[01/09/2023]

Centro de Investigación en Tecnologías de la Información y las Comunicaciones (CITIC)

Universidade da Coruña (UDC)

Final Report

Lista de autores

Oluwatosin Esther Odubanjo
Diego Andrade Canosa
Juan Touriño Domínguez
María José Martín Santamaría
Basilio Fraguela Rodríguez

XUNTA DE GALICIA

Unión Europea

Unha maneira de facer Europa.

CESGA GALICIA SUPERCOMPUTING CENTER

Despregamento dunha infraestructura baseada en tecnoloxías cuánticas da información que permita impulsar a I+D+i en Galicia. Apoiar a transición cara a una economía dixital.

Operación financiada pola Unión Europea, a través do FONDO EUROPEO DE DESENVOLVEMENTO REXIONAL (FEDER) como parte da resposta da Unión á pandemia da COVID-19

DATA	AUTHOR	CHANGES	VERSION
16 / 08 / 2023	Diego Andrade Canosa	Initial version and structure	0.1
28 / 08 / 2023	Oluwatosin Esther Odubanjo	First complete draft	0.5
30 / 08 / 2023	Oluwatosin Esther Odubanjo	Final version	1.0
31 / 08 / 2023	Oluwatosin Esther Odubanjo	Final version (CESGA specification format)	1.1

Table of Contents

1 Introduction **7**

2 Project Information **7**

3 Results of the Project **10**

3.1 Result 1: Study of the state of the art 10

3.1.1 Application-Centric Benchmarking 13

3.1.2 Compiler-Centric Benchmarking 14

3.1.3 Aggregated / Holistic Benchmarking 14

3.1.4 Volumetric Benchmarking (VB) 14

3.1.5 Hardware-Centric Benchmarking 14

3.2 Result 2: Selection of the cases 15

3.3 Result 3: Detailed definition of the cases 15

3.3.1 Probability Loading (PL) 15

3.3.2 Amplitude Estimation (AE) 17

3.3.3 Quantum Phase Estimation (QPE) 18

3.4 Result 4: Implementation of the cases 20

3.5 Result 5: Design and implementation of a prototype of the results repository 20

3.5.1 Design of the architecture and main components 20

3.5.2 Web Interface Design 21

List of Figures

1 Quantum Architecture Stack 13

List of Tables

List of Acronyms

1 Introduction

This document is the final report of the contract titled: "Research in the field of definition and execution of benchmarks for Quantum Computing eligible for funding by the European Union within the framework of the REACT EU Axis of the FEDER Galicia operational program 2014-2020, as part of the European Union's response to the COVID-19 pandemic" (reference CTS-2022-0053) between the "Fundación Publica Galega Centro Tecnolóxico de Supercomputación de Galicia - CESGA" and the "Fundación Centro de Investigación en Tecnoloxías da Información e as Comunicacións de Galicia - FCITICCG".

The objective of this contract has been the research for the definition and execution of Quantum Computing Benchmarks. More specifically:

- Study of the state of the art on benchmarking quantum computers oriented towards applications, considering the entire execution stack.
- Selection of cases to be implemented based on the analysis of the state of the art and applications of interest to be defined together with CESGA, considering the methodology defined by CESGA.
- Detailed definition of the cases.
- Implementation of the benchmark cases in the environments available at CESGA.
- Design and development of a prototype of a results repository that fits the needs for analyzing obtained results.
- Final report.

2 Project Information

The contract developed a project organized into 7 tasks and 4 milestones. The tasks are the definition of the different things to be done during the project, while the milestones define dates where the results of different tasks have to be delivered to CESGA. The 7 tasks were:

- Task 1: Study of the state of the art on benchmarking quantum computers oriented towards applications and considering the entire execution stack.
- Task 2: Selection of cases to implement based on the analysis of the state of the art and applications of interest, to be defined in collaboration with CESGA, considering the methodology defined by CESGA.
- Task 3: Detailed definition of the cases.
- Task 4: Implementation of the benchmark cases in the environments available at CESGA.
- Task 5: Design of a prototype results repository that fits the analysis needs of these.
- Task 6: Implementation of a prototype repository and its evaluation.
- Task 7: Final report.

The milestones are usually associated to the results of one of several of the 7 tasks that composed the project. The deliverable of a milestone is always a document describing the workforce effort and the tasks done, as well as other documents describing the results, and in some cases, the source code generated. The 4 milestones were:

- Milestone 1: Completion and contribution of the result of task 1: The result will be an updated study of the state of the art.
- Milestone 2: Completion and contribution of the result of task 3. The result is a report with a detailed definition of the cases.
- Milestone 3: Completion and contribution of the results of tasks 4, 5, and 6. The result to be provided is a copy of the developed code and the associated documentation, in digital format.
- Milestone 4: Contribution of the final report.

The results and documentation associated to each milestone were delivered on the following dates:

- Milestone 1: November 30th, 2022
- Milestone 2: March 31st, 2023
- Milestone 3: July 31st, 2023
- Milestone 4: It is the milestone associated to this document and it is due by August 31st, 2023

The workforce of the contract in CITIC was composed by 5 researchers, two of which acted as principal investigators of the project.

- Dr. Diego Andrade Canosa (**DAC**), co-IP of the research work and researcher at CITIC-UDC.
- Professor Juan Touriño Domínguez (**JTD**), co-IP of the research work and researcher at CITIC-UDC.
- Professor María José Martín Santamaría (**MJMS**), researcher at CITIC-UDC.
- Professor Basilio Bernardo Fraguela Rodríguez (**BBFS**), researcher at CITIC-UDC.
- Miss Oluwatosin Esther Odubanjo (**OEO**), researcher at CITIC-UDC.

The effort of the team is reported as a percentage of their total working time. The effort was also reported separately for each milestone, being:

- Milestone 1:
  - DAC: 11% of the time dedicated to the project
  - JTD: 3% of the time dedicated to the project
  - MJMS: 3% of the time dedicated to the project
  - BBFS: 3% of the time dedicated to the project
- Milestone 2:
  - DAC: 11% of the time dedicated to the project
  - JTD: 3% of the time dedicated to the project
  - MJMS: 3% of the time dedicated to the project
  - BBFS: 3% of the time dedicated to the project

- Milestone 3:
  - DAC: 11% of the time dedicated to the project
  - JTD: 3% of the time dedicated to the project
  - MJMS: 3% of the time dedicated to the project
  - BBFS: 3% of the time dedicated to the project
  - OEO: 100% of the time dedicated to the project
- Milestone 4:
  - DAC: 11% of the time dedicated to the project
  - JTD: 3% of the time dedicated to the project
  - MJMS: 3% of the time dedicated to the project
  - BBFS: 3% of the time dedicated to the project
  - OEO: 100% of the time dedicated to the project

3 Results of the Project

The results of the project can be identified with the 7 tasks of the project.

- Result 1: Study of the state of the art on benchmarking of quantum computers, associated with Task 1 and reported as part of the results of milestone 1 (see Appendix 1)
- Result 2: Selection of the cases to be implemented following the benchmarking methodology proposed by CESGA. This result is associated with Task 2 and was reported as part of the results of milestone 2 (see Appendix 2).
- Result 3: Detailed definition of the selected cases. This result is associated with Task 3 and was reported as part of the results of milestone 2 (see Appendix 2).
- Result 4: Implementation of the cases. This result is associated with Task 4 and was reported as part of the results of milestone 3 (see Appendix 3).
- Result 5: Design and implementation of a prototype of the results repository. This result is associated with Task 5 and 6 and was reported as part of the results of milestone 3 (see Appendix 3).
- Results 6: This final report.

  Now, we make a detailed and separated comment on Results 1-5.

3.1 Result 1: Study of the state of the art

Current Quantum Architectures are based on the Noisy Intermediate-Scale Quantum technology (NISQ), permitting only about 50 to a few hundred qubits [27]. The Quantum Architecture (QA) has high degrees of freedom ranging from the qubit's technology and operations to the diversity of possible applications to the stack of the QA; each component varies independently, as well as widely, and their interactions affects the rate of performance which a quantum system may possess at a time, thus resulting in a complex and varied behaviour of the quantum system. Perhaps the most influencing degrees of freedom are the internal quantum noise and the external environment, which affects the stability of the quantum operations, coherence time, and the correctness of results. Admist the opinion that Quantum Architecture Benchmarking is a too-early initiative [3], current research shows researchers proposing methodologies to either isolate ([33], [8], [20], [17]), randomize ([31], [35], [5], [18], [15], [23], [29]) and even treat as a whole ([24], [7]) the degrees of freedom of the QA.

Quantum Architecture Benchmarking (QArB) can be broadly classified into Quantum Operation Benchmarking (QOB), Quantum Application Benchmarking (QAB) and Quantum Synthetic Benchmarking, QSB;

Quantum Operation Benchmarking (QOB): QOB is concerned with testing how reliable a quantum operation is when applied to a quantum workload in the presence of quantum noise and imperfect controls [33], it aims at providing information pertaining to the practicality (accuracy), fidelity and effectiveness (error rates) of quantum operations / gates. Benchmark results can be obtained for techniques of: Quantum Process tomography, Quantum State tomography, Gate set tomography, Direct Fidelity Estimation, randomized benchmarking and cycle benchmarking [31], [7], [9].

1. **Quantum Process Tomography (QPT):** This technique is used to reconstruct the complete description of a quantum operation, including how it affects different input states. It provides a detailed characterization of the behavior of quantum gates. An adaptation and a more efficient variant of QPT is Compressed Quantum Process Tomography (CQPT), which is designed to overcome some of the challenges posed by the computational complexity of full tomography.

CQPT aims to reduce the number of measurements required for tomography by using clever techniques, often inspired by concepts from compressed sensing and machine learning. These techniques allow researchers to gather sufficient information about the quantum process using fewer measurements, thus saving time and resources.

2. **Quantum State Tomography (QST):** This technique is used to reconstruct the density matrix that describes the quantum state of a qubit or a set of qubits. It helps to assess the quality and accuracy of prepared quantum states.

3. **Gate Set Tomography (GST):** This technique is an extension of QPT; it involves characterizing a complete set of quantum gates and their behavior. It goes beyond analyzing individual gates and considers the interactions and relationships among multiple gates.

4. **Direct Fidelity Estimation (DFE):** This technique is used to directly estimate the fidelity or similarity between a desired quantum operation and the actual operation executed on a quantum device. It provides a way to quantitatively measure how well a quantum gate or operation matches the intended ideal behaviour.

5. **Randomized Benchmarking (RB):** This technique is a scalable technique for quantitative characterization of the probability of error rates of quantum operators (gates) [31], [35] by using random sequences of gate operations and measuring average sequence fidelity of these operations, [35], [22]. By analyzing the fidelity decay as sequence length increases, it provides valuable insights into the error rates and performance of quantum operators (gates). RB can be approached from various standards and techniques [5], [18], [15], [23], [29].

6. **Cycle Benchmarking (CB):** This is a practical and scalable technique for characterizing the errors and fidelity of a quantum process by examining the global and local properties of cycles in a quantum circuit.

Quantum Application Benchmarking (QAB): QAB is user-inspired and uses real-world applications to characterize the performance of a QA.

Quantum Synthetic Benchmarking (QSB): QSB characterizes specially structured and constructed workloads with specific properties to stress a specific component in the Quantum Architecture, as such these workloads can be artificially generated or may be inspired by applications. Depending on performance assessment, their holistic degree can be influenced by the decision to focus on specific components of the QA stack, which means that certain components can be isolated and tested as in the form of a controlled experimentation. In this case, an understanding of individual components of a QA is obtained, but it does not provide a comprehensive assessment of a QA's overall capabilities, leading to a less holistic evaluation. The holistic degree QSB, can be extended by taking into account the full stack of the QA. Benchmark results can be obtained for techniques of: Heavy Output Generation Benchmarking, cross-entropy Benchmarking, [24], [8].

We have used this definition to capture QSB in the context of the structure and specific properties of the workloads as well as its holistic degree which is a measure of controlled experimentation (hence, it is slightly different from and an extension of the definition in [33]), thereby allowing us to further divided QSB into headings we have termed Quantum Random Modeled Benchmarking (QRMB) and Quantum Application Modeled Benchmarking (QAMB).

1. **Quantum Random Modeled Benchmarking (QRMB):** This category of QSB characterizes benchmarks that are based on random model of circuits; these quantum workloads are artificially generated, which means that they are not based on real-world applications and they are designed to focus on components of the QA stack at the lower level (such as gates,

error correction, or noise characteristics), and the compilation level of the QA stack.

The IBM's Quantum volume, QV, metric evaluation, [7], is an example of QRMB that provides a holistic measure [34], of the *degree of generality* [1] of a quantum computer; it takes into account hardware performance such as fidelity, error rates, crosstalk, etc.; design parameters such as connectivity of qubits and gate set and also includes the software for circuit optimization. It fills in the gap of techniques of QOB techniques which do not account for errors generated by the interactions of spectator qubits [2]. The main drawback of this benchmark is in its random-fixed-generic nature of the quantum circuit, non-scalability and restriction to the square structured quantum circuit model used.

Another example QRMB is the quantum LINPACK [8], which is analogous to the classical LINPACK benchmark; it measures performance of a QA to solve random systems of linear equations, using the RAndom Circuit Block-Encoded Matrix (RACBEM) as a model. This benchmark does not require an explicit compilation, hence it offers controlled experimentation; however it still makes use of a random matrix model and a random quantum circuit model, hence it does not reflect the behaviour of real-world applications and is also not scalable.

2. **Quantum Application Modeled Benchmarking (QAMB):** This category of QSB characterizes benchmarks that are pillared on real-world applications and reflect actual scenarios relevant to various fields; they are based on application-modeled circuits that have a systematic structure. However, they are not real-world applications but are constructed to evaluate QA's on tasks that have practical significance and hence are more holistic in nature than QRMB's. They aim to bridge the gap between theoretical capabilities and practical impact, allowing for the extrapolation of likely real-world application performance from the benchmark results. An example work is the work of (Daniel Mills, et al., 2021) [24], which uses a set of quantum class circuits inspired by structured circuits which reflects practicality in near-term machine learning and chemistry applications, random circuits, product formula circuits, and VQE state preparation circuits. This work takes inspiration from the IBM's quantum volume metric [7] for square modeled circuits and extends to models for deep and shallow quantum circuits reflecting practical significance. The proposed methodology is based on modifying the variable components of the QA stack, therefore, it offers controlled experimentation.

All benchmarking styles complement each other, they are not mutually exclusive, thus, there can be an overlap between them: QOB can be applied to user-inspired or component-inspired quantum workloads when assessing the performance of quantum gates in the workloads; QAB is user-inspired, so it is realistic and reflects the actual user experience; however, it can be time consuming especially in the event of repeating performance measurements, and it is also restrictive, as it is harder for the experimenter to isolate specific component of the QA from the application and in the case of hybrid quantum-classical applications, the classical section from the quantum section. On the other hand, QSB, is not restrictive; the experimenter is allowed to isolate and test specific components of the QA resulting in a controlled experimentation, which also allows for easier repeatability of performance measurements; however, it may be unrealistic and may not reflect actual user experience, and it may also be limited to a specific component of the QA, that is, it may be holistic or not holistic.

More specifically, depending on the measured performance metric, the structure of the quantum circuit evaluated and the target QA stack component (figure, 1), QArB be application-centric, compiler-centric, aggregated / holistic, and physical/hardware-centric.

[1] the degree to which the QA performs as a general purpose QA

[2] qubits that are not actively involved in the computation itself but are instead used to monitor or measure the state of other qubits

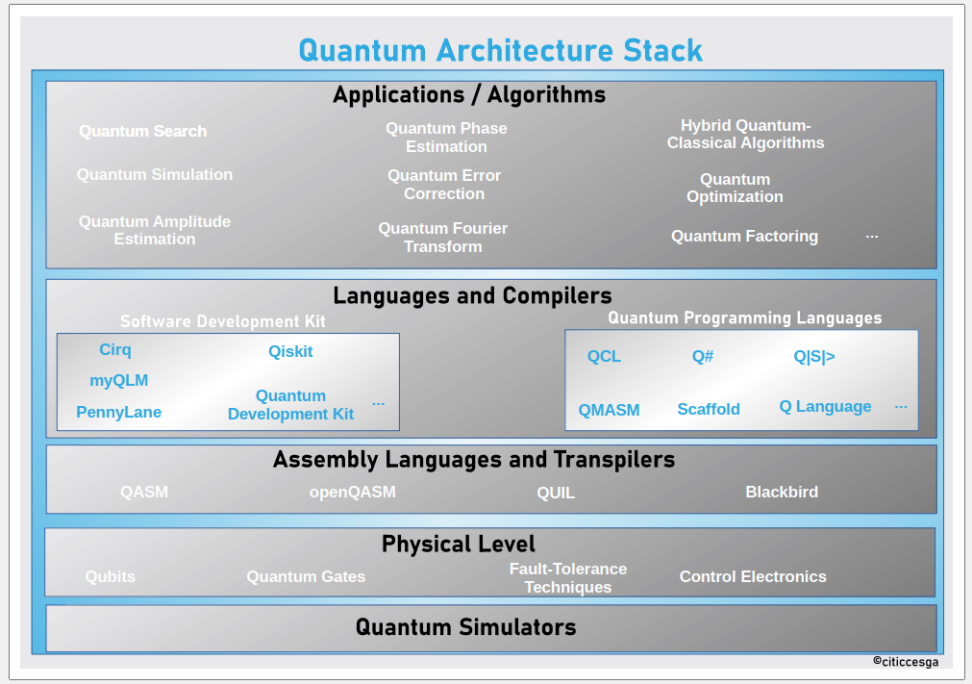

Figure 1: Quantum Architecture Stack

3.1.1 Application-Centric Benchmarking

This type of Benchmarking falls under the QAB category of QArB. Quantum applications and algorithms are the pivots of the benchmark; they are used to characterize the performance of a Quantum system in terms of its efficiency and effectiveness in executing a specific quantum workload using application-level performance metrics such as total execution time, quantum execution time, best approximation error, Q-Score, performance and accuracy, etc. Application-Centric Benchmarks are contextual, giving information about the performance of a system based on real-world applications; hence, they are informative to the end-users and investors.

The work of (Teague Tomesh, et al., 2022), SupermaQ [33], is an open-source scalable, hardware-agnostic quantum benchmark suite which applies techniques from classical benchmarking to quantum application benchmarking. In this work, the authors quantify the coverage of the suite using feature vectors; with selected, test cases addressing the issues of: entanglement, quantumness of nature, combinatorial optimization, ground state energy, and Hamiltonian simulation, benchmark results are collected from the IBM, IonQ and AQT@LBNL platforms. A major challenged faced in this work is that of varying compiler optimizations across chosen cloud services: IBM Qiskit, AWS Braket cloud services, and the Lawrence Berkeley National Lab's Advanced Quantum Testbed (AQT); the authors solve this by using the Close Division method, which specifies a restriction for optimization to obtain a lower-bound for the performance of the system.

QASMBench [20], is another open-source application-centric low-level benchmark suite based on the OpenQASM assembly representation; this suite aims at evaluating and characterizing the properties of emerging NISQ devices, estimating the potential of optimization from quantum transpilers, and benchmarking the performance of classical quantum simulators. The authors propose four circuit metrics to assess the execution efficiency, susceptibility to NISQ error, and potential gain from low-level optimizations: gate density, retention lifespan, measurement density, and entanglement variance. Selected test cases are from the domains of chemistry, simulation, linear algebra, searching, optimization, arithmetic, machine learning, etc. For measuring performance, the authors

measure the execution fidelity of a subset of QASMBench applications on 12 IBM-Q machines using density matrix state tomography, which is made up of 25K circuit evaluations; they also compare the fidelity of executions among the IBM-Q machines, the IonQ QPU and the Rigetti Aspen M-1 system.

3.1.2 Compiler-Centric Benchmarking

This type of benchmark is concerned with the compiler component of the QA stack; the work flow of the quantum compilation process is studied for possible optimization. The work of (Y. Kharkov, et al., 2022), Arline [17] is an example of an open-source compiler-centric benchmark for NISQ devices, it compares several quantum compilation frameworks based on a set of metrics such as the compression factor, post-optimization gate counts, circuit depth, hardware-dependent circuit cost function, compiler run time etc. In this work, the authors perform a variety of compiler tests for random and structured circuits for quantum algorithms in the domain of chemistry, finance, quantum dynamics and searching, using compilation subroutines and pipelines from Qiskit, Pytket, PyZX, Cirq and Arline Benchmarks. Benchmark reports are obtained for Rigetti Aspen 16q, IBM Rueshlikon 16q, IBM Falcone 27q and Google Sycamore 53q quantum hardware architectures.

3.1.3 Aggregated / Holistic Benchmarking

This approach of benchmarking is concerned with characterizing the performance of QA as a whole. IBM benchmark metrics are examples of this benchmarking framework: the quantum volume metric ([2], [7]) for summarizing the performance of a QA based on the number of qubits, number of gates qubit connectivy and number of operations as well as the holistic speed benchmark, the Circuit Layer Ops Second (CLOPS) metric for evaluating the number of QV circuits that can run on a QA in a given time. Other examples of a holistic benchmark are the application-inspired benchmark (Daniel Mills, et al., 2021 [24]) and the mirror circuit benchmarks proposed by [28].

3.1.4 Volumetric Benchmarking (VB)

VB is a structural benchmarking framework. Taking inspiration from IBM's QV metric, it defines a family of rectangular quantum circuits for studying time/space performance trade-offs [4]. The VB framework studies a set of random, periodic and application circuits used in quantum operation, synthetic and application benchmarking approaches. For example, the work of (Timothy Proctor, et al., 2022) shows the mirror circuit benchmarks executed under the framework of VB.

3.1.5 Hardware-Centric Benchmarking

Is concerned with techniques from Quantum Operation Benchmarking, QOB, for the characterization of: 1. quantum operators and operations in terms of error rates, fault tolerance and fidelity, and 2. classical control hardware.

1. **Quantum Operator and Operations Characterization, QOOC:** This evaluation provides insights into how well the quantum hardware is performing in terms of executing quantum operations accurately.

2. **Classical Control Hardware Characterization, CCHC:** This is the evaluation of the classical control infrastructure that interacts with the QA. It is important for: 1. Effectively mapping qubit operations to physical qubit. 2. Coordinating the sequence of quantum operations performed on qubits. 3. Finding an optimal sequence for the execution of quantum operations to minimize errors, improve fidelity, or achieve specific computational goals. 4. Monitoring the behaviour of quantum operations to detect deviations from the ideal operation.

3.2 Result 2: Selection of the cases

The benchmarking task defined within the scope of this contract is aimed at the quantitative evaluation of the technological advancement in Quantum Hardware Research (QHR), with a focus on real-world applications and their use cases in industry. A collection of test cases for characterizing the performance of a QA is defined within the framework of an heterogeneous benchmark suite[3] and is based on quantum subroutines which are often used: 1. in the domains of quantum chemistry, finance, machine learning; 2. in quantum algorithms such as the shor's algorithm, quantum algorithm for solving linear systems of equations, the VQE, quantum principal component analysis, amplitude estimation algorithms, e.t.c.

We characterize these quantum subroutines as kernels and define them on the basis of a high-level definition, using a mathematical and algorithmic approach that is agnostic of the QA. Hence, each test case in the benchmark suite is defined by a kernel (quantum subroutine) marked by the following yardsticks:

Yardstick 1 - Application: The kernel must be generic; thus can be applied across several quantum algorithms and applications.

Yardstick 2 - Definition: The kernel can be structured and constructed mathematically or algorithmically and can be defined for a benchmark test case that characterizes scalability and accuracy (yardsticks 3 and 4).

Yardstick 3 - Computation: The kernel can be defined for a benchmark test case which is scalable, in other words, can be executed for an increasing number of qubits. While, making it possible for a classical simulation to ensure verification of results (see yardstick 4).

Yardstick 4 - Verification: The kernel can be defined for a benchmark test case to stress specific metrics that characterize accuracy. The obtained result from executing the benchmark test case defined by the kernel must be verifiable analytically or by classical simulation.

Yardstick 5 - Performance: A set of metrics can be defined to characterize execution performance of the benchmark test case.

In the next section, 3.3, we give a detailed definition of the test cases: Probability Loading, Amplitude Estimation and Quantum Phase Estimation.

3.3 Result 3: Detailed definition of the cases

In this section, we give a detailed definition of the test cases with respect to the **yardsticks** of the kernel defined in section 3.2

3.3.1 Probability Loading (PL)

The **PL Benchmark Test Case** is characterized by the **PL kernel** and fulfills all yardsticks presented in section 3.2.

Application: The **PL kernel** is generic to many different quantum algorithms like the Harrow-Hassidim-Lloyd (**HHL**) [14], quantum principal component analysis (**PCA**) [21], quantum amplitude estimation algorithms[6], etc. In these quantum algorithms, the **PL kernel** is an initialization step; it is computationally intensive because its number of operations typically scales as $\sim 2^n$, where n is the number of qubits to be initialized.

[3] a suite comprising a set of test cases from diverse domains

3.3 Result 3: Detailed definition of the cases

Definition: Given a list of normalized vectors,

$$\mathbf{V} = \{v_0, v_1, \cdot, v_{2^n-1}\}, v_i \in \mathbb{C}^{2^n}; \sum_{i=0}^{2^n} |v_i|^2 = 1 \tag{1}$$

The main task of the **PL kernel** is to generate an operator **U**, from the normalised vector **V**, which satisfies the equation:

$$\mathbf{U}|0\rangle_n = \sum_{i=0}^{2^n-1} v_i |i\rangle_n \tag{2}$$

Hence, the **PL kernel** described above, can be used to define a **Benchmark Test Case** for loading of a fixed normal probability density function (**PDF**). Such that, the equation 1 can be re-expressed as:

$$\mathbf{P} = \{p_0, p_1, \cdot, p_{2^n-1}\}, p_i \in [0, 1]; \sum_{i=0}^{2^n} |p_i|^2 = 1 \tag{3}$$

thus, the equation 2 then becomes:

$$\mathbf{U}|0\rangle_n = \sum_{i=0}^{2^n-1} \sqrt{p_i} |i\rangle_n \tag{4}$$

Computation: For n-number of qubits, the action of the operator **U** is translated into a quantum program. The Operator, **U** is obtained by encoding a list of normalized probabilities, $\mathbf{P_{norm}}(\mathbf{x})$, into a n-qubits quantum circuit - this list of normalized probabilities, $\mathbf{P_{norm}}(\mathbf{x})$ is created from an array, $\mathbf{x}$, which is bounded by a fixed mean, $\tilde{\mu}$ and fixed standard deviation, $\tilde{\sigma}$; it serves as an input to be loaded into a particular **PL** algorithm (for example, algorithms in [32], [12]), which generates the **U** operator. The quantum program can be represented mathematically as:

$$\mathbf{U}|0\rangle_n = \sum_{i=0}^{2^n-1} \sqrt{P_{norm}(x_i)} |i\rangle_n \tag{5}$$

By executing the quantum program, $\mathbf{U}|0\rangle_n$ in equation 5, and measuring all the n-qubits n_{shots} $(= (min(10^6, \frac{100}{min(P_{norm}(x_i)}))$ times, a measured probability distribution, **Q**, is obtained. **Q** is defined as the ratio of the the number of times in which a state $|i\rangle_n$ is obtained to the number of shots, n_{shots}.

Verification: A characterization of accuracy for the **PL Benchmark Test Case** is obtained by comparing the values of the measured probability distribution, **Q** with that of the normalized probability distribution, $\mathbf{P_{norm}}$, using two metrics:

- *The Kolmogorov-Smirnov*, (**KS**), between **Q** and $\mathbf{P_{norm}}$.

$$\mathbf{KS} = \max\left(\left|\sum_{j=0}^{i} P_{norm}(x_j) - \sum_{j=0}^{i} Q_j\right|, \ \forall i = 0, 1, \cdots, 2^n - 1\right) \tag{6}$$

3.3 Result 3: Detailed definition of the cases

- *The Kullback-Leibler divergence.*

$$\mathbf{KL(Q/P_{norm})} = \sum_{j=0}^{2^n-1} P_{norm}(x_j) \ln \frac{P_{norm}(x_j)}{\max(\epsilon, Q_k)} \tag{7}$$

where $\epsilon = \min(P_{norm}(x_j)) * 10^{-5}$ which guarantees the logarithm exists when $Q_k = 0$

Also, a characterization of relationship between $\mathbf{Q}$ and $\mathbf{P_{norm}}$, is obtained with the verification metric - the Chi- Square test, χ^2, which generates a p-value from $n_{shots}\mathbf{Q}$ and $n_{shots}\mathbf{P_{norm}}$. If the p-value is lower than 0.05, the obtained result from executing the Benchmark Test Case is invalid.

Performance: In addition to the metrics for characterizing accuracy and relationship, the elapsed and quantum times are obtained for executing the Benchmark Test Case. These metrics characterizes the performance of the QA executing the **PL Benchmark Test Case**.

3.3.2 Amplitude Estimation (AE)

The **AE Benchmark Test Case** is characterized by the **AE kernel** and fulfills all yardsticks presented in section 3.2.

Application: The **AE kernel** is generic to many different applications of quantum computation in the domains of finance [30, 38, 13], chemistry [19, 1], machine learning [36, 37] and can also be used for generic tasks such as numeric integration [25].

Definition: Given a unitary operator, $\mathbf{A}$, which acts on an initial n-qubit state, $|0\rangle^{\otimes n}$,

$$|\Psi\rangle = \mathbf{A}|0\rangle_n = \sum_{i=0}^{2^n-1} a_i|i\rangle_n \tag{8}$$

Now we are interested in the sub-state composed by some basis states $J = \{j_0, j_1, \cdots, j_l\}$, given in equation 9:

$$|\Psi\rangle = \mathbf{A}|0\rangle_n = \sum_{j\in J} a_j|j\rangle_n + \sum_{i\notin J} a_i|i\rangle_n \tag{9}$$

If we define the sub-states $|\Psi_0\rangle$ and $|\Psi_1\rangle$ using (10):

$$|\Psi_0\rangle = \frac{1}{\sqrt{a}} \sum_{j\in J} a_j|j\rangle_n \;\; and \;\; |\Psi_1\rangle = \frac{1}{\sqrt{1-a}} \sum_{i=0, i\notin J}^{2^n-1} a_i|i\rangle_n \tag{10}$$

Then the final state, $|\Psi\rangle$, can be expressed as (11):

$$|\Psi\rangle = \mathbf{A}|0\rangle_n = \sqrt{a}|\Psi_0\rangle + \sqrt{1-a}|\Psi_1\rangle \tag{11}$$

The main task of the **AE kernel** is to find an estimation of the amplitude of $|\Psi_0\rangle$: a.

The **AE kernel** described above can be used to define a **Benchmark Test Case** for finding the integral of a function $f(x)$, in a closed interval $[a, b] \subset \mathbb{R}$; in our case the sine function, given as:

$$\mathbf{F} = \int_a^b \sin(x)dx = -\cos x|_a^b = \cos(a) - \cos(b) \tag{12}$$

Computation: For n-number of qubits, the action of the operator, $\mathbf{A}$, given in equation 11 is translated into a quantum program. The operator, $\mathbf{A}$, is constructed such that the value of the desired integral, $\mathbf{F}$, is encoded into the amplitude, $\sqrt{a}$, of the state given in equation 11 as a Riemann sum, $\mathbf{S}$ of $\mathbf{f}(x_i)$ over $[a, b] \subset \mathbb{R}$ and serves as in input to be loaded into a particular **AE** algorithm (for example, the algorithm in [6]), which returns the value of the estimated integral $\tilde{\mathbf{F}}$.

For the particular case of the sine function in equation 12, the function is first discretized to obtain $f(x_i)$ in 2^n intervals; the desired integral for each integral is then estimated as a Riemann sum, $S_{[a,b]}$:

$$S_{[a,b]} = \sum_{i=0}^{2^n-1} f_{x_i} \cdot (x_{i+1} - x_i) \tag{13}$$

Following, the discretized sine function can then be encoded into a n-qubit quantum circuit, by first normalizing $f(x_i)$ and then creating the operator $\mathbf{A}^I$ (where $I = \{0, 1\}$ is the interval where the integral can be estimated) which encodes $f(x_i)$ into the n-qubit quantum circuit and also serves as an input to an AE algorithm which estimates, $\tilde{a}$, as a Riemann sum approximation, $\tilde{S}^I_{[a,b]}$, obtained by measuring the probability of obtaining the state, $|\Psi_0\rangle$.

Verification: For characterizing the accuracy and quality of the AE estimation of the integral, two metris are used:

- *Sum absolute error*: between the *AE* estimator and the Riemann sum computed using (13): $\epsilon = |\tilde{S}^I_{[a,b]} - S_{[a,b]}|$
- *Oracle calls*: total number of calls of the operator $\mathbf{A}^I$

Performance: In addition to the metrics for characterizing the accuracy and quality of the **AE** estimation, the elapsed, run and quantum times are obtained for executing the Benchmark Test Case. These metrics characterizes the performance of the QA executing the **AE Benchmark Test Case**.

3.3.3 Quantum Phase Estimation (QPE)

The **QPE Benchmark Test Case** is characterized by the **QPE kernel** and fulfills all yardsticks presented in section 3.2.

Application: The **QPE kernel** is used in the computation of the eigenvalues of a unitary operator [16], [10]; it is generic to many different applications and algorithms of quantum computation such as quantum integer decomposition, quantum solving of linear systems, Shor's algorithm, Hamiltonian energy VQE solver, etc.

Definition: Given a n-unitary operator, $\mathbf{U}$, the eigenvalues of this operator are phases that can be represented as $e^{2i\pi\lambda_j}$ for $j = 0, 1, 2, \cdots 2^m - 1$.

For a particular *eigenstate* $|\psi_j\rangle$, then:

$$\mathcal{U}\,|\psi_j\rangle = e^{2i\pi\lambda_j}\,|\psi_j\rangle\,; 0 \leq \lambda_j < 1. \tag{14}$$

3.3 Result 3: Detailed definition of the cases

The main task of the **QPE kernel** is to obtain an estimation of the different eigenvalues λ_j.

Hence, a **Benchmark Test Case** for computing the eigenvalues, λ_j, of a n qubits unitary operator $R_z^n(\vec{\theta}) = \otimes_{i=1}^{n} R_z(\theta_i)$, for a vector of n angles $\vec{\theta} = \{\theta_i\}$ $i = 0, 1, \cdots, n-1$, is the defined case characterized by the **QPE kernel**.

Where: the $R_z(\theta)$ operator is a Z-axis rotation gate given by (15).

$$R_z(\theta) = \begin{pmatrix} e^{-i\frac{\theta}{2}} & 0 \\ 0 & e^{i\frac{\theta}{2}} \end{pmatrix} = e^{-i\frac{\theta}{2}} |0\rangle\langle 0| + e^{i\frac{\theta}{2}} |1\rangle\langle 1| \tag{15}$$

And the angle θ is selected based on a random approach between the interval $[0, \pi]$ or the exact approach in which the angle selection depends on the number of auxiliary qubits for the **QPE** and is given by (16) where $\delta\theta = \frac{4\pi}{2^m}$ and a will be a binary random variable that can be $\{-1, 1\}$

$$\theta_{i+1} = \theta_i + a * \delta\theta \tag{16}$$

Computation: A reference probability distribution, $P^{th}_{\lambda,m}(\frac{k}{2^m})$, is computed by obtaining the associated eigenvalue λ_j for each 2^n possible state, $|i\rangle$ and generating a histogram for a complete list of of eigenvalues, λ^i. The histogram has a range given as [0, 1], a 2^m number of bins labeled as $\frac{k}{2^m}$ (where k is the number of the bin [k = 0, 1, 2 ,..., $2^m - 1$]) and the frequency of the eigenvalues in each bin, k is computed as $f_{\lambda k}$

Similarly, for n-number of qubits, a quantum circuit is executed and measured n_{shots} ($= \frac{1000}{0.81 \times f_{\lambda lf}}$; $f_{\lambda lf}$ = less frequent eigenvalue) times for generating a complete list of eigenvalues, λ_j^{QPE}, of the $R_z^n(\vec{\theta})$ operator loaded into a **QPE** algorithm initialized to an equiprobable combination of all 2^n possible states given as

$$|\Psi_0\rangle = \frac{1}{\sqrt{2^n}} \sum_{i=0}^{n-1} |i\rangle \tag{17}$$

This complete list of eigenvalues computed by the $QPE(\lambda_j^{QPE})$ is then used to draw a histogram which generates a measured probability distribution of the eigenvalues $P^{QPE}_{\lambda,m}(\frac{k}{2^m})$, obtained by the **QPE** algorithm. The histogram has the same properties as that of the reference probability distribution.

Verification: The **QPE Benchmark Test Case** is characterized for accuracy by comparing the two discrete probability distributions, $P^{th}_{\lambda,m}$ and $P^{QPE}_{\lambda,m}$ using two metrics:

- The Kolmogorov-Smirnov (KS) between $P^{th}_{\lambda,m}$ and $P^{QPE}_{\lambda,m}$. This is the maximum of the absolute difference between the cumulative distribution functions of $P^{th}_{\lambda,m}$ and $P^{QPE}_{\lambda,m}$ computed following 18, This will be the comparative metric for the angles from **random** case.

$$KS = \max\left(\left|\sum_{k=0}^{i} P^{th}_{\lambda,m}(\frac{k}{2^m}) - \sum_{k=0}^{i} P^{QPE}_{\lambda,m}(\frac{k}{2^m})\right|, \forall k = 0, 1, \cdots, 2^m - 1\right) \tag{18}$$

- The *fidelity*. In this case the distributions, ($P^{th}_{\lambda,m}$ and $P^{QPE}_{\lambda,m}$) will be considered as vectors and the **fidelity** will be the cosine of the angle between them that will be computed following (19). $|P^{th}_{\lambda}|$ and $|P^{QPE}_{\lambda}|$ will be the norm of the vectors corresponding to the *theoretical* and QPE distributions respectively. This will be the comparative metric for the angles from **exact** case.

$$fidelity = \frac{\sum_{k=0}^{n-1} P^{th}_{\lambda,m}(\frac{k}{2^m}) * P^{QPE}_{\lambda,m}(\frac{k}{2^m})}{|P^{th}_{\lambda}| * |P^{QPE}_{\lambda}|} \quad (19)$$

Performance: In addition to the metrics for characterizing accuracy of the result of the **QPE Benchmark Test Case**, the elapsed and quantum times are also obtained for executing the Benchmark Test Case. These metrics characterize the performance of the QA executing the **QPE Benchmark Test Case**.

3.4 Result 4: Implementation of the cases

Each proposed **Benchmark Test Case** must be complemented by a complete QLM-compatible software implementation and a complete result report into a separate JSON file.

A complete description of the QLM-reference implementation, the execution procedure as well as the benchmark report format of the Benchmark Test Cases: Probability Loading, PL, Amplitude Estimation, AE, and Quantum Phase Estimation, QPE, can be found in Appendix 2. The QLM-reference implementation of the Probability Loading and Amplitude Estimation Benchmark Test Cases is based on the Quantum Quantitative Finance Library (QQuantLib), which allows for the implementation of QLM-compatible Probability Loading algorithms and several Amplitude Estimation algorithms used in quantum finances ([11], [13], [26]) as well as the computation of integrals and the expectation value of functions, while that of the Quantum Phase Estimation Benchmark Test Case is based on the rz library which allows for the computation of the eigenvalue of a n qubits Z-axis rotation unitary operator.

3.5 Result 5: Design and implementation of a prototype of the results repository

The Quantum Computing Benchmark Repository (QCBencRepo) is a centralized storage location to collect and store the results of the benchmark suite reported for different Quantum Computing (QC) platforms by different manufacturers or operators.

3.5.1 Design of the architecture and main components

The repository is composed of:

- **Web Front-end** - The visual and interactive aspect of the web application that users can see and interact with through their web browsers. It's implementation is with the React framework.
- **REST API**: This is an interface that allows for the interaction between web services and Python Clients and also for reporting the benchmark results of the suite. It is implemented with the Django framework, equipped with a legacy implementation for a user authentication system and for an OAuth authorization system; the user authentication system allows manufacturers or reporters to register in the repository, while the OAuth authorization system is responsible for user identification in the REST API as well as for the user submission of authorized requests.
- **Database (DB)**: For storing information. Since, the benchmark suite generates the results as JSON files, the database system is divided into
  1. Non-Relational (NoSQL) DB : To store JSON files
  2. Relational (SQL) DB: To store the rest of the information.

3.5.2 Web Interface Design

The web user interface (UI) is a front-end whose back-end is the same REST API used for reporting, retrieving, and updating the information of the SQL and NoSQL databases of the repository; it serves as a visualization platform for private reporting and public report viewing and also allows a reporter (user) to sign up in order to report benchmark results obtained for a QC platform. The benchmark results for a QC platform are directly reported using a Python script through the REST API.

Through the web UI, registered users can access a private area section and a public area section, while non-registered users have access to only the public area section for report viewing. A report is a detailed analysis of the benchmark result for a QC platform; a report that can be viewed publicly is one that has been previously signed and submitted by a reporter. Generally, a report can be classified as:

1. **Private report:** This is a signed analysis of a benchmark result for a QC platform of a registered user, with read-write rights to the registered user and read-only rights to a non-registered user. It is accessible through the private area section of the web UI.
2. **Public report:** This is a signed and submitted analysis of a benchmark result for a QC platform of a registered user with read-only rights to a non-registered user and read-write rights to a registered user. It is accessible through the public area section of the web UI.

Finally, the categories of users that can access benchmark reports are explained explicitly as follows:

1. **Registered Users**: The user authentication system allows manufacturers or reporters to register in the repository; users can register through the web interface, which also serves as a visualization platform for the user's report and the reports of other reporters. The registered users have access to the use of OAuth, which assigns a Client Key for signing all their reports; this key is stored and associated in the database with the user. Also, in addition to having access to public reports, the registered users have access to a private area section of the web UI for managing -reports, linked users and existing benchmark reports as well as for adding new users to the reporting organization.
2. **Non-registered Users:** These users only have access to public reports, which are the analysis of already reported results.

Bibliography

[1] Alán Aspuru-Guzik, Anthony D. Dutoi, Peter J. Love, and Martin Head-Gordon. Simulated quantum computation of molecular energies. Science, 309(5741):1704–1707, 2005. doi: 10.1126/science.1113479.

[2] Lev S. Bishop, Sergey Bravyi, Andrew Cross, Jay M. Gambetta, and John Smolin. Quantum Volume. 2017. URL https://storageconsortium.de/content/sites/default/files/quantum-volumehp08co1vbo0cc8fr.pdf.

[3] Robin Blume-Kohout and Kevin Young. Metrics and Benchmarks for Quantum Processors: State of Play. 2018.

[4] Robin Blume-Kohout and Kevin Young. A volumetric framework for quantum computer benchmarks. 2020. URL https://quantum-journal.org/papers/q-2020-11-15-362/.

[5] Kristine Boone, Arnaud Carignan-Dugas, Joel J. Wallman, and Joseph Emerson. Randomized Benchmarking under Different Gatesets. 2018. URL https://arxiv.org/pdf/1811.01920.pdf.

[6] Gilles Brassard, Peter Hoyer, Michele Mosca, and Alain Tapp. Quantum amplitude amplification and estimation. AMS Contemporary Mathematics Series, 305, 2000. doi: 10.1090/conm/305/05215.

[7] Andrew W. Cross, Lev S. Bishop, Sarah Sheldon, Paul D. Nation, and Jay M. Gambetta. Validating quantum computers using randomized model circuits. 2019. URL https://arxiv.org/pdf/1811.12926.pdf.

[8] Yulong Dong and Lin Lin. Random circuit block-encoded matrix and a proposal of quantum linpack benchmark. 2021. URL https://arxiv.org/pdf/2006.04010.pdf.

[9] Alexander Erhard, Joel J. Wallman, Lukas Postler, Michael Meth, Roman Stricker, Esteban A. Martinez, Philipp Schindler, Thomas Monz, Joseph Emerson, and Rainer Blatt. Characterizing large-scale quantum computers via cycle benchmarking. 2019. URL https://arxiv.org/pdf/1902.08543.pdf.

[10] Muhammad Faizan and Muhammad Faryad. Simulation and ananlysis of quantum phase estimation algorithm in the presence of incoherent quantum noise channels. 2023.

[11] Gonzalo Ferro, Alberto Manzano, Andrés Gómez, Alvaro Leitao, María R. Nogueiras, and Carlos Vázque. D5.4: Evaluation of quantum algorithms for pricing and computation of var, 2022.

[12] Lov Grover and Terry Rudolph. Creating superpositions that correspond to efficiently integrable probability distributions. arXiv e-prints, 2002. doi: 10.48550/arXiv.quant-ph/0208112.

[13] Andrés Gómez, Alvaro Leitao Rodriguez, Alberto Manzano, Maria Nogueiras, Gustavo Ordóñez, and Carlos Vázquez. A survey on quantum computational finance for derivatives pricing and var. Archives of Computational Methods in Engineering, 03 2022. doi: 10.1007/s11831-022-09732-9.

[14] Aram W. Harrow, Avinatan Hassidim, and Seth Lloyd. Quantum algorithm for linear systems of equations. Physical Review Letters, 103(15), oct 2009. doi: 10.1103/physrevlett.103.150502.

[15] J. Helsen, I. Roth, E. Onorati, A.H. Werner, and J. Eisert. General Framework for Randomized Benchmarking. 2022. URL https://journals.aps.org/prxquantum/pdf/10.1103/PRXQuantum.3.020357.

[16] Duo Jiang, Xiaonan Liu, Huichao Song, and Haoshan Xie. An survey: Quantum phase estimation algorithms. In 2021 IEEE 5th Information Technology,Networking,Electronic and Automation Control Conference (ITNEC), volume 5, pages 884–888, 2021. doi: 10.1109/ITNEC52019.2021.9587010.

[17] Y. Kharkov, A. Ivanova, E. Mikhantiev, and A. Kotelnikov. Arline Benchmarks: Automated Benchmarking Platform for Quantum Compilers. 2022. URL https://arxiv.org/abs/2202.14025.

[18] E. Knill, D. Leibfried, R. Reichle, J. Britton, R. B. Blakestad, J. D. Jost, C. Langer, R. Ozeri, S. Seidelin, and D. J. Wineland. Randomized Benchmarking of Quantum Gates. 2007. URL https://arxiv.org/pdf/0707.0963.pdf.

[19] Emanuel Knill, Gerardo Ortiz, and Rolando D. Somma. Optimal quantum measurements of expectation values of observables. Physical Review A, 75:012328, 2007. doi: 10.1103/PhysRevA.75.012328.

[20] Ang Li, Samuel Stein, Sriram Krishnamoorthy, and James Ang. Qasmbench: A Low-level Quantum Benchmark Suite for Nisq Evaluation and Simulation. 2022. URL https://arxiv.org/abs/2005.13018.

[21] Seth Lloyd, Masoud Mohseni, and Patrick Rebentrost. Quantum principal component analysis. Nature Physics, 10(9):631–633, jul 2014. doi: 10.1038/nphys3029.

[22] Easwar Magesan, J. M. Gambetta, and Joseph Emerson. Robust randomized benchmarking of quantum processes. 2010. URL https://arxiv.org/pdf/1009.3639.pdf.

[23] David C. McKay, Andrew W. Cross, Christopher J. Wood, and Jay M. Gambetta. Correlated Randomized Benchmarking. 2020. URL https://arxiv.org/abs/2003.02354.

[24] Daniel Mills, Seyon Sivarajah, Travis L. Scholten, and Ross Duncan. Application-Motivated, Holistic benchmarking for a Full Quantum Computing Stack. 2021. URL https://quantum-journal.org/papers/q-2021-03-22-415/pdf/.

[25] Ashley Montanaro. Quantum speedup of monte carlo methods. Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences, 471(2181):20150301, 2015.

[26] María Nogueiras, Gustavo Ordónez Sanz, Carlos Vázquez Cendón, Alvaro Leitao Rodríguez, Alberto Manzano Herrero, Daniele Musso, and Andrés Gómez. D5.1: Review of state-of-the-art forpricing and computation of var, 2021.

[27] John Preskill. Quantum Computing in the NISQ era and beyond. 2018. URL https://arxiv.org/abs/1801.00862.

[28] Timothy Proctor, Kenneth Rudinger, Kevin Young, Erik Nielsen, and Robin Blume-Kohout. Measuring the capabilities of quantum computers. 2022. URL https://arxiv.org/pdf/2008.11294.pdf.

[29] Timothy J. Proctor, Arnaud Carignan-Dugas, Kenneth Rudinger, Erik Nielsen, Robin Blume-Kohout, and Kevin Young. Direct randomized benchmarking for multi-qubit devices. 2018. URL https://arxiv.org/abs/1807.07975.

[30] Patrick Rebentrost, Brajesh Gupt, and Thomas R. Bromley. Quantum computational finance: Monte Carlo pricing of financial derivatives. Physical Review A, 98(2), 2018. ISSN 2469-9934.

[31] Salonik Resch and Ulya R. Karpuzcu. Benchmarking Quantum Computers and the Impact of Quantum Noise. 2021. URL https://arxiv.org/abs/1912.00546.

[32] V.V. Shende, S.S. Bullock, and I.L. Markov. Synthesis of quantum-logic circuits. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 25(6):1000–1010, 2006. doi: 10.1109/tcad.2005.855930.

[33] Teague Tomesh, Pranav Gokhale, Victory Omole, Gokul Subramanian Ravi, Kaitlin N. Smith, Joshua Viszlai, Xin-Chuan Wu, Nikos Hardavellas, Margaret R. Martonosi, and Frederic T. Chong. Supermarq: A Scalable Quantum Benchmark Suite. 2022. URL https://arxiv.org/abs/2202.11045.

[34] Andrew Wack, Hanhee Paik, Ali Javadi-Abhari, Petar Jurcevic, Ismael Faro, Jay M. Gambetta, and Blake R. Johnson. Scale, quality, and speed: three key attributes to measure the performance of near-term quantum computers. 2021. URL https://arxiv.org/pdf/2110.14108.pdf.

[35] Junchao Wang, Guoping Guo, and Zheng Shan. Sok: Benchmarking the performance of a quantum computer. 2022. URL https://doi.org/10.3390/e24101467.

[36] Nathan Wiebe, Ashish Kapoor, and Krysta M. Svore. Quantum algorithms for nearest-neighbor methods for supervised and unsupervised learning. Quantum Info. Comput., 15(3–4):316–356, 2015. ISSN 1533-7146.

[37] Nathan Wiebe, Ashish Kapoor, and Krysta M. Svore. Quantum deep learning. Quantum Info. Comput., 16(7–8):541–587, 2016. ISSN 1533-7146.

[38] Stefan Woerner and Daniel J. Egger. Quantum risk analysis. npj Quantum Information, 5(1), 2019. doi: 10.1038/s41534-019-0130-6.