VeriQBench: A Multi-Type Circuit Benchmark for Quantum Hardware and Formal Verification Tools

Kean Chen[1,2,3], Wang Fang[4,2,3], Ji Guan[2], Xin Hong[2], Mingyu Huang[2,3], Riling Li[2], Junyi Liu[7,2,3], Qisheng Wang[4,8,5], Mingsheng Ying[6]

[1] School of Engineering and Applied Science, University of Pennsylvania, United States
[2] Institute of Software, Chinese Academy of Sciences, China
[3] University of Chinese Academy of Sciences, China
[4] School of Informatics, University of Edinburgh, United Kingdom
[5] Department of Computer Science and Technology, Tsinghua University, China
[6] Centre for Quantum Software and Information, University of Technology Sydney, Australia
[7] Joint Center for Quantum Information and Computer Science, University of Maryland, United States
[8] Graduate School of Mathematics, Nagoya University, Japan

Abstract. Existing quantum circuit benchmarks predominantly assess hardware and software performance along two dimensions, qubit number and gate count, and focus mainly on combinational circuits. This severely limits their usefulness for evaluating formal verification and analysis tools, which must handle advanced circuit types now supported by contemporary platforms, including dynamic circuits with mid-circuit measurements and feedback, sequential circuits with loops, and variational circuits used in hybrid quantum–classical algorithms. We introduce *VeriQBench*, a publicly available quantum circuit benchmark specifically designed to support the development and evaluation of quantum formal methods tools. *VeriQBench* systematically incorporates combinational, dynamic, sequential, and variational circuits, spans problem sizes from small to medium–large scales, and is distributed in OpenQASM together with scripts for generating scalable instances and converting to common verification formats. *VeriQBench* increases circuit coverage by 26% over state-of-the-art benchmarks, indicating substantially improved structural diversity beyond qubit and gate counts. We demonstrate the utility of *VeriQBench* through two case studies: benchmarking quantum circuit equivalence-checking tools across different circuit types, and benchmarking the noise behaviour of superconducting quantum devices. The observed variation in verification performance and noise characteristics across circuit types shows that type diversity is a crucial dimension for assessing and designing reliable quantum systems, and *VeriQBench* provides a practical benchmark resource for the FM community.

Keywords: Quantum Computing, Quantum Circuits, Benchmarking, Equivalence Checking, Sequential Quantum Circuits

1 Introduction

Quantum circuits play a vital role in quantum computing by serving as the foundation for encoding, processing, and manipulating quantum information to accomplish computational tasks on quantum computers. Recently, researchers have developed various methods based on quantum circuits to assess and evaluate quantum hardware and software. These methods typically involve implementing and simulating quantum circuits on quantum and classical computers, respectively, to analyze factors such as quantum hardware noise levels [53] and the performance of quantum circuit equivalence-checking software tools [88,95,18,17,43].

In order to standardize the quantum circuit-based methods used for characterizing quantum systems, a variety of quantum circuit benchmarks have been introduced to establish a consistent evaluation framework. Each benchmark is accompanied by specific evaluation metrics, such as quantum volume, fidelity, and Q-score, tailored to different purposes and is based on quantum circuits. Examples of such benchmarks include QASMBench [50], the quantum LINPACK benchmark [31], and SupermarQ [87]. These benchmarks encompass circuits from various domains, such as random protocols [8], quantum algorithms [64,24], variational quantum algorithms [70,33,40,86,20], and classical reversible circuits [2]. Primarily, these benchmarks consist of quantum circuits with varying complexities in terms of qubit number (the number of quantum bits) and gate number (the number of quantum gates), aiming to assess the performance of quantum hardware and software in handling increasingly intricate circuits.

While existing benchmarks primarily concentrate on static or combinational quantum circuits (fixed sequences of quantum gate operations), the evolution of quantum devices has introduced advanced circuit forms like dynamic and sequential quantum circuits, and correspondingly, quantum software tools have been developed to optimize and verify these circuit types, especially in the field of quantum chip design automation. These new circuit types support time-dependent operations and adaptive logic, enabling the implementation of more complex quantum algorithms. Specifically, dynamic quantum circuits, which incorporate measurements and classical feedback during execution, offer adaptive behaviour, which supports the implementation of quantum teleportation and phase estimation algorithms, and sequential circuits involve looping and state-dependent processes crucial for algorithms such as quantum walks or iterative quantum algorithms. These advanced features must be taken into account when characterizing or benchmarking quantum hardware and software. Therefore, there is a growing need to expand benchmark frameworks beyond traditional complexity metrics (qubit number and gate number) to accommodate the structural diversity of quantum circuits.

To address this gap, we introduce *VeriQBench*, an extensive quantum circuit benchmark designed to enhance existing benchmarks by incorporating various circuit types while maintaining current circuit complexity in terms of qubit number and gate number. *VeriQBench* covers four main categories of circuit types:

1. **Combinational Quantum Circuits** – These circuits follow predetermined sequences of gate operations, representing fundamental quantum algorithms.
2. **Dynamic Quantum Circuits** – These circuits involve intermediate measurements and feedback mechanisms, reflecting hybrid quantum-classical protocols.
3. **Sequential Quantum Circuits** – These circuits execute time-dependent processes with feedback loops, enabling more advanced quantum algorithms such as quantum walks and repeat-until-success.
4. **Variational Quantum Circuits** – Also known as "parameterized quantum circuits," these circuits support optimization-based algorithms used in quantum machine learning and variational quantum eigensolvers.

As a result of integrating the multi-type circuit approach, *VeriQBench* achieves an improvement of at least 26% in the circuit coverage [87] compared to the state-of-the-art quantum circuit benchmarks, such as SupermarQ [87], Benchpress [47], and QASMBench [50].

VeriQBench is designed to support a range of evaluation metrics (e.g., fidelity and runtime) for the assessment of both quantum hardware (e.g., superconducting, neutral atom, photonic chips) and software (e.g., equivalence checking, noise mitigation, optimization tools) systems that utilize quantum circuits. In addition to accommodating various types of quantum circuits (combinational, dynamic, sequential, and variational), *VeriQBench* includes several practical features to enhance its effectiveness:

- **Representivity:** The circuits in *VeriQBench* are collected from research in a variety of fields, such as hardware performance evaluation [54], quantum algorithm design [80], equivalence checking [43,89], circuit testing [22], and circuit optimizing [7,63]. This diverse selection of circuits enhances the benchmark's practical applicability and adaptability. Most of the combinational quantum circuits have been involved in the existing quantum circuit benchmarks.
- **Scalability:** The benchmark comprises circuits ranging from 2 qubits to over 50 qubits, classified into small-scale (<20 qubits), medium-scale (20-50 qubits), and large-scale (>50 qubits) categories. This diversity in circuit size results in varying qubit numbers and gate numbers (which increase with qubit numbers), keeping the circuit complexity considered in the existing benchmarks
- **Usability:** All circuits in the benchmark are defined in the OpenQASM quantum assembly language and are provided as ".qasm" files. Moreover, most circuits can be converted to other formats such as Q#, PyQuil, Cirq, etc., using the online *q-convert* tool accessible at: `http://quantum-circuit.com/qconvert`.
- **Evolvability:** Users are offered scripts to create quantum circuits with different qubit numbers within the benchmark, ensuring adaptability as quantum technology advances.

To ensure the credibility of our benchmark, all quantum circuits included in it have been executed and verified using simulators from Qiskit [72], an open-source software toolkit for quantum computing, and QCOR [59], a programming language and compiler tailored for the hybrid quantum-classical computation model. Moreover, we employ our benchmark to assess the performance of quantum software and hardware in equivalence checking and practical use, respectively. The results of the above experiments show that the above software and hardware systems exhibit variations in performance across different types of quantum circuits, with some unable to cope with more complex circuit types, such as sequential circuits. This underscores the importance of incorporating circuit types as a metric dimension in quantum circuit benchmarking, a feature that is integrated within our benchmark, *VeriQBench*. The *VeriQBench* is publicly available at `https://github.com/Veri-Q/Benchmark`.

1.1 Related Work

In both classical and quantum computing, benchmarks serve as essential tools for evaluating hardware and software performance across a range of metrics and applications.

Classical Multi-type Circuit Benchmarks: In classical computing, multi-type circuit benchmarks have long provided standardized methods for comprehensive testing. By covering combinational, sequential, and dynamic circuits, these benchmarks enable a thorough evaluation of both functional and performance aspects, addressing a range of computational demands from static logic operations to adaptive and time-dependent processes. For instance, ISCAS benchmarks — including ISCAS'85 [41] for combinational circuits and ISCAS'89 [16] for sequential circuits — offer foundational circuit collections for assessing digital circuit reliability, fault tolerance, and synthesis optimization. These suites evaluate performance in both fixed operations (combinational) and state-dependent operations (sequential), providing insights into hardware handling loops, feedback, and memory elements. The ITC'99 suite [29] extends this by including testable sequential circuits with scan chains, designed specifically for testing and debugging tools in complex sequential logic circuits.

More recent benchmarks, like the EPFL combinational suite [5], focus primarily on logic synthesis optimization for combinational circuits, while LGSynth'91 [96] covers mixed circuit types, integrating high-level functional descriptions with detailed structural elements. These benchmarks address diverse computational needs, including loop handling and adaptive mechanisms, essential for time-dependent computations and applications involving sequential processes.

Quantum Circuit Benchmarks: In quantum system benchmarking, circuit-based methods similarly provide critical evaluation metrics for quantum systems, generally divided into two main categories: benchmarks focused on assessing quantum hardware capabilities and those that offer a versatile collection of circuits for general-purpose testing.

For hardware benchmarking, Thomas et al. [54] introduced a benchmark suite using circuits like Grover's algorithm to determine Quantum Volume, offering a standardized metric to assess a device's processing power. Alternatively, SupermarQ by Tomesh et al. [87] evaluates specific quantum applications, such as the Quantum Approximate Optimization Algorithm (QAOA) and Variational Quantum Eigensolver (VQE). Rudi et al. [34] take a practical approach by benchmarking industrial tasks such as robot path and vehicle routing optimizations, comparing quantum and classical solutions across different infrastructure types to highlight real-world hardware performance. While these benchmarks provide valuable insights, they focus primarily on individual aspects of hardware performance and lack a broader framework suitable for evaluating complex software stacks.

Additionally, several studies have compiled circuit collections aimed at general-purpose benchmarking. RevLib [92] offers a suite of reversible circuits foundational for basic quantum studies, while QASMBench [50] provides OpenQASM-based circuits that are versatile and accessible for diverse applications. MQT-Bench [75] spans various abstraction levels, from functional descriptions to hardware implementations, and LIMPACK [31] focuses on generating circuits for solving systems of linear equations.

These benchmark suites collectively offer broad evaluations across multiple use cases, yet they predominantly focus on qubit number and gate number as complexity metrics, overlooking the importance of circuit type. With advances in quantum hardware, including support for dynamic and sequential circuits on platforms like IBM's Qiskit, it is essential to incorporate these advanced circuit types. Dynamic and sequential circuits enable time-dependent operations and adaptive logic, which are essential for implementing complex algorithms like quantum walks and iterative processes, thus making circuit type a crucial dimension in quantum benchmarking.

To address these requirements, we propose *VeriQBench*, a quantum circuit benchmark incorporating combinational, dynamic, sequential, and variational circuits, covering all three complexity dimensions: circuit type, qubit number, and gate number. By integrating current evaluation metrics such as fidelity and runtime, *VeriQBench* complements existing benchmarks, providing a more comprehensive understanding of quantum system performance. For a detailed comparison of supported circuit types across current benchmarks and *VeriQBench*, see Table 1.

Supported Circuit Types	**Combinational**	**Dynamic**	**Sequential**	**Variational**
Revlib[92]	✓	×	×	×
LINPACK[31]	✓	×	×	×
SupermarQ[87]	✓	✓	×	✓
Quark[34]	✓	×	×	✓
Application-Oriented[54]	✓	✓	×	✓
QASMBench[50]	✓	×	×	✓
MQTBench[75]	✓	×	×	✓
VeriQBench	✓	✓	✓	✓

Table 1. Comparison between existing quantum circuit benchmarks and *VeriQBench*.

2 VeriQBench

In this section, we elaborate on the introduction of our quantum circuit benchmark *VeriQBench* in four parts: combinational quantum circuits, dynamic quantum circuits, sequential quantum circuits, and variational quantum circuits. Table 2 displays the quantum circuits included in the benchmark, categorized by circuit types. We have made available the OpenQASM files for these circuits at various scales, and additionally, you can create circuits with a custom number of qubits for most of them using the scripts provided in our repository.

In the following part, we introduce these circuits in more detail.

Combinational	**Dynamic**	**Sequential**	**Variational**
Quantum Algorithms	*Teleportation*	*Repeat-Until-Success*	*Eigensolver*
(bv,qft,grover,adder...)	*Fourier Transform*	*Quantum Walk*	*QAOA*
Reversible Circuits	*Phase Estimation*	*Quantum Control*	*QCNN*
Qubit Mapping	*Error Correction*		*Efficient SU2*
Random Circuits	*State Injection*		

Table 2. List of quantum circuits in *VeriQBench* categorized by circuit types.

2.1 Combinational Quantum Circuits

Combinational quantum circuits refer to quantum circuits that follow predetermined sequences of gate operations without intermediate measurements or feedback, serving as the foundation for implementing fundamental quantum algorithms. The combinational quantum circuits in *VeriQBench* cover core fundamental scenarios of quantum computing, encompassing four major categories: quantum algorithms (Bernstein-Vazirani algorithm [11], Quantum Fourier Transform [64], Phase Estimation [64], Grover's algorithm [38], quantum adder [21], etc.), reversible circuits [56], dedicated circuits for qubit mapping, and random circuits (random Clifford circuits [14,1,15], quantum volume circuits [30], quantum supremacy-related circuits [13]). These selected circuits are derived from key research fields such as quantum algorithm design and hardware performance evaluation. They not only include classical combinational circuit types from existing benchmarks but also supplement practical scenario requirements like qubit mapping, achieving both completeness and representativeness. This provides a comprehensive and representative test carrier for evaluating the performance of quantum hardware and software in scenarios involving basic fixed logic operations

2.2 Dynamic Quantum Circuits

Dynamic quantum circuits are quantum computation models that enable flexible hybrid quantum-classical execution, characterized by mid-circuit measurements and classical feedback mechanisms to adjust subsequent quantum operations—an essential form for reducing quantum resource costs on NISQ devices.

Experiments in [27] show that dynamic quantum circuits can indeed offer a 'substantial and tangible advantage' on current noisy quantum hardware. [94] also shows that dynamic quantum circuits can be used to tackle real-world problems, such as portfolio optimization problems, on near-term quantum hardware. In addition, software platforms such as IBM's Qiskit, Google's Cirq, and Amazon's Braket all added their support for dynamic quantum circuits [78].

The dynamic quantum circuits in *VeriQBench* focus on hybrid quantum-classical computing scenarios, consisting of five core types: quantum teleportation [9], semiclassical Fourier transform [37], iterative phase estimation [27], quantum error correction [64], and state injection [76]. These circuits are core carriers for implementing key algorithms such as quantum teleportation and phase estimation, and have been verified for support in existing quantum software platforms (e.g., Qiskit, Cirq). The selection of circuits not only covers the core technical paths of dynamic quantum computing but also includes typical scenarios in practical applications. Their completeness meets the performance evaluation needs of hybrid quantum-classical protocols, while their representativeness ensures the reference value of test results for real-world scenarios

2.3 Sequential Quantum Circuits

Sequential quantum circuits are advanced quantum circuits integrated with clock signals, featuring time-dependent processes and feedback loops—they can be seen as a generalization of classical synchronous logic circuits and are critical for algorithms with loops [55,89]. The sequential quantum circuits in *VeriQBench* target advanced quantum algorithms with loops and feedback, including three major categories: Repeat-Until-Success circuits [66,89], quantum walk circuits [3,49,6], and classical control circuits [89,90]. These circuits introduce clock signals and state-dependent processes, acting as core implementation forms for complex applications such as quantum walks and iterative quantum algorithms, and have been experimentally verified on hardware platforms like superconducting quantum processors and photonic chips [60,99]. The selection of circuits not only covers the core technical scenarios of sequential quantum computing but also aligns with the support directions of quantum programming frameworks (e.g., isQ [39], QPanda [32] and Qiskit [72]). Combining completeness and representativeness, they provide a key test basis for evaluating the ability of quantum hardware and software to handle time-dependent quantum logic

2.4 Variational Quantum Circuits

Variational quantum circuits (also known as parameterized quantum circuits) are quantum circuits with adjustable parameters, where parameters are optimized by classical algorithms to support optimization-based tasks—they are the core of hybrid quantum-classical algorithms like quantum machine learning and quantum chemistry calculations. The variational quantum circuits in *VeriQBench* focus on optimization-based quantum algorithms, including four major categories: Variational Quantum Eigensolver (VQE) [86], Quantum Approximate Optimization Algorithm (QAOA) [33], Quantum Convolutional Neural Network (QCNN) [26], and Efficient SU2 ansatz. These circuits are core tools in fields such as quantum machine learning and quantum chemistry computing, and their parameter optimization characteristics pose unique challenges to the adaptability and performance of quantum hardware and software. The selection of circuits covers mainstream technical routes and typical application scenarios of variational quantum computing, reflecting comprehensive coverage of current research hotspots in quantum algorithms while ensuring the representativeness of test scenarios. This provides a complete and reasonable test set for evaluating the performance of quantum systems in optimization-driven tasks.

3 Credibility

In this section, we introduce the procedure of validating the credibility of all quantum circuits in *VeriQBench*. All our benchmark circuits are validated by the simulators from Qiskit [72] and QCOR [59].

For OpenQASM 2.0 descriptions, Qiskit's simulator is used. To run an example named `in_file.qasm`, simply navigate to its directory and execute the codes in Python:

```
qc = qiskit.QuantumCircuit.from_qasm_file('./in_file.qasm')
simulator = qiskit_aer.Aer.get_backend('qasm_simulator')
qc = qiskit.transpile(qc, simulator)
result = simulator.run(qc, shots=1, memory=True).result()
```

For OpenQASM 3.0 descriptions, QCOR is used. To run an example named `in_file.qasm`, simply navigate to its directory and execute:

```
qcor --shots 1024 in_file.qasm
./a.out
```

The simulation can be accelerated using TNQVM [58], which leverages tensor network theory to simulate quantum circuits. To run the above example with TNQVM acceleration, simply execute:

```
qcor --qpu tnqvm[tnqvm-visitor:exatn]
--shots 1024 in_file.qasm
./a.out
```

For more details, we refer the readers to the official documentation and user guides [4].

4 Coverage

In this section, we detail the quantitative advantages of VeriQBench in evaluating quantum systems through circuit coverage [87]. VeriQBench outperforms other benchmarks by achieving a 26% higher coverage, leading to a more comprehensive evaluation. Various circuit types exhibit strengths in different areas that enhance circuit coverage, facilitating a holistic assessment that encompasses a wide range of quantum system characteristics.

In evaluating a circuit, [87] considered six features: program communication (PC), critical-depth (CD), entanglement-ratio (Ent), parallelism (Par), liveness (Liv), and mid-circuit measurement (Mea). Each feature is represented by a value between 0 and 1. By examining the six-dimensional feature vectors of different circuits, distinct characteristics of each circuit can be observed. The coverage, defined as the volume of the convex hull created by these feature vectors across all circuits, serves as a crucial benchmark indicator. A higher coverage value indicates a more comprehensive evaluation of quantum software or hardware.

In Fig. 1, we computed the average feature vectors for each circuit type and presented them in a radar chart. The chart illustrates how distinct circuit types excel in various features. For instance, combinational quantum circuits tend to exhibit higher PC and Ent values on average, whereas dynamic quantum circuits excel in mid-circuit measurement (Mea). Sequential quantum circuits showcase the highest CD value, and variational quantum circuits demonstrate enhanced parallelism (Par). This suggests that different circuit types yield varying impacts on the evaluation of quantum software and hardware. To further elucidate the characteristics of each circuit type, we have identified circuits that attain the maximum value in each dimension for every circuit type and depicted them in radar graphs in Fig. 2 (a)-(d) in the appendix. For example, 'bv_2' has the biggest PC value among all combinational circuits in *VeriQBench*, so we display it in 2 (a) with the blue line.

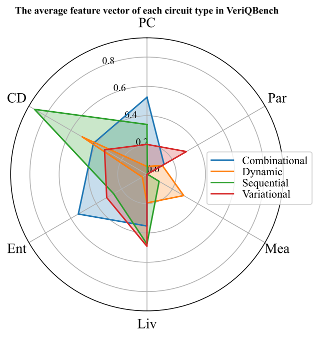


Fig. 1. The average feature vectors of each type of circuit.

Furthermore, we assessed the circuit coverage in *VeriQBench* and compared it with various widely recognized benchmarks. The findings are presented in Table 3. The coverage achieved by *VeriQBench* is 1.14e-02, surpassing the state-of-the-art benchmarks by at least 26%. Additionally, we conducted an analysis of the coverage for each circuit type within *VeriQBench*. The four circuit

Suite	Volume	Circuits
VeriQBench (this work)	1.14e-02	822
SupermarQ [87]	9.0e-03	52
Benchpress[47]	6.8e-03	655
QASMBench [50]	4.0e-03	62
CBG2021 [28]	1.6e-08	10476
MQTBench [75]	9.0e-10	1942
TriQ [61]	4.1e-14	12
PPL+2020 [68]	1.0e-15	9

Table 3. Coverage comparison of different benchmark suites.

Example Applications	Quantum Circuits Used in *VeriQBench*
Equivalence Checking [18,42,89]	QFT, Grover, Dynamic (Sec. 2.2), Sequential (Sec. 2.3)
Fault Simulation [45]	BV, QFT, VQE, QAOA
Circuit Optimization [63,7]	QFT, Reversible (Sec. 2.1)
ATPG [10,22]	Grover, BV, QFT, QV
Qubit Mapping [84,83]	Generated Circuits (Sec. 2.1)
Model Checking [35]	Teleportation
Hardware Performance [54,87]	Grover, QAOA, VQE

Table 4. Example applications.

types—combinational, dynamic, sequential, and variational—exhibited coverages of 1.13e-09, 7.71e-06, 7.87e-08, and 9.71e-15, respectively. It is evident from the data that individual circuit types exhibit relatively low coverage values; however, when amalgamated, they collectively contribute to a higher coverage rate. This outcome is attributed to the distinct positioning of each circuit type within the feature vector space.

5 Applications

In this section, we demonstrate practicality through the exploration of various example applications and emphasize the use of benchmark circuits from *VeriQBench*. A brief summary of these applications is provided in Table 4. Subsequently, we utilize *VeriQBench* to further validate its applicability by assessing the effectiveness of tools for verifying quantum circuit equivalence and evaluating the performance of superconducting quantum processors. The empirical findings suggest that these quantum systems demonstrate diverse performance across different circuit types, thus validating that the unique feature of *VeriQBench* introduces a variety of quantum circuits for benchmarking purposes.

Equivalence Checking of Quantum Circuits: Equivalence checking is to check if two quantum circuits are functionally equivalent [88,95,18,17,42,43,89]. The circuits for quantum Fourier transform and Grover's algorithms have been used in work [18] to demonstrate the effectiveness of their equivalence checking methods. Also, the circuits given in Section 2.2 have been used in testing the algorithm developed in [42] for checking the equivalence of dynamic quantum circuits, and the circuits such as Bernstein-Vazirani and quantum Fourier transform and quantum volume have been used in the approximate equivalence checking of noisy quantum circuits [43]. The circuits given in Section 2.3 have been used in equivalence checking of sequential quantum circuits [89].

Fault Simulation of Quantum Circuits: Given a quantum circuit and its faulty verision, fault simulation is to calculate the fidelity between the expected output state and the actually obtained output state [46,23,48,45]. The work is performed on combinational quantum circuits, including Bernstein-Vazirani and quantum Fourier transform with realistic fault models proposed in [44].

Circuit Optimization: Quantum circuit optimization [57,71,77,7,63] aims to reduce the complexity of a quantum circuit, where the complexity, depending on the scenario, can be quantified by the size, depth, T-count or T-depth of the quantum circuit. The reversible circuits, such as divisibility checkers, have been used to benchmark the quantum circuit optimization algorithms [63,7].

Automatic Test Pattern Generation (ATPG): Quantum ATPG algorithms [67,10,22] aim to generate specific test patterns based on the structure and fault model of the quantum circuit, where the test patterns include a set of input vectors and corresponding outputs vectors. The combinational quantum circuits, including Grover's algorithm, Bernstein-Vazirani, quantum Fourier transform and quantum volume with the unitary fault model, have been used to demonstrate the effectiveness of the ATPG algorithms in [10,22].

Qubit Mapping: The target of qubit mapping is to assign a physical qubit on a quantum chip to every logical qubit in a quantum circuit while optimizing the performance of the circuit [57,79,52,93,69,81,100,25,91,51,12,62,85,82]. One of the most important performance metrics of quantum circuits is the depth of the circuit. The method for generating qubit mapping problems with optimal solutions for depth in this benchmark was proposed in [84], and was used to evaluate the qubit mapping tools in [83].

Model Checking Quantum Systems: Model-checking is one of the most successful verification techniques with numerous applications in hardware and software industries. It has been generalized to check various properties of quantum systems, including quantum cryptographic protocols [36], quantum programs [98], and quantum circuits [97]. The teleportation circuit in Section 2.2 has been used in the probabilistic model checking algorithm of quantum protocols [35]. *Evaluating Hardware Performance*: Evaluating the performance of hardware is very important for quantum devices, especially in the current NISQ (noisy intermediate-scale quantum) computing era. Combinational quantum circuits such as Grover's algorithm have been used to determine the quantum volume of quantum devices [54], and Variational quantum circuits, including Quantum Approximate Optimization Algorithm (QAOA) and variational Quantum Eigensolver (VQE), have been used in evaluating the performance of quantum hardware on specific quantum applications [87].

The above applications further validate the representativeness of quantum circuits within our benchmark, as some of these applications have employed them. Nevertheless, these applications have not previously regarded circuit type as a critical parameter for assessing the performance of quantum systems. In the following discussion, we will investigate the significant role of circuit type in benchmarking quantum systems, with examples such as equivalence checking of quantum circuits and evaluating hardware performance.

5.1 Equivalence Checking

In this section, we present examples illustrating how *VeriQBench* can be utilized to benchmark equivalence-checking tools in quantum computing in greater depth. We evaluate QCEC [19] and TDD-based [42] approaches for combinational, dynamic, and variational quantum circuits. For sequential quantum circuits, as only one method is proposed in [89], we adopt the tensor network-based method as a reference point. All experiments are conducted on a computer equipped with an i9-13900H processor and 64GB of RAM.

Circuit Type	Circuit Name	Qubit Number	Gate Number	QCEC	TDD
combinational	adder_n31	31	40	0.07	0.2
	bv_31	31	93	0.05	0.18
	grover_11	11	50	0.033	0.37
	pe_10	11	75	0.026	0.001
	qft_10	10	55	0.035	0.49
	16QBT_16CYC_32GN_1.0P2_0	16	32	0.038	0.12
	qv_n10_d10	10	511	0.05	>300
	rand_cliff_10_AG	10	134	0.047	4.54
	rd84d1	15	28	0.037	0.001
dynamic	dqc_pe_10	11	85	0.034	0.054
	dqc_qft_10	10	65	0.033	0.047
	dqc_bitflip_code	6	12	0.021	0.002
	dqc_phaseflip_code	6	18	0.027	0.008
	dqc_state_injection_S	2	5	0.007	0.001
	dqc_state_injection_T	2	5	0.009	0.001
	dqc_teleportation	3	8	0.015	0.005
variational	HEA_5	5	110	0.033	4.53
	SPA_5	5	58	0.031	0.095
	UCC_5	5	833	0.045	2.37
	qaoa_6	6	110	0.037	0.058
	hf_6_0_5	6	155	0.037	0.16

Table 5. Equivalence checking of combinational, dynamic, and variational quantum circuits.

Combinational, Dynamic, and Variational Quantum Circuits
We choose several circuits from every category and then compile them through the Qiskit Transpiler according to the *qasm_simulator* backend in Qiskit Aer. We check the equivalence of the circuits before and after the compilation using QCEC and TDD. Since the maximum number of qubits supported by the Qiskit Transpiler is 31, we only test circuits under 31 qubits. The corresponding experimental data is shown in Table 5. In this table, we give the circuit type, name, qubit number, and gate number of the circuit in the first four columns,

and then we record the time (seconds) of checking their equivalence using QCEC and TDD in the last two columns. For TDD, we simply calculate the decision diagram representation of the two circuits and compare them, except that the same quantum gates that appeared at the beginning of the two circuits will be omitted.

From the table, we can see that QCEC and TDD perform differently for different circuit types. For combinational quantum circuits and variational quantum circuits, QCEC is faster than TDD in most cases. But for dynamic quantum circuits, TDD outperforms QCEC on average. Both QCEC and TDD fail to support the equivalence checking of sequential quantum circuits. This underscores the significance of considering circuit type as a crucial factor when evaluating equivalence-checking tools.

Sequential Quantum Circuits
For sequential quantum circuits, we also choose several circuits and then construct Mealy machines and check their equivalence as defined in [89]. The method of [89] (sqcircuit, implemented in Python with Numpy module), and a tensor network-based method are compared. The corresponding experimental data is shown in Table 6. In this table, we record the number of gates in the loop. From the table, we can see that when the number of qubits is small, sqcircuit is normally faster than the tensor network-based method, but when the number of qubits grows, the tensor network-based method outperforms the sqcircuit.

Circuit Type	Circuit Name	Qubit Number	Gate Number (loop)	sqcircuit	TN
sequential	qft-small	6	10	0.5	0.61
	qft-large	9	28	43.99	25.96
	qwalk-small	4	7	0.14	0.23
	qwalk-large	9	15	>300	0.27
	qrus-1	2	12	0.01	0.02
	qctrl-1	6	13	2.07	2.79

Table 6. Equivalence checking of sequential quantum circuits.

5.2 Hardware Experiments

Subsequently, we evaluated diverse superconducting quantum processors by executing circuits in *VeriQBench* and quantifying the fidelity between the actual output and ideal output state. The processors experimented include Xiaohong from QuantumCTek [74], Dongling from BAQIS [65], ibm_brisbane, ibm_kyiv and ibm_sherbrooke from IBM [73].

We evaluate three types of quantum circuits—combinational, variational, and dynamic—using a diverse set of benchmarks: a 5-qubit Bernstein-Vazirani circuit (*bv_5*), a 5-qubit Grover circuit with 3 iterations (*grover_5*), a 7-qubit phase estimation circuit (*pe_6*), a 10-qubit adder (*adder_10*), 4-qubit variational circuits (*eff_k_4*, k=1,. . . ,3) based on Qiskit's *EfficientSU2* ansatz, a 4-qubit quantum convolutional neural network (*qcnn_4*), dynamic phase estimation circuits (*dqc_pe_10*, *dqc_pe_12*, *dqc_pe_15*), and a 6-qubit bit-flip code circuit

Type	Circuit Name	Qubit Number	Gate Number			Xiaohong	Dongling	ibm_brisbane	ibm_kyiv	ibm_sherbrooke
			Original	Xiaohong Dongling ISA	IBM ISA					
combinational	bv_5	5	15	14	73	0.68	0.48	0.62	0.83	0.85
	grover_5	5	62	185	1269	0.13	0.16	0.13	0.11	0.13
	pe_6	7	40	113	604	0.15	0.11	0.02	0.01	0.00
	adder_10	10	12	72	442	0.31	0.19	0.05	0.01	0.06
variational	eff_1_4	4	19	19	51	0.80	0.61	0.72	0.95	0.88
	eff_2_4	4	19	19	51	0.77	0.64	0.78	0.93	0.87
	eff_3_4	4	19	19	45	0.82	0.66	0.79	0.91	0.87
	qcnn_4	4	66	43	243	0.69	0.66	0.74	0.63	0.70
dynamic	dqc_pe_10	11	76	-	225	-	-	0.45	0.44	0.47
	dqc_pe_12	13	103	-	251	-	-	0.42	0.32	0.36
	dqc_pe_15	16	151	-	314	-	-	0.47	0.41	0.42
	dqc_bit_flip_6	6	9	-	128	-	-	0.62	0.59	0.40

Table 7. Hardware benchmarking of multi-type quantum circuits (rounded to 2 decimal places).

(*dqc_bit_flip_6*). Each circuit begins from the all-zero state and is designed to output a specific computational basis state.

We ran each circuit 1024 times on all devices and measured fidelity based on the probability of obtaining the correct output. Table 7 summarizes the results, where "–" indicates unsupported circuits. The "Gate Number" column compares total gate counts before and after compilation under three ISAs: the original OpenQASM, the Xiaohong & Dongling ISA (supporting H, X, T, Pauli-rotation, and CNOT), and the IBM ISA (supporting ECR, Identity, Pauli-Z rotation, SX, X, and control-flow operations). Differences in native gate sets lead to varying compiled gate counts.

The table illustrates the varying behaviour of these hardware platforms across different circuit types. Xiaohong's hardware outperforms Dongling and all three IBM's hardware in most combinational quantum circuits, while IBM and Xiaohong's hardware surpass Dongling in variational quantum circuits. IBM's hardware stands out as the only one capable of executing dynamic quantum circuits, yet all platforms encounter challenges when executing sequential quantum circuits with loops, which is why the results for sequential circuits are omitted in Table 7. This underscores the importance of considering circuit type as a benchmark factor for quantum hardware evaluation within our proposed *VeriQBench*. The parameters of the devices are listed in Table 8.

Devices	2Q Error (layered)	Median readout error	Median T1 (μs)	Median T2 (μs)
Xiaohong	1.74%	5.71%	34.73	3.26
Dongling	3.71%	5.52%	59.54	37.26
ibm_brisbane	1.79%	1.43%	223.26	143.89
ibm_kyiv	1.86%	0.79%	277.04	117.71
ibm_sherbrooke	1.48%	1.26%	266.64	152.22

Table 8. The hardware parameters of the quantum processors.

To further assess IBM quantum processors using *VeriQBench*, we evaluated a broader set of circuits with varying qubit counts. As shown in Table 9, `ibm_brisbane` consistently performs best on most combinational and dynamic circuits (see also Table 7), while `ibm_kyiv` leads in most variational cases. Additionally, `ibm_sherbrooke` excels in specific instances such as *bv_10* and *qcnn_16*. These results underscore the importance of circuit type as a key benchmarking dimension when comparing quantum devices within the same family under the *VeriQBench* framework.

Type	Circuit Name	Qubit Number	Gate Number		brisbane	kyiv	sherbrooke
			Original	ISA			
combinational	bv_10	10	30	166	0.02	0.01	0.09
	bv_11	11	33	181	0.03	0.01	0.01
	bv_12	12	36	243	0.04	0.00	0.00
	bv_13	13	39	225	0.00	0.00	0.00
variational	eff_10	10	49	131	0.31	0.61	0.17
	eff_11	11	54	141	0.13	0.42	0.02
	eff_12	12	59	153	0.00	0.11	0.02
	eff_13	13	64	173	0.02	0.07	0.01
	qcnn_8	8	154	799	0.53	0.52	0.54
	qcnn_16	16	330	1833	0.47	0.52	0.53

Table 9. IBM hardware evaluations on larger multi-type quantum circuits (rounded to 2 decimal places).

6 Conclusion

This paper presented *VeriQBench*, a comprehensive quantum circuit benchmark that expands existing approaches by incorporating combinational, dynamic, sequential, and variational circuits. Moving beyond traditional metrics like qubit number and gate number, *VeriQBench* addresses the need for structural diversity, reflecting the advanced capabilities of modern quantum hardware. Designed for both hardware and software evaluation, it includes features like scalability and OpenQASM compatibility.

We validated *VeriQBench* with Qiskit and QCOR simulators and conducted experiments across different quantum hardware and equivalence-checking tools, revealing notable performance differences based on circuit type. By offering a diverse, adaptable framework, *VeriQBench* supports a thorough evaluation of quantum systems, increasing circuit coverage by over 26% compared to current quantum circuit benchmarks. This advancement contributes to the advancement of reliable quantum technologies.

References

1. Scott Aaronson and Daniel Gottesman. Improved simulation of stabilizer circuits. *Physical Review A*, 70(5):052328, 2004.
2. Nabila Abdessaied and Rolf Drechsler. Reversible and quantum circuits. *Google Scholar Google Scholar Digital Library Digital Library*, 2016.
3. Y. Aharonov, L. Davidovich, and N. Zagury. Quantum random walks. *Physical Review A*, 48(2):1687, 1993.
4. AIDE-QC. AIDE-QC: Documentation and user guides. `https://aide-qc.github.io/deploy/`.
5. Luca Gaetano Amarù, Pierre-Emmanuel Gaillardon, and Giovanni De Micheli. The epfl combinational benchmark suite. In *IWLS 2015*, 2015.
6. A. Ambainis, E. Bach, A. Nayak, A. Vishwanath, and J. Watrous. One-dimensional quantum walks. In *Proceedings of the Thirty-Third Annual ACM Symposium on Theory of Computing*, pages 37–49, 2001.
7. Matthew Amy, Dmitri Maslov, and Michele Mosca. Polynomial-time t-depth optimization of clifford+ t circuits via matroid partitioning. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 33(10):1476–1489, 2014.
8. Frank Arute, Kunal Arya, Ryan Babbush, Dave Bacon, Joseph C. Bardin, Rami Barends, Rupak Biswas, Sergio Boixo, Fernando G. S. L. Brandao, David A. Buell, Brian Burkett, Yu Chen, Zijun Chen, Ben Chiaro, Roberto Collins, William Courtney, Andrew Dunsworth, Edward Farhi, Brooks Foxen, Austin Fowler, Craig Gidney, Marissa Giustina, Rob Graff, Keith Guerin, Steve Habegger, Matthew P. Harrigan, Michael J. Hartmann, Alan Ho, Markus Hoffmann, Trent Huang, Travis S. Humble, Sergei V. Isakov, Evan Jeffrey, Zhang Jiang, Dvir Kafri, Kostyantyn Kechedzhi, Julian Kelly, Paul V. Klimov, Sergey Knysh, Alexander Korotkov, Fedor Kostritsa, David Landhuis, Mike Lindmark, Erik Lucero, Dmitry Lyakh, Salvatore Mandrà, Jarrod R. McClean, Matthew McEwen, Anthony Megrant, Xiao Mi, Kristel Michielsen, Masoud Mohseni, Josh Mutus, Ofer Naaman, Matthew Neeley, Charles Neill, Murphy Yuezhen Niu, Eric Ostby, Andre Petukhov, John C. Platt, Chris Quintana, Eleanor G. Rieffel, Pedram Roushan, Nicholas C. Rubin, Daniel Sank, Kevin J. Satzinger, Vadim Smelyanskiy, Kevin J. Sung, Matthew D. Trevithick, Amit Vainsencher, Benjamin Villalonga, Theodore White, Z. Jamie Yao, Ping Yeh, Adam Zalcman, Hartmut Neven, and John M. Martinis. Quantum supremacy using a programmable superconducting processor. *Nature*, 574(7779):505–510, Oct 2019.
9. Charles H Bennett, Gilles Brassard, Claude Crépeau, Richard Jozsa, Asher Peres, and William K Wootters. Teleporting an unknown quantum state via dual classical and einstein-podolsky-rosen channels. *Physical review letters*, 70(13):1895, 1993.
10. Debajyoti Bera. Detection and diagnosis of single faults in quantum circuits. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 37(3):587–600, 2017.
11. Ethan Bernstein and Umesh Vazirani. Quantum complexity theory. *SIAM Journal on Computing*, 26(5):1411–1473, 1997.
12. Debjyoti Bhattacharjee, Abdullah Ash Saki, Mahabubul Alam, Anupam Chattopadhyay, and Swaroop Ghosh. Muqut: Multi-constraint quantum circuit mapping on nisq computers. In *2019 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pages 1–7. IEEE, 2019.

13. Sergio Boixo, Sergei V Isakov, Vadim N Smelyanskiy, Ryan Babbush, Nan Ding, Zhang Jiang, Michael J Bremner, John M Martinis, and Hartmut Neven. Characterizing quantum supremacy in near-term devices. *Nature Physics*, 14(6):595–600, 2018.
14. Sergey Bravyi and Dmitri Maslov. Hadamard-free circuits expose the structure of the clifford group. *IEEE Transactions on Information Theory*, 67(7):4546–4563, 2021.
15. Sergey Bravyi, Ruslan Shaydulin, Shaohan Hu, and Dmitri Maslov. Clifford circuit optimization with templates and symbolic pauli gates. *Quantum*, 5:580, 2021.
16. F. Brglez, D. Bryan, and K. Kozminski. Combinational profiles of sequential benchmark circuits. In *1989 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 1929–1934 vol.3, 1989.
17. Lukas Burgholzer and Robert Wille. Advanced equivalence checking for quantum circuits. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 40(9):1810–1824, 2020.
18. Lukas Burgholzer and Robert Wille. Improved dd-based equivalence checking of quantum circuits. In *2020 25th Asia and South Pacific Design Automation Conference (ASP-DAC)*, pages 127–132. IEEE, 2020.
19. Lukas Burgholzer and Robert Wille. Qcec: A jkq tool for quantum circuit equivalence checking. *Software Impacts*, 7:100051, 2021.
20. M. Cerezo, Andrew Arrasmith, Ryan Babbush, Simon C. Benjamin, Suguru Endo, Keisuke Fujii, Jarrod R. McClean, Kosuke Mitarai, Xiao Yuan, Lukasz Cincio, and Patrick J. Coles. Variational quantum algorithms. *Nature Reviews Physics*, 3(9):625–644, 2021.
21. Amlan Chakrabarti and Susmita Sur-Kolay. Designing quantum adder circuits and evaluating their error performance. In *2008 International Conference on Electronic Design*, pages 1–6, 2008.
22. Kean Chen and Mingsheng Ying. Automatic test pattern generation for robust quantum circuit testing. *ACM Transactions on Design Automation of Electronic Systems*, 29(6):1–36, 2024.
23. Song Cheng, Chenfeng Cao, Chao Zhang, Yongxiang Liu, Shi-Yao Hou, Pengxiang Xu, and Bei Zeng. Simulating noisy quantum circuits with matrix product density operators. *Phys. Rev. Research*, 3:023005, Apr 2021.
24. Andrew M Childs. Lecture notes on quantum algorithms. *Lecture notes at University of Maryland*, 2017.
25. Andrew M Childs, Eddie Schoute, and Cem M Unsal. Circuit transformations for quantum architectures. *arXiv preprint arXiv:1902.09102*, 2019.
26. Iris Cong, Soonwon Choi, and Mikhail D Lukin. Quantum convolutional neural networks. *Nature Physics*, 15(12):1273–1278, 2019.
27. Antonio D Córcoles, Maika Takita, Ken Inoue, Scott Lekuch, Zlatko K Minev, Jerry M Chow, and Jay M Gambetta. Exploiting dynamic quantum circuits in a quantum algorithm with superconducting qubits. *Physical Review Letters*, 127(10):100501, 2021.
28. Arjan Cornelissen, Johannes Bausch, and András Gilyén. Scalable Benchmarks for Gate-Based Quantum Computers, April 2021. arXiv:2104.10698 [quant-ph].
29. F. Corno, M.S. Reorda, and G. Squillero. Rt-level itc'99 benchmarks and first atpg results. *IEEE Design & Test of Computers*, 17(3):44–53, 2000.
30. Andrew W Cross, Lev S Bishop, Sarah Sheldon, Paul D Nation, and Jay M Gambetta. Validating quantum computers using randomized model circuits. *Physical Review A*, 100(3):032328, 2019.

31. Yulong Dong and Lin Lin. Random circuit block-encoded matrix and a proposal of quantum linpack benchmark. *Physical Review A*, 103(6):062412, 2021.
32. Menghan Dou, Tianrui Zou, Yuan Fang, Jing Wang, Dongyi Zhao, Lei Yu, Boying Chen, Wenbo Guo, Ye Li, Zhaoyun Chen, and Guoping Guo. Qpanda: high-performance quantum computing framework for multiple application scenarios. *CoRR*, abs/2212.14201, 2022.
33. Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. A quantum approximate optimization algorithm. *arXiv preprint arXiv:1411.4028*, 2014.
34. Jernej Rudi Finžgar, Philipp Ross, Leonhard Hölscher, Johannes Klepsch, and Andre Luckow. Quark: A framework for quantum computing application benchmarking. In *2022 IEEE International Conference on Quantum Computing and Engineering (QCE)*, pages 226–237, 2022.
35. Simon Gay, Rajagopal Nagarajan, and Nikolaos Papanikolaou. Probabilistic model–checking of quantum protocols. *arXiv preprint quant-ph/0504007*, 2005.
36. Simon J Gay, Rajagopal Nagarajan, and Nikolaos Papanikolaou. Qmc: A model checker for quantum systems. In *International Conference on Computer Aided Verification*, pages 543–547. Springer, 2008.
37. Robert B Griffiths and Chi-Sheng Niu. Semiclassical fourier transform for quantum computation. *Physical Review Letters*, 76(17):3228, 1996.
38. Lov K Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, pages 212–219, 1996.
39. Jingzhe Guo, Huazhe Lou, Jintao Yu, Riling Li, Wang Fang, Junyi Liu, Peixun Long, Shenggang Ying, and Mingsheng Ying. isq: An integrated software stack for quantum programming. *IEEE Transactions on Quantum Engineering*, 4:1–16, 2023.
40. Stuart Hadfield, Zhihui Wang, Bryan O'gorman, Eleanor G Rieffel, Davide Venturelli, and Rupak Biswas. From the quantum approximate optimization algorithm to a quantum alternating operator ansatz. *Algorithms*, 12(2):34, 2019.
41. M.C. Hansen, H. Yalcin, and J.P. Hayes. Unveiling the iscas-85 benchmarks: a case study in reverse engineering. *IEEE Design & Test of Computers*, 16(3):72–80, 1999.
42. Xin Hong, Yuan Feng, Sanjiang Li, and Mingsheng Ying. Equivalence checking of dynamic quantum circuits. In *Proceedings of the 41st IEEE/ACM International Conference on Computer-Aided Design*, pages 1–8, 2022.
43. Xin Hong, Mingsheng Ying, Yuan Feng, Xiangzhen Zhou, and Sanjiang Li. Approximate equivalence checking of noisy quantum circuits. In *2021 58th ACM/IEEE Design Automation Conference (DAC)*, pages 637–642. IEEE, 2021.
44. Cheng-Yun Hsieh, Chen-Hung Wu, Chia-Hsien Huang, His-Sheng Goan, and James Chien Mo Li. Realistic fault models and fault simulation for quantum dot quantum circuits. In *2020 57th ACM/IEEE Design Automation Conference (DAC)*, pages 1–6, 2020.
45. Mingyu Huang, Ji Guan, Wang Fang, and Mingsheng Ying. Approximation algorithm for noisy quantum circuit simulation. In *2024 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 1–6. IEEE, 2024.
46. Yipeng Huang, Steven Holtzen, Todd Millstein, Guy Van den Broeck, and Margaret Martonosi. Logical abstractions for noisy variational quantum algorithm simulation. In *Proceedings of the 26th ACM International Conference on Architectural Support for Programming Languages and Operating Systems*, ASPLOS 2021, page 456–472, New York, NY, USA, 2021. Association for Computing Machinery.

47. IBM. Qiskit benchpress. `https://github.com/Qiskit/benchpress/tree/main`.
48. Sergei V. Isakov, Dvir Kafri, Orion Martin, Catherine Vollgraff Heidweiller, Wojciech Mruczkiewicz, Matthew P. Harrigan, Nicholas C. Rubin, Ross Thomson, Michael Broughton, Kevin Kissell, Evan Peters, Erik Gustafson, Andy C. Y. Li, Henry Lamm, Gabriel Perdue, Alan K. Ho, Doug Strain, and Sergio Boixo. Simulations of quantum circuits with approximate noise using qsim and cirq, 2021.
49. J. Kempe. Quantum random walks: an introductory overview. *Contemporary Physics*, 44(4):307–327, 2003.
50. Ang Li, Samuel Stein, Sriram Krishnamoorthy, and James Ang. Qasmbench: A low-level quantum benchmark suite for nisq evaluation and simulation. *ACM Transactions on Quantum Computing*, 4(2), February 2023.
51. Gushu Li, Yufei Ding, and Yuan Xie. Tackling the qubit mapping problem for nisq-era quantum devices. In *Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems*, pages 1001–1014, 2019.
52. Chia-Chun Lin, Susmita Sur-Kolay, and Niraj K Jha. Paqcs: Physical design-aware fault-tolerant quantum circuit synthesis. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 23(7):1221–1234, 2014.
53. Yunchao Liu, Matthew Otten, Roozbeh Bassirianjahromi, Liang Jiang, and Bill Fefferman. Benchmarking near-term quantum computers via random circuit sampling. *arXiv preprint arXiv:2105.05232*, 2021.
54. Thomas Lubinski, Sonika Johri, Paul Varosy, Jeremiah Coleman, Luning Zhao, Jason Necaise, Charles H. Baldwin, Karl Mayer, and Timothy Proctor. Application-oriented performance benchmarks for quantum computing. *IEEE Transactions on Quantum Engineering*, 4:1–32, 2023.
55. M. Lukac and M. Perkowski. Quantum finite state machines as sequential quantum circuits. In *Proceedings of the 39th International Symposium on Multiple-Valued Logic*, pages 92–97, 2009.
56. Dmitri Maslov. Reversible logic synthesis benchmarks page. `https://reversiblebenchmarks.github.io`.
57. Dmitri Maslov, Gerhard W Dueck, D Michael Miller, and Camille Negrevergne. Quantum circuit simplification and level compaction. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 27(3):436–444, 2008.
58. Alexander McCaskey, Eugene Dumitrescu, Mengsu Chen, Dmitry Lyakh, and Travis Humble. Validating quantum-classical programming models with tensor network simulations. *PLOS ONE*, 13(12):1–19, 12 2018.
59. Alexander Mccaskey, Thien Nguyen, Anthony Santana, Daniel Claudino, Tyler Kharazi, and Hal Finkel. Extending c++ for heterogeneous quantum-classical computing. *ACM Transactions on Quantum Computing*, 2(2):1–36, 2021.
60. M. S. Moreira, G. G. Guerreschi, W. Vlothuizen, J. F. Marques, J. van Straten, S. P. Premaratne, X. Zou, H. Ali, N. Muthusubramanian, C. Zachariadis, J. van Someren, M. Beekman, N. Haider, A. Bruno, C. G. Almudever, A. Y. Matsuura, and L. DiCarlo. Realization of a quantum neural network using repeat-until-success circuits in a superconducting quantum processor. *npj Quantum Information*, 9(1), November 2023.
61. Prakash Murali, Norbert Matthias Linke, Margaret Martonosi, Ali Javadi Abhari, Nhung Hong Nguyen, and Cinthia Huerta Alderete. Full-stack, real-system quantum computer studies: architectural comparisons and design insights. In *Proceedings of the 46th International Symposium on Computer Architecture*, pages 527–540, Phoenix Arizona, June 2019. ACM.

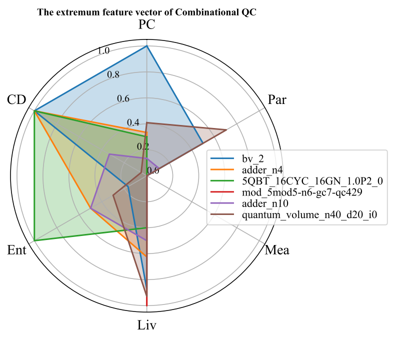
62. Prakash Murali, Norbert Matthias Linke, Margaret Martonosi, Ali Javadi Abhari, Nhung Hong Nguyen, and Cinthia Huerta Alderete. Full-stack, real-system quantum computer studies: Architectural comparisons and design insights. In *2019 ACM/IEEE 46th Annual International Symposium on Computer Architecture (ISCA)*, pages 527–540. IEEE, 2019.
63. Yunseong Nam, Neil J Ross, Yuan Su, Andrew M Childs, and Dmitri Maslov. Automated optimization of large quantum circuits with continuous parameters. *npj Quantum Information*, 4(1):1–12, 2018.
64. Michael A Nielsen and Isaac Chuang. Quantum computation and quantum information, 2002.
65. Beijing Academy of Quantum Information Sciences (BAQIS). Quafu quantum computing cloud platform. `https://quafu.baqis.ac.cn/`. Accessed: 2024-10-31.
66. A. Paetznick and K. M. Svore. Repeat-until-success: non-deterministic decomposition of single-qubit unitaries. *Quantum Information & Computation*, 14(15–16):1277–1301, 2014.
67. Alexandru Paler, Ilia Polian, and John P Hayes. Detection and diagnosis of faulty quantum circuits. In *17th Asia and South Pacific Design Automation Conference*, pages 181–186. IEEE, 2012.
68. Tirthak Patel, Abhay Potharaju, Baolin Li, Rohan Basu Roy, and Devesh Tiwari. Experimental Evaluation of NISQ Quantum Computers: Error Measurement, Characterization, and Implications. In *SC20: International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–15, Atlanta, GA, USA, November 2020. IEEE.
69. Massoud Pedram and Alireza Shafaei. Layout optimization for quantum circuits with linear nearest neighbor architectures. *IEEE Circuits and Systems Magazine*, 16(2):62–74, 2016.
70. Alberto Peruzzo, Jarrod McClean, Peter Shadbolt, Man-Hong Yung, Xiao-Qi Zhou, Peter J Love, Alán Aspuru-Guzik, and Jeremy L O'brien. A variational eigenvalue solver on a photonic quantum processor. *Nature communications*, 5(1):1–7, 2014.
71. Aditya K Prasad, Vivek V Shende, Igor L Markov, John P Hayes, and Ketan N Patel. Data structures and algorithms for simplifying reversible circuits. *ACM Journal on Emerging Technologies in Computing Systems (JETC)*, 2(4):277–293, 2006.
72. Qiskit. Qiskit. `https://qiskit.org`.
73. IBM Quantum. Eagle quantum processor performance. `https://www.ibm.com/quantum/blog/eagle-quantum-processor-performance`. Accessed: 2024-10-31.
74. QuantumCTek. Quantumctek cloud platform. `https://quantumctek-cloud.com/`. Accessed: 2024-10-31.
75. Nils Quetschlich, Lukas Burgholzer, and Robert Wille. MQT Bench: Benchmarking Software and Design Automation Tools for Quantum Computing. *Quantum*, 7:1062, July 2023.
76. Colm A Ryan, Blake R Johnson, Diego Ristè, Brian Donovan, and Thomas A Ohki. Hardware for dynamic quantum computing. *Review of Scientific Instruments*, 88(10):104703, 2017.
77. Mehdi Saeedi and Igor L Markov. Synthesis and optimization of reversible circuits—a survey. *ACM Computing Surveys (CSUR)*, 45(2):1–34, 2013.
78. Manuel A. Serrano, José A. Cruz-Lemus, Ricardo Perez-Castillo, and Mario Piattini. Quantum Software Components and Platforms: Overview and Quality Assessment. *ACM Comput. Surv.*, 55(8), dec 2022.

79. Alireza Shafaei, Mehdi Saeedi, and Massoud Pedram. Qubit placement to minimize communication overhead in 2d quantum architectures. In *2014 19th Asia and South Pacific Design Automation Conference (ASP-DAC)*, pages 495–500. IEEE, 2014.
80. Changpeng Shao, Yang Li, and Hongbo Li. Quantum Algorithm Design: Techniques and Applications. *Journal of Systems Science and Complexity*, 32(1), February 2019.
81. Marcos Yukio Siraichi, Vinícius Fernandes dos Santos, Caroline Collange, and Fernando Magno Quintão Pereira. Qubit allocation. In *Proceedings of the 2018 International Symposium on Code Generation and Optimization*, pages 113–125, 2018.
82. Seyon Sivarajah, Silas Dilkes, Alexander Cowtan, Will Simmons, Alec Edgington, and Ross Duncan. t— ket¿: a retargetable compiler for nisq devices. *Quantum Science and Technology*, 6(1):014003, 2020.
83. Bochen Tan and Jason Cong. Optimal layout synthesis for quantum computing. In *2020 IEEE/ACM International Conference On Computer Aided Design (ICCAD)*, pages 1–9. IEEE, 2020.
84. Bochen Tan and Jason Cong. Optimality study of existing quantum computing layout synthesis tools. *IEEE Transactions on Computers*, 70(9):1363–1373, 2020.
85. Swamit S Tannu and Moinuddin K Qureshi. Not all qubits are created equal: a case for variability-aware policies for nisq-era quantum computers. In *Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems*, pages 987–999, 2019.
86. Jules Tilly, Hongxiang Chen, Shuxiang Cao, Dario Picozzi, Kanav Setia, Ying Li, Edward Grant, Leonard Wossnig, Ivan Rungger, George H. Booth, and Jonathan Tennyson. The variational quantum eigensolver: A review of methods and best practices. *Physics Reports*, 986:1–128, 2022. The Variational Quantum Eigensolver: a review of methods and best practices.
87. Teague Tomesh, Pranav Gokhale, Victory Omole, Gokul Subramanian Ravi, Kaitlin N. Smith, Joshua Viszlai, Xin-Chuan Wu, Nikos Hardavellas, Margaret Martonosi, and Fred Chong. Supermarq: A scalable quantum benchmark suite. *2022 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*, pages 587–603, 2022.
88. George F Viamontes, Igor L Markov, and John P Hayes. Checking equivalence of quantum circuits and states. In *2007 IEEE/ACM International Conference on Computer-Aided Design*, pages 69–74. IEEE, 2007.
89. Qisheng Wang, Riling Li, and Mingsheng Ying. Equivalence checking of sequential quantum circuits. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 41(9):3143–3156, 2022.
90. Qisheng Wang, Junyi Liu, and Mingsheng Ying. Equivalence checking of quantum finite-state machines. *Journal of Computer and System Sciences*, 116:1–21, 2021.
91. Robert Wille, Lukas Burgholzer, and Alwin Zulehner. Mapping quantum circuits to ibm qx architectures using the minimal number of swap and h operations. In *2019 56th ACM/IEEE Design Automation Conference (DAC)*, pages 1–6. IEEE, 2019.
92. Robert Wille, Daniel Große, Lisa Teuber, Gerhard W. Dueck, and Rolf Drechsler. Revlib: An online resource for reversible functions and reversible circuits. In *38th International Symposium on Multiple Valued Logic (ismvl 2008)*, pages 220–225, 2008.

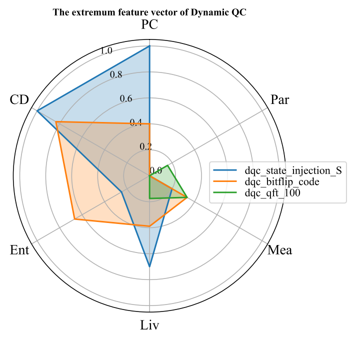
93. Robert Wille, Aaron Lye, and Rolf Drechsler. Optimal swap gate insertion for nearest neighbor quantum circuits. In *2014 19th Asia and South Pacific Design Automation Conference (ASP-DAC)*, pages 489–494. IEEE, 2014.
94. Romina Yalovetzky, Pierre Minssen, Dylan Herman, and Marco Pistoia. Solving linear systems on quantum hardware with hybrid HHL++. *Scientific Reports*, 14(1), September 2024.
95. Shigeru Yamashita and Igor L Markov. Fast equivalence-checking for quantum circuits. In *2010 IEEE/ACM International Symposium on Nanoscale Architectures*, pages 23–28. IEEE, 2010.
96. S. Y. Yang. Logic synthesis and optimization benchmarks user guide version 3.0. In *IWLS 1991*, 1991.
97. Mingsheng Ying. Model checking for verification of quantum circuits. In *International Symposium on Formal Methods*, pages 23–39. Springer, 2021.
98. Mingsheng Ying and Yuan Feng. *Model Checking Quantum Systems: Principles and Algorithms.* Cambridge University Press, 2021.
99. Wen-Hao Zhou, Xiao-Wei Wang, Ruo-Jing Ren, Yu-Xuan Fu, Yi-Jun Chang, Xiao-Yun Xu, Hao Tang, and Xian-Min Jin. Multi-particle quantum walks on 3D integrated photonic chip. *Light: Science & Applications*, 13(1), October 2024.
100. Alwin Zulehner, Alexandru Paler, and Robert Wille. An efficient methodology for mapping quantum circuits to the ibm qx architectures. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 38(7):1226–1236, 2018.

A Feature vectors

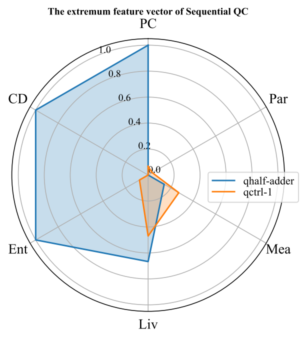
The extremum feature vectors for each circuit type in *VeriQBench* are shown below.



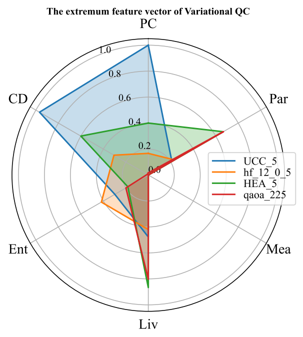
(a)



(b)



(c)



(d)

Fig. 2. The extremum feature vectors of each type of circuit.

B Detailed Circuit Introduction

A detailed introduction to the circuits included in *VeriQBench* is given below.

B.1 Combinational Quantum Circuits

Quantum Algorithms

Bernstein-Vazirani Algorithm. Bernstein-Vazirani algorithm [11] is an algorithm that can be used to find the hidden string s given a boolean function $f(x)$, where $f(x) = \langle s, x\rangle = s_0x_0 \oplus s_1x_1 \oplus \cdots \oplus s_nx_n$. For the classical algorithm, it normally needs $\mathcal{O}(n)$ times to complete this task using a bit-by-bit inquiring method. But, it only needs $\mathcal{O}(1)$ times using the Bernstein-Vazirani algorithm. Suppose that you are given an oracle O_s, where $O_s |x\rangle |y\rangle = |x\rangle |f(x) \oplus y\rangle$.

First, apply a series of Hadamard gates to state $|0\rangle \cdots |0\rangle |1\rangle$ to change the state to $\frac{1}{\sqrt{2^n}} \sum_{x=0}^{2^n-1} |x\rangle \otimes \frac{|0\rangle - |1\rangle}{\sqrt{2}}$. Then, applying the oracle, the state will become $\frac{1}{\sqrt{2^n}} \sum_{x=0}^{2^n-1} (-1)^{\langle s,x\rangle} |x\rangle \otimes \frac{|0\rangle - |1\rangle}{\sqrt{2}}$. Finally, apply a series of Hadamard gates, and the final state will change to $|s\rangle |1\rangle$. At the end, measuring the first n qubits gives the value of s.

For $s = 101$ and $s = 111$, the corresponding circuits are shown in Fig. 3. In these two circuits, an X gate is added at the last qubit to change the initial state from $|0\rangle$ to $|1\rangle$, and the $CNOT$ gates are used for implementing the oracle. For these circuits, setting input state $|0\rangle \cdots |0\rangle$ and measuring at the end will give the hidden string s.

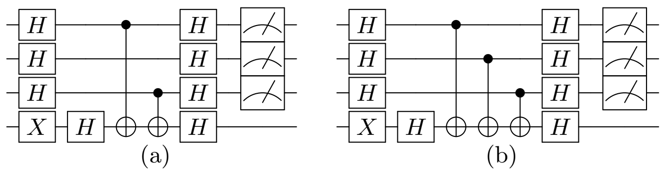


Fig. 3. Two examples of the Bernstein-Vazirani algorithm, where (a) $s = 101$ and (b) $s = 111$.

Quantum Fourier Transform. Quantum Fourier transform [64] is a commonly used algorithm in quantum computing. It turns an input state $|j\rangle$ to its Fourier transform $QFT\,|j\rangle = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} e^{2\pi ijk/N} |k\rangle$.

The circuit for the 3-qubit quantum Fourier transform is shown in Fig. 4. It is easy to observe from this circuit that the quantum Fourier transform can be implemented using a series of Hadamard gates and a set of Control-R_k gates, where $R_k = \begin{bmatrix} 1 & 0 \\ 0 & e^{2\pi i/2^k} \end{bmatrix}$.

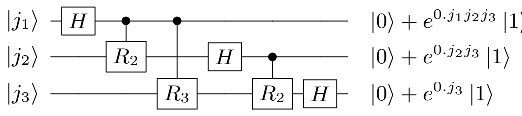

$|j_1\rangle$
$|j_2\rangle$
$|j_3\rangle$
H
R_2
R_3
$|0\rangle + e^{0.j_1j_2j_3}|1\rangle$
$|0\rangle + e^{0.j_2j_3}|1\rangle$
$|0\rangle + e^{0.j_3}|1\rangle$

Fig. 4. Quantum circuit for Fourier transform.

Quantum Phase Estimation. Phase estimation [64] is an algorithm that can be used to estimate the phase φ in an eigenvalue of a unitary U, where $U|\psi\rangle = e^{2\pi i\varphi}|\psi\rangle$ for some $|\psi\rangle$. The corresponding circuit is shown in Fig. 5.

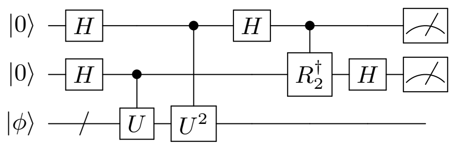


Fig. 5. Quantum circuit for phase estimation.

The process can be conducted by first constructing a superposition state and then applying a series of controlled-U^{2^k} gates. Then, an inverse quantum Fourier transform can be used to extract the corresponding value of the phase. In this circuit, we suppose that U is a single qubit diagonal matrix $U = \begin{bmatrix} 1 & 0 \\ 0 & e^{2\pi i\varphi} \end{bmatrix}$, and here we take $\varphi = 1/1024$ as an example.

Grover's Algorithm. Grover's algorithm [38] is one of the most commonly used algorithms in quantum computing to search for the solution of an integer function.

Suppose you are given an oracle O such that $O|x\rangle = (-1)^{f(x)}|x\rangle$, then initialise the state to be the equal superposition state $|\psi\rangle = \frac{1}{N^{1/2}} \sum_{x=0}^{N-1} |x\rangle$, and iteratively apply the Grover operator $G = (2|\psi\rangle\langle\psi| - I)O$, and finally measure all the qubits, you will get a solution of the function $f(x)$ with a probability close to 1.

Fig. 6 gives an example of the Grover's algorithm. Here, $f(x) = x_1 \cdot x_2$, running this circuit and measuring at the end will obtain 11, which is the solution of this function.

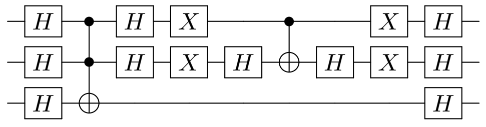


Fig. 6. An example of Grover's algorithm. Here, $f(x)$ is a 2-bit function and $f(x) = x_1 \cdot x_2$.

Quantum Adder. A quantum adder is a quantum circuit that implements the add operation on two-bit strings. For example, if we compute '2+3=5', then we represent the input string as '010' and '011', and the expected output bit string is '101'. The implementation of the quantum adder circuit is illustrated as follows (Fig. 7), and the additional qubits are used to store the carry bit [21].

Reversible Circuits

A classical n-bit reversible gate is a bijective mapping f from the set $\{0,1\}^n$ of n-bit data onto itself. Thus the vector of input states can always be reconstructed from the vector of output states. A combinational logic circuit is reversible if it only contains reversible gates and has no fan-out. Classical reversible circuits

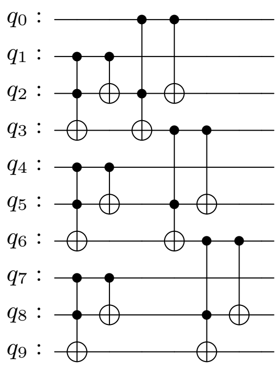


Fig. 7. Circuit of the 3-bit adder.

may be implemented in quantum technology and have important applications in many quantum algorithms, such as the arithmetic module of Shor's Algorithm and the oracle of Grover's Algorithm. In our benchmark, we collect the classical reversible circuits in the Reversible Logic Synthesis Benchmarks Page [56].

Qubit Mapping

On the current superconducting quantum processors, 2-qubit gates are usually unavailable for arbitrary pairs of qubits but only for a small part of them. In order to make all the 2-qubit gates in a circuit available on a specific quantum chip, we have to map the qubits in the circuit to those on the quantum chip and insert some SWAP gates. Meanwhile, we want to make sure that the modified circuit is optimal in depth or the number of inserted SWAP gates. However, since the above problem, known as the qubit mapping problem, is NP-complete, it is difficult to theoretically evaluate the performance of different algorithms. Instead, the performance can be evaluated through benchmarks. [84] presents an algorithm to generate benchmarks of the qubit mapping problems on specific quantum processors along with optimal solutions. We have implemented the generating algorithm in Python. The input and output of the implementation are described below:

The set of qubit pairs that are available for 2-qubit gates is given by the edge set E of a graph G. Given depth d, gate number N, and the proportion of 2-qubit gates p_2, a random quantum circuit that can be executed on the graph G is generated along with an optimal qubit mapping of depth d.

Random Circuits

Random Clifford Circuits. Clifford operation plays an important role in quantum error correction, randomized benchmarking protocols and quantum circuit simulation. By definition, the Clifford operation is a unitary operation taking elements of G_n to elements of G_n, where G_n is the Pauli group on n qubits. Any n-qubit Clifford operation can be simulated using $O(n^2)$ Hadamard, phase and controlled-NOT gates. Clifford group elements are important and frequently encountered subsets of physical-level and fault-tolerant quantum circuits [15], and sometimes an entire quantum algorithm can be a Clifford circuit (e.g., Bernstein–Vazirani [64]).

The Clifford group is a unitary 2-design. That is, a random uniformly distributed element of the Clifford group has exactly the same second-order moments as the Haar random unitary operation. This means the random Clifford operations can serve as a substitute for Haar random unitaries in any application that depends only on the second-order moments. In Qiskit, the random Clifford is sampled using the method of [14]. Then the Clifford circuit is synthesized by the method in [1] and optimized by the method in [15]. In our repository, we included these circuits.

Quantum Volume. Quantum volume [30] is a metric that can be used to measure the capabilities and error rates of a quantum computer. It quantifies the largest random circuit of equal width and depth that the computer can successfully implement. The circuit model used for measuring quantum volume is as follows (Fig. 8).

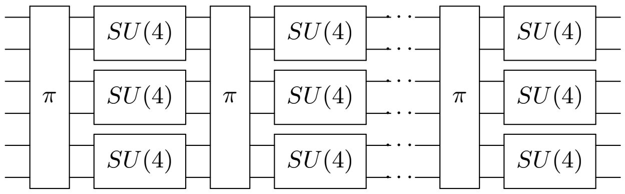


Fig. 8. The circuit model for quantum volume.

Here, π is a permutation of qubits, and every $SU(4)$ represents a 2-qubit unitary gate sampled from the Haar measure on $SU(4)$. There will be d layers of this module if the depth of the circuit is d, and if the number of qubits in the circuit is odd, then every layer will have an idle qubit.

In our benchmark, we give a series of circuits for quantum volume constructed on basic quantum gates. And the $SU(4)$ gates are decomposed using Qiskit.

Supremacy Circuits. Random circuits have been widely used in works related to quantum supremacy [13]. Since there are already some benchmarks with such circuits, we just include the existing and commonly used circuits directly in our benchmark. One of such benchmark is GRCS [9]. GRCS provides a lot of random circuits, but all in a '.txt' format and not given in the standard OpenQASM language. In our benchmark, we give the OpenQASM version of the random circuits shown in GRCS.

B.2 Dynamic Quantum Circuits

Quantum Teleportation

[9] `https://github.com/sboixo/GRCS`

One of the simplest examples of dynamic quantum circuits is teleportation. Teleportation is a protocol for transmitting a qubit between two users by sending two classical bits of information [9].

The corresponding circuit is shown in Fig. 9. The first qubit belongs to Alice, and the last qubit belongs to Bob. By first applying a series of quantum gates and sending the measurement result to Bob, Bob can obtain the state of the first qubit of Alice by applying a series of gates according to the measurement information. The dashed box shows the mid-circuit measurement and classical control needed to accomplish the protocol.

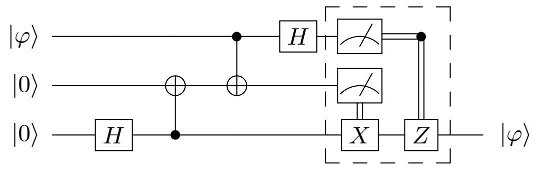


Fig. 9. Dynamic quantum circuit for teleportation.

Semiclassical Fourier Transform

The quantum Fourier transform can also be represented as a dynamic quantum circuit form [37]. The following gives detailed information on the dynamic version of the quantum Fourier transform. From Fig. 4 and Fig. 10, you can see that all the controlled-R_k gates are replaced by a measurement and classically controlled R_k gates.

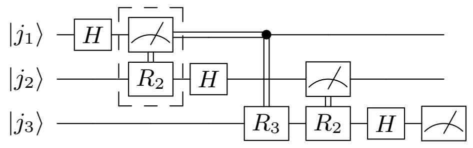


Fig. 10. Dynamic quantum circuit for Fourier transform.

The advantage of using this dynamic quantum circuit is that there is no need to apply 2-qubit gates during the process, all the 2-qubit gates can be replaced by single-qubit gates chosen according to the measurement results.

Iterative Phase Estimation

Since the quantum phase estimation uses the inverse quantum Fourier transform as a subroutine, it can also be executed as a dynamic quantum circuit. The dynamic version of quantum phase estimation is shown in Fig. 11.

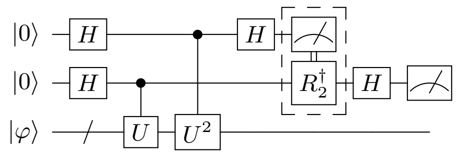


Fig. 11. Quantum circuit for phase estimation.

The advantage of the phase estimation implemented by the dynamic quantum circuit is that the circuit can be executed with only 2-qubit gates, and more details are shown in [27].

Quantum Error correction
Another example of dynamic quantum circuits is the circuit for quantum error correction, where a quantum state must first experience a period of syndrome measurement, and then be recovered according to the measurement result.
Fig. 12 gives the circuit model of error correction

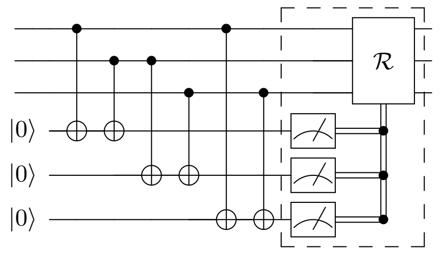


Fig. 12. A circuit model for quantum error correction.

State Injection State injection is the technique for implementing some quantum gates using a dynamic scheme, as shown in Fig. 13. The basic idea is that the effect of this gate can be implemented by using a special state and a series of gates that can be implemented more simply.

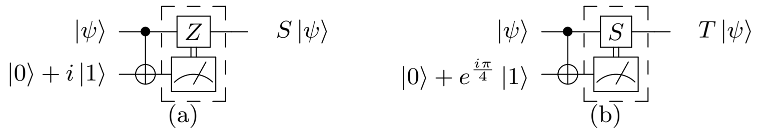


Fig. 13. The quantum circuits for state injection, (a) implementation of S gate, (b) implementation of T gate.

B.3 Sequential Quantum Circuits

Repeat-Until-Success

Repeat-Until-Success circuits [66] are a bunch of efficient implementations of quantum logic gates which make good use of intermediate measurements as feedback. Fig. 14 shows a sequential quantum circuit for repeat-until-success implementation of quantum gate $V_3 = \frac{I+2iZ}{\sqrt{5}}$.

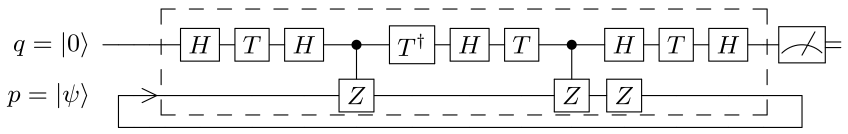


Fig. 14. A sequential quantum circuit for repeat-until-success implementation of V_3 which is taken from [89].

In Fig. 14, q is the only input variable (qubit) while p is the only internal variable (qubit). At each time step, the qubit q will be measured in the computational basis, indicating whether the implementation succeeds.

Quantum Walk

Quantum walks [3,49,6] are quantum generalizations of classical random walks, and have many applications in quantum algorithms. As an illustrative example (see Fig. 15), we consider a quantum walk on a circle (with positions $|0\rangle_p, |1\rangle_p, |2\rangle_p, |3\rangle_p$) with an absorbing boundary $|3\rangle_p$. Here, the coin operator is given by a Hadamard gate H and *Toss* is the toss operator

$$\begin{aligned} \mathit{Toss} : |0\rangle_c |i\rangle_p &\to |0\rangle_c |(i+1) \bmod 4\rangle_p , \\ |1\rangle_c |i\rangle_p &\to |1\rangle_c |(i-1) \bmod 4\rangle_p , \end{aligned}$$

which translates the position conditioned on the coin state.

Classical Control

For the convenience of controlling the behaviour of quantum systems, we consider a class of sequential quantum circuits with their input variables in the computational basis, which can be understood as a generalization of the quantum Mealy machine considered in [90]. For example, in Fig. 16, the input variables q_1 and q_2 control which kind of quantum gates is performed on p_1, p_2, p_3. To retrieve some information from the sequential quantum circuit, we use a detective qubit (an input variable) d with it being initialized to $|0\rangle$.

B.4 Variational Quantum Circuits

Variational Quantum Eigensolver

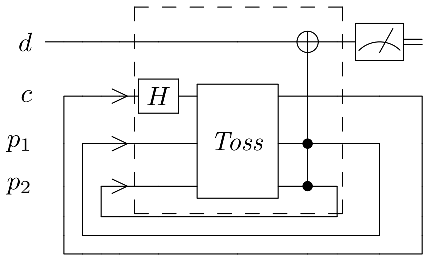


Fig. 15. A sequential quantum circuit for a quantum walk. This figure is taken from [89].

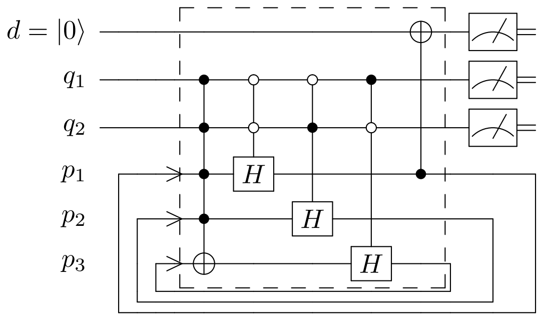


Fig. 16. A sequential quantum circuit for classical control. This figure is taken from [89].

Solving the ground state energy (minimum eigenvalue) of a Hamiltonian is fundamental in quantum chemistry and condensed matter physics. By using various VQCs to model wavefunctions, VQE converts the minimum eigenvalue problem into optimization over the parameters of VQCs. The reader can refer to a review [86] for comprehensive knowledge. In the benchmark, we provided a Python script to generate OpenQASM files for VQCs used in VQE literature. Different ansatz can be assigned in this script.

Quantum Approximate Optimization Algorithm

The Quantum Approximate Optimization Algorithm (QAOA) is a quantum algorithm designed to solve combinatorial optimization problems by approximating solutions to complex problems like the Max-Cut or other NP-hard tasks [33]. It combines quantum mechanics with classical optimization by utilizing a quantum circuit comprising a series of parameterized gates that evolve between two Hamiltonians: one encoding the problem's constraints and another driving state transitions. By adjusting these parameters, QAOA aims to produce quantum states that yield high-quality solutions upon measurement. The algorithm's iterative nature allows for tunable accuracy, where more iterations often result in better approximations, balancing computational complexity with solution quality.

Quantum Convolutional Neural Network

A Quantum Convolutional Neural Network (QCNN) is a quantum algorithm designed to mimic the structure and benefits of classical convolutional neural networks for tasks in quantum machine learning, such as pattern recognition and classification of quantum states [26]. Inspired by convolutional layers, QCNNs apply a series of quantum operations that reduce the quantum state's dimensionality through quantum pooling layers, allowing them to process large amounts of data with fewer qubits efficiently. QCNNs are particularly effective for analyzing quantum phases of matter and performing quantum feature extraction, leveraging quantum entanglement and superposition to capture complex data patterns with potential advantages over classical neural networks in certain contexts. Fig. 17 illustrates a 4-qubit Quantum Convolutional Neural Network (QCNN) architecture; the OpenQASM file of the QCNN circuit can be found in our repository in GitHub.

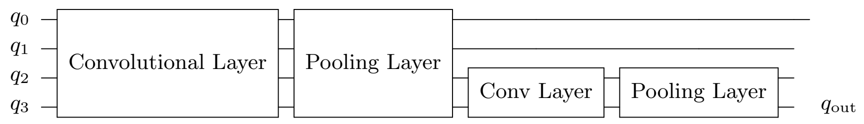


Fig. 17. A 4-qubit Quantum Convolutional Neural Network (QCNN) architecture.

Efficient SU2 Ansatz

The EfficientSU2 circuit in IBM's Qiskit is a variational quantum circuit designed with layers of single-qubit gates that span the SU(2) group (i.e., 2x2 unitary matrices with a determinant of 1, such as Pauli rotation gates) combined with CNOT gates. This structure makes EfficientSU2 a versatile, heuristic circuit well-suited for preparing trial wave functions in variational quantum algorithms and for serving as a classification circuit in quantum machine learning. Its layered design allows for flexible parameter tuning, enabling effective approximation of target quantum states and efficient encoding of patterns in quantum data. In our repository, we use the EfficientSU2 ansatz to train five 4-qubit circuits that output all one state. An example of the circuit is illustrated in Fig. 18.

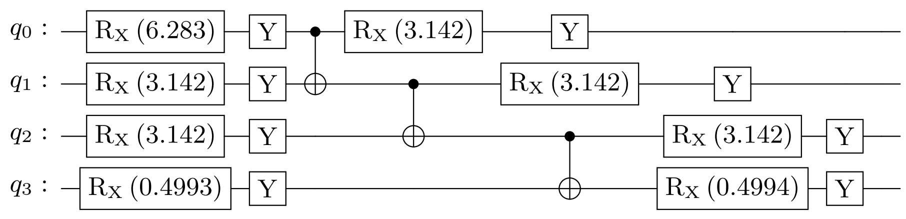


Fig. 18. A 4-qubit EfficientSU2 circuit.