in the IQPE algorithm) does no longer make sense. This is due to the non-unitary nature of the dynamic circuit primitives, that no longer allow to deterministically simulate the quantum circuit in one go using quantum circuit simulators such as [36]–[38]. For example, each time a reset operation is encountered this would technically require the calculation of the partial trace of the system over the particular qubit (and reinitializing it to $|0\rangle$). However, the partial trace is an operation that maps pure states to mixed states. One possible approach for solving this problem would be to repeatedly simulate the dynamic circuit and stochastically realize dynamic circuit primitives such as measurements and resets. However, one would have to perform huge amounts of individual runs in order to reason about the output distribution in a statistically significant way. Another approach requires leaving the pure state picture and using a density matrix simulator (such as, e.g., [39]–[41]). Although these simulators can naturally handle resets, mid-circuit measurements, and classic-controlled operations, they also do not allow to determine the complete distribution of (intermediate) measurement outcomes via a single simulation run, but only the density matrix for a particular set of measurements.

In the following, we propose a technique that allows to extract the complete set of measurement probabilities for a dynamic circuit given a particular input state. To this end, consider a quantum circuit G involving m measurements. Then, each measurement during the circuit simulation constitutes a branching point where the

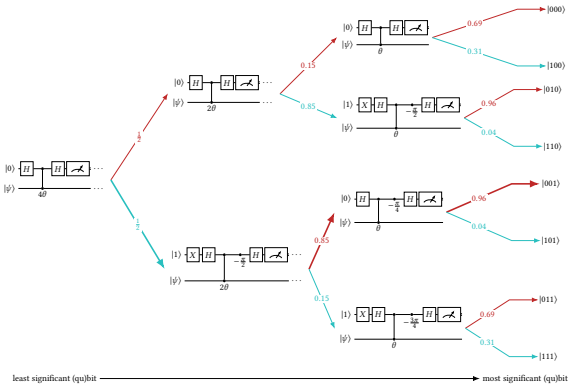


Figure 4: Measurement outcome distribution extraction for the IQPE circuit with $\theta = \frac{3\pi}{8}$. Red and blue arrows denote the $|0\rangle$- and $|1\rangle$-successor, respectively.

probabilities of the qubit to be measured are check-pointed and the simulation splits into two independent simulations: one assuming the measurement outcome is $|0\rangle$ and the other one assuming the outcome is $|1\rangle$. Depending on the outcome being $|0\rangle$ or $|1\rangle$, a subsequent reset operation is translated to a no-op or an X gate, while any classically-controlled operation is ignored or applied, respectively. The probability of observing a particular basis state $|i\rangle = |(i_{m-1} \dots i_0)_2\rangle$ can then be reconstructed from the product of the check-pointed probabilities along the path of simulations corresponding to the outcomes i_0 to i_{m-1}.

Example 7. *Consider again the IQPE algorithm for estimating the phase θ of $U = p(\frac{3\pi}{8})$ corresponding to the eigenstate $|\psi\rangle = |1\rangle$ up to a precision of three bits, as shown in Fig. 2. The circuit contains a total of $m = 3$ measurements (necessary for the 3-bit precisison) and uses the fixed input state $|000\rangle \otimes |\psi\rangle = |0001\rangle$. Iteratively simulating the circuit, check-pointing the probabilities at each of the measurements, and adjusting the subsequent circuit parts to be simulated accordingly, results in a computational flow as illustrated in Fig. 4. There, red arrows denote the $|0\rangle$-successor, while blue arrows denote the $|1\rangle$-successor, i.e., the subsequent computations upon measuring $|0\rangle$ or $|1\rangle$, respectively. The path indicated in bold represents the extraction of the probability for the $|001\rangle$ basis state—resulting in $\frac{1}{2} * 0.85 * 0.96 \approx 0.408$.*

Extracting the distribution of measurement outcomes of a dynamic circuit in this fashion naturally requires a total of 2^m individual simulations, where m is the number of mid-circuit measurements. However, large parts of the simulations can be shared in between simulation runs. For example, the circuit up until the first checkpoint only needs to be simulated once, while two simulations are necessary up until the second checkpoint, and so on. In general, the k^{th} sub-circuit needs to be simulated in at most 2^k variations. If any measurement along a path produces a probability of zero, further simulations along that path need not be started at all. In addition, the individual simulations in between checkpoints are completely independent from another and, hence, are embarrassingly parallelizable. On top of that, each of these sub-circuits consists of a much smaller number of gates and acts on far fewer qubits than the whole dynamic circuit's static counterpart. As a consequence, the complete measurement outcome distribution can be efficiently extracted in many cases, even though exponentially many simulations might be required in the worst case.

Table 1: Experimental Evaluations

Static		Dynamic		Full Functional Verification		Fixed Input State	
n	$\|G\|$	n	$\|G\|$	t_{trans} [s]	t_{ver} [s]	$t_{extract}$ [s]	t_{sim} [s]
Bernstein-Vazirani							
121	300	2	539	0.0003	0.028	0.0003	0.011
122	303	2	544	0.0003	0.028	0.0003	0.011
123	305	2	548	0.0003	0.026	0.0003	0.011
124	307	2	552	0.0003	0.028	0.0003	0.012
125	310	2	557	0.0003	0.027	0.0003	0.012
126	312	2	561	0.0004	0.028	0.0003	0.012
127	314	2	565	0.0003	0.027	0.0003	0.013
128	317	2	570	0.0003	0.030	0.0004	0.012
Quantum Fourier Transform							
23	276	1	321	0.0002	0.003	24.8242	0.001
24	300	1	347	0.0002	0.003	52.4281	0.001
25	325	1	374	0.0002	0.004	107.3160	0.001
26	351	1	402	0.0002	0.004	223.4190	0.002
125	5664	1	8124	0.0432	0.366	—	0.150
126	5723	1	8252	0.0440	0.364	—	0.150
127	5782	1	8381	0.0454	0.380	—	0.156
128	5841	1	8511	0.0465	0.380	—	0.158
Quantum Phase Estimation							
43	988	2	1071	0.0011	0.101	0.0004	0.011
44	1033	2	1118	0.0012	0.252	0.0004	0.012
45	1079	2	1166	0.0013	0.674	0.0005	0.013
46	1125	2	1215	0.0014	3.045	0.0005	0.013
47	1173	2	1265	0.0015	8.306	0.0005	0.014
48	1220	2	1316	0.0016	19.720	0.0006	0.016
49	1266	2	1368	0.0016	71.937	0.0006	0.016
50	1314	2	1421	0.0017	173.294	0.0006	0.017

n: Number of qubits $|G|$: Number of gates
t_{trans}: Runtime of the transformation scheme from Section 4
t_{ver}: Runtime of the subsequent equivalence check
$t_{extract}$: Runtime of the scheme from Section 5 for dynamic circuit
t_{sim}: Runtime of classical simulation for static circuit

6 EXPERIMENTAL EVALUATION

The methods proposed above can be implemented on top of any existing verification or simulation tool, respectively. In order to demonstrate that the proposed schemes indeed allow to efficiently handle non-unitaries in equivalence checking flows, we exemplarily implemented them on top of the open-source quantum circuit equivalence checking tool *QCEC* [42] that is publicly available at https://github.com/iic-jku/qcec. The tool supports complete functional verification (as considered in Section 4) as well as simulative verification (as considered in Section 5).

As benchmarks we consider various instances of the famous Bernstein-Vazirani algorithm [43], the Quantum Fourier Transform, and the QPE algorithm, which was used as running example throughout this work. For each static algorithm, a dynamic realization has been derived [30], [44], [45]. These algorithms are a good fit for evaluating the overhead of the proposed schemes, as they feature all the hurdles of dynamic quantum circuits that have to be overcome for verifying their equivalence. All evaluations have been conducted on a machine equipped with an AMD Ryzen 9 5950X CPU and 64 GiB RAM running Ubuntu 20.04. Table 1 summarizes the obtained results. To this end, it first lists the number of qubits n as well as the number of gates $|G|$ of the original (static) and the dynamic circuit, respectively. Then, the runtime t_{trans} of the transformation scheme (as proposed in Section 4) is listed along the time t_{ver} it took to verify the equivalence of both circuits. Finally, $t_{extract}$ denotes the runtime of the extraction scheme (proposed in Section 5) applied to the dynamic circuit, while t_{sim} denotes the runtime of the classical simulation of the original (static) circuit.

In a first series of evaluations, we employ the scheme proposed in Section 4 to eliminate the non-unitaries from the dynamic circuit and, afterwards, apply the generic "proportional" strategy of

QCEC for checking the equivalence of the resulting circuit with the corresponding original circuit. As can be seen from the results, transforming the dynamic circuit using the proposed scheme incurs practically no overhead (t_{trans} is on the order of 1 ms for all tested instances) and allows to successfully verify the full functional equivalence of the Bernstein-Vazirani and QFT algorithms with up to 128 qubits in a fraction of a second. Even the QPE instances with up to 50 qubits can be verified in less than 3 min.

In a second series of evaluations, we use the iterative simulation scheme proposed in Section 5 (without employing paralellization) to extract the distribution of measurement outcomes from the dynamic circuits. In addition, we classically simulate the static counterpart in order to judge the runtime overhead of the proposed scheme. As expected from the discussion at the beginning of Section 5, only checking the equivalence for a fixed input is an easier task compared to checking the full functional equivalence, in general. The results for the Bernstein-Vazirani and QPE algorithm show, that extracting the complete measurement probabilities of a dynamic circuit can, in fact, be faster than classically simulating the static counterpart by more than an order of magnitude. This is in line with the discussions at the end of Section 5 and can be attributed to the fact, that the respectively resulting state vectors are sparse, i.e., feature only few non-zero amplitudes. In contrast, the state vector resulting from the Quantum Fourier Transform is dense, which immediately reflects in the runtime of the extraction scheme, i.e., it roughly doubles with every added qubit. Thus, the scheme from Section 4 should be preferred in this case.

7 CONCLUSIONS

In this work, we discussed the upcoming challenges that are currently emerging with the introduction of dynamic quantum circuits. We showed that, due to their non-unitary nature, most existing solutions cannot be used for these circuits anymore in an out-of-the-box fashion. Afterwards, we presented dedicated schemes that eventually allow to treat the involved circuits as if they did not contain non-unitaries at all. More precisely, the usage of established verification techniques for dynamic quantum circuits is enabled by handling non-unitaries either through

(1) transforming the dynamic circuit primitives by substituting reset operations with "fresh" qubits and applying the deferred measurement principle (see Section 4), or
(2) using classical simulation techniques to extract the distribution of measurement outcomes from a dynamic quantum circuit—given a fixed input (see Section 5).

These schemes form a generic solution for handling non-unitaries in verifying the equivalence of quantum circuits that, as demonstrated in our evaluations, is applicable to any existing verification framework.

Acknowledgements

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 BMWK on the basis of a decision by the German Bundestag through project QuaST.

REFERENCES

[1] A. W. Cross *et al.*, *OpenQASM 3: A broader and deeper quantum assembly language*, 2021. arXiv: 2104.14722.
[2] IBM Quantum, "Quantum circuits get a dynamic upgrade with the help of concurrent classical computation," 2021. [Online]. Available: https://www.ibm.com/blogs/research/2021/02/quantum-phase-estimation/.
[3] S. Yamashita and I. L. Markov, "Fast equivalence-checking for quantum circuits," in *Int'l Symp. on Nanoscale Architectures*, 2010.
[4] L. Burgholzer and R. Wille, "Advanced equivalence checking for quantum circuits," *IEEE Trans. on CAD of Integrated Circuits and Systems*, 2021.
[5] L. Burgholzer, R. Raymond, and R. Wille, "Verifying results of the IBM Qiskit quantum circuit compilation flow," in *Int'l Conf. on Quantum Computing and Engineering*, 2020.
[6] G. F. Viamontes, I. L. Markov, and J. P. Hayes, "Checking equivalence of quantum circuits and states," in *Int'l Conf. on CAD*, 2007.
[7] P. Niemann, R. Wille, and R. Drechsler, "Equivalence checking in multi-level quantum systems," in *Int'l Conf. of Reversible Computation*, 2014.
[8] 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*, 2008.
[9] K. N. Smith and M. A. Thornton, "A quantum computational compiler and design tool for technology-specific targets," in *Int'l Symp. on Computer Architecture*, 2019.
[10] M. Amy, "Towards large-scale functional verification of universal quantum circuits," in *International Conference on Quantum Physics and Logic*, 2019.
[11] X. Hong *et al.*, *A tensor network based decision diagram for representation of quantum circuits*, 2020. arXiv: 2009.02618.
[12] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information*. Cambridge University Press, 2010.
[13] A. Barenco *et al.*, "Elementary gates for quantum computation," *Phys. Rev. A*, 1995.
[14] P. W. Shor, "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer," *SIAM J. Comput.*, 1997.
[15] A. W. Harrow, A. Hassidim, and S. Lloyd, "Quantum algorithm for linear systems of equations," *Phys. Rev. Lett.*, 2009.
[16] S. Lloyd, M. Mohseni, and P. Rebentrost, "Quantum principal component analysis," *Nat. Phys.*, 2014.
[17] D. Maslov, "On the advantages of using relative phase Toffolis with an application to multiple control Toffoli optimization," *Phys. Rev. A*, 2016.
[18] R. Wille, M. Soeken, C. Otterstedt, and R. Drechsler, "Improving the mapping of reversible circuits to quantum circuits using multiple target lines," in *Asia and South Pacific Design Automation Conf.*, 2013.
[19] A. M.-v. de Griend and R. Duncan, *Architecture-aware synthesis of phase polynomials for NISQ devices*, 2020. arXiv: 2004.06052.
[20] M. Y. Siraichi, V. F. dos Santos, S. Collange, and F. M. Q. Pereira, "Qubit allocation," in *Int'l Symp. on Code Generation and Optimization*, 2018.
[21] A. Zulehner, A. Paler, and R. Wille, "An efficient methodology for mapping quantum circuits to the IBM QX architectures," *IEEE Trans. on CAD of Integrated Circuits and Systems*, 2019.
[22] R. Wille, L. Burgholzer, and A. Zulehner, "Mapping quantum circuits to IBM QX architectures using the minimal number of SWAP and H operations," in *Design Automation Conf.*, 2019.
[23] G. Li, Y. Ding, and Y. 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.
[24] S. Sivarajah *et al.*, "T|ket>: A retargetable compiler for NISQ devices," *Quantum Sci. Technol.*, 2020.
[25] T. Itoko, R. Raymond, T. Imamichi, and A. Matsuo, "Optimization of quantum circuit mapping using gate transformation and commutation," *Integration*, 2020.
[26] G. Vidal and C. M. Dawson, "Universal quantum circuit for two-qubit transformations with three controlled-NOT gates," *Phys. Rev. A*, 2004.
[27] K. Hietala *et al.*, *A verified optimizer for quantum circuits*, 2019. arXiv: 1912.02250.
[28] D. Janzing, P. Wocjan, and T. Beth, ""Non-identity check" is QMA-complete," *Int. J. Quantum Inform.*, 2005.
[29] C. H. Bennett *et al.*, "Teleporting an unknown quantum state via dual classical and Einstein-Podolsky-Rosen channels," *Phys. Rev. Lett.*, 1993.
[30] M. Dobsicek, G. Johansson, V. S. Shumeiko, and G. Wendin, "Arbitrary accuracy iterative phase estimation algorithm as a two qubit benchmark," *Phys. Rev. A*, 2007.
[31] A. D. Corcoles *et al.*, *Exploiting dynamic quantum circuits in a quantum algorithm with superconducting qubits*, 2021. arXiv: 2102.01682.
[32] M. Ying, "Floyd–hoare logic for quantum programs," *ACM Trans. Program. Lang. Syst.*, 2011.
[33] *Foundations of Quantum Programming*. 2016.
[34] Q. Wang, R. Li, and M. Ying, *Equivalence checking of sequential quantum circuits*, 2021. arXiv: 1811.07722.
[35] X. Hong, Y. Feng, S. Li, and M. Ying, *Equivalence checking of dynamic quantum circuits*, 2021. arXiv: 2106.01658.
[36] A. Zulehner and R. Wille, "Advanced simulation of quantum computations," *IEEE Trans. on CAD of Integrated Circuits and Systems*, 2019.
[37] G. G. Guerreschi, J. Hogaboam, F. Baruffa, and N. P. D. Sawaya, "Intel Quantum Simulator: A cloud-ready high-performance simulator of quantum circuits," *Quantum Sci. Technol.*, 2020.
[38] B. Villalonga *et al.*, "A flexible high-performance simulator for verifying and benchmarking quantum circuits implemented on real hardware," *Npj Quantum Inf.*, 2019.
[39] G. F. Viamontes, I. L. Markov, and J. P. Hayes, "Graph-based simulation of quantum computation in the density matrix representation," in *Quantum Information and Computation II*, 2004.
[40] T. Grurl, J. Fuß, and R. Wille, "Considering decoherence errors in the simulation of quantum circuits using decision diagrams," in *Int'l Conf. on CAD*, 2020.
[41] A. Li, O. Subasi, X. Yang, and S. Krishnamoorthy, "Density matrix quantum circuit simulation via the BSP machine on modern GPU clusters," in *Int'l Conf. for High Performance Computing, Networking, Storage and Analysis*, 2020.
[42] L. Burgholzer and R. Wille, "QCEC: A JKQ tool for quantum circuit equivalence checking," *Softw. Impacts*, 2021.
[43] E. Bernstein and U. Vazirani, "Quantum complexity theory," *SIAM J. Comput.*, 1997.
[44] P. Nation and B. Johnson. "How to measure and reset a qubit in the middle of a circuit execution," IBM Research Blog. (2021), [Online]. Available: https://www.ibm.com/blogs/research/2021/02/quantum-mid-circuit-measurement/.
[45] R. B. Griffiths and C.-S. Niu, "Semiclassical Fourier transform for quantum computation," *Phys. Rev. Lett.*, 1996.