Measuring what matters: A scalable framework for application-level quantum benchmarking

Willie Aboumrad, Claudio Girotto, Joshua Goings, Luning Zhao, Miguel Angel Lopez-Ruiz, Daiwei Zhu, Ananth Kaushik, Sayonee Ray, Samwel Sekwao, Jason Iaconis, Andrew Arrasmith, Andrii Maksymov, Yvette de Sereville, Felix Tripier, Far McKon, Coleman Collins, Evgeny Epifanovsky, Masako Yamada, Martin Roetteler

IonQ Inc, College Park, MD, 20740, USA

April 14th, 2026

Abstract

As quantum computing systems continue to mature, there is an increasing need for benchmarking methodologies that capture performance in terms of meaningful, application-level metrics. In this work, we present a scalable framework for application-level quantum benchmarking that is designed to support internal system evaluation and cross-platform comparison across technology providers. Our framework is guided by a set of core principles, including measurability, simplicity, scalability, and extensibility. We present 13 benchmark families that reflect realistic workloads across multiple domains. This enables the systematic evaluation of the quality of solutions, the total execution time, total used energy, as well as Time-to-Solution. The benchmarks are designed to be reproducible, interpretable across stakeholder groups, and adaptable to evolving system capabilities. The framework aims to bridge the gap between low-level performance metrics and real-world value, providing a unified approach to assessing quantum systems. The resulting benchmarks support development and validation and contribute to the foundation of industry-wide benchmarking standards.

Contents

1 Introduction

The development of application-level benchmarks is essential for systematically evaluating and communicating the performance of quantum computing systems. Benchmarks serve multiple roles across internal development, external communication, and industry-wide comparison, each with distinct objectives and requirements. The benefits of benchmarks include the ability to monitor system readiness and guide platform development, perform engineering tests at the system level ("integration tests"), communicate system value to customers and stakeholders, and enable cross-platform comparisons across technology providers. Considerable efforts have already been made to identify and develop application-level benchmarks, see e.g. [1, 2, 3, 4, 5, 6, 7, 8]. The following principles guided the choices made in the development of the application-level benchmarking framework presented in this document:

- **Measurability**: Benchmarks must be inexpensive enough to run frequently for continuous monitoring. Physical quantities such as the execution time, execution energy, and Time-to-Solution should be reported. Metrics should be actionable and be usable to directly inform system characterization, integration testing, and improvement efforts.
- **Simplicity**: Outputs should be easy to understand, mathematically precise. It should be easy to integrate them into dashboards, internal reports, build pipelines. There should be a simple score produced by each benchmark which allows to assess performance.
- **Scalability**: Benchmarks should consist of families of instances for various input problem sizes and be able to be scaled to larger problems, covering the NISQ regime but also be extensible into the FTQC regime.
- **Extensibility**: The benchmark collection should be extensible and the community should be able to compete for each reported benchmark result and be able to report their own findings.

A key challenge in selecting benchmarks is ensuring that they capture system-level performance in ways that are meaningful to end users, i.e., they must reflect real-world computational workloads, results must be reproducible across similarly capable hardware platforms, and performance must be reported consistently in terms that connect directly to the cost and effort of obtaining a solution. To address these needs, we selected a total collection of 13 benchmark families which span a wide variety of both, underlying application technology as well as use cases. Table 1 gives an overview of the selection.

Application benchmarks should evaluate the full system stack: hardware, compiler, runtime, performance management, and software layers, all working in concert as this reflects real-world workflows. The term performance management is used in this document broadly, encompassing error mitigation, error correction, circuit optimization, and other algorithmic optimizations, each of which represents a distinct capability that can enhance how well a system executes a workload in practice. Just like with MLPerf, all techniques are acceptable, as long as they are actually available as value to users and are disclosed in the benchmark.

Component-level metrics such as gate fidelity, qubit count, and logical qubit count are critical and meaningful in their own right, but in isolation they do not capture the complete picture in terms of impact on applications or commercial value. When designing quantum computers, there is a complex trade-off space across component-level metrics driven by architectural choices. For

example, higher gate fidelity may come at the expense of slower gate speeds, increasing logical qubit count through error correction consumes physical qubits, and compiler optimizations interact with hardware constraints in ways that vary by algorithm. An application-level benchmark suite of sufficient size and coverage measures how all of these components work in concert across the full problem stack, inclusive of the hybrid quantum-classical components of the workload. This is the complete articulation of performance that component-level metrics alone fail to provide.

The framework we have chosen is inspired by MLPerf [9], the standard for AI benchmarking, due to its emphasis on fairness, transparency, and real-world relevance. Following MLPerf's structure, this framework distinguishes between two benchmark divisions:

- **Closed benchmarks** fix the implementation and enable direct cross-platform, cross-vendor comparison. The variable is the system, not the algorithm.
- **Open benchmarks** hold the success criterion constant but permit algorithmic innovation beyond the reference implementation, allowing teams to demonstrate their latest results without sacrificing intellectual property.

Both divisions are represented in the benchmark collection.[1] Like MLPerf, this framework is designed to evolve alongside the technology it measures. Benchmark problems can be proposed and launched as new application domains emerge, and existing problems can be modified or retired as hardware capabilities and relevant business problems shift. This keeps the framework grounded in what quantum systems are actually being asked to do at any given point in the technology's development. We developed a Python-based framework that comes with a code runner, a collection of instances, reference implementations of all benchmarks, as well as reference measurements of all benchmarks. The code, covering both quantum and classical components, is made available in a public repository at `github.com/ionq-publications/apps-benchmark` using a standard development environment (Qiskit), so that anyone can (a) reproduce the results on IonQ hardware, (b) run these benchmarks on other systems for independent comparison, and (c) contribute their measured performance metrics back to the benchmark. Next, we give a short overview of the main metrics reported.

Solution quality. The primary performance metric for this framework is *solution quality*, defined as the figure of merit that determines what constitutes a valid answer in a given benchmark. Solution quality takes different forms depending on the problem. For quantum chemistry benchmarks, it could be energy error relative to an exact reference. For combinatorial optimization, it is often an approximation ratio. For machine learning applications, it could be classification accuracy. The quality threshold is established before results are reported. Meeting it is the condition for a benchmark to be considered as having reached a solution. As solution quality generally is a multi-dimensional metric, we distill the overall collection of solution quality performance indicators into a single numerical score.

Execution time. Execution time measures the compute time a program spends running against the hardware, i.e., the active time the QPU spends executing the quantum programs. It inten-

[1] Note the slightly counterintuitive use of the terms "open" and "closed," which is the reverse of their meaning in the open-source software context. Closedness of a benchmark refers to the fact that the actual implementation is locked down (closed), whereas openness of a benchmark refers to the fact that only the problem definition is given and it is left unspecified (open) as to how exactly the problem is solved.

tionally does not include waiting times, pre-flight times, compilation times, and similar overheads. Execution time is best for evaluating algorithm efficiency, while time to solution dictates user experience and task deadlines. Further detail is provided in Section 1.2

Execution energy. Execution energy measures the total energy consumed to reach the solution quality threshold across both quantum and classical components. IonQ's Forte and Forte Enterprise systems are instrumented to measure power consumption using meters and overall electric energy consumption of these systems is reported.

Time-to-solution. In contrast to execution time, the time-to-Solution (TTS), often also called "wall-clock time" or "response time," measures the total elapsed time from task start to completion, including IO, waiting, and setup. We included measurements of TTS where the particular benchmark structure permits a meaningful definition of a task to be measure completion for. Typically, these tasks are more complex than a single run and may involve, for example, searching for a solution of a pre-defined quality. Some benchmarks in this suite involve hybrid quantum-classical algorithms. Both quantum execution time and classical co-processing are included in TTS measurements where applicable. Further detail and examples are provided in Section 1.3. Similar to TTS, total Energy-to-Solution (ETS) and Cost-to-Solution (CTS) can be defined.

Extensibility. The framework is designed to incorporate new metrics without structural change. An example is Cost-to-Solution (CTS), which is not currently reported. CTS treats time and energy as direct inputs to the economic cost of running a quantum workload. It connects hardware performance to procurement and operational decisions in terms that matter to quantum computing end users. We expect that future iterations of the benchmarking framework will incorporate CTS and other metrics.

1.1 Overview of the Benchmark Families

The benchmarks in this suite span optimization (QAOA, LR-QAOA, varQITE), quantum chemistry (VQE with the UpCCD ansatz, QC-AFQMC), machine learning (QCNN, quantum copula), data loading (tensor network-based image loading), simulation (Quantum Lattice Boltzmann), and foundational subroutines (QFT, Hidden Shift, Fixed-Point Amplitude Amplification, High Energy Physics). Each benchmark is described in terms of its rationale for inclusion, the algorithm itself, problem instances, and results. Hardware results are reported for IonQ systems using standardized problem instances and shot counts.

The benchmarking code is publicly available, and any party wishing to perform cross-platform comparisons may do so using the tools provided in the repository (using their own system access arrangement). Because performance degrades with circuit depth on all NISQ platforms, we present the results as a function of problem size and circuit depth so that the noise regime of each system is visible rather than obscured by selective reporting. For Open Benchmarks (where the benchmark code can be modified to demonstrate algorithmic innovation), we provide the definition of the problem and a description of the method we followed, but not necessarily IonQ's proprietary algorithm - it is up to each benchmarking participant to try and outperform through innovation. Error mitigation has been used throughout the benchmarks, following broadly the methods outlined in [10]. These methods are either already integrated into IonQ's compiler and runtime system, or will be made available in future releases.

The goal is a benchmarking framework that scales with the technology, evolves with new real-world workflows and algorithm developments, and helps the ecosystem with information they need to make real development, procurement and deployment decisions. We aim to advance a shared understanding of where quantum systems and algorithms perform today and what remains to be solved.

Algorithm	Name	Relevant Industries	Type	Division
VQE with UpCCD ansatz	Variational Quantum Eigensolver (hydrogen chains)	Pharmaceuticals Chemical Manufacturing Materials Science	Quality	Closed
QAOA	Quantum Approximate Optimization Algorithm (MaxCut)	Financial Services Logistics Energy	Quality	Closed
LR-QAOA	Linear-Ramp QAOA (MaxCut)	Financial Services Logistics Manufacturing	Quality Time to Solution	Closed
FAA	Fixed Point Amplitude Amplification	Financial Services Pharmaceuticals Cybersecurity	Quality	Closed
QCNN	Quantum Convolutional Neural Network (image classification)	Healthcare Automotive Defense	Quality	Closed
Quantum Copula	Quantum Circuit Born Machine Copula (portfolio risk/VaR)	Financial Services Insurance	Quality	Closed
Image Loading	Tensor Network-based MPS image loading	Automotive Healthcare Financial Services	Quality	Closed
QFT	Quantum Fourier Transform (Cosine, and Hidden Phase)	Cybersecurity Financial Services Pharmaceuticals	Quality Time to Solution	Closed
HSBP	Hidden Shift Benchmark Problem	Quantum Hardware Defense Government	Quality Time to Solution	Closed
VarQITE	Variational Quantum Imaginary-Time Evolution (MaxCut)	Manufacturing Energy Financial Services	Quality	Open
QC-AFQMC	Quantum-Classical Auxiliary Field Quantum Monte Carlo	Pharmaceuticals Chemical Manufacturing Materials Science	Quality	Open
QLBM	Quantum Lattice Boltzmann Method (advection-diffusion)	Aerospace Automotive Energy	Quality	Open
High Energy Physics	Neutrinoless double beta decay simulation (lattice QCD)	National Labs Defense Nuclear Energy	Quality	Open

Table 1: Overview of the families of application benchmarks analyzed in this paper. For each benchmark, we report a Quality score and the execution time. For certain benchmarks that look for a 'needle-in-a-haystack' solution, we provide Time-to-Solution (TTS) measurements. We distinguish between two Divisions: Closed benchmarks in which an open source implementation of the entire benchmark along with a complete specification of all circuits is provided, and Open benchmarks in which the target problem is specified but in which the implementation is left open.

1.2 Execution Time Definition

Execution time, represented in green in this figure, is recorded on the QPU and represents the total time the job spent on the QPU, including codegen of waveforms with templated calibration parameters, execution on-ions via the real-time subsystem, and measurement thresholding and aggregation.

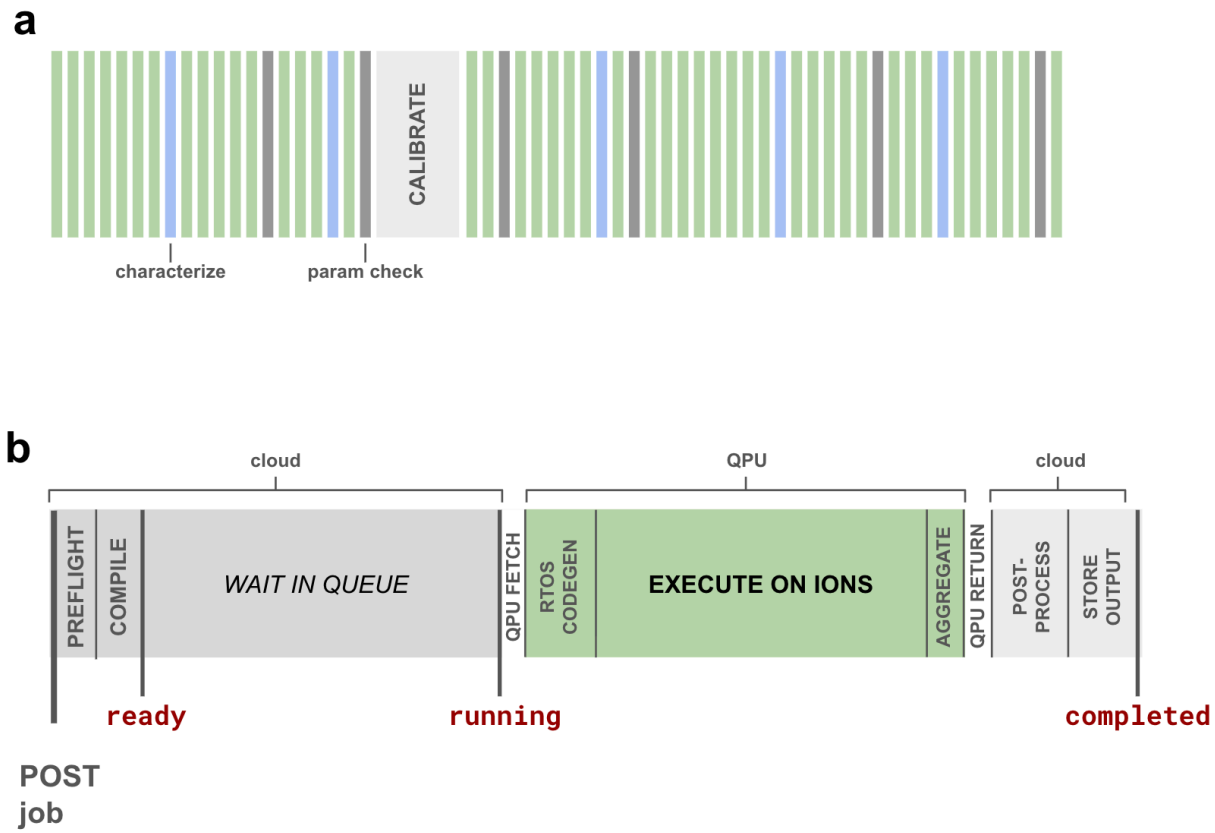

Figure 1: Illustration of execution time as reported in the framework. Shown in a) is the execution duty cycle from the perspective of the QPU — individual end-user jobs are interleaved with periodic characterization jobs (blue) and parameter check jobs (dark gray). If a parameter check fails, the calibration system kicks off a short unscheduled calibration (order minutes) to bring the parameter back into an acceptable range. Forte-class systems see approximately 40–50 minutes of aggregate execution time per hour, depending on systematic drift, job structure, and other variable factors. Shown in b) are the end-to-end steps the job goes through from submission to results retrieval. All portions shown in gray are parallelized via horizontally-scaling cloud infrastructure. In both figures, bar lengths are representative but not to scale.

1.3 Time-to-Solution Definition

Although raw shot rate is a useful measure of hardware throughput, it does not by itself determine application-level performance. For practical quantum computing, the relevant question is how

quickly a system produces a sample of sufficiently high quality, not simply how quickly it produces any sample at all. In this sense, shot quality is the more consequential metric of utility: a device that returns lower-quality samples at a high rate may be less useful in practice than one that produces high-quality samples more reliably. We therefore use Time-to-Solution (TTS), and in particular confidence time-to-first-solution, as the central metric for assessing quantum utility in real applications.

For each target quality threshold p, we estimate the probability $\hat{q}(p)$ of obtaining a sample with quality at least p from the empirical histogram of samples returned by the device. An expectation-based TTS proxy is then obtained by taking $1/\hat{q}(p)$ as the expected number of shots required to observe the first successful sample, giving

$$\mathrm{TTS}(p) = \frac{t_{\mathrm{shot}}}{\hat{q}(p)},$$

where t_{shot} is the effective time per shot. We also compute a confidence time-to-first-solution, defined as the runtime required to obtain at least one sample of quality at least p with confidence level c. Assuming independent shots, this yields

$$\mathrm{TTS}_c(p) = t_{\mathrm{shot}} \frac{\log(1-c)}{\log\big(1-\hat{q}(p)\big)}.$$

This confidence-based TTS is the primary metric used below, as it captures the quantity most relevant to end users: the time required to obtain a solution of practical value.

1.4 Organization of the Results

The rest of the paper is structured as follows: In Section 2 we present the methodology and the results of 9 Closed benchmarks. Python code for these benchmarks, including a benchmark running for streamlined execution and automatic computation of scores is available as described in Appendix A. In Section 3 we present the methodology and the results of 4 Open benchmarks. Appendix A also provides further detail on how to contribute your own measurements and the requirements on reporting in the Closed and Open benchmark divisions. Appendix B has a collection of measurements that were obtained on IonQ's quantum hardware.

2 Results: Closed Benchmarks

2.1 VQE for Chemistry with Unitary Pair-Coupled Cluster Doubles (UpCCD)

2.1.1 Rationale

This benchmark measures a quantum system's ability to compute molecular ground-state energies (specifically, the energy of hydrogen chains at their lowest quantum state)[11]. The core task is to execute a deep Variational Quantum Eigensolver (VQE) quantum circuit with pre-optimized parameters and return the expectation value $\langle\Psi| H |\Psi\rangle$ within chemical accuracy (± 1.6 mHa).

The benchmark addresses the electronic structure problem, a foundational challenge in quantum chemistry focused on calculating a molecule's ground-state energy. We use one-dimensional chains of hydrogen atoms as our model system. While seemingly simple, these hydrogen chains are

a conventional and powerful tool for studying electronic correlation (the quantum entanglement between electrons that makes many molecules classically intractable)[12]. By tuning the chain length and atomic spacing, we can create a spectrum of problems that range from weakly- to strongly-correlated, mimicking the diverse challenges found in industrial applications like drug discovery and materials science.

To rigorously benchmark the core quantum hardware, the VQE protocol is designed as an optimization-free test. By providing pre-optimized parameters, we isolate and measure the system's ability to execute a deep circuit faithfully. This directly assesses the accumulation of hardware errors—the true bottleneck for achieving chemical accuracy and near-term quantum advantage.

2.1.2 Algorithm Description

The UpCCD ansatz [13, 14, 15] constructs the quantum state through a specific pattern of two-electron excitations.

This operator moves electron pairs from occupied orbitals (i) to virtual orbitals (a), maintaining spin pairing throughout. The restriction to paired excitations (what physicists call the "seniority-zero subspace") dramatically reduces circuit complexity while capturing the dominant physics of closed-shell molecules under bond breaking scenarios.

The circuit depth scales favorably with system size. For N qubits[2], single-qubit gate counts go as $(N^2 + 7N)$ while two-qubit gates go as N^2. This polynomial scaling contrasts with the exponential growth of the full configuration space, making UpCCD tractable for near-term devices.

The molecular Hamiltonian, after mapping to qubits via the paired-electron formalism, is partitioned for efficient measurement. All diagonal terms (containing only Z and Identity operators) can be measured simultaneously in the computational basis. The off-diagonal terms (the XX and YY components) are grouped separately and require basis-change rotations before measurement. This well-known grouping strategy limits the total measurement overhead to a small, constant number of circuit executions (three in this case), regardless of the molecule's size. This is in contrast to the more general UCC Ansätze where the number of terms to measure grows prohibitively.

The benchmark measures success through the absolute error in energy: Error $= |E_{\text{measured}} - E_{\text{DOCI}}|$ Here, E_{DOCI} represents the exact ground-state energy within the paired-electron model. Because UpCCD operates entirely within this restricted space, doubly-occupied configuration interaction (DOCI) provides the theoretical limit of accuracy. Any deviation from E_{DOCI} directly measures hardware imperfection rather than ansatz approximation.

2.1.3 Problem Instances

The specific instances consist of linear chains of 2 to 18 hydrogen atoms, with the interatomic distance varying from 0.75 to 2.0. Each instance is delivered as a self-contained JSON file that encodes the complete computational task, including the molecular geometry, the qubit Hamiltonian, and importantly, the pre-optimized ansatz parameters, which eliminates classical optimization variance and ensures reproducible benchmarking.

[2] At half-filling, as is the case for the hydrogen chains in minimal basis STO-3G. This is an upper bound, though the asymptotics hold for the average case as well.

Code Listing 1: A representative JSON data structure for a problem instance from the quantum chemistry benchmark suite. This example, h002_chain_1_25, defines the simulation of a 2-qubit hydrogen molecule (H2) at a 1.25 bond distance using the STO-3G basis. The data includes the qubit Hamiltonian, optimizer settings, and final results from a noiseless Variational Quantum Eigensolver (VQE) simulation.


```
{
"benchmark_category": "chemistry",
"problem_type": "hydrogen_chain_vqe",
"instance_name": "h002_chain_1_25",
"solution_algorithms": ["vqe_puccd"],
"num_qubits": 2,
"data": {
  "geometry": [["H", 0.0, 0.0, -0.625], ["H", 0.0, 0.0, 0.625]],
  "description": "H2 1D chain, 1.25 Angstroms",
  "basis": "sto-3g", "mapper": "PairedElectron",
  "hf_energy": -0.989113814090892, "nuclear_repulsion_energy": 0.42334176873600005,
  "num_spatial_orbitals": 2, "num_alpha": 1, "num_beta": 1,
  "paired_hamiltonian_dict": {
    "II": -0.13566483543472385,
    "IZ": 0.22512501915719244,
    "ZI": -0.14722118724272235,
    "ZZ": 0.4811027722562538,
    "XX": 0.10655120065707539,
    "YY": 0.10655120065707539},
  "reference_energy_doci": -1.045783144549802, "reference_energy_fci":
      -1.0457831445498016,
  "optimizer_config": {...},
  "vqe_final_energy": -1.0457831445494497, "optimal_parameters": [0.25990952197965067],
  "optimization_info": {...},
 }
}
```


The difficulty of these instances scales in two ways. First, increasing the system size (the number of atoms) requires deeper circuits with quadratically more gates, amplifying the effects of decoherence and gate errors. Second, stretching the bond lengths (particularly in the $1.5 - 2.0$ range) increases the electronic correlation, creating more complex quantum states that are inherently more sensitive to hardware noise. This dual scaling of size and correlation allows the benchmark to comprehensively probe different hardware failure modes, from predictable error accumulation with gate count to the state's increased sensitivity to phase errors under strong correlation. The benchmark thus answers a precise question: given perfect classical pre-processing (optimal parameters), can a quantum system execute the required circuit faithfully enough to achieve chemical accuracy?

2.1.4 Results

The VQE benchmark sets a rigorous bar, with the "solved" criterion defined as achieving accuracy within 1 mHa of the exact solution. As this standard currently remains unmet across the industry, the benchmark provides a critical measure of hardware fidelity. To isolate and quantify hardware noise, we execute a fixed, optimized circuit and measure the absolute deviation from the known exact classical answer, ensuring minimal statistical error. This method provides a direct, unvarnished

view of a system's true error accumulation.

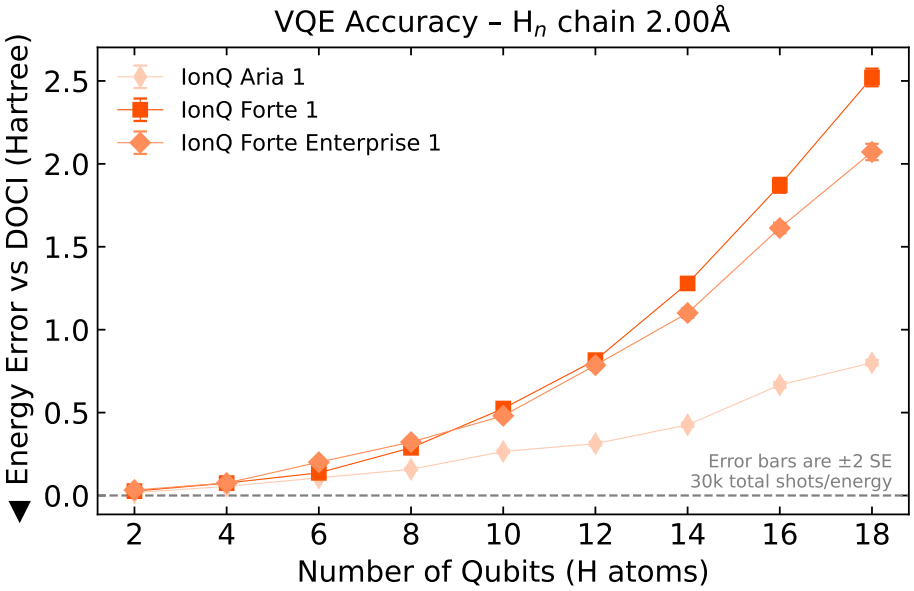

Figure 2: VQE benchmark performance for hydrogen chains at 2.00 bond distance across IonQ QPU backends. Absolute energy error relative to exact DOCI energy versus system size (2–18 qubits). Three backends are compared: `Aria-1`, `Forte-1`, and `Forte-Enterprise-1`. Error bars represent ±2 standard errors; each evaluation uses 30,000 total shots across three measurement bases (ZZ, YY, XX) by nature of the pUCCD ansatz.

2.2 Quantum Approximate Optimization Algorithm (QAOA)

2.2.1 Rationale

QAOA is a universal hybrid (classical-quantum) algorithm capable of addressing a wide range of combinatorial optimization problems relevant to many industries [16]. For this benchmark, we focus specifically on the MaxCut problem, a canonical NP-hard optimization problem that can be directly mapped onto a quantum Hamiltonian, providing a clear and rigorous testbed for evaluating quantum hardware performance.

QAOA's performance is notably sensitive to underlying hardware characteristics, such as qubit connectivity, gate fidelities, and coherence times. A key parameter influencing this sensitivity is the number of circuit layers p, which directly controls circuit depth, impacting both solution quality and susceptibility to hardware-induced errors. Therefore, QAOA serves as an effective benchmark for characterizing and comparing quantum computing systems [17].

2.2.2 Algorithm Description

QAOA is a variational approach that finds approximate solutions to optimization problems by encoding them into a cost Hamiltonian H_c and then preparing a parameterized quantum state $|\psi(\boldsymbol{\gamma}, \boldsymbol{\beta})\rangle$, to minimize the cost function $C(\boldsymbol{\gamma}, \boldsymbol{\beta})$ defined by the expectation value

$$C(\boldsymbol{\gamma}, \boldsymbol{\beta}) = \langle \psi(\boldsymbol{\gamma}, \boldsymbol{\beta}) | \, H_c \, | \psi(\boldsymbol{\gamma}, \boldsymbol{\beta}) \rangle \, .$$

The trial state, or ansatz, $|\psi(\boldsymbol{\gamma}, \boldsymbol{\beta})\rangle$ is prepared by applying p alternating layers of unitary operators to an initial uniform superposition state, $|+\rangle^{\otimes n}$. The circuit is parametrized by $2p$ angles: $\boldsymbol{\gamma} = (\gamma_1, \ldots, \gamma_p)$ and $\boldsymbol{\beta} = (\beta, \ldots, \beta_p)$. The operators of the kth layer are the cost unitary, $U(H_c, \gamma_k) = \exp[-i\gamma_k H_c]$, and the mixer unitary, $U(H_B, \beta_k) = \exp[-i\beta_k H_B]$, where H_B is chosen such that its ground state is easily prepared, e.g. $H_B = \sum_{i=1}^{n} X_i$ whose ground state is the initial state $|+\rangle^{\otimes n}$. The full p-layer QAOA ansatz is thus given by

$$|\psi(\boldsymbol{\gamma}, \boldsymbol{\beta})\rangle = U(H_B, \beta_p) U(H_c, \gamma_p) \ldots U(H_B, \beta_1) U(H_c, \gamma_1) \, |+\rangle^{\otimes n} \, .$$

While QAOA typically employs classical optimization methods to iteratively update the parameters $(\boldsymbol{\gamma}, \boldsymbol{\beta})$, this benchmark employs an optimization-free variant known as the fixed-angle conjecture [18, 19]. In this approach, pre-determined sets of angles are used, proven to yield universally strong performance guarantees for specific classes of graphs. For example, for MaxCut on 3-regular graphs, these angles are established up to $p = 11$ layers, and for 4-regular graphs up to $p = 5$. Eliminating the classical optimization step reduces overhead and variability, enabling more direct comparisons of quantum hardware performance.

The primary score metric for this benchmark is the approximation ratio $\mathrm{AR} \in [0, 1]$, defined as

$$\mathrm{AR} = \frac{C(\boldsymbol{\gamma}, \boldsymbol{\beta})}{C_{\mathrm{opt}}}, \tag{1}$$

where C_{opt} is the exact optimal solution obtained classically, which is feasible for these problem sizes. Ideally, without hardware noise, the AR should monotonically approach one as the number of layers p increases. However, without error correction or mitigation, hardware noise typically causes performance degradation at larger circuit depths, observed as a drop in the AR after reaching a peak. Thus, this optimization-free QAOA benchmark provides direct insights into quantum hardware performance independent of classical optimization overhead.

2.2.3 Problem Instances

The MaxCut problem involves partitioning the vertices of an undirected graph $G(V, E)$ with vertex set V and edge set E, into two disjoint subsets to maximize the number of edges connecting these subsets. The objective function of the problem can be mapped to a cost Hamiltonian of the form

$$H_c = \frac{1}{2} \sum_{(i,j) \in E} (Z_i Z_j - 1).$$

For this benchmark, we consider randomly generated 3- and 4-regular graphs with sizes ranging from 8 to 36 vertices. These are, in turn, mapped to Hamiltonians acting on 8 to 36 qubits, respectively. Each problem instance is thus defined by a d-regular graph of n vertices, which is solved using a fixed p-layer QAOA ansatz of n qubits.

2.2.4 Results

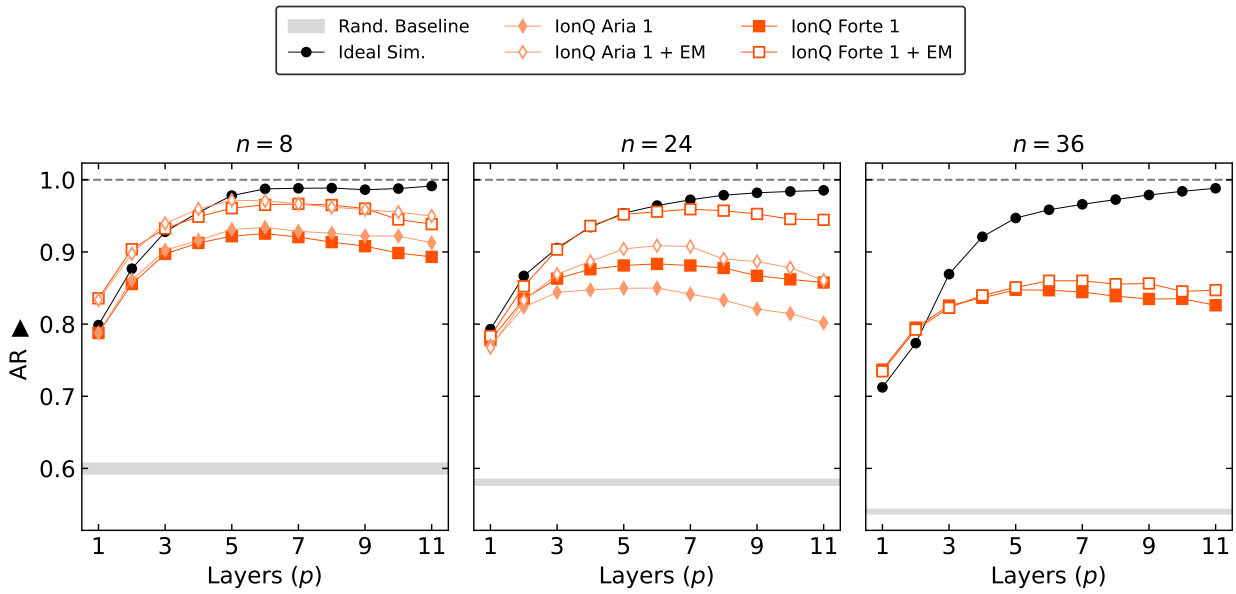

Figure 3: Performance of the fixed-angle QAOA benchmark for solving the MaxCut problem on 3-regular graphs of varying sizes ($n = 8$, 24, and 36 vertices). The results in each subplot compare ideal noiseless simulations with empirical data from IonQ devices, presented in both raw and error-mitigated (EM) forms. Each hardware execution used 5,000 measurement shots. A dashed horizontal line indicates the theoretical maximum AR = 1, where the optimal solution is consistently sampled, and the shaded area shows the $\pm 3\sigma$ confidence interval for a random-sampling baseline. The ideal simulations were performed using a statevector simulator for the $n = 8, 24$ cases and a Matrix Product State (MPS) simulation with a bond dimension of $\chi = 256$ for the $n = 36$ case.

The results in Figure 3 show that the IonQ backends follow the expected ideal trend at small depth, where the approximation ratio initially improves as p increases, and then gradually depart from the noiseless curves as circuit depth grows. Across all three problem sizes, the measured approximation ratios remain well above the random-sampling band over a substantial depth range, demonstrating that the hardware retains a clear optimization signal rather than collapsing to random output. Here, the random baseline corresponds to the AR obtained by drawing bit strings uniformly at random with the same number of shots, and the shaded band indicates its $\pm 3\sigma$ confidence interval.

The error-mitigated curves further narrow the gap to ideal performance in the shallow-to-intermediate depth regime, while the eventual rollover beyond roughly $p \approx 6$ reflects the accumulated impact of hardware noise at larger depths. This makes the benchmark useful both for identifying the onset of the noise-dominated regime and for quantifying how much optimization headroom above random sampling survives on quantum hardware.

2.3 Linear-Ramp Quantum Approximate Optimization Algorithm (LR-QAOA)

2.3.1 Rationale

The Linear-Ramp QAOA (LR-QAOA) offers a powerful, non-variational approach for characterizing quantum hardware. Instead of a layer-by-layer parameter search, it employs a pre-defined linear schedule for its angles. This schedule is governed by two fixed global parameters that scale the linear ramp of the individual layer angles across the layers of the circuit, completely eliminating the classical optimization loop. By removing the classical optimizer, LR-QAOA provides a method to benchmark different hardware platforms on an equal footing. Moreover, since every circuit instance is fully specified a priori, the protocol allows for a direct probe of how performance scales with circuit depth.

Although non-variational, LR-QAOA functions as a universal quantum optimization algorithm, capable of tackling different problems using the same or a similar parameter schedule [20, 21]. Furthermore, it has been shown that effective schedules for large problem instances can be estimated by extrapolating from the results of smaller, classically simulable problems [22]. These features make LR-QAOA an attractive tool for both robust hardware characterization and resource-limited applications.

2.3.2 Algorithm Description

The LR-QAOA protocol uses the same ansatz as the standard QAOA, but it replaces the variational parameter optimization with a fixed, linear schedule. The parameters γ_k and β_k are determined by a linear ramp that depends on only two global hyperparameters, Δ_γ and Δ_β, and the total number of layers p. The schedule for the parameters at layer k (for $k = 0, \ldots, p1$) is given by

$$\beta_k = \left(1 - \frac{k}{p}\right)\Delta_\beta, \qquad \gamma_k = \frac{k+1}{p}\Delta_\gamma.$$

Since the parameters are pre-determined, the algorithm is non-variational and can be executed directly on the hardware without a classical optimization overhead.

Similarly to the QAOA benchmark, the score metric we use is also the approximation ratio AR

$$\mathrm{AR} = \frac{C(\boldsymbol{\gamma}, \boldsymbol{\beta})}{C_{\mathrm{opt}}},$$

where C_{opt} is the exact optimal solution obtained with a classical solver.

In addition to reporting the raw approximation ratio, we also use an effective approximation ratio, denoted $\mathrm{AR}_{\mathrm{eff}}$, which normalizes performance against a random baseline. Following the LR-QAOA benchmarking protocol of Ref. [21], we first construct a random-sampling baseline by drawing batches of bit strings uniformly at random (with the same number of samples as the hardware experiment) and computing their approximation ratios. Let μ_{rand} and σ_{rand} be the mean and standard deviation of these random-batch scores; we define a conservative random threshold

$$\mathrm{AR}_{\mathrm{rand}} = \mu_{\mathrm{rand}} + 3\sigma_{\mathrm{rand}},$$

corresponding to the upper edge of a 3σ confidence interval for a purely random sampler. For each instance we then take the best hardware performance across depths,

$$\mathrm{AR}_{\max} = \max_{p} \mathrm{AR}(p),$$

and define

$$\mathrm{AR}_{\mathrm{eff}} = \frac{\mathrm{AR}_{\max} - \mathrm{AR}_{\mathrm{rand}}}{1 - \mathrm{AR}_{\mathrm{rand}}}.$$

By construction, $\mathrm{AR}_{\mathrm{eff}} = 0$ means the best observed performance is statistically indistinguishable from random sampling, while $\mathrm{AR}_{\mathrm{eff}} = 1$ indicates that the device effectively closes the entire gap between the random baseline and the optimal solution for that instance. Values $0 < \mathrm{AR}_{\mathrm{eff}} < 1$ quantify the fraction of the available "headroom" over random that survives hardware noise. Conversely, $\mathrm{AR}_{\mathrm{eff}} < 0$ indicates that the device performs worse than the 3σ random threshold (e.g., due to decoherence or strong bias), and is therefore considered to fail the benchmark. In this way, $\mathrm{AR}_{\mathrm{eff}}$ provides a simple, single scalar score that both normalizes performance across instances of differing difficulty and supports a clear pass/fail criterion.

2.3.3 Problem Instances

To maintain continuity with the QAOA benchmark, we again use the MaxCut problem. The problem instances consist of the same randomly generated 3- and 4-regular graphs with vertex counts of 24 and 36. We also add fully connected weighted graphs (FCW), since these instances can accurately gauge qubit connectivity in quantum hardware. The number of QAOA layers p is varied up to a maximum of 53. This range is sufficient to observe the full performance curve, from the initial increase in the approximation ratio to the eventual decline into the noise-dominated regime on current NISQ hardware.

The global parameters for the linear schedule were determined empirically. Following an initial approximation of $\Delta_\gamma = \Delta_\beta$, a parameter search was conducted across different layers to find a single value that consistently optimized the AR. This search yielded an approximated optimal value of $\Delta_\gamma = \Delta_\beta = 1.25$, which was found to be robust across different problem sizes for both 3- and 4-regular graphs and was therefore used for all experiments.

2.3.4 Results

Table 2: Effective approximation ratio $\mathrm{AR}_{\mathrm{eff}}$ for IonQ backends and problem instances.

Problem Instance	`aria-1`	`forte-1`
3-reg, $n = 24$	–	0.8390
3-reg, $n = 36$	–	0.5280
4-reg, $n = 24$	–	0.8668
4-reg, $n = 36$	–	0.8009
FCW, $n = 12$	0.3728	0.3821
FCW, $n = 16$	–	0.2504

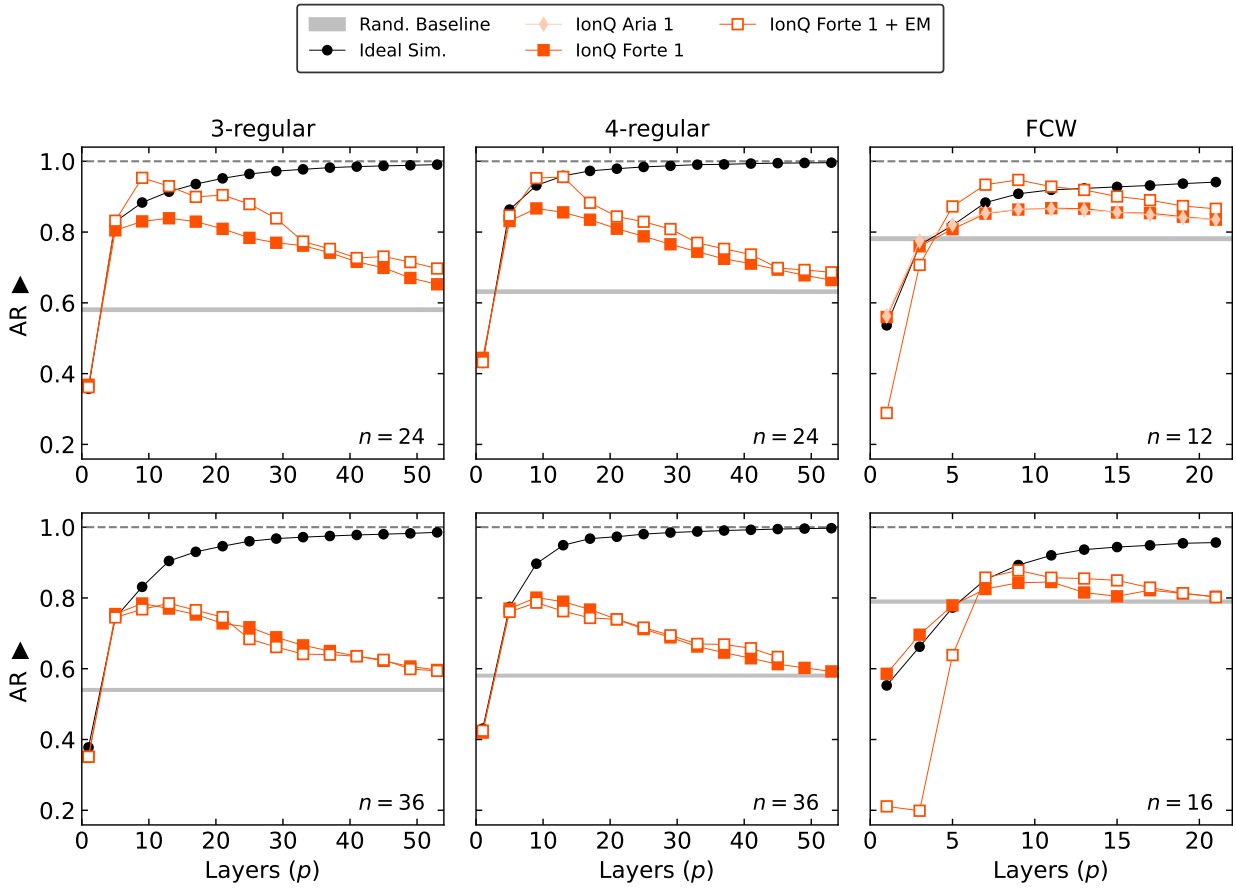

Figure 4: Performance of the LR-QAOA benchmark for solving the MaxCut problem on 3- and 4-regular graphs and on fully-connected weighted graphs (FCW) of varying sizes (n = 12, 16, 24, and 36 vertices). The results in each subplot compare ideal noiseless simulations against empirical hardware data. Each hardware execution used 5,000 measurement shots. The horizontal dashed line at AR = 1 represents the theoretical maximum, and the shaded area delineates the $\pm 3\sigma$ confidence interval for a random-sampling baseline.

The LR-QAOA results in Figure 4 show the same qualitative pattern across the three problem families: the approximation ratio rises above the random-sampling band at small-to-intermediate depth, reaches a peak, and then gradually decays as circuit noise accumulates at larger p. Because the figure includes 3-regular, 4-regular, and FCW graph instances, it provides a broader view of how the hardware behaves across problems with different connectivity structure and depth requirements.

These trends are compactly summarized by the effective approximation ratio $\mathrm{AR}_{\mathrm{eff}}$ in Table 2. IonQ backends achieve positive $\mathrm{AR}_{\mathrm{eff}}$ values across all reported instances, including the 3-regular, 4-regular, and FCW cases. Forte-1 reaches $\mathrm{AR}_{\mathrm{eff}} = 0.8390$ and 0.8668 on the 24-qubit 3- and 4-regular graphs, respectively, remains strongly above the random threshold for the 36-qubit instances, and retains a positive margin over random on the FCW graphs. Aria-1 also passes the reported FCW instance at $n = 12$. Taken together, these results show that LR-QAOA preserves a substantial fraction of the available optimization headroom over random sampling on IonQ hardware across multiple graph classes.

2.4 Fixed Point Amplitude Amplification (FAA)

2.4.1 Rationale

At the heart of many quantum algorithms lies a powerful technique called amplitude amplification, which is the generalized underpinning of Grover's search algorithm [23]. It allows for the significant increase of the probability of measuring a desired quantum state, known as the "target state." However, standard amplitude amplification has a critical drawback of "overshoot" which occurs when the amplification is performed for more than the required number of times which can result in the probability of finding the target state actually decreasing. The fixed-point amplitude amplification (FAA) algorithm [24, 25] is a refined version of this technique that solves this "overshoot" problem, guaranteeing that the probability of measuring the target state is confined within a predefined window close to 1.

The robustness of fixed-point amplitude amplification makes it a valuable tool in various areas of quantum computing, such as:

- Quantum Search: This is the most direct application. FAA allows for the creation of more reliable quantum search algorithms that find a marked item in an unstructured database without needing to know the number of target items in advance. This is a significant improvement over Grover's algorithm, which requires this prior knowledge to determine the optimal number of iterations [23, 24].
- Quantum Simulation: In simulations of quantum systems, it is often necessary to prepare specific initial states. FAA can be employed to efficiently prepare these states by amplifying the amplitude of the desired state from a more easily preparable superposition. This is crucial for simulations in areas like materials science and drug discovery [26].
- Quantum Optimization: In optimization problems, the goal is to find the best solution from a large set of possibilities. FAA can be used in conjunction with algorithms like the Quantum Approximate Optimization Algorithm (QAOA) to increase the probability of measuring the state that corresponds to the optimal solution [27].

2.4.2 Algorithm Description

Fixed-Point Amplitude Amplification (FAA) is designed to increase the probability of measuring a desired ("target") quantum state. It's a crucial modification of the standard Grover-style amplitude amplification which guarantees that the probability of success converges and remains within a predefined window close to 1, eliminating the need to know the number of target states beforehand to avoid "overshooting" the solution. This is achieved by altering the standard reflection operators with controlled phase shifts using quantum signal processing (QSP), effectively adjusting the rotation angle at each step to ensure a steady approach to the target state.

The phase shifts are determined using the quantum signal processing package pyqsp [25, 28, 29, 30, 31]. For this study, we used 5 layers of amplification, which resulted in 10 computed phase shift angles [−1.44174911, 2.96208034, 3.64950635, 2.62339909, 5.22425252, 5.22425252, 2.62339909, 3.64950635, 2.96208034, −26.57449034]. These angles are applied to each layer of amplification, alternating between the oracle and diffusion unitaries per layer. The multi-controlled CNOT gates are decomposed using the v-chain algorithm, which utilizes ancillae qubits but reduces the overall circuit depth compared to a naive decomposition of the multi-controlled gates [32].

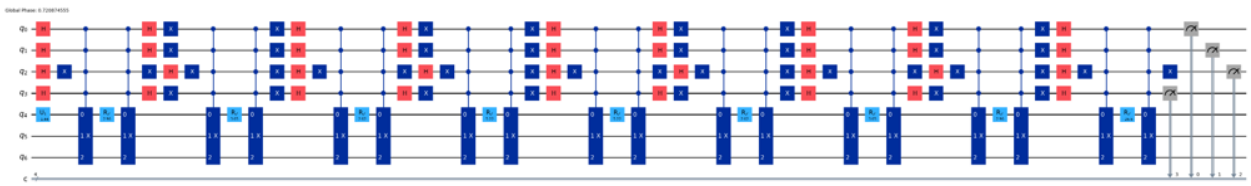

Figure 5: A quantum circuit implementing 5 layers of Fixed Point Amplitude Amplification (FAA) on 4 qubits. The circuit shows the extra ancillae qubits used for the v-chain decomposition of the multi-controlled CNOT gates in the oracle and diffusion unitaries.

2.4.3 Problem Instances

The problem instances involve all possible binary bit strings that can be generated for a given number of qubits. The range of qubit sizes we studied here are 4, 5, and 6. So, this results in 16, 32, and 64 possible bit strings, respectively. Each bit string defines a problem instance and becomes the target for amplitude amplification. The FAA algorithm is applied to each problem instance with 5 layers of amplification, and the circuit is measured to sample the target bit string.

The score is computed as the ratio of the sampling probability of the target bit string and the maximum possible probability achievable for the target bit string with the same set number of layers of amplification. This range of qubits and problem instances is sufficient to observe the performance of the algorithm in the noise-dominated regime on current NISQ hardware.

2.4.4 Results

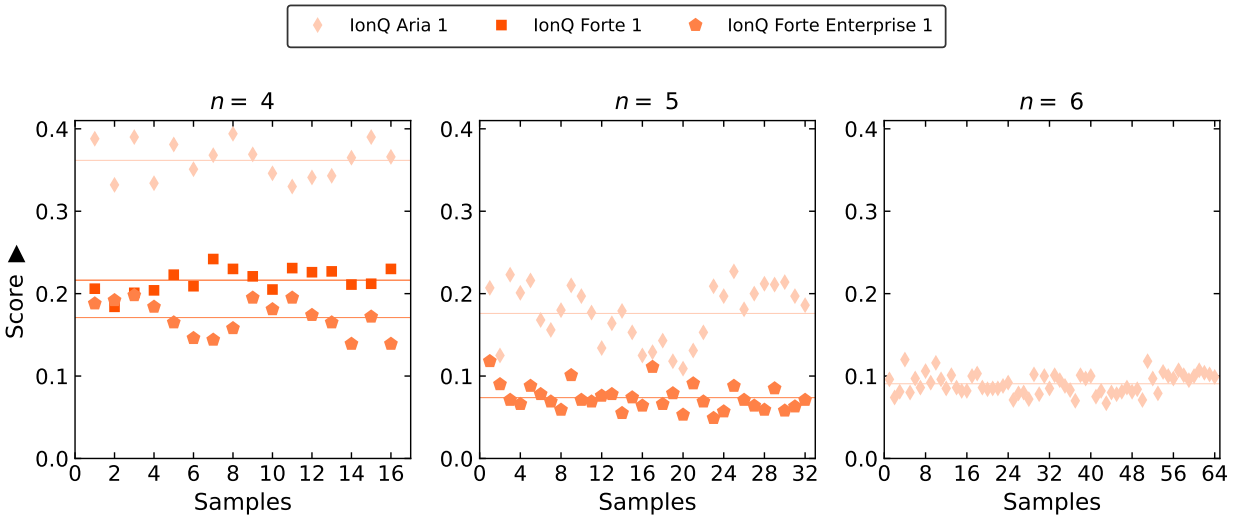

Figure 6: Performance of the Fixed Point Amplitude Amplification (FAA) benchmark for unstructured search on 4, 5, and 6 qubits. The results in each subplot compare empirical data from IonQ quantum hardware. Each hardware execution utilized 1,000 measurement shots. The solid horizontal lines depict the average performance score across all target bit strings.

The FAA results are shown in Figure 6. For all the problem sizes and across all bit strings measured, IonQ backends consistently produce high-quality results. At smaller problem sizes (4 qubits), all the backends perform better (produce higher scores) than at larger problem sizes due to the relatively short circuit depth. As the problem size increases to 5 qubits and 6 qubits, the circuit for the FAA algorithm gets deeper and deeper. Thus, the performance score of all backends degrades due to increased QPU noise. The separation in performance between the IonQ Aria and IonQ Forte backends decreases noticeably. This is because the gate fidelity of the IonQ Aria backend decays more rapidly into the noise-dominated region of increased circuit depth than the gate fidelity of IonQ Forte backends. Although the performance score is low (around 0.1) on the plot on the right ($n = 6$), which has the deepest circuits, the variation in the score across different bitstrings tested is small showing robustness in the IonQ quantum hardware.

2.5 Image Classification with Quantum Convolutional Neural Network (QCNN)

2.5.1 Rationale

This benchmark is based on the quantum Convolutional Neural Network (QCNN) circuit, which is trained for binary classification of MNIST handwritten digits. QCNNs are variational quantum circuits whose architecture is motivated by classical Convolutional Neural Networks (CNNs) [33, 34]. They have been widely researched and studied for image processing type tasks like recognition, classification, or object detection [35, 36]. The quantum hardware is benchmarked by evaluating the inference performance of a pretrained QCNN, specifically measuring its test accuracy. QCNNs have shown certain promising features like good generalization capability over unseen datasets and requiring fewer training parameters. QCNNs have also shown competitive performance on widely relevant problems [37, 38, 39, 40] and are a promising application for quantum computing. In order for any ML-based algorithm to perform well on the quantum computer, the pre-trained circuit needs to perform well on the hardware and with low or reasonable latency. That is what this benchmark based on inference demonstrates.

2.5.2 Algorithm Description

In the particular QCNN structure we used, the original images, which were of 28×28 shape, are resized into 3×3 and 4×4 images using standard Dataloader modules in pyTorch. These images are then vectorized and encoded into the circuit by Angle Encoding, where the pixel values are encoded in single-qubit rotation gates. It is then followed by layers of unitary gates, which act as the convolutional layers, and controlled rotation gates, which act as the pooling function [11]. Depending on the complexity of the task, we can add more layers of the trainable unitaries before the final pooling. At the end, we perform X, Y, and Z measurements on the first qubit, which are then fed into a final feed-forward neural network as shown in the diagram below.

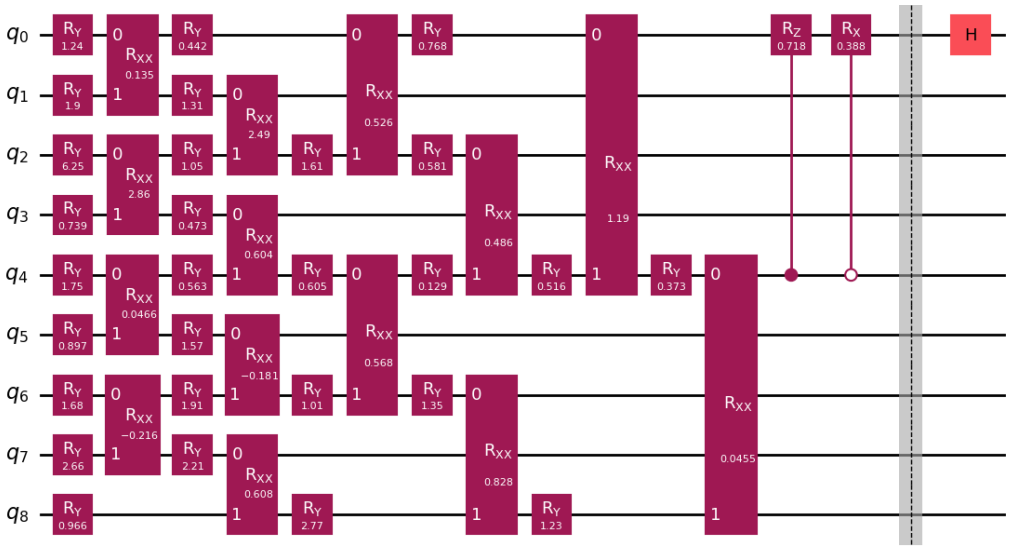

Figure 7: A typical QCNN circuit. Once the data is loaded onto the qubit register, it is transformed through a sequence of convolutional layers. Then measurements are performed on the top qubits in X, Y, and Z bases. The final output is produced by a neural network layer which outputs the label.

The network is trained with 20-40 resized images and with cross-entropy loss. We used COBYLA as the optimizer. To bypass the problem of vanishing gradients, we restarted the optimizer after every 400 iterations. We ran the optimization through 10 such restarts, thus, for 4,000 iterations. We initialized all the parameters, both classical and quantum, randomly.

2.5.3 Problem Instances

We perform two sets of classification: one where we classify between 0-1 classes and the other between 1-7 classes in the MNIST dataset. In the ideal case, the 1-7 classification shows lower test accuracy than 0-1 classification with the specific algorithm used. In each set, we have two circuit sizes. One with 9 qubits and $\sim 30-40$ two qubit gates. The other with 16 qubits and $\sim 70-80$ two qubit gates. We only use inference for benchmarking. There are 4 sets of problem instances, where each classification type has two circuit sizes. Each set consists of the final trained parameters of the corresponding QCNN model, the corresponding test images and their labels. Note that the 9 qubit circuit is trained on 3X3 images and 16 qubit circuits are trained on 4×4 images. The only pre-processing we use on the images before loading them on the quantum circuit is resizing. The quantum circuit has the QCNN structure with ~ 40 parameters in the 9 qubit instance and ~ 80 parameters in the 16 qubit instance. The output of the quantum circuit are single qubit X, Y and Z measurements, which are then post-processed by a small classical neural network (NN) with ~ 14 parameters to predict the class label. The models were trained on 20-40 resized images and tested on 50 resized images. The score of the benchmark is the test accuracy.

In this application, we benchmark the hardware results against ideal results. We do not benchmark performance scaling with increasing hardware size, because the ideal model itself did not show any particular trend between the two circuit sizes. The reason for this is that the two models

are different, being trained on two different image sizes and, hence, image types.

2.5.4 Results

In Figure 8, we demonstrate the QPU performance along with noisy simulation and ideal results. The target of this benchmark is to achieve 100% accuracy in each problem instance.

We see that Aria, with custom error mitigation, performs as well as the ideal simulation in the 1-0 classification task ($\sim 95 - 100\%$ accuracy), for both the 9q and 16q models. In this category, we emphasize that the hardware has achieved the target, as shown by the two models. However, the 1-7 category of classification shows reduced performance on both ideal and noisy simulators, and expectedly, on the QPU as well.

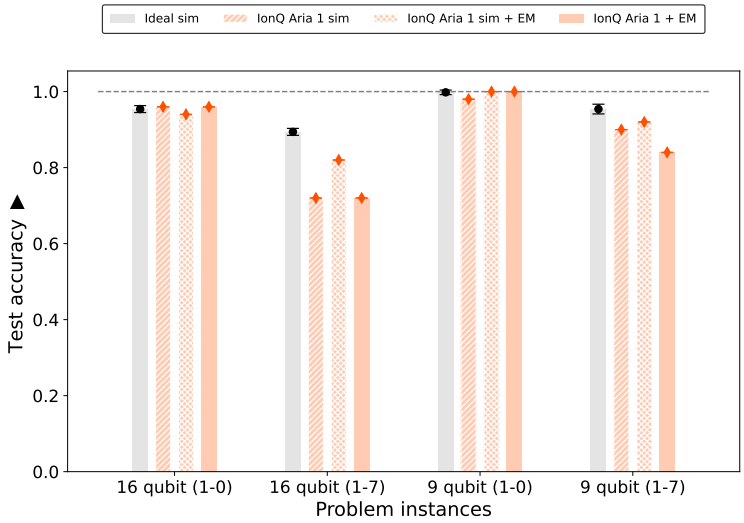

Figure 8: Test accuracy of the 4 QCNN problem instances with 50 test images.

2.6 Portfolio Risk Analysis Using Quantum Copulas

2.6.1 Rationale

Copulas are mathematical tools used to model multivariate joint probability distributions [41]. Copulas enable one to treat the marginal distributions of each of the random variables independently, and have been used in various fields such as quantitative finance, engineering, signal processing, and medicine [42, 43, 44, 45].

The three main classes of copulas are Archimedean, vine, and elliptical copulas. The most commonly used copulas in practice are empirical copulas, which are a mixture of copulas. Empirical copulas are usually modeled using parametric methods such as maximum likelihood estimation [46]. As a result, it becomes more computationally expensive to model empirical copulas in high dimensions.

It was shown that a Quantum Circuit Born Machine (QCBM) can be trained to model the joint probability distribution of a portfolio of up to 4 indices [47]. It was also shown that the QCBM copula can outperform classical methods when used to estimate the value of at risk (VaR) of the portfolios. The VaR is a metric commonly used in the financial sector to estimate risk associated

with a portfolio over a specific time frame at a given confidence interval. Given a portfolio with loss L, time horizon, and confidence interval α, the VaR of the portfolio is the smallest number l such that the probability that $L > l$ is at most $1 - \alpha$. Mathematically, VaR is defined as

$$\text{VaR}_\alpha = inf\{l \in \Re : F_L(l) > \alpha\}, \tag{2}$$

where F_L is the cumulative distribution of the loss function. Note that the VaR is simply the quantile of the loss distribution. In previous works, the QCBM copula was trained by minimizing the Kullback-Leibler (KL) divergence between the distribution of the training data and that of the data generated by the QCBM. This training method, however, does not scale because the KL divergence becomes intractable as the number of indices increases. In this work, we implement a novel training method that overcomes the scaling issues of the previous iterations of QCBM copulas, and show that quantum copulas can now be used to model the joint probability distributions of up to 10 indices.

2.6.2 Algorithm Description

QCBM copulas are trained to estimate the out of sample 95% VaR of equally weighted portfolios. Portfolios of 5-10 variables (stock indices) were built from the following individual returns: AAPL, ADBE , AMZN, FORD, INTC, JPM, KO, MCD, MSFT, and PFE. The trading period is from 01/04/2001 to 12/30/2020. The last 1,000 days of the trading period are used for VaR estimation, and the rest of the data is used to train the QCBM models.

The quantum models were trained by minimizing the maximum mean discrepancy (MMD) between the training data and samples from the QCBM models in copula space. The MMD is a metric used to measure the distance between two distributions. Given distributions X and Y, the MMD between the distributions is given by

$$MMD(X, Y) = \mathbb{E}_{x,x' \sim X}[k(x, x')] + \mathbb{E}_{y,y' \sim Y}[k(y, y')] - 2\mathbb{E}_{x \sim X, y \sim Y}[k(x, y)], \tag{3}$$

where $k(x, y)$ is a kernel function. In this work, we use the Gaussian kernel with width $\sigma = 1$ to train the QCBM models. The training data is converted into copula space by using marginal distribution F_j for each index. In the financial industry, it is customary to use student's t-distributions to estimate the marginal distributions F_j. Measurements from the QCBM circuit can be converted into copula space by using a specified number of qubits per variable m. As an example, for an n variable portfolio with m qubits per variable, the total number of qubits used in a QCBM circuit is mn. Sampling from the circuit will generate bit strings of length mn, and each m bit strings are converted into fractions, yielding copula samples $u_1, ..., u_n$ with $u_i \in [0, 1)$.

We considered two types of ansatz for the QCBM models. The first type (ansatz 1) has layers of single qubit rotation gates $R_x(\theta)$, $R_z(\phi)$, and two qubit entangling gates $R_{xx}(\psi)$ with all-to-all connectivity. An example of ansatz 1 used to model a 2-variable copula using $m = 3$ qubits per variable is shown in Figure 9. The second type of ansatz (ansatz 2) is constructed by forming a maximally entangled Greenberger–Horne–Zeilinger (GHZ) state with one qubit from each register (one register per variable) followed by layers of single qubit rotation gates $R_z(\phi)$, $R_x(\theta)$, and two qubit entangling gates $R_{zz}(\psi)$ with next-nearest neighbor connectivity. An example of ansatz 2 used to model a 2-variable copula using $m = 3$ qubits per variable is shown in Figure 10.

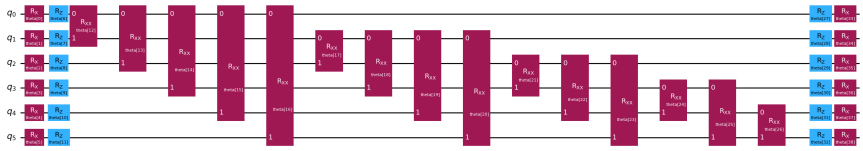

Figure 9: QCBM ansatz 1 used to model a 2-variable copula with 3 bits per variable.

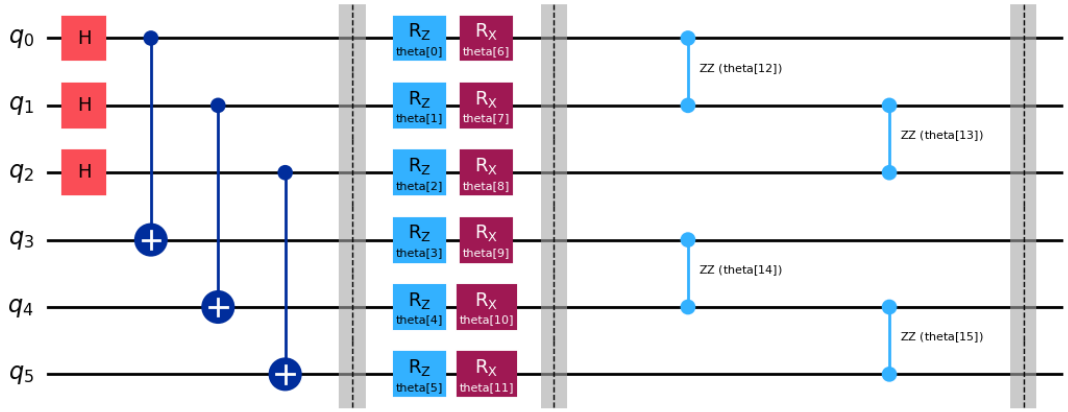

Figure 10: QCBM ansatz 2 used to model a 2-variable copula with 3 bits per variable.

Copula samples u_j from the fully trained QCBM circuits are converted into real space x_j by using the inverse transformation $x_j = F_j^{-1}(u_j)$. The VaR for the generated data is then calculated by using the generated loss distribution. This algorithm returns a score, which is the ratio between the VaR calculated from the data generated by QCBM to the actual VaR on the test data. The score is taken to be the minimum of VaR and 1/VaR so as to ensure it lies in the range [0,1].

2.6.3 Problem Instances

#vars	#q	#1q (ansatz 1)	#2q (ansatz 1)	#1q (ansatz 2)	#2q (ansatz 2)
5	15	60	105	33	22
6	18	72	153	39	27
7	21	84	210	45	32
8	24	96	276	51	37
9	27	108	351	57	42
10	30	120	435	63	47

Table 3: Quantum copula: Overview of the number of variables (#vars), number of qubits (#q), as well as number of 1-qubit gates (#1q), and 2-qubit gates (#2q) for the problem instances and the 2 ansätze considered.

This benchmark considers portfolios of 5-10 variables. QCBM models were trained for each model using the two ansatz discussed in the previous section, using $m = 3$ qubits per register (variable). As a result, the total number of problem instances in this benchmark is 12. Note that for portfolios of n variables, the number of qubits required in the QCBM ansatz is mn. The number of qubits and gates used for each of the ansatz, for each problem instance, is shown in Table 3.

2.6.4 Results

The performance of QCBM copulas trained using the two ansatz for each of the problem instances is shown in Figure 11. The figure shows score versus number of variables for ideal simulations, noisy simulations, and simulations on IonQ's Aria-1 and Forte-1 quantum hardware (with and without error mitigation). Also shown on the plot is the performance of a classical baseline, which generates copula samples randomly from a uniform distribution. Due to the limited number of qubits, only portfolios of up to 8 variables were simulated on Aria-1 (see Table 3). The error bars are obtained from 10 independent runs for ideal and noisy simulations, 3 runs for Aria-1 simulations of ansatz 1, 2 runs for Aria 1 simulations of ansatz 2, 5 runs for Aria-1 simulations of ansatz 1 with error mitigation (EM), 3 runs for Aria-1 simulations of ansatz 2 with error mitigation, and 10 runs for random sampling. The custom error mitigation parameters used for this benchmark are $power = 1.5$ and $threshold = 0.0$.

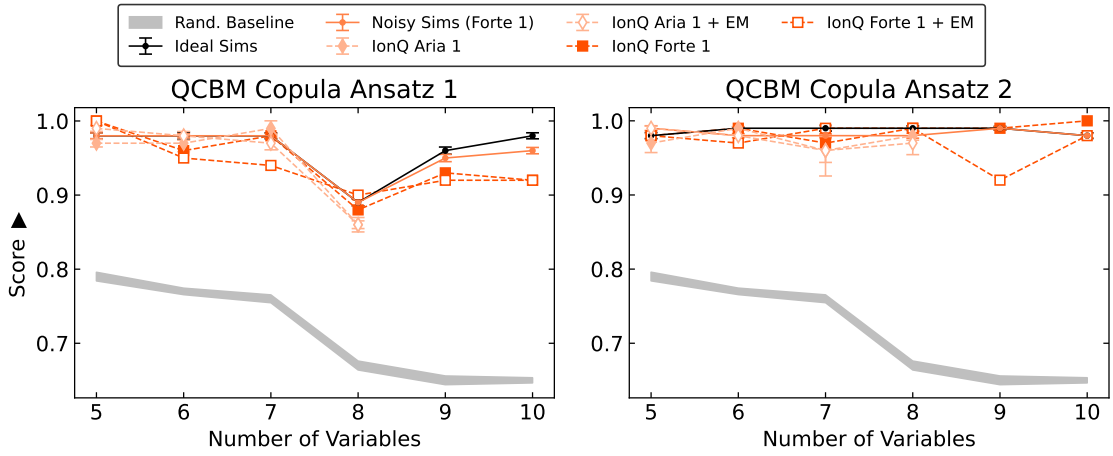

Figure 11: Benchmark scores versus number of portfolio variables for QCBM ansatz 1 (left) and 2 (right panel). Quantum copula results shown for ideal simulations, noisy simulations, and simulations on IonQ's quantum hardware. Results for the random sampling baseline are also included for comparison.

Figure 11 shows that QCBM copulas outperform random sampling in estimating the out-of-sample VaRs of all the portfolios considered. This shows the effectiveness of trained QCBM copulas for portfolio risk assessment. For ansatz 1 (see left panel of Figure 11), Forte 1 hardware results without error mitigation are statistically similar to ideal simulation results up to 8 variables. For 9 and 10 variables, the higher number of 2-qubit gates causes an increase in quantum errors, resulting in slightly lower Forte 1 scores compared to ideal simulations. For ansatz 2, all hardware scores are similar to ideal simulation scores for all portfolios. This result is expected because the significantly

smaller number of 2-qubit gates makes ansatz 2 more resilient to quantum errors. Figure 11 also shows that simulations on quantum processing units (QPUs) with error mitigation do not seem to improve scores for either ansatz.

2.7 Image Loading with Tensor Network-Based Quantum Circuits

2.7.1 Rationale

Generating circuits to prepare an initial quantum wave function is a critical step in a wide range of quantum algorithms. Preparing an arbitrary wave function, $|\Psi_0\rangle$, is notoriously difficult in the most general case. Preparing an arbitrary initial state on N qubits typically requires a circuit with $\mathrm{O}(2^N)$ quantum gates. This presents particular challenges in the field of quantum machine learning, where it is often needed to prepare an initial state that transforms classical data into quantum data by representing the data as a quantum state. In the context of QML, this state preparation method is known as data loading. While there are many possible choices of how to encode this classical data, one particularly popular method, due to its qubit efficiency, is known as the dense amplitude encoding, where we prepare the wave function

$$|\Psi_0\rangle = \frac{1}{Z}\sum_{i=0}^{M}\sqrt{x_i}\,|i\rangle\,,$$

where x_i are the M real numbers which make up a classical data point x, and is the normalization constant.

For an arbitrary data point x, it is clear that $O(M)$ quantum gates are required to perform this data loading step. On NISQ devices, this data loading step quickly becomes impractical for even moderately large input data, and can be a computational bottleneck even on fully fault-tolerant devices. However, in many cases, the input data is not an arbitrary vector but instead has an internal structure that can be taken advantage of to greatly reduce the number of quantum operations needed to perform the data loading. This is the case in computer vision applications, where the input data is an image that may be fed into a supervised learning algorithm to perform, for example, a classification task. In this case, for the data point x, the index i can represent the (x, y) coordinates of the pixel location, and the value of x_i represents the pixel intensity. In most images, the pixel intensities are not randomly distributed, but instead form a smooth function in the two-dimensional space represented by the (x, y) pixel coordinates.

Recent advances in quantum data loading have found that such smooth distributions can be approximately represented rather efficiently using mathematical objects known as tensor networks. A particular kind of tensor network, the matrix product state (MPS), has been shown to be particularly useful for these applications, both due to the efficiency of performing classical manipulations on these objects and also due to the development of algorithms to generate low-depth quantum circuits that can approximately prepare these MPS states. Using these methods, a quantum circuit which approximately loads 2D images can be constructed with O(DN) quantum gates, where $D \sim O(1)$ is the depth of the circuit. The quality of approximation of the image representation improves as D increases, however device noise also increases with D.

In this benchmark, we test the ability of the quantum device to load grayscale 2D images of various complexities. We measure the fidelity of the final reconstructed image, measured from

the output of the MPS image loading circuits, with the original image and test the ability of the quantum hardware to improve the image fidelity as the circuit depth increases.

2.7.2 Algorithm Description

The benchmark for the image loading algorithm involves generating a series of quantum circuits that approximately prepare a matrix product state wavefunction whose amplitudes encode a 2D grayscale image.

A matrix product state wave function has the form

$$|\Psi_{mps}\rangle = \frac{1}{Z}\sum_{\sigma_i}\prod_{i=0}^{N} M^{\sigma_i}_{\alpha_i\beta_{i+1}}|\sigma\rangle\,,$$

where $M^{\sigma_i}_{\alpha_i\beta_{i+1}}$ is a 3-component tensor for each qubit σ_i . For a particular spin configuration of the i^{th} qubit, this becomes a matrix with indices α_i, β_{i+1}, which can take values $1,\ldots,\chi$. The variable χ, is the linear size of the matrix and is known as the bond dimension. It has been shown that a fixed χ is sufficient to perform the image loading to a fixed fidelity, independent of the size of the image. However, the value of χ does depend on the complexity of the original image.

As described in Refs. [48, 49], the matrices, M, can be viewed as isometries, which can then be expanded and converted directly into unitary matrices, and then directly form the set of quantum gates that prepare the MPS state on a quantum computer. These unitary matrices act on $log_2(\chi)+1$, qubits. While this construction is already efficient in theory, in practice, we can reduce the circuit depth even further by applying an iterative algorithm for approximating a bond-dimension χ MPS state.

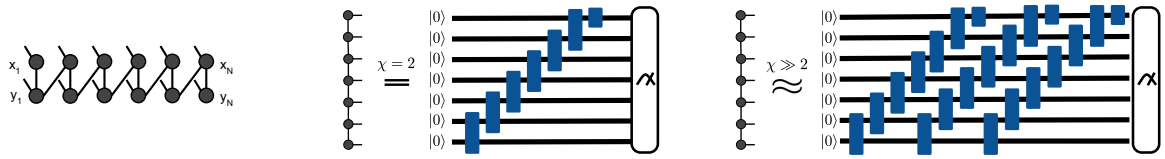

Figure 12: The tensor network diagram used to represent a 2D image *(left)* . A single layer quantum circuit *(middle)* can exactly construct an MPS state with bond dimension $\chi = 2$. A depth D circuit *(right)* approximately constructs a $\chi > 2$ MPS state.

For each image, the method proceeds by first approximating the image as a matrix product state with bond dimension χ. This MPS is then used to construct a quantum circuit with D layers, each layer consisting of $N-1$ general $O(4)$ matrices. The $O(4)$ matrices are optimized using the methods of Ref. [49] to approximate the tensor network. As the number of layers D increases, the quality of the MPS approximation and the fidelity of the quantum state measured in the computational basis with the input image also increases. However, on noisy quantum hardware, the increased circuit depth of higher D circuits leads to increased levels of noise and a reduction in the fidelity with the input image. Therefore, we expect to see an optimal depth D, where both sources of error are comparable to each other. The optimal value of D depends both on the size and complexity of the input image, which affects the quality of the MPS approximation, and on the hardware noise, which

affects the ability to prepare the target quantum state. The final score is the mean-squared-error (MSE) between the reconstructed image and the original image. That is

$$MSE = \sum_{i=0}^{2^N} \left(x_i^{(Q)} - x_i^{(I)} \right)^2 \tag{4}$$

where $x_i^{(Q)}$ is the measured amplitude from the quantum state of the bit string for pixel i, and $x_i^{(I)}$ is the corresponding pixel intensity of the real image I. In both cases, we normalize the pixel values so that $\sum_i (x_i^I)^2 = 1$.

2.7.3 Problem Instances

We benchmark the MPS loading method on two grayscale images, which are respectively of low and medium image complexity. The low complexity image is a 32×32 grayscale image of the number 5 taken from the MNIST handwritten digit database. The medium complexity image is a 64×64 grayscale image of the outline of a shark taken from the ImageNet-Sketch database. For both images, we generate circuits for increasing number of layers, D, and measure the output distribution in the computational basis, giving a score for each value of D. The quantum circuit for the $M \times M$ circuit with D layers uses $2\log_2(M)$ qubits and approximately $2.5D \times (\log_2(M) - 1)$ CNOT gates. The score is the mean-squared-error between the reconstructed image and the original image.

The overall score is the lowest MSE achieved for the given image, taken at the depth D that gives the best score. The hardware passes the benchmark for a given image if it is able to achieve a score below $MSE \sim 0.1$

2.7.4 Results

QPU Comparison: A comparison of the hardware results between the IonQ machines is shown in Figure 13. The left shows the performance on the easy difficulty MNIST handwritten digit test case, and the right shows the performance on the medium difficulty ImageNet-Sketch image. The IonQ Forte QPU showed the best performance up to depth $D = 3$ for the MNIST image, achieving an MSE ~ 0.15. The IonQ Aria QPU performed similarly to IonQ Forte when error-mitigation methods were applied to the result. In both cases, the best MSE score on the ImageNet-Sketch image was achieved for the lowest depth $D = 3$.

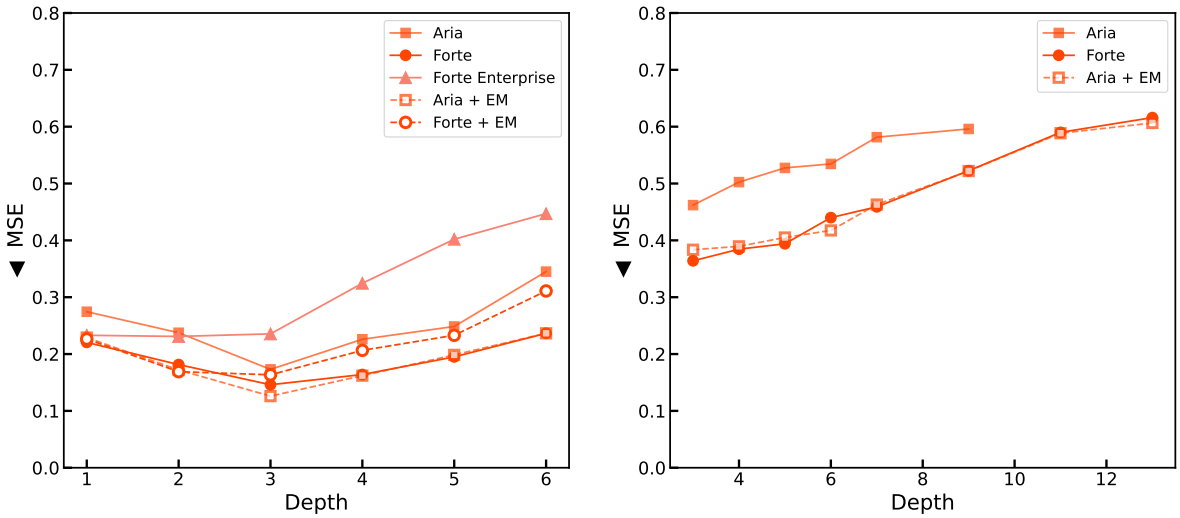

Figure 13: The Mean-Squared error score of the MNIST images (left) and the ImageNet-Sketch Images (right), prepared on the IonQ QPUs, as a function of depth. Note that lower is better.

In Figures 14 and 15, we show the output images at all layer depths D for the best-performing QPUs on the IonQ hardware. In the MNIST case, one can see the initial improvement in image quality as D increases due to the improved MPS approximation, followed by a degradation of image quality at higher values of D when hardware noise begins to dominate. For the ImageNet-Sketch image, the best performing model occurs at depth $D = 3$.

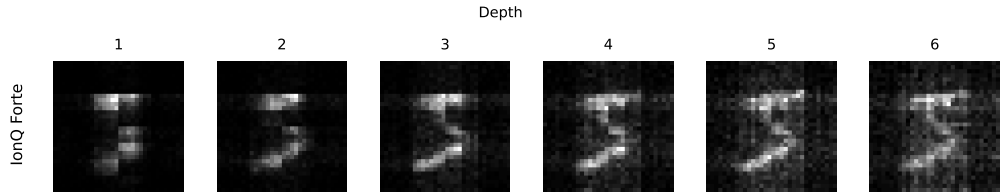

Figure 14: The images generated on IonQ Forte for the MNIST images at different depths.

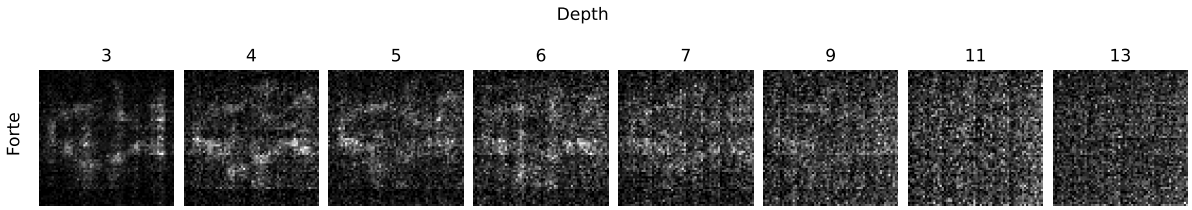

Figure 15: The generated ImageNet-Sketch images sampled from the output of the best-performing IonQ QPU backend.

2.8 Quantum Fourier Transform (QFT)

2.8.1 Rationale

QFT implementations are of fundamental interest in the world of quantum computing. The QFT represents one of the few known subroutines for which fault-tolerant quantum computers are guaranteed to deliver computational advantage (from a complexity theory perspective) over the best known classical algorithms; concretely, quantum computers can compute Fourier transform operations on discrete vectors of length N in $O(\log N)$ quantum operations, whereas the best classical algorithms run in $O(N \log N)$.

QFT subroutines feature prominently in well-known algorithms poised to deliver quantum advantage, like Shor's and HHL. Thus, benchmarking the ability of today's hardware to execute QFT subroutines is fundamental for gauging progress towards the practical implementation of algorithms set to deliver quantum advantage.

As suggested above, the goal of this benchmark is to measure the fidelity of a quantum computer's execution of a QFT subroutine. This is challenging for at least two reasons, and care must be taken in designing suitable benchmark challenges. On one hand, when the input state is supported on a few computational basis states, modern compilers employing advanced techniques can effectively compress the benchmark circuit, which results in a poor challenge that fails to reflect the performance a user might encounter when executing an algorithm that relies on QFT subroutines. In particular, compilers are particularly effective at dealing with subcircuits, placing QFT gates and their inverse in sequence, even when there are additional gates in between. On the other, it is non-trivial to design a complex input state that results in a compact output that can readily be compared against an expected, baseline result. Together, this means QFT subroutine benchmarks must balance the complexity of the input state with that of the output state, so that the input is sufficiently "interesting," so the benchmark circuit is not "compiled away," and yet the expected output state is sufficiently "simple" so that it can be readily interpreted into a consistent benchmark score.

We find that existing benchmarks are codified in quantum circuits whose gate count can be significantly reduced by advanced compilation techniques, and therefore fail to serve as adequate proxies for the performance of QFT subroutines employed in more complex workflows in practice. In this paper, we introduce three families of QFT challenges intended to address this concern, while producing output states whose correctness can be readily verified.

In each of the following benchmarks, each problem instance consists of a single quantum circuit that features at least one QFT gate.

2.8.2 Algorithm Description

The QFT benchmark actually consists of 2 separate problems, Cosine QFT, and Hidden Phase QFT, both of which have in common that they use the QFT circuit as the key subroutine. The difference between the tasks is the preparation of the initial state to which the QFT is applied and the definition of what success looks like for the respective tasks.

Cosine QFT Challenge This challenge uses a quantum circuit to load a cosine plane wave onto the amplitudes of the qubit register and then applies a QFT gate to recover the frequency of the loaded cosine. The input to the final QFT gate is a plane wave, so the expected output

state is supported on exactly two computational basis states; the benchmark score is computed by measuring the fidelity of the sampled output against the known reference.

The novelty here is how the challenge circuits load cosine waves onto the qubit registers. Mathematically, the implementation relies on properties of the Discrete Fourier Transform (DFT) that relate the real part of an N-vector x_n in the time domain to its representation $\widehat{x}_k$ in the frequency domain, and conversely, the real part of $\widehat{x}_k$ to x_n:

$$\mathrm{Re}(x_n) = \frac{1}{2}\big(\widehat{x}_k + \widehat{x}^*_{N-k}\big), \quad \text{and} \quad \mathrm{Re}(\widehat{x}_k) = \frac{1}{2}\big(x_n + x_{N-n}\big). \tag{5}$$

Concretely, the n-qubit challenge circuit loads the cosine plane wave with a given frequency s as follows. First, the circuit prepares a uniform superposition of the states $|0\rangle$ and $|N-1\rangle$ using a single Hadamard gate and $n-1$ CNOT operations. Then the circuit uses up to $n-1$ additional CNOT gates to transform $|0\rangle$ into $|N-2s-1\rangle$ by applying X gates targeting the active bits in the binary representation of s, controlled by the Most Significant Bit (MSB). We assume the target frequency s satisfies $N/4 < s < N/2$, so that $N-2s-1$ represents a positive integer in the two's complement representation. The latter guarantees that the MSB in the binary representation of $N-2s-1$ is not active. In symbols, and keeping the two's complement representation for negative integers in mind, we can express this initial state preparation procedure as follows:

$$|0\rangle^{\otimes n} \to \frac{1}{\sqrt{2}}\big(\,|0\rangle + |-1\rangle\,\big) \to \frac{1}{\sqrt{2}}\big(\,|N-2s-1\rangle + |-1\rangle\,\big).$$

The challenge circuit then performs Draper-style addition to load the desired cosine wave. In particular, the circuit features a QFT gate followed by a sequence of controlled single-qubit rotations to implement the following transformation:

$$\to \frac{1}{\sqrt{2}}\big(|\widehat{N-2s-1}\rangle + |\widehat{-1}\rangle\big) \to \frac{1}{\sqrt{2}}\big(|\widehat{-s}\rangle + |\widehat{s}\rangle\big) = \sum_k \cos\left(\frac{2\pi k}{N}s\right)|k\rangle\,.$$

The second transformation shifts each Fourier basis vector by $s+1$, and the equality follows by the first relation in Equation (5) and the definition of the QFT.

Finally, the challenge circuit applies a QFT gate to the loaded plane wave to recover the encoded frequency s:

$$\sum_k \cos\left(\frac{2\pi k}{N}s\right)|k\rangle \to \sum_k \cos\left(\frac{2\pi k}{N}s\right)\widehat{|k\rangle} \to \frac{1}{\sqrt{2}}\big(\,|s\rangle + |-s\rangle\,\big).$$

The last calculation follows from the second relation in Equation (5) and the definition of the QFT.

Figure 16 depicts a typical challenge circuit diagram. The identification $\frac{1}{\sqrt{2}}\big(|\widehat{-s}\rangle + |\widehat{s}\rangle\big) = \sum_k \cos\big(\frac{2\pi k}{N}s\big)|k\rangle$ makes it easy to produce a compactly supported output reference state in principle, and yet makes it difficult for compilers to render challenge circuits trivial.

Hidden Phase QFT Challenge Each Hidden Phase QFT challenge circuit consists of a Hadamard and a Fourier transform, followed by a sequence of controlled single-qubit rotations, and an additional set of Fourier and Hadamard transforms. The overall operation is trivial for the work qubit register, with the core computation occurring on an ancilla qubit. This ancilla qubit's rotation

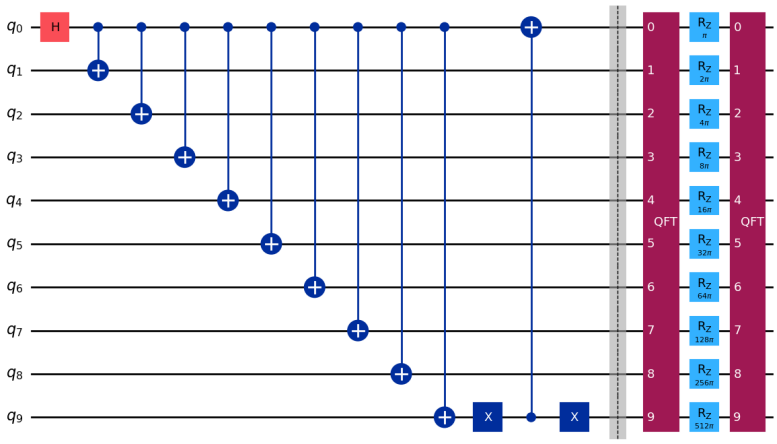

Figure 16: A typical Cosine QFT Challenge circuit. The circuit first loads a cosine plane wave of a given frequency onto the amplitudes of the qubit register, and then it applies a QFT gate to recover the encoded frequency of the cosine wave. The circuit loads the cosine wave by applying a number of entangling gates to a superposition of two computational basis states, and then performing Draper-style addition.

frequency, which is set by the controlled single-qubit gates, must be determined by sampling the output state. The expected output state is thus compactly supported on the computational basis, and the benchmark is scored by estimating the fidelity of the output state against the reference state.

In symbols, we implement the following qubit state transformations. Given a work register with n qubits and a hidden frequency $0 \leq k^\star < N$, with $N = 2^n$, we begin by loading a uniform superposition over all Fourier basis vectors $\widehat{|x\rangle}$. We define these vectors as the image of the computational basis vectors $|x\rangle$ under the QFT gate:

$$|0\rangle^{\otimes(n+1)} \to \frac{1}{\sqrt{2N}} \sum_{x=0}^{N} \big(|x\rangle |0\rangle + |x\rangle |1\rangle \big) \to \frac{1}{\sqrt{2N}} \sum_{x=0}^{N} \big(\widehat{|x\rangle} |0\rangle + \widehat{|x\rangle} |1\rangle \big).$$

Next, we set $\lambda_k = \frac{2\pi}{N} k$ and apply cyclic shift operators (implemented as controlled single-qubit R_Z rotations) to obtain

$$\to \frac{1}{\sqrt{2N}} \sum_{x} \big(\widehat{|x + k^\star\rangle} |0\rangle + \widehat{|x - k^\star\rangle} |1\rangle \big) = \frac{1}{\sqrt{2N}} \sum_{x} \big(\omega_N^{k^\star} \widehat{|x\rangle} |0\rangle + \omega_N^{-k^\star} \widehat{|x\rangle} |1\rangle \big),$$

with $\omega_N = e^{2\pi i/N}$ denoting a primitive N-th root of unity. The last equality relies on well-known properties of the discrete Fourier transform. Finally, we apply additional Fourier and Hadamard transforms to recover the initial state of the work qubit register:

$$\begin{aligned} &\to \frac{1}{\sqrt{2N}} \sum_{x} \big(\omega_N^{k^\star} |x\rangle |0\rangle + \omega_N^{-k^\star} |x\rangle |1\rangle \big) \to \frac{1}{\sqrt{2}} |0\rangle \otimes \big(\omega_N^{k^\star} H |0\rangle + \omega_N^{-k^\star} H |1\rangle \big) \\ &\to |0\rangle \otimes \big(\cos(\lambda_{k^\star}) |0\rangle + \sin(\lambda_{k^\star}) |1\rangle \big). \end{aligned} \tag{6}$$

When all is said and done, the work qubit register is mapped back to the initial $|0\rangle$ state, and the ancilla qubit has picked up the hidden phase; it's in the superposition

$$HR_Z(\lambda_{k^\star})H\,|0\rangle\,.$$

In principle, we can recover the correct hidden phase by sampling the output state.

In practice, we compute the benchmark score as the Hellinger fidelity between the observed output distribution and the reference state, which is given by Equation (6).

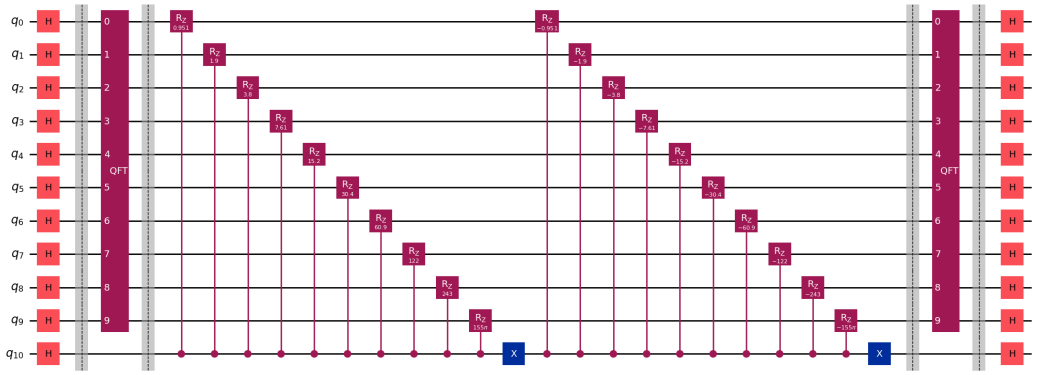

Figure 17: A typical Hidden Phase QFT challenge circuit. The circuit first loads a superposition over all qubits, then applies a QFT gate on the work register, here illustrated as the top 9 wires. Then the circuit applies two Draper-adder-type operations: it shifts the vectors with ancilla state $|0\rangle$ *upwards* by the hidden phase k^*, and shifts those with ancilla state $|1\rangle$ *downwards* by k^*. Then the circut applies another QFT gate on the work registered, followed by a Hadamard transformation on all the qubits. The work register returns to the initial $|0\rangle$ state, and the ancilla qubit has picked up a hidden phase as described above.

2.8.3 Problem Instances

For Cosine QFT, in principle each problem instance is characterized by two integers: a positive integer n denoting the number of qubits in the challenge circuit, and an integer $2^{n/4} < s < 2^{n/2}$ denoting the cosine frequency. In practice, we've set the cosine frequency as $s = 2^{n/2} - 1$ as a safeguard to prevent artificially high scores on machines whose qubits tend to decohere to the $|0\rangle$ state.

For Hidden Phase QFT, each problem instance is characterized by two integers: a positive integer n denoting the number of work qubits in the challenge circuit, and an integer $2^{n/2} \leq k^* < 2^n$ denoting the *hidden frequency*. For each challenge, the hidden frequency k^* was generated as a uniform random sample in $[2^{n/2}, 2^n)$. We chose to avoid frequencies in $[0, 2^{n/2})$ since lower frequency values could result in ancilla states with no significant amplitude in the $|1\rangle$ state.

2.8.4 Results

For Cosine QFT, Figure 18 plots the observed benchmark scores for the Cosine QFT challenge. Each circuit was measured $1,000$ times. Recall that a noiseless quantum device would always achieve the maximum score of 1.0. IonQ Forte shows significant performance degradation with

increasing qubit count, with fidelity scores nearing zero as we push the system to its maximum circuit width. The observed scores seem to follow an exponential trend.

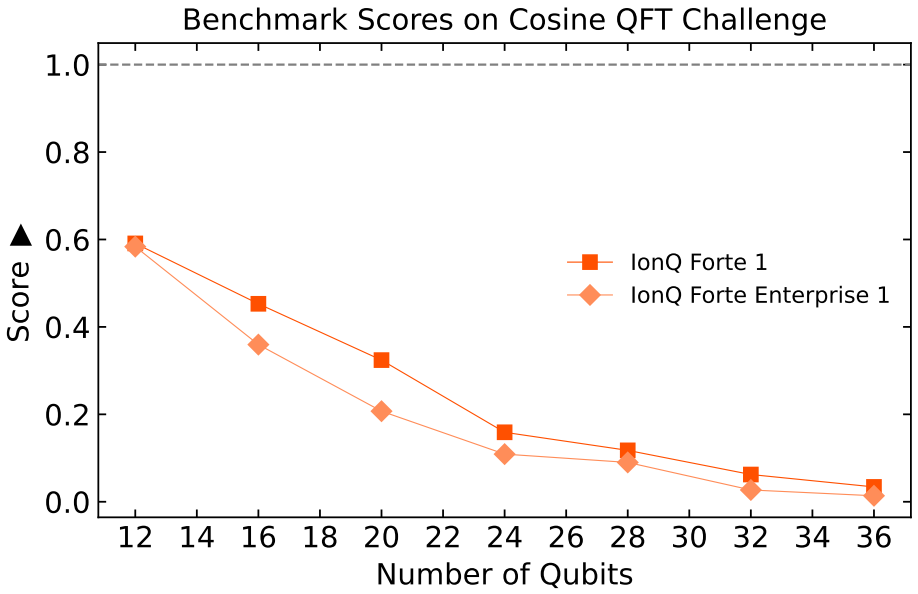

Figure 18: Benchmark score for the Cosine QFT challenge obtained using IonQ Forte and IonQ Forte Enterprise.

For Hidden Phase QFT, Figure 19 plots the observed benchmark scores for the Hidden Phase QFT Challenge. Each circuit was measured $1,000$ times. Recall that a noiseless quantum device would always achieve the maximum score of 1.0. IonQ Forte shows significant performance degradation with increasing qubit count, with fidelity scores nearing zero as we push the system to its maximum circuit width. The observed scores seem to follow an exponential trend, with some deviations. The latter are likely due to the slight differences in challenge difficulty that arise from the particular choice of hidden frequency.

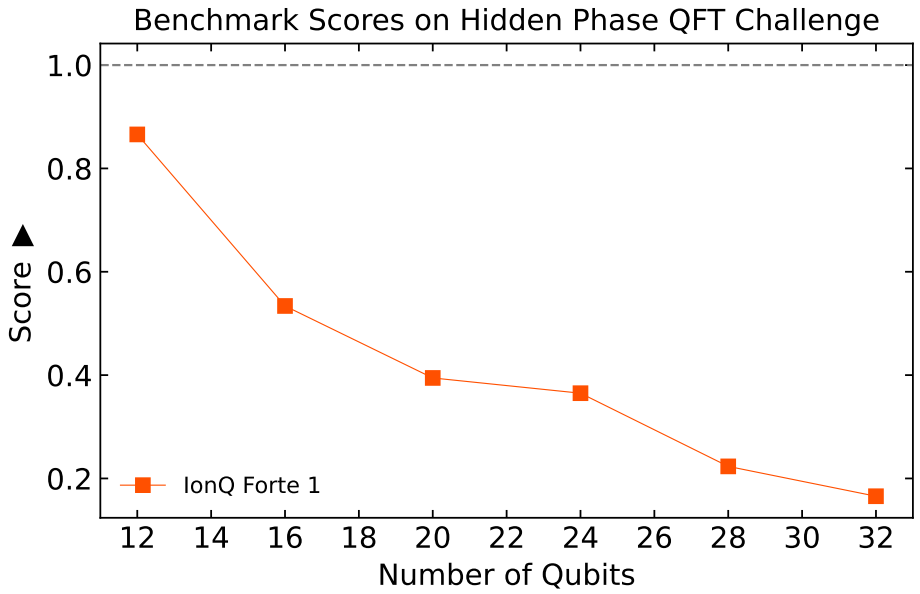

Figure 19: Benchmark score for the Hidden Phase QFT challenge.

2.9 Hidden Shift Benchmark Problem (HSBP)

2.9.1 Rationale

The hidden shift problem is a natural source of challenges where a quantum computer may demonstrate advantage over classical computational resources. In particular, the hidden shift problem for Boolean functions introduced in [50] defines a blueprint for benchmark challenges whose difficulty can be tweaked at will, by increasing the number of CNOT and Toffoli gates used in permutation operations. The algorithm introduced in [50] shows an oracle separation between P and BQP and has since been widely used for benchmarking classical simulation algorithms.

The benchmark challenge we introduce below is based on Theorem 6 in [50], and it serves as a sort of universal benchmark for quantum computers in the sense that it tests the computer's ability to execute circuits with varying numbers of entangling gates in different configurations, albeit it is not directly connected to any practical application as of yet. While Theorem 6 in [50] applies to the class of bent functions, we focus on the subclass of Majorana-McFarland bent functions of the form

$$h_\pi(x, y) = x \cdot \pi(y)^t = x_1 y_{\pi_1} \oplus \cdots \oplus x_n y_{\pi_n},$$

with $\pi \colon \mathbb{Z}_2^m \to \mathbb{Z}_2^m$ denoting an arbitrary permutation on m points. The dual of h_π is conveniently given by $\widetilde{h_\pi}(x, y) = \pi^{-1}(x) \cdot y^t$. In what follows below, we let $n = 2m$ and consider the function $f_\pi \colon \mathbb{Z}_2^n \to \mathbb{Z}_2$ given by $f(x) = h_\pi(x_o, x_e)$, with $x_o = (x_1, x_3, \ldots, x_{2m-1})$ denoting the odd-indexed components, and $x_e = (x_2, x_4, \ldots, x_{2m})$ denoting the even-indexed components of x. Explicitly,

$$f(x) = x_1 x_{\pi_2} \oplus \cdots \oplus x_{2m-1} x_{\pi_{2m}}.$$

2.9.2 Algorithm Description

Each HSBP challenge consists of a number of quantum circuits. For each circuit width, the challenge circuits share a common structure, and they correspond to different choices of hidden shift and permutation. We use multiple circuits in order to improve the precision of performance metrics by reducing idiosyncratic noise in the benchmark scores associated with the particular choice of hidden shifts or permutation gates.

Each challenge circuit is constructed as follows. First, we rotate the qubits into a uniform superposition over all computational basis states. Next we compute a shifted version of f_π into the amplitudes of the qubit register, apply a Hadamard transform, and then compute the dual function $\widetilde{f_\pi}$ into the amplitudes of the resulting state. Finally, we apply another Hadamard transform to recover the hidden shift.

The output state is just that: a pure state supported on exactly one bit-string, which describes the hidden shift used to calculate the shifted version of f_π. The benchmark score is computed as the probability of sampling the desired hidden shift bit-string in the output state.

The permutation π is a key component of our benchmarking algorithm: it provides a mechanism for tuning the computational complexity, and thereby the difficulty, of each benchmark challenge. In practice, we implement every given permutation using a collection of CNOT and Toffoli gates; each problem instance is in part defined by the choice of π, or equivalently, the choice of CNOT and Toffoli gates used to implement it in the challenge circuit.

Figure 20 illustrates the anatomy of a typical challenge circuit. Given an even number of qubits $n = 2m$, a hidden shift $s \in \{0,1\}^m$, and a permutation π on m points, the circuit first applies a Hadamard transform to prepare a uniform superposition over all computational basis states, and then it computes the shifted bent function $g(x) = f_\pi(x \oplus s)$ into the amplitudes of the qubit register, resulting in the transformation

$$|0\rangle^{\otimes n} \to \frac{1}{\sqrt{2^n}} \sum_x |x\rangle \to \frac{1}{\sqrt{2^n}} \sum_x (-1)^{f_\pi(x \oplus s)} |x\rangle \,.$$

The U_g block represents the computation of g. The circuit then applies a Hadamard transformation to obtain

$$\begin{aligned}
\frac{1}{\sqrt{2^n}} \sum_x (-1)^{f_\pi(x \oplus s)} |x\rangle &\to \frac{1}{2^n} \sum_{x,y} (-1)^{xy^t + f_\pi(x + \oplus s)} |y\rangle \\
&= \sum_y (-1)^{ys^t} \left(\frac{1}{2^n} \sum_z (-1)^{yz^t + f(z)} \right) |y\rangle \\
&= \sum_y (-1)^{ys^t} \widehat{f_\pi(y)} \, |y\rangle \\
&= \frac{1}{\sqrt{2^n}} \sum_y (-1)^{ys^t} (-1)^{\widetilde{f_\pi}(y)} \, |y\rangle \,,
\end{aligned}$$

where $\widehat{f_\pi(y)} = \frac{1}{2^n} \sum_z (-1)^{yz^t + f(z)}$ denotes a Fourier coefficient. Note that the last equality follows by definition of the dual bent function [50]. The circuit then computes the dual function $\widetilde{f_\pi}$ onto the amplitudes of the qubit register and finally it applies a third Hadamard transformation to recover

the hidden shift s:

$$\begin{aligned}\frac{1}{\sqrt{2^n}}\sum_y(-1)^{ys^t}(-1)^{\widetilde{f_\pi}(y)}\,|y\rangle &\to \frac{1}{\sqrt{2^n}}\sum_y(-1)^{ys^t}\,|y\rangle \\ &\to \frac{1}{2^n}\sum_{x,y}(-1)^{(x+s)y^t}\,|x\rangle \\ &= \sum_x \widehat{\delta}(x+s)\,|x\rangle \\ &= |s\rangle\,.\end{aligned}$$

In the first equality δ denotes the discrete delta function whose output is trivial unless the input is null.

In practice, each challenge circuit computes f_π and related functions onto the amplitude of its qubit register using a collection of CNOT and Toffoli gates to implement the permutation π, and a stack of CZ gates to compute the required signs: note that the CZ gate transforms $|x_1x_2\rangle$ into $(-1)^{x_1x_2}\,|x_1x_2\rangle$.

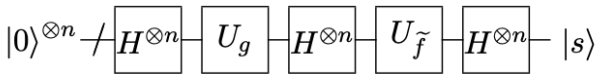

Figure 20: Structure of a hidden shift circuit. The objective is to recover the hidden shift s through a sequence of operations involving Hadamard transforms and the computation of bent functions into the amplitudes of a qubit register.

2.9.3 Problem Instances

Each problem instance consists of a number of challenge circuits, as described above, and each circuit is defined by an even integer $n = 2m$ together with a hidden shift $s \in \{0,1\}^m$ and a permutation π on m points. For every challenge, we generate hidden shifts using a sampling procedure where we generate each bit independently, drawing Bernouilli variables with probability of success set to 0.75.

While arbitrary permutations can be used to define HSBP challenges, in this paper, we focus on four families of permutations for ease of characterization. The first family consists of those permutations whose action is implemented by a single multi-controlled X gate. The associated challenges are labeled by MCX and their permutation blocks consist of a single multi-controlled X gate targeting the bottom wire and controlled by the rest of the even-indexed qubits, which is then transpiled into a sequence of CNOT gates and single-qubit rotations.

The second family consists of those permutations whose action is implemented by a "ladder" of CNOT gates; for these challenges,

$$\pi = \prod_{j=1}^{m-1} CNOT(x_{2j}, x_{2j+2}).$$

The third family is similar, but instead of a CNOT ladder, we use a ladder of Toffoli, or CCX

gates. Concretely, in these challenges,

$$\pi = \prod_{j=1}^{m-2} CCX(x_{2j}, x_{2j+2}, x_{2j+4}).$$

Each problem instance in each of these challenge families consists of 10 quantum circuits of the same width, each circuit corresponding to a different hidden shift.

The fourth challenge family leverages randomly generated permutations. Each of these permutations consists of a fixed number of CNOT gates. Each such permutation was generated by sampling pairs of integers representing the target and control qubits from the possible universe uniformly at random. Each problem instance in this challenge family consists of $9 = 3 \times 3$ quantum circuits: we use three hidden shifts and three random permutations, and each circuit corresponds to a possible hidden shift and permutation pair. This structure allows for averaging benchmark scores across a minimal number of hidden shift and random permutation choices. For the purposes of this paper we tested random permutation challenges with $10, 20, \ldots, 100$ CNOT gates.

2.9.4 Results

Figure 21 plots the benchmark scores obtained on IonQ Forte for each of the challenge families considered. The MCX challenge is the most difficult, as expected, since the corresponding circuits require the highest number of entangling gates.

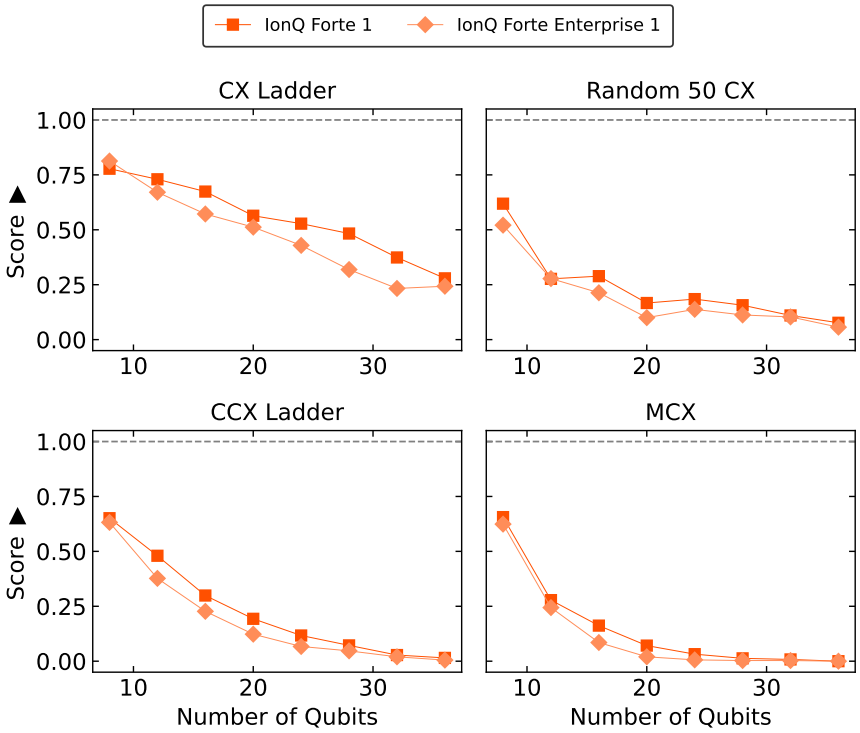

Figure 21: Benchmark scores computed on IonQ Forte class systems for the HSBP challenge families considered: CX ladder, random permutation with 50 CNOT gates, CCX ladder, and MCX permutations.

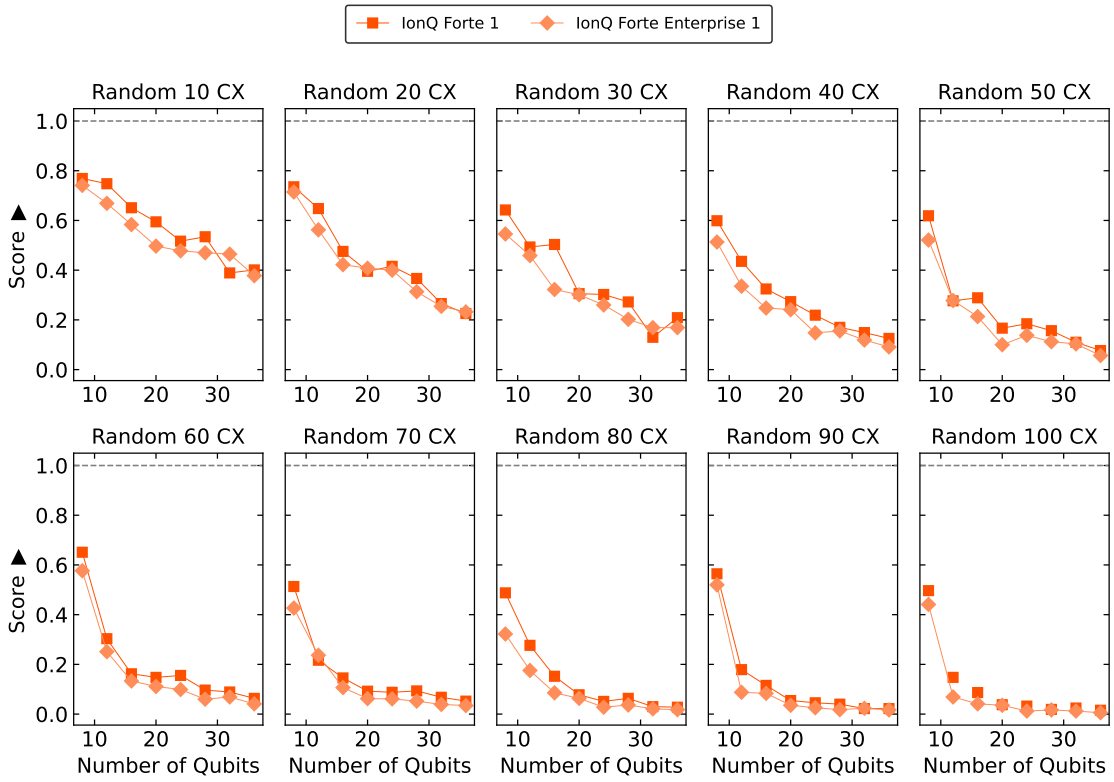

Figure 22: Benchmark scores obtained on IonQ Forte when executing HSBP random permutation challenges with $20, 40, 60, 80$, and 100 CNOT gates. This plot illustrates the increasing difficulty of benchmark challenges as a function of the number of entangling in the permutations.

3 Results: Open Benchmarks

3.1 Variational Quantum Imaginary-Time Evolution (varQITE)

3.1.1 Rationale

The variational quantum imaginary-time evolution approach is an interesting alternative to QAOA for solving combinatorial optimization problems [51]. Like QAOA, varQITE is a general-purpose variational algorithm that seeks to minimize the energy of a Hamiltonian by tuning circuit parameters in order to obtain the optimal solution to a difficult combinatorial problem.

Two main advantages of varQITE over QAOA include

- it can leverage shallow problem- and hardware-specific ansätze; and
- it obviates the need for a classical optimizer to tune circuit parameters by providing an update rule based on time-marching a linear differential equation.

Aside from being sensitive to hardware characteristics like qubit connectivity, gate fidelities, and coherence times, the performance for varQITE on a given problem heavily depends on the choice of ansatz. For this benchmark, we again focus on the MaxCut problem, using the same

quantum Hamiltonian formulation that we leverage in our QAOA and related benchmarks. In fact, we use the same problem instances as well, for the sake of comparing the performance of different algorithms on the same problems.

We leverage the ansatz layout that was introduced in [52] in the context of the graph partitioning problem (GPP).

3.1.2 Algorithm Description

Much like QAOA, varQITE is a variational approach that finds approximate solutions to optimization problems by encoding them into a Hamiltonian H and minimizing the cost function $C(\theta)$ defined by the expectation value

$$C(\theta) = \langle \psi(\theta) | \, H_c \, | \psi(\theta) \rangle \, ,$$

with $|\theta\rangle$ denoting the chosen problem-specific ansatz.

The varQITE algorithm defines an explicit parameter update rule that does not rely on classical optimization routines; instead, the update rule enforces quantum imaginary-time evolution under the action of the problem Hamiltonian H for a chosen set of observables. For details, see [51].

For our benchmark challenges, we use the so-called Heavy Neighbors Ansatz introduced in [52]. This ansatz is well-suited for the MaxCut problem since it selectively reinforces entanglement between qubits by leveraging structural properties of the underlying problem graph. Concretely, the ansatz features parametrized entangling gates connecting qubits that represent nodes connected by edges with high weight, which allows for effective cut size tuning by tweaking gate parameters.

In any case, each benchmark challenge consists of a single quantum circuit. The benchmark challenge measures the performance of quantum hardware upon executing an optimized ansatz to retrieve high-quality approximations to the MaxCut problem.

We prepare each challenge circuit by running the variational portion of the varQITE algorithm on an ideal simulator until convergence. This "pre-training" yields the parameter values that are fixed for each challenge circuit.

Upon executing each challenge circuit, we measure the resulting AR as defined by Equation (1) and report the value as the challenge score.

3.1.3 Problem Instances

For the sake of comparing the performance of different solution algorithms on the same problems, we consider the universe of MaxCut instances described in Section 2.2. In particular, we focus on the subset of 3-regular graphs and their corresponding MaxCut instances.

To execute varQITE, we use the two-layer Heavy Neighbors Ansatz defined by the problem graph with $3n$ entangling gates per layer, with n denoting the number of nodes in the problem graph. The ansatz parameters are obtained by executing varQITE in an ideal simulator until the absolute change in Hamiltonian energy falls below 10^{-4}.

3.1.4 Results

Figure 23 plots benchmark scores obtained on IonQ Forte. Each challenge circuit was executed 1,000 times. The plot illustrates a decreasing trend in performance as the circuit width, and hence the circuit depth, increases.

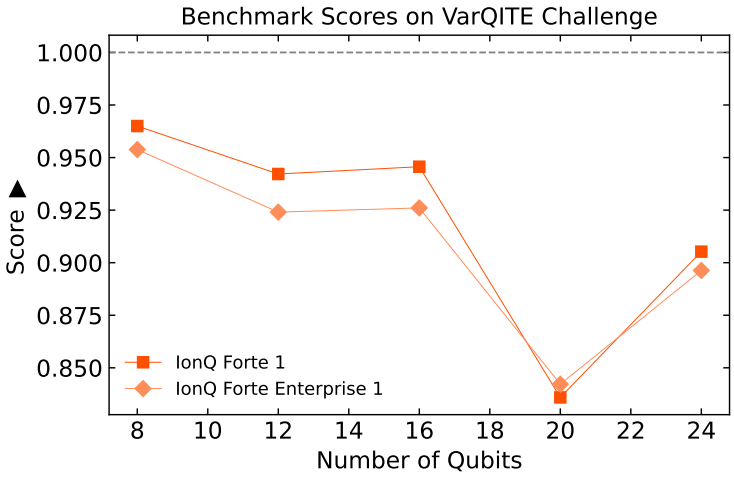

Figure 23: Benchmark score for varQITE challenges testing the solution quality of MaxCut problems on 3-regular graphs of varying sizes.

3.2 Quantum-Classical Auxiliary Field Quantum Monte Carlo (QC-AFQMC)

3.2.1 Rationale

Auxiliary-field quantum Monte Carlo (AFQMC) is a projector-based quantum Monte Carlo approach particularly effective in accurately modeling ground-state electronic structures, especially when electron correlation effects are significant [53, 54, 55, 56]. Its strength stems from systematically improvable accuracy achievable through enhanced trial wavefunctions, controlled imaginary-time propagation, and careful walker population management. Quantum-classical AFQMC (QC-AFQMC) is an extension of classical AFQMC that utilizes efficient quantum circuits for the trial state that would otherwise be inefficient to simulate classically [57, 58, 59, 60, 61]. Because the quantum trial states can be made systematically more expressive at polynomial circuit depth, they can capture multi-reference and strong-correlation structure that is prohibitively expensive for classical ansätze, while still entering AFQMC only through quantities that are efficient to estimate on quantum hardware (notably overlaps and local energies). In this way, the quantum device supplies a compact, high-fidelity trial state that improves the phaseless constraint and reduces bias, and the classical Monte Carlo projector supplies the long-imaginary-time refinement needed to recover the ground state with controllable cost and error.

To align with this benchmarking framework's emphasis on practical performance metrics, QC-AFQMC is evaluated explicitly by measuring the end-to-end wall time (TTS) required to attain a predefined energy accuracy (ΔE) relative to a classical benchmark, along with the corresponding energy utilized (EPS). Rather than relying on proxy circuit metrics, this approach directly assesses the ability of the quantum-classical workflow to achieve stated accuracy goals and quantifies the computational resources (time and energy) needed to do so.

The hybrid QC-AFQMC structure clearly separates tasks between quantum and classical processors. The quantum device efficiently prepares highly correlated trial wavefunctions ($|\Psi_T\rangle$) and generates measurement data through matchgate shadows, which are a specialized form of randomized measurement suitable for efficient classical post-processing [58, 60, 61]. Classical computational infrastructure then performs the AFQMC projection steps, implements importance sampling, and

accumulates statistical estimators. This task division ensures practical applicability by explicitly exposing the entire quantum-classical algorithm stack to rigorous TTS and EPS benchmarking.

3.2.2 Algorithm Description

QC-AFQMC is fundamentally built upon the phaseless AFQMC methodology, employing imaginary-time propagation to project onto the ground state. The Hubbard–Stratonovich transformation decomposes the propagation into stochastic one-body operations, sampled using an ensemble of walkers represented as Slater determinants. Importance sampling strategies involving force biases are used to reduce statistical variance. To manage phase instabilities inherent in quantum Monte Carlo methods, a phaseless constraint is introduced that projects walker phases toward the trial wavefunction $|\Psi_T\rangle$, creating a controllable, trial-dependent bias that diminishes as trial wavefunction quality improves. Energies are computed primarily using mixed estimators, with optional back-propagation methods available to enhance accuracy. Operational parameters include imaginary-time step size ($\Delta\tau$), walker populations, reconfiguration frequency, and orbital reorthogonalization intervals.

The quantum enhancement in QC-AFQMC is realized through matchgate shadows,[58] which facilitate efficient quantum-classical integration. The quantum processor generates high-quality trial wavefunctions, typically the pair-coupled cluster doubles variant of the unitary coupled-cluster (UpCCD) ansätze. Random matchgate basis rotation operations are appended to the trial circuit in order to operationalize the entire circuit for (partial) shadow tomography and observable estimation. Matchgate shadows yield randomized quantum measurements from which classical processing reconstructs essential quantities, such as overlaps $\langle\Psi_T|\phi\rangle$, force biases, and local energies. Classical reconstruction exploits Pfaffian algebra, ensuring computational feasibility even as system sizes scale. This procedure scales polynomially, at approximately $\mathcal{O}(n^5)$ per shadow, driven mainly by Pfaffian and tensor contractions. Empirically, maintaining variance at fixed accuracy requires the number of shadows N_{sh} to scale modestly as $n^{0.5}\log n$, resulting in a total classical processing complexity of approximately $\mathcal{O}(n^{5.5}\log n)$ [61].

3.2.3 Problem Instances

We benchmark QC-AFQMC on linear hydrogen chains in the STO-3G basis at 2.0 Å. The instances reported here are H_4 and H_6, with exact FCI references -1.898 Ha and -2.847 Ha, respectively. The H_4 instance is treated in a CAS(4,4) active space mapped to 8 qubits, and H_6 in CAS(6,6) mapped to 12 qubits. Trial states are prepared with the UpCCD circuit family and accessed through matchgate shadows for overlap, force-bias, and local-energy estimation.

Unless otherwise noted, all QC-AFQMC runs use an imaginary-time step $\Delta\tau = 0.01$ with 256 walkers and a projection length of 80 blocks $\times$ 20 steps ($\tau_{\max} = 16$ a.u.). The first 10 blocks are discarded as equilibration. Statistical uncertainties are computed by reblocking over the final 70 blocks and reported as 95% confidence intervals.

Two-body operators are represented by a Cholesky decomposition with tolerance 10^{-7}; this yields 7 auxiliary fields for H_4 and 11 for H_6. Per platform, we acquire $N_{\text{sh}} = 21{,}080$ matchgate shadow samples for H_4 and $N_{\text{sh}} = 28{,}289$ samples for H_6.

3.2.4 Results

Table 4 summarizes QC-AFQMC total energies for linear H_4 and H_6 chains in STO-3G at 2.0 Å. The corresponding FCI references are −1.898 Ha (H_4) and −2.847 Ha (H_6). The H_4 calculations employ a CAS(4,4) active space mapped to 8 qubits with 21,080 matchgate shadow samples, while H_6 uses CAS(6,6) on 12 qubits with 28,289 samples.

On the ideal simulator, QC-AFQMC yields −1.868 Ha for H_4 (18.79 kcal/mol above FCI) and −2.768 Ha for H_6 (49.98 kcal/mol above FCI). These residuals are expected: the phaseless constraint introduces a trial-state-dependent bias that vanishes only as $|\Psi_T\rangle \to |\Psi_0\rangle$, and the UpCCD trial ansatz is restricted to paired excitations. The larger residual for H_6 is consistent with stronger multireference character in the CAS(6,6) manifold. These simulator results therefore provide noise-free baselines for assessing QPU-prepared trial states.

For H_4, Aria-1 agrees with the simulator within uncertainty, giving −1.866 Ha (95% CI 14.5 mHa). Forte-Enterprise-1 reaches −1.858 Ha (25.05 kcal/mol above FCI), while Forte-1 yields −1.818 Ha (50.09 kcal/mol above FCI). The imaginary-time traces in Figure 24(a) show larger projection-time fluctuations for Forte-1. Aria-1 exhibits a transient spike near $\tau = 13$–14 a.u., attributable to numerical instability in the walker renormalization matrix R during finite-difference force-bias evaluation [61].

For H_6, Forte-Enterprise-1 reaches −2.738 Ha (68.80 kcal/mol above FCI; 95% CI 7.4 mHa) and Forte-1 yields −2.716 Ha (82.07 kcal/mol above FCI; 95% CI 8.6 mHa). Relative to the simulator baseline, the QPU-derived trial states lie 18.82 kcal/mol (Forte-Enterprise-1) and 32.09 kcal/mol (Forte-1) higher, a larger degradation than for H_4, consistent with the deeper circuits required at 12 qubits. Backend ordering is preserved across both systems, with Forte-Enterprise-1 outperforming Forte-1. The H_6 convergence traces in Figure 25(a) show an equilibration transient over the first ~ 2 a.u. before reaching steady-state fluctuations.

Figures 24 and 25(b) report the final reblocked estimates and 95% confidence intervals for each backend.

Table 4: QC-AFQMC total energies for linear H_4 and H_6 chains in STO-3G at 2.0 Å for simulated and QPU-generated trial states.

System	Backend	E (Ha)	95% CI (mHa)	ΔE_{FCI} (kcal/mol)
H_4	Simulator (ideal)	−1.868 049	7.2	18.79
	Aria-1 QPU	−1.865 905	14.5	20.14
	Forte-1 QPU	−1.818 176	12.4	50.09
	Forte-Enterprise-1 QPU	−1.858 087	7.4	25.05
H_6	Simulator (ideal)	−2.767 543	8.4	49.98
	Forte-1 QPU	−2.716 399	8.6	82.07
	Forte-Enterprise-1 QPU	−2.737 551	7.4	68.80

Uncertainties are 95% confidence intervals from reblocking over the final 70 blocks (first 10 discarded as equilibration).
ΔE_{FCI} is relative to exact FCI references: −1.898 Ha (H_4) and −2.847 Ha (H_6).

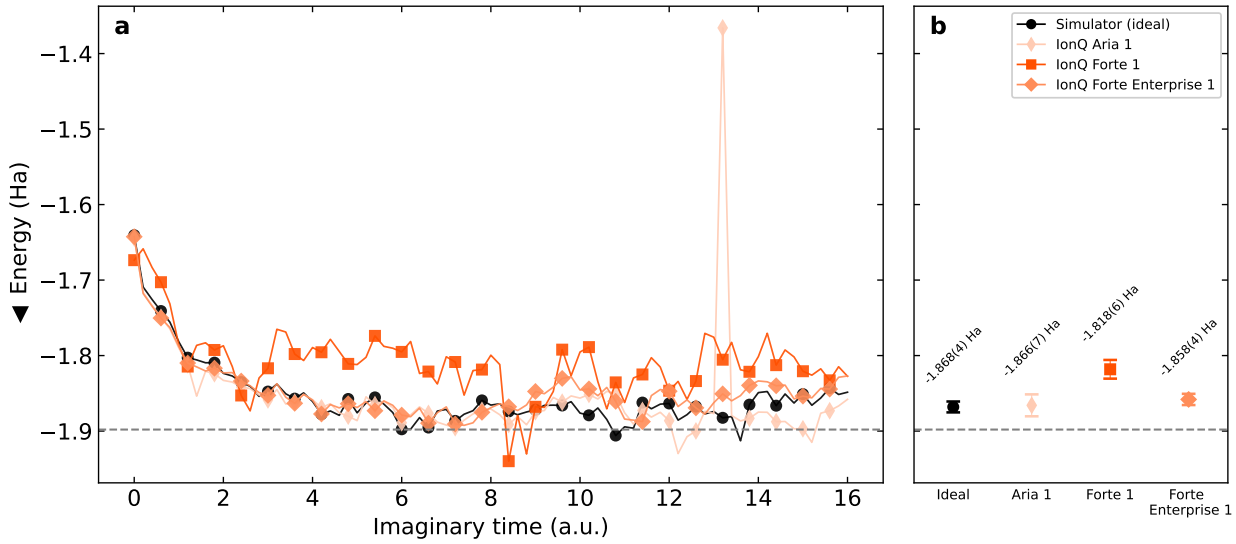

Figure 24: QC-AFQMC results for linear H_4 in STO-3G at 2.0 Å using CAS(4,4) (8 qubits). (a) Imaginary-time evolution of the total energy; the dashed line marks the FCI reference (−1.898 Ha). (b) Final reblocked energies for the ideal simulator and the `Aria-1`, `Forte-1`, and `Forte-Enterprise-1` QPUs; error bars denote 95% confidence intervals (numbers in parentheses indicate the corresponding uncertainty in the last digits).

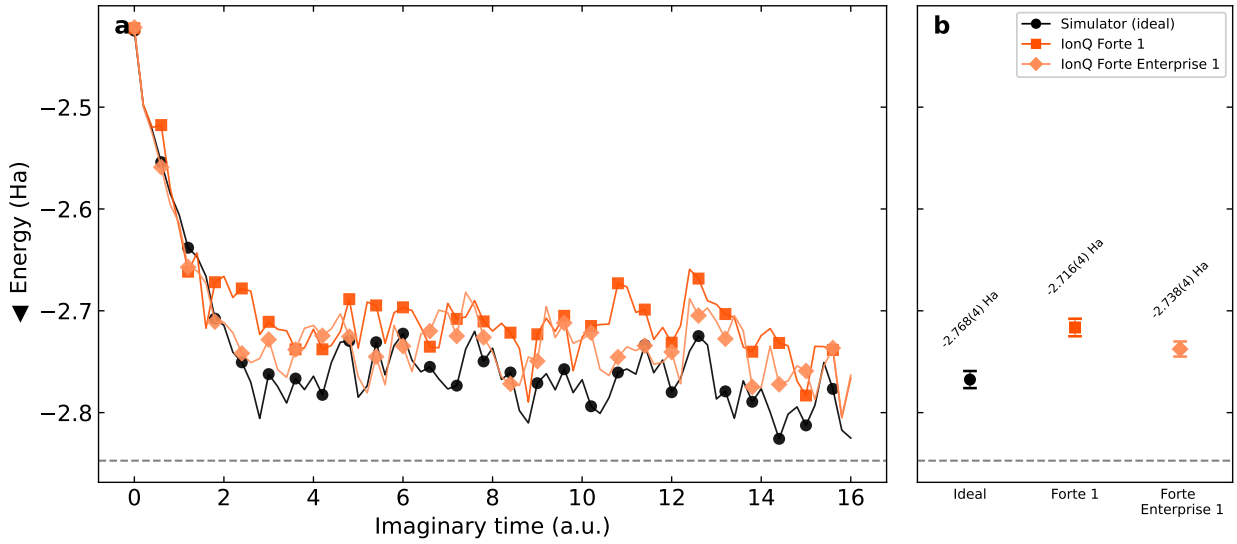

Figure 25: QC-AFQMC results for linear H_6 in STO-3G at 2.0 Å using CAS(6,6) (12 qubits). (a) Imaginary-time evolution of the total energy; the dashed line marks the FCI reference (−2.847 Ha). (b) Final reblocked energies for the ideal simulator and the `Forte-1` and `Forte-Enterprise-1` QPUs; error bars denote 95% confidence intervals (numbers in parentheses indicate the corresponding uncertainty in the last digits).

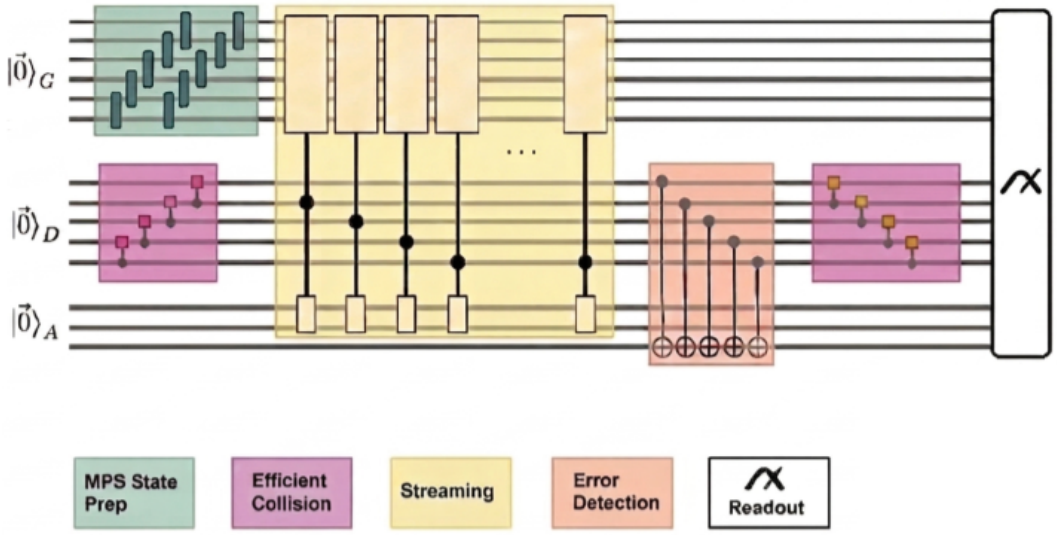

Figure 26: Overview of the QLBM pipeline, showing the D2Q5 lattice Boltzmann formulation to simulate macroscopic fluid dynamics. The quantum algorithm takes as input a smooth density distribution (a 2D Gaussian in this case) and applies the QLBM quantum circuit for T iterations. The input distribution is loaded using a MPS state preparation circuit. The collision operator is prepared using a unary state preparation circuit on the direction qubits. The streaming step involves a series of operations on the grid qubits which are controlled on the direction qubits. Ancilla qubits are used to reduce the total gate count of the multi-control unitary gates. We perform error detection via additional ancilla qubits during the circuit execution.

3.3 Quantum Lattice Boltzmann Method for Differential Equations

3.3.1 Rationale

This benchmark is based on a Quantum Lattice Boltzmann (QLBM) algorithm for solving the advection diffusion equation in 2D [62]. The QLBM is an iterative algorithm where a macroscopic density function is described as a sum of individual microscopic particle densities. The algorithm pipeline for each time step includes: initial state encoding, collision and streaming operations, recovering the macroscopic density from the individual densities, and implementing the appropriate boundary conditions. The final state obtained after each such evolution at time t, serves as the initial state for the next iteration. To benchmark the quantum hardware we use quantum state fidelity during the evolution as the metric.

3.3.2 Algorithm Description

We follow the algorithm developed and implemented in [62], represented in Fig. 26, which is based on the quantum lattice Boltzmann method (qLBM) for simulating a density function $\Phi(\mathbf{x}, t)$ evolving under a linear advection-diffusion equation [63, 64, 65]. The equation is given by,

$$\partial_t \Phi = D\nabla^2 \Phi - \nabla\left(\mathbf{u}\Phi\right), \tag{7}$$

where, $\mathbf{u}(\mathbf{x})$ is the velocity field and D is a constant diffusion coefficient.

In this approach, the macroscopic density $\Phi(\mathbf{x}, t)$ evolution is modeled as that of individual species $f_i(\mathbf{x}, t)$ of microscopic particle densities, $\Phi(x, t) = \sum_i f_i(\mathbf{x}, t)$. At time t, this approximation is given by [66],

$$f_i(\mathbf{x} + c_i\Delta t, t + \Delta t) = (1 - \omega)f_i(\mathbf{x}, t) + \omega f_i^{eq}(\mathbf{x}, t), \tag{8}$$

, where Δt is the time step, f_i^{eq} is the equilibrium density distribution and $\omega = \Delta t/\tau$ with relaxation time τ. f_i^{eq} is given by the Bhatnagar-Gross-Krook (BGK) formulation [67] as,

$$f_i^{eq} = \omega_i \Phi \left[1 + \frac{c_i \cdot u_i}{c_s^2}\right]. \tag{9}$$

We set the relaxation constant $\omega = 1$, which simplifies Equations (8) and (9) to,

$$\Phi(x, t) = \sum_i k_i \Phi\left[\mathbf{x} - c_i\Delta t, t - \Delta t\right], \tag{10}$$

$$\text{where } k_i = \omega_i \left[1 + \frac{c_i \cdot u_i}{c_s^2}\right]. \tag{11}$$

Following are the modules in the final algorithm:

1. ***Encoding:*** $\Phi(\mathbf{x}, t)$ is encoded as a quantum state $|\Phi_t\rangle_G$,

$$|\Phi_t\rangle_G := \frac{1}{\|\Phi_t\|_2} \sum_{\mathbf{x}} \Phi(\mathbf{x}, t)|x\rangle_G, \tag{12}$$

   where $\|\Phi_t\|_2 := \sqrt{\sum_{\mathbf{x}} \Phi(\mathbf{x}, t)^2}$ where each gridpoint $\mathbf{x}$ is represented by a computational basis state $|x\rangle$. This state is prepared on n_G 'grid qubits' to represent $N = L^d$ lattice sites in d dimensions.

2. ***Collision:*** The velocity is considered to be uniform and the collision operator is applied via state preparation of a quantum state $|k\rangle$ with the help of an auxiliary qubit, where

$$|k\rangle = \frac{1}{\sqrt{2^{n_D}}} \sum_i k_i |i\rangle_D |0\rangle_a + |\chi_0\rangle_D |1\rangle_a, \tag{13}$$

   for some orthogonal state $|\chi_0\rangle$. The state $|k\rangle$ is prepared on n_D 'direction qubits' for M directions.

3. ***Streaming:*** The streaming operator U_S is created via controlled unitaries,

$$U_S = \prod_i \left(\mathbf{1}_G \otimes (\mathbf{1} - |i\rangle\langle i|)_D + S_i \otimes (|i\rangle\langle i|)_D\right). \tag{14}$$

   where the controls are on direction qubits and unitaries S_i are given by, $S_i|x\rangle_G = |x + c_i\Delta t\rangle_G$. Applying U_S on the prepared input state $|\Phi_t\rangle_G|k\rangle_D$ yields,

$$\frac{1}{\|\Phi_t\|_2\sqrt{2^{n_D}}} \sum_{x,i} k_i |i\rangle_D \Phi(x - c_i\Delta t, t)|x\rangle_G |0\rangle_a + |\chi_1\rangle_{D,G}|1\rangle_a, \tag{15}$$

   where $|\chi_1\rangle_{D,G}$ is some orthogonal state.

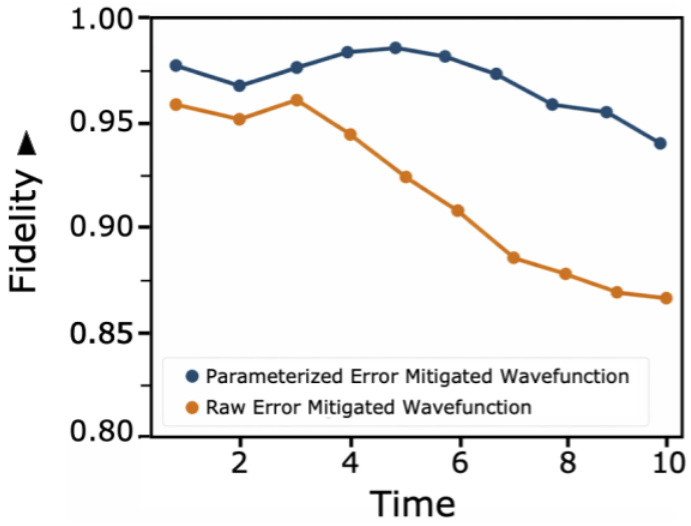

Figure 27: The overlap, $|\langle\psi|\phi\rangle|$, between the QPU density and the exact classical LBM solution (orange curve). Further improvement in fidelity is achieved by measuring observables (instead of doing full state tomography from raw shots) and reconstructing the 2D Gaussian distribution, where fidelity remains $\sim 94\%$ even after 10 time steps (blue curve).

4. ***Macroscopic quantities:*** The final step involves applying Hadamard gates on all the direction qubits which essentially adds all the amplitudes over the direction qubits. The final state is given by:

$$\begin{aligned}
&\frac{1}{2^{n_D}}\sum_x\left(\sum_i k_i\Phi(x-c_i\Delta t,t)\right)|x\rangle_G|0\rangle_{D,a}+|\chi_2\rangle\\
=&\frac{1}{2^{n_D}}\sum_x\Phi(x,t+\Delta t)|x\rangle_G|0\rangle_{D,a}+|\chi_2\rangle\\
=&\frac{\|\Phi_{t+1}\|_2}{2^{n_D}\|\Phi_t\|_2}|\Phi_{t+1}\rangle_G|0\rangle_{D,a}+|\chi_2\rangle
\end{aligned}\tag{16}$$

$|\chi_2\rangle$ is an orthogonal state such that, $(\mathbb{1}_G\otimes\langle 0|_{D,a})|\chi_2\rangle=0$. The final evolved state $|\Phi_{t+1}\rangle$ is obtained by post-selecting on the $|0\rangle_D$ state.

3.3.3 Problem Instances

We implemented the QLBM algorithm on hardware to solve the advection-diffusion equation for a 2D initial Gaussian density distribution on a 16×16 grid. The quantum circuit used 19 qubits and ~ 260 two-qubit gates. The results of this execution are shown in Figure 27, and demonstrate agreement with the classical LBM simulation as shown by the fidelity values. We used 10k shots and custom error mitigation strategies to achieve this performance.

3.3.4 Results

The benchmark metric in this algorithm is fidelity, which is measured with respect to the exact classical solution at every time step. Note that the fidelity values differ from the target value of 1

not only because of the hardware noise but also due to shot noise and approximate state loading techniques like tensor network methods. The orange curve in Figure 27 demonstrates fidelity when the quantum state is constructed using shots from the quantum circuit readout. The fidelity can be further improved by reading out observables like mean, variance and covariance from the shots and reconstructing the state from these observables, as demonstrated by the blue curve. This algorithm achieves fidelity upto 0.95 at the end of 10 time steps under the advection-diffusion equation on IonQ Forte.

3.4 High Energy Physics

3.4.1 Rationale

Future quantum computers could enable *ab initio* investigations in nuclear and high-energy physics, revolutionizing these fields, particularly in simulating nuclear reactions and dynamics on yocto-second timescales, similar to femtosecond imaging in chemistry. Meanwhile, current NISQ computers, despite limitations, are valuable for simulating out-of-equilibrium dynamics and quantum many-body phenomena.

The work presented in [68] establishes a benchmark for exploring a potential exotic nuclear decay relevant to new physics searches. The stability of matter provides strong constraints on physics beyond the Standard Model, with observations of proton decay, neutron-antineutron oscillations, and nuclear beta decay setting limits on fundamental symmetry breaking. Specifically, neutrinoless double beta decay ($0u\beta\beta$-decay) of certain nuclei indicates lepton number violation. Calculating these decay rates is challenging due to the Majorana-neutrino induced process, which involves two charged-current weak interactions connected by a near-massless neutrino within the nucleus. Simulating this doubly-weak decay processes is a significant challenge for classical computing due to the necessity of coherently tracking the dynamics in a strongly-interacting nucleus. While progress has been made with Euclidean-space lattice QCD simulations, quantum computation may be better suited for tracking low-energy nuclear excitations and strong correlations.

In this benchmark we test the ability of quantum computers to simulate rare fundamental physics processes by simulating the $0\nu\beta\beta$-decay of a simple nucleus in 1+1D lattice QCD and observing the lepton-number violating dynamics. The simulation involves two baryons confined to two spatial sites, incorporating strong and weak interactions, and explicitly violating lepton-number conservation with a Majorana neutrino mass term.

3.4.2 Algorithm Description

We perform the simulation of the neutrinoless double-beta ($0\nu\beta\beta$) decay of a nucleus using a 1+1D lattice Quantum Chromodynamics (QCD) model. This model incorporates dynamical quarks (up and down) and leptons (electrons and neutrinos). The simulation utilizes a lattice with periodic boundary conditions (PBCs) and two spatial lattice sites, which are mapped onto 32 qubits. A minimum of two spatial sites is essential to accommodate the degrees of freedom generated during $0\nu\beta\beta$-decay.

Weak interactions are represented by an effective four-Fermi interaction that locally couples quarks and leptons. The hadronic states in 1+1D QCD form isospin multiplets, with the lowest-lying baryon multiplet having an isospin of $I = 3/2$. This multiplet is labeled $\Delta^{++}, \Delta^{+}, \Delta^{0}, \Delta^{-}$ due to its resemblance to the Δ resonance observed in 3+1D QCD.

Parameters within the Hamiltonian, including a Majorana mass term, are tuned to facilitate the $0\nu\beta\beta$-decay of an initial $|\Delta^-\Delta^-\rangle$ state. To simulate this decay, a quantum circuit is employed. This circuit first initializes the lepton vacuum and the $|\Delta^-\Delta^-\rangle$ state. Subsequently, it undergoes time evolution driven by a Hamiltonian that encompasses strong and weak interactions, as well as free fermion terms. The inclusion of a lepton-number breaking neutrino Majorana mass in the free fermion Hamiltonian enables the $0\nu\beta\beta$ decay channel. Following the time evolution, the lepton qubits are measured to detect the occurrence of weak decays.

We include techniques like symmetrization, twirling, amplification, extrapolation, regression, and post-selection to approximate ideal circuit outputs by leveraging application and device symmetries to reduce noise-induced biases with lower overhead. Combining specific circuit variants with observable-specific post-selection rules allows for precise error detection and higher shot efficiency.

The overall approach integrates debiasing through symmetrization, post-selection via symmetry checks, and a new parameterized nonlinear filtering method on the lepton qubit register, utilizing spare qubits for flag-based mid-circuit symmetry checks and leakage error detection.

Specific for the hardware, the circuits were optimized by merging two-qubit gate blocks thanks to the all-to-all connectivity and native RZZ(θ) gates, reducing entangling gates by 15% and efficiently reducing bias through varying qubit-to-ion assignments, and employing phase-flip twirling. The circuit also utilizes bit-flip symmetrization of readout.

In post-processing, measurement statistics from different twirled variants are combined, filtering out outlier bit strings based on a filter threshold. This threshold depends on device noise, the number of twirled variants, and shots per variant; a higher threshold better mitigates hardware noise but requires more resources, while a too-high threshold for fixed resources can lead to information loss.

3.4.3 Problem Instances

We simulate $0\nu\beta\beta$-decay by first preparing the initial state $|\psi_{\text{init}}\rangle = |\psi_{\text{vac}}^{(\text{lep})}\rangle, |\Delta^-\Delta^-\rangle$ using the Scalable Circuit ADAPT-VQE (SC-ADAPT-VQE) algorithm. This prepares the lepton vacuum and the quark wavefunction, where $|0\rangle^{\otimes 6}$ represents the fully occupied d quark register. The SC-ADAPT-VQE uses a single parameterized circuit $e^{i\theta\hat{O}}$ on a two-site lattice, built from the commutator of mass and kinetic terms for u quarks, ensuring QCD Hamiltonian symmetries.

Next, we time-evolve the initial state with $e^{-i\hat{H}t}$ using first-order Trotterization, simplifying the first step as the initial state is an eigenstate of $(\hat{H}_{\text{free}} + \hat{H}_{\text{Maj}} + \hat{H}_{\text{glue}})$. We developed a new, highly parallelizable method for constructing time-evolution circuits, overcoming Jordan-Wigner transformation challenges by designing circuits without Pauli-$\hat{Z}$ strings, then incorporating them via CZ conjugation.

The quantum circuit for $0\nu\beta\beta$-decay simulation involves preparing the initial state with SC-ADAPT-VQE, followed by two Trotterized time evolution steps, and finally flagging leakage events with ancilla-coupled lepton qubits before z-basis measurement.

In hardware experiments on 36 qubit quantum computers, we used 32 qubits to prepare the initial state and perform two steps of first-order Trotterized time evolution. We reduced two-qubit gates by truncating chromoelectric interaction and only keeping weak interaction terms acting on valence fermions, and removed small-angle two-qubit rotations. The compiled circuits require 470 R_{ZZ} gates.

We optimized circuit performance using noise tailoring, error mitigation (Pauli twirling, XY4 dynamic decoupling, measurement twirling via bit flipping), and error detection. Each circuit was

compiled into 96 twirled variants with unique qubit assignments and bit flips for readout error symmetrization, each with 150 shots (160 for $t = 2$), totaling 24,000 shots. The remaining four qubits flagged qubit leakage, and measured bit strings were post-selected based on ancilla states to indicate no leakage and conserve color and total electric charges.

3.4.4 Results

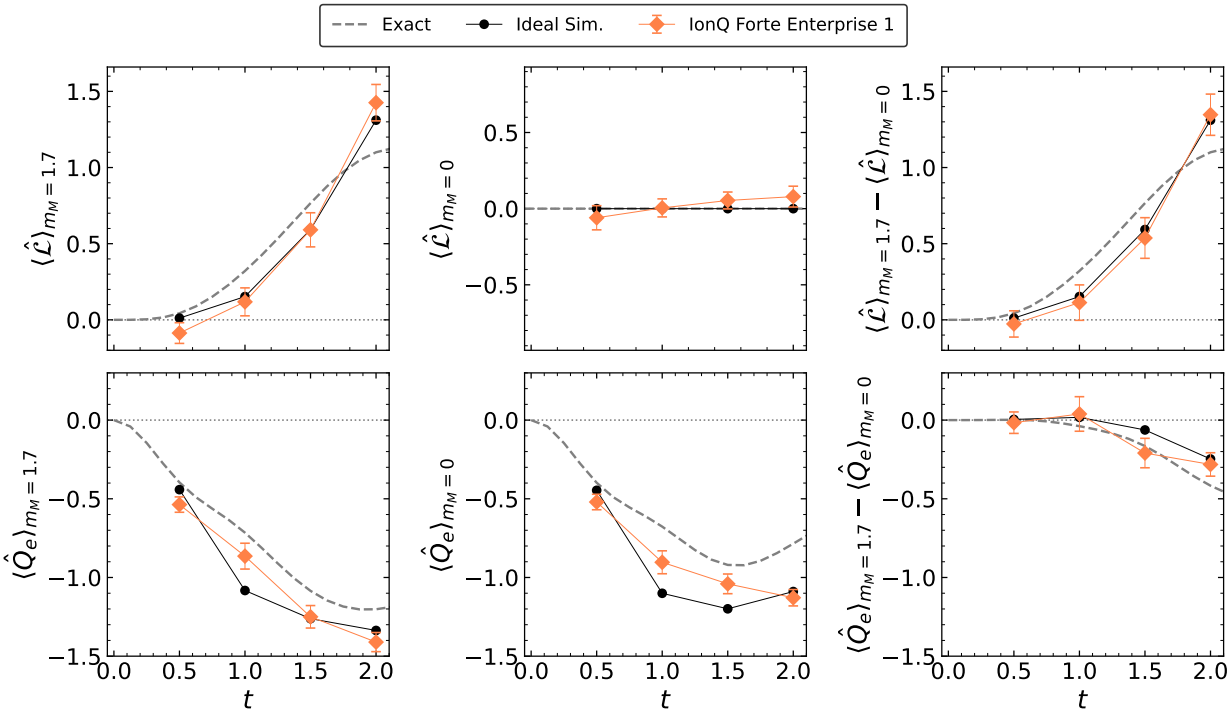

Figure 28: Lepton number time evolution during $|\Delta^-\Delta^-\rangle$ two-baryon state decay in 1+1D QCD. The circuit used two first-order Trotter steps with approximate valence-fermion weak interactions (470 two-qubit gates). Panels show results for $m_M = 1.7$ (left), $m_M = 0$ (center), and the difference between them (right). Data was obtained from IonQ Forte Enterprise. Black points indicate ideal simulations; the gray dotted line is for reference.

Weak decays can be detected and classified by measuring the total electric charge of the electrons $\hat{Q}_e$ and the lepton number $\hat{\mathcal{L}}$. Both are zero in the initial state; non-zero $\hat{Q}_e$ signals a decay, while non-zero $\hat{\mathcal{L}}$ indicates a neutrino Majorana mass effect.

Figure 28 displays IonQ Forte Enterprise results for $t = \{0.5, 1.0, 1.5, 2.0\}$ and $m_M = \{0.0, 1.7\}$. We utilized 14,400 shots for $t \leq 1.5$ and 24,000 shots for $t = 2.0$. Roughly 10% of shots survived filtering based on leakage and charge conservation. Error bars represent bootstrap resampling [68]. Ideal noiseless classical simulations, also plotted, show excellent agreement with the experimental data. Notably, for $m_M = 1.7$, we find a statistically significant signal for lepton number violation—a 10σ difference at $t = 2.0$ compared to the $m_M = 0$ case.

For $m_M = 0$, the lepton number remains zero, consistent with single β-decay and $2\nu\beta\beta$-decay, although electric charge is produced. Due to the confined simulation volume, baryons and leptons

interact continuously, preventing separation and sustaining a non-zero lepton electric charge density. Analyzing the time dependence of $\hat{\mathcal{L}}$ and $\hat{Q}_e$ could allow us to probe the underlying reaction mechanism. Predicting these values with accuracy requires the coherent summation of interfering reaction pathways—a task for which quantum computers are uniquely suited. Wavefunction evolution at the yocto-scale can provide key information to identify dominant decay pathways.

4 Results: Time-to-Solution Benchmarks

4.1 Time-To-Solution for Hidden Shift and Cosine QFT

The time it takes to hit high quality solutions for the first time when iteratively trying to sample solutions can be used to assess TTS performance on various benchmarks. For instance, in the Hidden Shift with permutations (HSBP) challenge at 36 qubits, we use permutations consisting of 80 randomly assigned CX gates. Sampling bit-strings close to the target shift becomes increasingly difficult when using a random sampler, and the confidence time-to-first-sample increases super exponentially. We compare against binonmial random samplers where each measured qubit is a perfect $p = 0.5$ bit-flip.

The Cosine QFT challenge at 36 qubits similarly shows that sampling bit-strings close to the known target solutions becomes increasingly difficult for leading superconducting chips, and the confidence time-to-first-sample increases super exponentially. Upon 1 million circuit executions, the random sampler failed to sample a single bit-string with less than 3 bit-flip errors, while IonQ Forte manages to sample the correct solution in minutes.

These examples illustrate that while raw shot-rate remains a valid indicator of hardware-level performance, shot "quality" as measured by the confidence time-to-first solution proxy, is the critical metric of system utility for real applications. The per-shot time for the Cosine QFT was set to 472us per shot and for Hidden Shift to 337us per shot. This corresponds to a 2q-gate time of around 200ns.

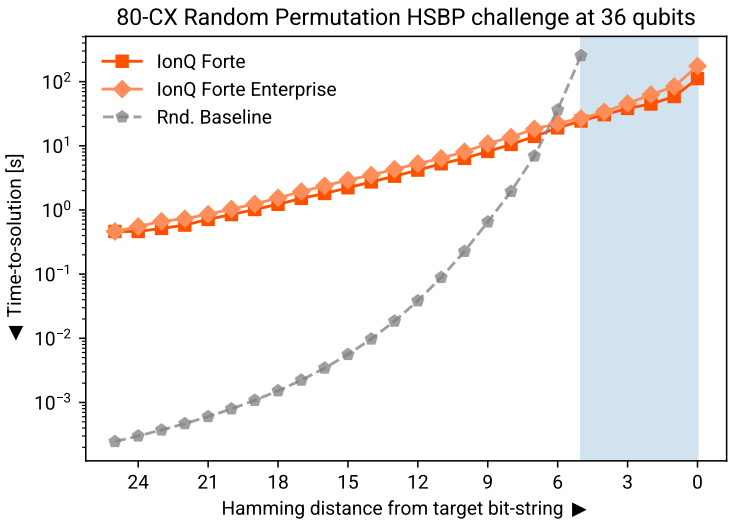

Figure 29: Time-to-solution for the HSBP challenge at 36 qubits with permutations consisting of 80 randomly assigned CX gates. The plot illustrates that sampling bit-strings close to the target shift becomes increasingly difficult for a random sampler and the confidence time-to-first-sample increases super exponentially. The random baseline was generated by sampling 1 million bit-string uniformly at random and computing the Hamming distance from the target shift; the distribution is binomial. On the contrary, IonQ Forte manages to sample the correct solution in minutes.

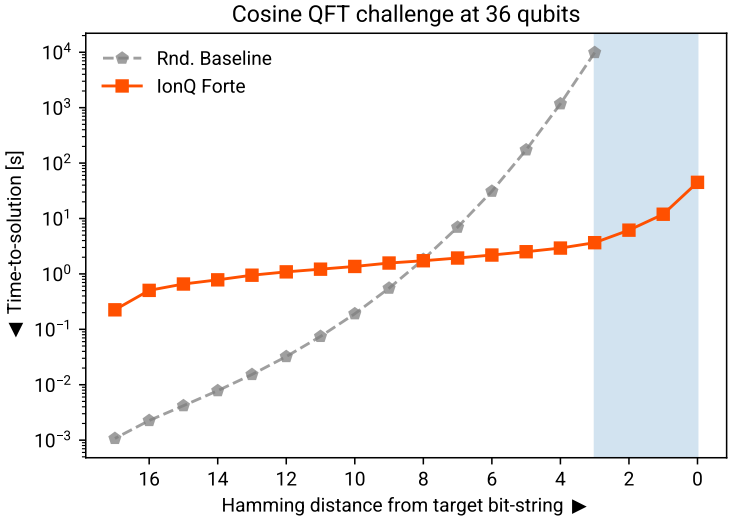

Figure 30: Time-to-solution for the Cosine QFT challenge at 36 qubits. This plot illustrates that sampling bit-strings close to the known target solutions to the Cosine QFT challenge becomes increasingly difficult for a random sampler and the confidence time-to-first-sample increases super exponentially. On the contrary, IonQ Forte manages to sample the correct solution in minutes.

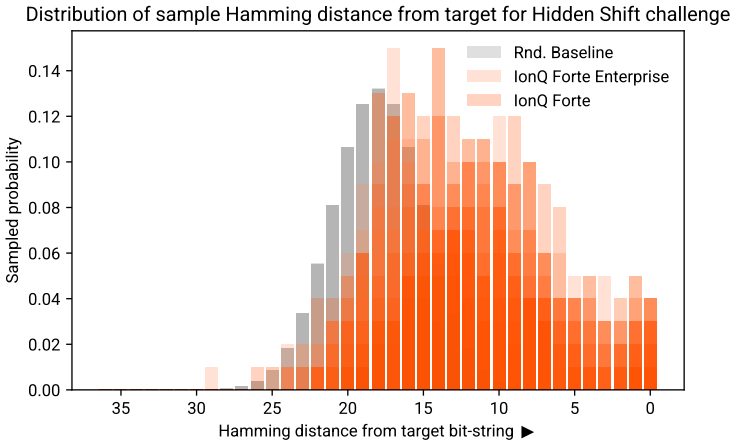

Figure 31: Shown are the quasi-probability distributions of the Hamming weight of bitvectors for the hidden shift problem. The shown distributions are taken from IonQ Forte (5k shots, orange) and the random sampler (1M shots, blue). As can be seen from the distributions, the IonQ device is able to sample from a heavy tail near approximation ratio (AR) of 1.0, whereas the random sampler has no mass there at all, even after taking 1M samples.

The underlying reason for the advantage illustrated in Fig. 29 is the difference in the ability to sample from long tails of the hidden shift bitvector distribution. As shown in Fig. 31, the IonQ device is able to find 'needles-in-a-hackstack' bitvectors that have high approximation ratio, whereas the random sampler fails, i.e., the long tails are sampled with exponentially low probability.

4.2 Time-To-Solution for LR-QAOA

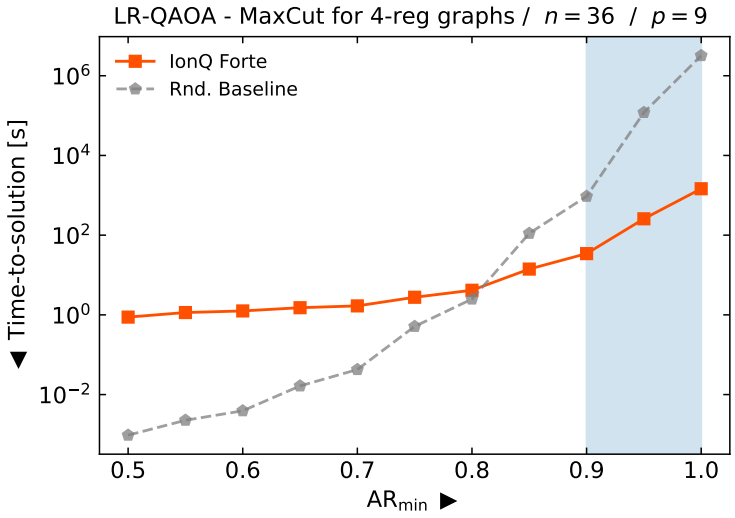

Figure 32: Time-to-solution (TTS) at 99% confidence as a function of the minimum approximation ratio $\mathrm{AR}_{\min}$ for $p = 9$ LR-QAOA on a 36-qubit, 4-regular MaxCut instance. The figure compares IonQ Forte data at 5,000 shots against the exact random-sampling baseline. IonQ Forte achieves finite TTS up to $\mathrm{AR}_{\min} = 1.0$, while the random baseline grows rapidly at high target approximation ratios.

To quantify the practical performance of quantum hardware relative to a non-algorithmic baseline, we compute the Time-to-Solution (TTS) metric at 99% confidence (see Figure 32). For a given success threshold $\mathrm{AR}_{\min}$, let p_s denote the probability that a single circuit execution yields an approximation ratio $\mathrm{AR} \geq \mathrm{AR}_{\min}$. The number of independent shots required to observe at least one such solution with a confidence level of 0.01 (i.e., 99% confidence) is

$$N_{\text{shots}} = \frac{\ln 0.01}{\ln(1 - p_s)}, \tag{17}$$

and the Time-to-Solution is then $\mathrm{TTS} = N_{\text{shots}} \cdot t_{\text{shot}}$, where t_{shot} is the wall-clock time per circuit execution. The TTS thus answers the question: how long must one run the device before being 99% certain of having sampled at least one bit string with approximation ratio at or above a given target?

We benchmark a 4-regular MaxCut instance on $n = 36$ qubits using a $p = 9$ layer LR-QAOA circuit. The depth $p = 9$ was selected because it yielded the highest raw approximation ratio in the IonQ Forte dataset among all tested depths, without any error mitigation applied. The IonQ Forte data comprises $5,000$ shots with $t_{\text{shot}} = 0.89$ s.

As a point of comparison, we also compute the TTS for a uniformly random sampler. Rather than drawing finite random samples, we evaluate the approximation ratio of every bit string in the full 2^{36}-dimensional Hilbert space and construct the exact AR distribution, so that $p_s = (\text{number of states with AR} \geq \mathrm{AR}_{\min})/2^{36}$. The per-shot time for this baseline is set to $t_{\text{shot}} = 445$ μs to give the random sampler a deliberately generous advantage. This corresponds to a 2q-gate time of around 200ns.

Relative to this exact random baseline, IonQ Forte remains strongly separated at high AR thresholds. With only $5,000$ shots, the device samples 14 bit strings that achieve the optimal cut ($\mathrm{AR} = 1.0$), yielding a finite TTS even at the most stringent threshold. At $\mathrm{AR}_{\mathrm{min}} = 0.90$, Forte's TTS is approximately 34 s, compared to 942 s for the random baseline. The gap widens further as the target threshold approaches the optimum, showing that the observed output distribution retains meaningful algorithmic structure rather than collapsing toward uniform random sampling. This TTS analysis therefore complements the raw AR and $\mathrm{AR}_{\mathrm{eff}}$ metrics by translating the separation from random sampling into an operational measure of solution cost.

To illustrate the projected scaling of the superconducting qubit system TTS beyond its observed range, we fit a model of the form $\mathrm{TTS}(\mathrm{AR}_{\mathrm{min}}) = e^{a\,x^2+b\,x+c}$ to its data in the regime $\mathrm{AR}_{\mathrm{min}} \in [0.50, 0.90]$, where $x \equiv \mathrm{AR}_{\mathrm{min}}$. Extrapolating this fit to $\mathrm{AR}_{\mathrm{min}} = 1.0$ yields a projected TTS on the order of 10^5 s, underscoring the impracticality of the noisy superconducting backend for this circuit depth and problem size. The shaded region in Figure 32 highlights the extrapolation domain where the superconducting qubit system has no observed data, reinforcing the conclusion that the trapped-ion device maintains a substantial advantage in solution quality for this benchmark.

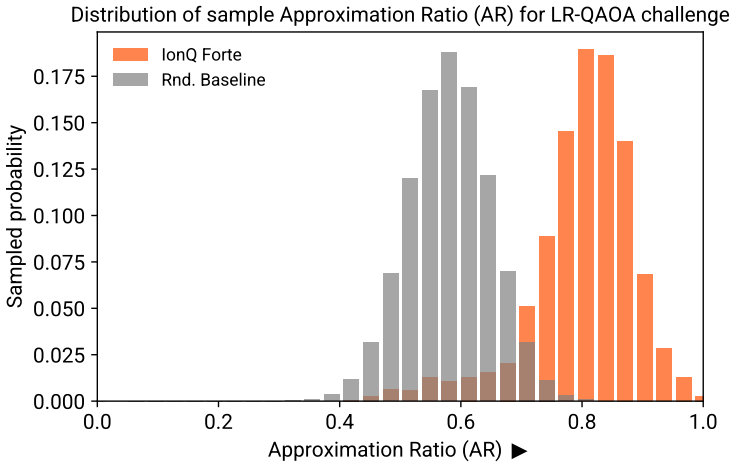

Figure 33: Shown are the quasi-probability distributions of the approximation ratio of bitvectors for the MaxCut problem. The shown distributions are taken from IonQ Forte (5k shots, orange) and the random sampler (1M shots, grey). As can be seen from the distributions, the IonQ device is able to sample from a heavy tail near approximation ratio (AR) of 1.0, whereas the random sampler has no mass there at all, even after taking 1M samples.

The underlying reason for the advantage illustrated in Fig. 32 is the difference in the ability to sample from long tails. As shown in Fig. 33, the IonQ device is able to find 'needles-in-a-hackstack' bitvectors that have high approximation ratio, whereas the random sampler fails, i.e., the long tails are sampled with exponentially low probability.

Finally, in Fig. **??**, we show that the IonQ device's ability to sample more effectively from tails of distributions translates into advantages over architectures such as superconducting qubits, which due to the mapping overhead to their 2D architecture behave closely to random samplers, i.e., they are not able to sample from valuable solutions from the long tails.

5 Conclusions

We introduced a scalable and principled framework for application-level quantum benchmarking that addresses the diverse needs of internal development, customer and community engagements, and comparative evaluation across quantum computing platforms. By focusing on application-driven metrics such as solution quality, Time-to-Solution, and cost proxies, the proposed framework provides a more meaningful characterization of system performance than traditional hardware-centric benchmarks.

A key contribution of this work is the identification of guiding principles—measurability, simplicity, scalability, and extensibility – that ensure benchmarks remain actionable, interpretable, and reproducible. These principles enable the construction of benchmark suites that can evolve alongside hardware improvements while maintaining consistency and credibility over time. Code that allow to independently run and extend the provided benchmarks was provided along with a detailed discussion of the benchmarks themselves.

The framework also highlights the importance of standardization and openness in benchmarking practices. By enabling full disclosure of implementation details and supporting reproducibility, it fosters trust among customers and facilitates collaboration across the broader quantum ecosystem. Moreover, the use of benchmark families allows for systematic scaling across problem sizes, ensuring relevance as quantum systems grow in capability.

Looking forward, this framework provides a foundation for the development of industry-wide benchmarking standards and the integration of application-level metrics into decision-making processes. It also opens the door to future extensions, including the incorporation of new application domains, improved cost models, and tighter integration with emerging quantum software and hardware stacks.

Ultimately, measuring what matters requires aligning benchmarking efforts with real-world use cases. The approach presented here represents a step toward that goal, enabling a more accurate and transparent assessment of quantum computing performance and its impact on practical applications.

References

[1] Noah Siekierski, Stefan Seritan, Neer Patel, Siyuan Niu, Thomas Lubinski, and Timothy Proctor. Software for creating scalable benchmarks from quantum algorithms, 2025.

[2] Thomas Lubinski, Joshua J. Goings, Karl Mayer, Sonika Johri, Nithin Reddy, Aman Mehta, Niranjan Bhatia, Sonny Rappaport, Daniel Mills, Charles H. Baldwin, Luning Zhao, Aaron Barbosa, Smarak Maity, and Pranav S. Mundada. Quantum algorithm exploration using application-oriented performance benchmarks, 2024.

[3] Neer Patel, Anish Giri, Hrushikesh Pramod Patil, Noah Siekierski, Avimita Chatterjee, Sonika Johri, Timothy Proctor, Thomas Lubinski, and Siyuan Niu. Platform-agnostic modular architecture for quantum benchmarking, 2025.

[4] Avimita Chatterjee, Sonny Rappaport, Anish Giri, Sonika Johri, Timothy Proctor, David E. Bernal Neira, Pratik Sathe, and Thomas Lubinski. A comprehensive cross-model framework for benchmarking the performance of quantum hamiltonian simulations. *IEEE Transactions on Quantum Engineering*, 2025.

[5] Thorsten Koch, David E. Bernal Neira, Ying Chen, Giorgio Cortiana, Daniel J. Egger, Raoul Heese, Narendra N. Hegade, Alejandro Gomez Cadavid, Rhea Huang, Toshinari Itoko, Thomas Kleinert, Pedro Maciel Xavier, Naeimeh Mohseni, Jhon A. Montanez-Barrera, Koji Nakano, Giacomo Nannicini, Corey O'Meara, Justin Pauckert, Manuel Proissl, Anurag Ramesh, Maximilian Schicker, Noriaki Shimada, Mitsuharu Takeori, Victor Valls, David Van Bulck, Stefan Woerner, and Christa Zoufal. Quantum optimization benchmarking library – the intractable decathlon, 2025.

[6] Thomas Lubinski, Carleton Coffrin, Catherine McGeoch, Pratik Sathe, Joshua Apanavicius, and David E. Bernal Neira. Optimization applications as quantum performance benchmarks, 2023.

[7] 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*, 2023.

[8] Nicolas P. D. Sawaya, Daniel Marti-Dafcik, Yang Ho, Daniel P. Tabor, David E. Bernal Neira, Alicia B. Magann, Shavindra Premaratne, Pradeep Dubey, Anne Matsuura, Nathan Bishop, Wibe A. de Jong, Simon Benjamin, Ojas Parekh, Norm Tubman, Katherine Klymko, and Daan Camps. Hamlib: A library of hamiltonians for benchmarking quantum algorithms and hardware. *Quantum*, 8:1559, 2024.

[9] Peter Mattson, Hanlin Tang, Gu-Yeon Wei, Carole-Jean Wu, Vijay Janapa Reddi, Christine Cheng, Cody Coleman, Gregory Diamos, David Kanter, Paulius Micikevicius, et al. Mlperf: An industry standard benchmark suite for machine learning performance. *IEEE Micro*, 40(2):8–16, 2020.

[10] Andrii Maksymov, Jason Nguyen, Yunseong Nam, and Igor Markov. Enhancing quantum computer performance via symmetrization, 2023.

[11] 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):4213, 23 July 2014.

[12] Nicholas H. Stair and Francesco A. Evangelista. Exploring hilbert space on a budget: Novel benchmark set and performance metric for testing electronic structure methods in the regime of strong correlation. *The Journal of Chemical Physics*, 153(10):104108, 09 2020.

[13] Joonho Lee, William J Huggins, Martin Head-Gordon, and K Birgitta Whaley. Generalized unitary coupled cluster wave functions for quantum computation. *Journal of chemical theory and computation*, 15(1):311–324, 8 January 2019.

[14] Vincent E Elfving, Marta Millaruelo, José A Gámez, and Christian Gogolin. Simulating quantum chemistry in the seniority-zero space on qubit-based quantum computers. *Physical review. A*, 103(3):032605, 11 March 2021.

[15] Luning Zhao, Joshua Goings, Kyujin Shin, Woomin Kyoung, Johanna I Fuks, June-Koo Kevin Rhee, Young Min Rhee, Kenneth Wright, Jason Nguyen, Jungsang Kim, and Sonika Johri. Orbital-optimized pair-correlated electron simulations on trapped-ion quantum computers. *npj quantum information*, 9(1):1–9, 23 June 2023.

[16] Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. A quantum approximate optimization algorithm, 2014.

[17] Deep Lall, Abhishek Agarwal, Weixi Zhang, Lachlan Lindoy, Tobias Lindström, Stephanie Webster, Simon Hall, Nicholas Chancellor, Petros Wallden, Raul Garcia-Patron, Elham Kashefi, Viv Kendon, Jonathan Pritchard, Alessandro Rossi, Animesh Datta, Theodoros Kapourniotis, Konstantinos Georgopoulos, and Ivan Rungger. A review and collection of metrics and benchmarks for quantum computers: definitions, methodologies and software, 2025.

[18] Jonathan Wurtz and Danylo Lykov. Fixed-angle conjectures for the quantum approximate optimization algorithm on regular maxcut graphs. *Phys. Rev. A*, 104:052419, Nov 2021.

[19] Ruslan Shaydulin and Marco Pistoia. Qaoa with $n \cdot p \geq 200$, 2023.

[20] J. A. Montanez-Barrera and Kristel Michielsen. Towards a universal qaoa protocol: Evidence of a scaling advantage in solving some combinatorial optimization problems, 2024.

[21] J. A. Montanez-Barrera, Kristel Michielsen, and David E. Bernal Neira. Evaluating the performance of quantum processing units at large width and depth, 2025.

[22] Vanessa Dehn, Martin Zaefferer, Gerhard Hellstern, Florentin Reiter, and Thomas Wellens. Extrapolation method to optimize linear-ramp qaoa parameters: Evaluation of qaoa runtime scaling, 2025.

[23] Lov K Grover. Fixed-point quantum search. *Physical Review Letters*, 95(15):150501, 2005.

[24] Theodore J Yoder, Guang Hao Low, and Isaac L Chuang. Fixed-point quantum search with an optimal number of queries. *Physical review letters*, 113(21):210501, 2014.

[25] John M Martyn, Zane M Rossi, Andrew K Tan, and Isaac L Chuang. Grand unification of quantum algorithms. *PRX quantum*, 2(4):040203, 2021.

[26] Alessandro Andrea Zecchi, Claudio Sanavio, Simona Perotto, and Sauro Succi. Improved amplitude amplification strategies for the quantum simulation of classical transport problems. *Quantum Science and Technology*, 10(3):035039, 2025.

[27] Zewen Zhang, Roger Paredes, Bhuvanesh Sundar, David Quiroga, Anastasios Kyrillidis, Leonardo Duenas-Osorio, Guido Pagano, and Kaden RA Hazzard. Grover-qaoa for 3-sat: Quadratic speedup, fair-sampling, and parameter clustering. *Quantum Science and Technology*, 10(1):015022, 2024.

[28] Rui Chao, Dawei Ding, Andras Gilyen, Cupjin Huang, and Mario Szegedy. Finding angles for quantum signal processing with machine precision. 2020.

[29] Jeongwan Haah. Product Decomposition of Periodic Functions in Quantum Signal Processing. *Quantum*, 3:190, October 2019.

[30] András Gilyén, Yuan Su, Guang Hao Low, and Nathan Wiebe. Quantum singular value transformation and beyond: exponential improvements for quantum matrix arithmetics. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, pages 193–204, 2019.

[31] Yulong Dong, Xiang Meng, K. Birgitta Whaley, and Lin Lin. Efficient phase-factor evaluation in quantum signal processing. *Physical Review A*, 103(4), April 2021.

[32] Ahmad Bennakhi, Gregory T Byrd, and Paul Franzon. Analyzing quantum circuit depth reduction with ancilla qubits in mcx gates. In *2024 IEEE International Conference on Quantum Computing and Engineering (QCE)*, volume 2, pages 510–511. IEEE, 2024.

[33] Keiron O'Shea and Ryan Nash. An introduction to convolutional neural networks, 2015.

[34] Iris Cong, Soonwon Choi, and Mikhail D. Lukin. Quantum convolutional neural networks. *Nat. Phys.*, 2019.

[35] Yanxuan Lü, Qing Gao, Jinhu Lü, Maciej Ogorzałek, and Jin Zheng. A quantum convolutional neural network for image classification. In *2021 40th Chinese Control Conference (CCC)*, pages 6329–6334, 2021.

[36] Tak Hur, Leeseok Kim, and Daniel K. Park. Quantum convolutional neural network for classical data classification. *Quantum Machine Intelligence*, 2022.

[37] Emily Jimin Roh, Chaemoon Im, Wonjun Jeong, and Soohyun Park. Computation-efficient quantum convolutional neural networks for autonomous driving applications. *The Journal of Supercomputing*, 2025.

[38] Chao-Han Huck Yang, Jun Qi, Samuel Yen-Chi Chen, Pin-Yu Chen, Sabato Marco Siniscalchi, Xiaoli Ma, and Chin-Hui Lee. Decentralizing feature extraction with quantum convolutional neural network for automatic speech recognition. In *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6523–6527, 2021.

[39] Hala Elhag, Tobias Hartung, Karl Jansen, Lento Nagano, Giorgio Menicagli Pirina, and Alice Di Tucci. Quantum convolutional neural networks for jet images classification, 2025.

[40] Seon-Geun Jeong, Quang-Vinh Do, Hae-Ji Hwang, Mikio Hasegawa, Hiroo Sekiya, and Won-Joo Hwang. Hybrid quantum convolutional neural networks for uwb signal classification. *IEEE Access*, 11:113726–113739, 2023.

[41] Martin J. Sklar. Fonctions de repartition a n dimensions et leurs marges. 1959.

[42] Alexander J. Mcneil. Dependence modeling with copulas, by harry joe. monographs on statistics and applied probability 134, published by crc press, 2015. total number of pages: 18 + 462. isbn: 978-1-4665-8322-1 (hardback). *Journal of Time Series Analysis*, 36(4):599–600, July 2015.

[43] Umberto Cherubini, Elisa Luciano, and Walter Vecchiato. *Copula methods in finance*. John Wiley & Sons, 2004.

[44] Régis Lebrun and Anne Dutfoy. An innovating analysis of the nataf transformation from the copula viewpoint. *Probabilistic Engineering Mechanics*, 24(3):312–320, 2009.

[45] Philippe Lambert and François Vandenhende. A copula-based model for multivariate non-normal longitudinal data: analysis of a dose titration safety study on a new antidepressant. *Statistics in Medicine*, 21(21):3197–3217, 2002.

[46] Joe Harry. *Multivariate models and dependence concepts / Harry Joe*. Chapman & Hall, London [etc., 1997.

[47] Daiwei Zhu, Weiwei Shen, Annarita Giani, Saikat Ray Majumder, Bogdan Neculaes, and Sonika Johri. Copula-based risk aggregation with trapped ion quantum computers, 2022.

[48] Manuel S Rudolph, Jing Chen, Jacob Miller, Atithi Acharya, and Alejandro Perdomo-Ortiz. Decomposition of matrix product states into shallow quantum circuits. *Quantum Science and Technology*, 9(1):015012, 2023.

[49] Jason Iaconis and Sonika Johri. Tensor network based efficient quantum data loading of images. *arXiv preprint arXiv:2310.05897*, 2023.

[50] Martin Rötteler. *Quantum algorithms for highly non-linear Boolean functions*, pages 448–457.

[51] Titus D. Morris, Ananth Kaushik, Martin Roetteler, and Phillip C. Lotshaw. Performant near-term quantum combinatorial optimization, 2024.

[52] Willie Aboumrad, Daiwei Zhu, Claudio Girotto, François-Henry Rouet, Jezer Jojo, Robert Lucas, Jay Pathak, Ananth Kaushik, and Martin Roetteler. Accelerating large-scale linear algebra using variational quantum imaginary time evolution, 2025.

[53] Joonho Lee, Hung Q Pham, and David R Reichman. Twenty years of auxiliary-field quantum Monte Carlo in quantum chemistry: An overview and assessment on main group chemistry and bond-breaking. *Journal of chemical theory and computation*, 18(12):7024–7042, 13 December 2022.

[54] Mario Motta and Shiwei Zhang. Ab initio computations of molecular systems by the auxiliary-field quantum Monte Carlo method. *Wiley interdisciplinary reviews. Computational molecular science*, 8(5):e1364, September 2018.

[55] Shiwei Zhang. Ab initio electronic structure calculations by auxiliary-field quantum Monte Carlo. In *Handbook of Materials Modeling*, pages 123–149. Springer International Publishing, Cham, 2020.

[56] James Shee, John L Weber, David R Reichman, Richard A Friesner, and Shiwei Zhang. On the potentially transformative role of auxiliary-field quantum Monte Carlo in quantum chemistry: A highly accurate method for transition metals and beyond. *The Journal of chemical physics*, 158(14):140901, 14 April 2023.

[57] William J Huggins, Bryan A O'Gorman, Nicholas C Rubin, David R Reichman, Ryan Babbush, and Joonho Lee. Unbiasing fermionic quantum Monte Carlo with a quantum computer. *Nature*, 603(7901):416–420, March 2022.

[58] Kianna Wan, William J Huggins, Joonho Lee, and Ryan Babbush. Matchgate shadows for fermionic quantum simulation. *Communications in mathematical physics*, 404(2):629–700, December 2023.

[59] Matthew Kiser, Anna Schroeder, Gian-Luca R Anselmetti, Chandan Kumar, Nikolaj Moll, Michael Streif, and Davide Vodola. Classical and quantum cost of measurement strategies for quantum-enhanced auxiliary field quantum Monte Carlo. *New journal of physics*, 26(3):033022, 1 March 2024.

[60] Benchen Huang, Yi-Ting Chen, Brajesh Gupt, Martin Suchara, Anh Tran, Sam McArdle, and Giulia Galli. Evaluating a quantum-classical quantum Monte Carlo algorithm with Matchgate shadows. *Physical review research*, 6(4):043063, 24 October 2024.

[61] Luning Zhao, Joshua J Goings, Willie Aboumrad, Andrew Arrasmith, Lazaro Calderin, Spencer Churchill, Dor Gabay, Thea Harvey-Brown, Melanie Hiles, Magda Kaja, Matthew Keesan, Karolina Kulesz, Andrii Maksymov, Mei Maruo, Mauricio Muñoz, Bas Nijholt, Rebekah Schiller, Yvette de Sereville, Amy Smidutz, Felix Tripier, Grace Yao, Trishal Zaveri, Coleman Collins, Martin Roetteler, Evgeny Epifanovsky, Arseny Kovyrshin, Lars Tornberg, Anders Broo, Jeff R Hammond, Zohim Chandani, Pradnya Khalate, Elica Kyoseva, Yi-Ting Chen, Eric M Kessler, Cedric Yen-Yu Lin, Gandhi Ramu, Ryan Shaffer, Michael Brett, Benchen Huang, Maxime R Hugues, and Tyler Y Takeshita. Quantum-classical auxiliary field quantum Monte Carlo with matchgate shadows on trapped ion quantum computers. *arXiv [quant-ph]*, 27 June 2025.

[62] Apurva Tiwari, Jason Iaconis, Jezer Jojo, Sayonee Ray, Martin Roetteler, Chris Hill, and Jay Pathak. Algorithmic advances towards a realizable quantum lattice boltzmann method, 2025.

[63] Ljubomir Budinski. Quantum algorithm for the advection–diffusion equation simulated with the lattice Boltzmann method. *Quantum Information Processing*, 20(2):57, February 2021.

[64] Ljubomir Budinski. Quantum algorithm for the Navier Stokes equations by using the stream-function vorticity formulation and the lattice Boltzmann method. *International Journal of Quantum Information*, 20(02):2150039, March 2022.

[65] David Wawrzyniak, Josef Winter, Steffen Schmidt, Thomas Indinger, Christian F. Janßen, Uwe Schramm, and Nikolaus A. Adams. A quantum algorithm for the lattice-Boltzmann method advection-diffusion equation. *Computer Physics Communications*, 306:109373, January 2025.

[66] A. A. Mohamad. *Lattice Boltzmann Method: Fundamentals and Engineering Applications with Computer Codes*. Springer London, London, 2019.

[67] P. L. Bhatnagar, E. P. Gross, and M. Krook. A model for collision processes in gases. i. small amplitude processes in charged and neutral one-component systems. *Phys. Rev.*, 94:511–525, May 1954.

[68] Ivan A Chernyshev, Roland C Farrell, Marc Illa, Martin J Savage, Andrii Maksymov, Felix Tripier, Miguel Angel Lopez-Ruiz, Andrew Arrasmith, Yvette de Sereville, Aharon Brodutch, et al. Pathfinding quantum simulations of neutrinoless double-β decay. *Nature Communications*, 2026.

Appendices

A Format and Requirements for Closed and Open Categories

The following tables collect results from a benchmarking campaign carried out on IonQ's Aria, Forte, and Forte Enterprise systems. Each line corresponds to a particular instance of the benchmark, characterized by a unique identifier as provided in the accompanying code repository at github.com/ionq-publications/apps-benchmark. The categories of data reported are the Domain to which the benchmark broadly belongs, the specific type of Problem that is addressed, and which Algorithm is used. Regarding the specific implementation, the total number of qubits (#q), the total number of different circuits (#qc), the total number of single-qubit gates (#1q) and two-qubit gates (#2q), as well as the total number of shots (#s) is reported. Backend specifies against which specific quantum machine the benchmark was run. The field EM indicates whether an error mitigation method was used. The score is a single numerical value that is defined for each benchmark and can be computed using the code provided. Exec. Time and Energy are the total QPU time and total energy spent to complete the benchmark as defined in Section 1.2. Execution time is reported in total number of seconds (s), Energy is reported in total kiloWatt-hours (kWh). Note that only Forte and Forte-Enterprise are equipped with instrumentation to measure the energy consumption, i.e., the corresponding values for Aria are left blank ("-").

The framework is designed to be extensible and we provide a guide (see `README.md` and `DIY_BACKEND.md`) on how to make your own backend connector for other QPUs. The following are requirements for reporting results for the Closed and Open divisions:

- The Closed Division is designed for fair, direct comparisons of hardware and software frameworks. This means the following: **Pre-processing/Post-processing:** Must be equivalent to the reference. **Algorithm change:** Prohibited. **Compliance Runs:** Must pass mandatory compliance tests to ensure score, which are verified by the benchmarking harness. **Dataset:** Must be identical. **Reproducibility:** Code used for the benchmark must be open-sourced.

- The Open Division encourages innovation in the approaches to tackle the problems. This means the following: **Quantum algorithm flexibility**: Allows changing the quantum circuits, error mitigation, model architecture, etc. **Pre-processing/Post-processing:** Allows arbitrary pre-processing or post-processing as long as it does not rely the solution of the problem. **Target Quality:** The model must still meet the defined target quality (accuracy) metric. **Software/Framework:** Can use different methods.

- General requirements for both Divisions: **System Availability:** The hardware and software stack must be available (or soon available) for purchase or access purchase, although preview hardware might be considered as long as the required metrics mentioned below are reported. **Reproducibility:** Methods that allow reproduction of the results (ideally, based on source code) must be shared. **Auditability:** Submissions are subject to a random or committee-selected audit process to verify compliance, especially for the Closed Division. **Metrics:** Must report all categories provided in the headers, in particular Score and Execution Time. Energy is optional, but reporting it is preferred.

For contributions to the framework, participants may open tickets (github Issues) against the underlying open source repository.

B Tables with Numerical Benchmark Results

Domain	Problem	Algorithm	#q	#qc	#1q	#2q	Shots	Backend	EM?	Score	Exec. Time (s)	Energy (kWh)
hidden shift	ccx ladder	hidden shift	8	10	1,510	80	1,000	IonQ Forte	N	0.651	73.385	0.10
hidden shift	ccx ladder	hidden shift	8	10	1,510	80	1,000	IonQ Forte-E	N	0.632	81.848	0.11
hidden shift	ccx ladder	hidden shift	12	10	2,200	120	1,000	IonQ Forte	N	0.48	127.524	0.17
hidden shift	ccx ladder	hidden shift	12	10	2,200	120	1,000	IonQ Forte-E	N	0.377	141.159	0.19
hidden shift	ccx ladder	hidden shift	16	10	2,940	160	1,000	IonQ Forte	N	0.299	180.439	0.24
hidden shift	ccx ladder	hidden shift	16	10	2,940	160	1,000	IonQ Forte-E	N	0.227	200.323	0.27
hidden shift	ccx ladder	hidden shift	20	10	3,700	200	1,000	IonQ Forte	N	0.193	234.91	0.32
hidden shift	ccx ladder	hidden shift	20	10	3,700	200	1,000	IonQ Forte-E	N	0.123	258.84	0.35
hidden shift	ccx ladder	hidden shift	24	10	4,490	240	1,000	IonQ Forte	N	0.117	287.983	0.39
hidden shift	ccx ladder	hidden shift	24	10	4,490	240	1,000	IonQ Forte-E	N	0.067	344.793	0.47
hidden shift	ccx ladder	hidden shift	28	10	5,110	280	1,000	IonQ Forte	N	0.072	342.27	0.46
hidden shift	ccx ladder	hidden shift	28	10	5,110	280	1,000	IonQ Forte-E	N	0.047	379.403	0.52
hidden shift	ccx ladder	hidden shift	32	10	5,910	320	1,000	IonQ Forte	N	0.028	394.945	0.53
hidden shift	ccx ladder	hidden shift	32	10	5,910	320	1,000	IonQ Forte-E	N	0.02	441.501	0.60
hidden shift	ccx ladder	hidden shift	36	10	6,550	360	1,000	IonQ Forte	N	0.015	448.729	0.61
hidden shift	ccx ladder	hidden shift	36	10	6,550	360	1,000	IonQ Forte-E	N	0.005	516.664	0.70
hidden shift	cx ladder	hidden shift	8	10	1,410	200	1,000	IonQ Forte	N	0.778	39.413	0.05
hidden shift	cx ladder	hidden shift	8	10	1,410	200	1,000	IonQ Forte-E	N	0.813	46.318	0.06
hidden shift	cx ladder	hidden shift	12	10	2,340	320	1,000	IonQ Forte	N	0.73	56.95	0.08
hidden shift	cx ladder	hidden shift	12	10	2,340	320	1,000	IonQ Forte-E	N	0.671	65.071	0.09
hidden shift	cx ladder	hidden shift	16	10	2,950	440	1,000	IonQ Forte	N	0.674	74.851	0.10
hidden shift	cx ladder	hidden shift	16	10	2,950	440	1,000	IonQ Forte-E	N	0.572	84.914	0.12
hidden shift	cx ladder	hidden shift	20	10	3,780	560	1,000	IonQ Forte	N	0.564	92.209	0.12
hidden shift	cx ladder	hidden shift	20	10	3,780	560	1,000	IonQ Forte-E	N	0.512	101.152	0.14
hidden shift	cx ladder	hidden shift	24	10	4,520	680	1,000	IonQ Forte	N	0.528	110.068	0.15
hidden shift	cx ladder	hidden shift	24	10	4,520	680	1,000	IonQ Forte-E	N	0.429	149.021	0.20
hidden shift	cx ladder	hidden shift	28	10	5,130	800	1,000	IonQ Forte	N	0.483	127.894	0.17
hidden shift	cx ladder	hidden shift	28	10	5,130	800	1,000	IonQ Forte-E	N	0.319	172.51	0.23
hidden shift	cx ladder	hidden shift	32	10	5,960	920	1,000	IonQ Forte	N	0.374	145.262	0.20
hidden shift	cx ladder	hidden shift	32	10	5,960	920	1,000	IonQ Forte-E	N	0.233	169.494	0.23
hidden shift	cx ladder	hidden shift	36	10	6,710	1,040	1,000	IonQ Forte	N	0.279	163.111	0.22
hidden shift	cx ladder	hidden shift	36	10	6,710	1,040	1,000	IonQ Forte-E	N	0.243	209.735	0.29
hidden shift	mcx	hidden shift	8	10	2,150	640	1,000	IonQ Forte	N	0.656	85.021	0.12
hidden shift	mcx	hidden shift	8	10	2,150	640	1,000	IonQ Forte-E	N	0.624	94.09	0.13
hidden shift	mcx	hidden shift	12	10	3,630	1,000	1,000	IonQ Forte	N	0.278	185.205	0.25
hidden shift	mcx	hidden shift	12	10	3,630	1,000	1,000	IonQ Forte-E	N	0.244	205.708	0.28
hidden shift	mcx	hidden shift	16	10	5,380	1,680	1,000	IonQ Forte	N	0.162	276.801	0.37
hidden shift	mcx	hidden shift	16	10	5,380	1,680	1,000	IonQ Forte-E	N	0.085	301.842	0.41
hidden shift	mcx	hidden shift	20	10	7,050	2,360	1,000	IonQ Forte	N	0.071	368.152	0.50
hidden shift	mcx	hidden shift	20	10	7,050	2,360	1,000	IonQ Forte-E	N	0.02	402.411	0.55
hidden shift	mcx	hidden shift	24	10	8,780	3,040	1,000	IonQ Forte	N	0.032	459.045	0.62
hidden shift	mcx	hidden shift	24	10	8,780	3,040	1,000	IonQ Forte-E	N	0.006	526.542	0.72
hidden shift	mcx	hidden shift	28	10	10,430	3,720	1,000	IonQ Forte	N	0.013	549.867	0.74
hidden shift	mcx	hidden shift	28	10	10,430	3,720	1,000	IonQ Forte-E	N	0.003	626.09	0.85
hidden shift	mcx	hidden shift	32	10	12,100	4,400	1,000	IonQ Forte	N	0.008	640.226	0.87
hidden shift	mcx	hidden shift	32	10	12,100	4,400	1,000	IonQ Forte-E	N	0.003	707.269	0.96
hidden shift	mcx	hidden shift	36	10	13,800	5,080	1,000	IonQ Forte	N	0	733.256	0.99
hidden shift	mcx	hidden shift	36	10	13,800	5,080	1,000	IonQ Forte-E	N	0	832.472	1.13
hidden shift	random 010 cx	hidden shift	8	9	1,452	432	900	IonQ Forte	N	0.7689	58.653	0.08
hidden shift	random 010 cx	hidden shift	8	9	1,452	432	900	IonQ Forte-E	N	0.7411	39.854	0.05
hidden shift	random 010 cx	hidden shift	12	9	2,118	468	900	IonQ Forte	N	0.7478	72.003	0.10
hidden shift	random 010 cx	hidden shift	12	9	2,118	468	900	IonQ Forte-E	N	0.6689	52.536	0.07
hidden shift	random 010 cx	hidden shift	16	9	2,814	504	900	IonQ Forte	N	0.6511	89.464	0.12
hidden shift	random 010 cx	hidden shift	16	9	2,814	504	900	IonQ Forte-E	N	0.5833	61.737	0.08
hidden shift	random 010 cx	hidden shift	20	9	3,480	540	900	IonQ Forte	N	0.5944	94.667	0.13
hidden shift	random 010 cx	hidden shift	20	9	3,480	540	900	IonQ Forte-E	N	0.4967	62.925	0.09
hidden shift	random 010 cx	hidden shift	24	9	4,176	576	900	IonQ Forte	N	0.5167	114.009	0.15
hidden shift	random 010 cx	hidden shift	24	9	4,176	576	900	IonQ Forte-E	N	0.4778	79.754	0.11
hidden shift	random 010 cx	hidden shift	28	9	4,782	612	900	IonQ Forte	N	0.5344	88.39	0.12
hidden shift	random 010 cx	hidden shift	28	9	4,782	612	900	IonQ Forte-E	N	0.47	75.913	0.10
hidden shift	random 010 cx	hidden shift	32	9	5,538	648	900	IonQ Forte	N	0.3889	90.026	0.12
hidden shift	random 010 cx	hidden shift	32	9	5,538	648	900	IonQ Forte-E	N	0.4644	81.291	0.11
hidden shift	random 010 cx	hidden shift	36	9	6,174	684	900	IonQ Forte	N	0.4011	100.799	0.14
hidden shift	random 010 cx	hidden shift	36	9	6,174	684	900	IonQ Forte-E	N	0.3778	94.624	0.13

Domain	Problem	Algorithm	#q	#qc	#1q	#2q	Shots	Backend	EM?	Score	Exec. Time (s)	Energy (kWh)
hidden shift	random 020 cx	hidden shift	8	9	1,422	792	900	IonQ Forte	N	0.7356	67.425	0.09
hidden shift	random 020 cx	hidden shift	8	9	1,422	792	900	IonQ Forte-E	N	0.7144	45.529	0.06
hidden shift	random 020 cx	hidden shift	12	9	2,058	828	900	IonQ Forte	N	0.6478	84.328	0.11
hidden shift	random 020 cx	hidden shift	12	9	2,058	828	900	IonQ Forte-E	N	0.5622	63.751	0.09
hidden shift	random 020 cx	hidden shift	16	9	2,784	864	900	IonQ Forte	N	0.4756	116.68	0.16
hidden shift	random 020 cx	hidden shift	16	9	2,784	864	900	IonQ Forte-E	N	0.4222	90.778	0.12
hidden shift	random 020 cx	hidden shift	20	9	3,450	900	900	IonQ Forte	N	0.3956	132.349	0.18
hidden shift	random 020 cx	hidden shift	20	9	3,450	900	900	IonQ Forte-E	N	0.4078	92.105	0.13
hidden shift	random 020 cx	hidden shift	24	9	4,326	936	900	IonQ Forte	N	0.4156	134.44	0.18
hidden shift	random 020 cx	hidden shift	24	9	4,326	936	900	IonQ Forte-E	N	0.4	92.949	0.13
hidden shift	random 020 cx	hidden shift	28	9	4,992	972	900	IonQ Forte	N	0.3667	130.604	0.18
hidden shift	random 020 cx	hidden shift	28	9	4,992	972	900	IonQ Forte-E	N	0.3133	111.503	0.15
hidden shift	random 020 cx	hidden shift	32	9	5,808	1,008	900	IonQ Forte	N	0.2656	141.552	0.19
hidden shift	random 020 cx	hidden shift	32	9	5,808	1,008	900	IonQ Forte-E	N	0.2544	127.912	0.17
hidden shift	random 020 cx	hidden shift	36	9	6,474	1,044	900	IonQ Forte	N	0.2256	146.75	0.20
hidden shift	random 020 cx	hidden shift	36	9	6,474	1,044	900	IonQ Forte-E	N	0.2322	129.617	0.18
hidden shift	random 030 cx	hidden shift	8	9	1,392	1,152	900	IonQ Forte	N	0.6422	92.494	0.13
hidden shift	random 030 cx	hidden shift	8	9	1,392	1,152	900	IonQ Forte-E	N	0.5456	70.539	0.10
hidden shift	random 030 cx	hidden shift	12	9	2,118	1,188	900	IonQ Forte	N	0.4933	123.976	0.17
hidden shift	random 030 cx	hidden shift	12	9	2,118	1,188	900	IonQ Forte-E	N	0.4589	97.78	0.13
hidden shift	random 030 cx	hidden shift	16	9	2,814	1,224	900	IonQ Forte	N	0.5033	149.784	0.20
hidden shift	random 030 cx	hidden shift	16	9	2,814	1,224	900	IonQ Forte-E	N	0.3222	113.674	0.15
hidden shift	random 030 cx	hidden shift	20	9	3,660	1,260	900	IonQ Forte	N	0.3056	160.547	0.22
hidden shift	random 030 cx	hidden shift	20	9	3,660	1,260	900	IonQ Forte-E	N	0.3011	123.755	0.17
hidden shift	random 030 cx	hidden shift	24	9	4,296	1,296	900	IonQ Forte	N	0.3022	176.474	0.24
hidden shift	random 030 cx	hidden shift	24	9	4,296	1,296	900	IonQ Forte-E	N	0.26	128.912	0.18
hidden shift	random 030 cx	hidden shift	28	9	4,962	1,332	900	IonQ Forte	N	0.2722	170.091	0.23
hidden shift	random 030 cx	hidden shift	28	9	4,962	1,332	900	IonQ Forte-E	N	0.2022	152.159	0.21
hidden shift	random 030 cx	hidden shift	32	9	5,538	1,368	900	IonQ Forte	N	0.13	177.212	0.24
hidden shift	random 030 cx	hidden shift	32	9	5,538	1,368	900	IonQ Forte-E	N	0.1689	152.406	0.21
hidden shift	random 030 cx	hidden shift	36	9	6,474	1,404	900	IonQ Forte	N	0.2089	185.852	0.25
hidden shift	random 030 cx	hidden shift	36	9	6,474	1,404	900	IonQ Forte-E	N	0.1689	165.155	0.22
hidden shift	random 040 cx	hidden shift	8	9	1,452	1,512	900	IonQ Forte	N	0.5989	98.713	0.13
hidden shift	random 040 cx	hidden shift	8	9	1,452	1,512	900	IonQ Forte-E	N	0.5133	76.05	0.10
hidden shift	random 040 cx	hidden shift	12	9	2,088	1,548	900	IonQ Forte	N	0.4356	137.502	0.19
hidden shift	random 040 cx	hidden shift	12	9	2,088	1,548	900	IonQ Forte-E	N	0.3356	116.662	0.16
hidden shift	random 040 cx	hidden shift	16	9	2,784	1,584	900	IonQ Forte	N	0.3244	176.273	0.24
hidden shift	random 040 cx	hidden shift	16	9	2,784	1,584	900	IonQ Forte-E	N	0.2478	132.006	0.18
hidden shift	random 040 cx	hidden shift	20	9	3,630	1,620	900	IonQ Forte	N	0.2733	186.788	0.25
hidden shift	random 040 cx	hidden shift	20	9	3,630	1,620	900	IonQ Forte-E	N	0.2411	141.119	0.19
hidden shift	random 040 cx	hidden shift	24	9	4,266	1,656	900	IonQ Forte	N	0.2189	228.36	0.31
hidden shift	random 040 cx	hidden shift	24	9	4,266	1,656	900	IonQ Forte-E	N	0.1478	162.888	0.22
hidden shift	random 040 cx	hidden shift	28	9	5,142	1,692	900	IonQ Forte	N	0.17	201.999	0.27
hidden shift	random 040 cx	hidden shift	28	9	5,142	1,692	900	IonQ Forte-E	N	0.1556	169.98	0.23
hidden shift	random 040 cx	hidden shift	32	9	5,568	1,728	900	IonQ Forte	N	0.1489	205.503	0.28
hidden shift	random 040 cx	hidden shift	32	9	5,568	1,728	900	IonQ Forte-E	N	0.1189	179.409	0.24
hidden shift	random 040 cx	hidden shift	36	9	6,474	1,764	900	IonQ Forte	N	0.1256	219.89	0.30
hidden shift	random 040 cx	hidden shift	36	9	6,474	1,764	900	IonQ Forte-E	N	0.09111	186.597	0.25
hidden shift	random 050 cx	hidden shift	8	9	1,392	1,872	900	IonQ Forte	N	0.6189	92.099	0.12
hidden shift	random 050 cx	hidden shift	8	9	1,392	1,872	900	IonQ Forte-E	N	0.5211	79.023	0.11
hidden shift	random 050 cx	hidden shift	12	9	2,118	1,908	900	IonQ Forte	N	0.2767	176.531	0.24
hidden shift	random 050 cx	hidden shift	12	9	2,118	1,908	900	IonQ Forte-E	N	0.2778	143.954	0.20
hidden shift	random 050 cx	hidden shift	16	9	2,754	1,944	900	IonQ Forte	N	0.2889	197.039	0.27
hidden shift	random 050 cx	hidden shift	16	9	2,754	1,944	900	IonQ Forte-E	N	0.2133	148.775	0.20
hidden shift	random 050 cx	hidden shift	20	9	3,570	1,980	900	IonQ Forte	N	0.1667	242.501	0.33
hidden shift	random 050 cx	hidden shift	20	9	3,570	1,980	900	IonQ Forte-E	N	0.1	174.372	0.24
hidden shift	random 050 cx	hidden shift	24	9	4,026	2,016	900	IonQ Forte	N	0.1844	254.145	0.34
hidden shift	random 050 cx	hidden shift	24	9	4,026	2,016	900	IonQ Forte-E	N	0.1378	189.326	0.26
hidden shift	random 050 cx	hidden shift	28	9	5,202	2,052	900	IonQ Forte	N	0.1567	234.343	0.32
hidden shift	random 050 cx	hidden shift	28	9	5,202	2,052	900	IonQ Forte-E	N	0.1122	198.223	0.27
hidden shift	random 050 cx	hidden shift	32	9	5,718	2,088	900	IonQ Forte	N	0.11	230.949	0.31
hidden shift	random 050 cx	hidden shift	32	9	5,718	2,088	900	IonQ Forte-E	N	0.1033	192.531	0.26
hidden shift	random 050 cx	hidden shift	36	9	6,564	2,124	900	IonQ Forte	N	0.07667	254.666	0.34
hidden shift	random 050 cx	hidden shift	36	9	6,564	2,124	900	IonQ Forte-E	N	0.05667	213.434	0.29
hidden shift	random 060 cx	hidden shift	8	9	1,422	2,232	900	IonQ Forte	N	0.6511	78.315	0.11
hidden shift	random 060 cx	hidden shift	8	9	1,422	2,232	900	IonQ Forte-E	N	0.5767	64.018	0.09
hidden shift	random 060 cx	hidden shift	12	9	2,118	2,268	900	IonQ Forte	N	0.3033	169.14	0.23
hidden shift	random 060 cx	hidden shift	12	9	2,118	2,268	900	IonQ Forte-E	N	0.2511	142.753	0.19
hidden shift	random 060 cx	hidden shift	16	9	2,904	2,304	900	IonQ Forte	N	0.1622	238.543	0.32
hidden shift	random 060 cx	hidden shift	16	9	2,904	2,304	900	IonQ Forte-E	N	0.1333	198.549	0.27
hidden shift	random 060 cx	hidden shift	20	9	3,450	2,340	900	IonQ Forte	N	0.1478	240.822	0.33
hidden shift	random 060 cx	hidden shift	20	9	3,450	2,340	900	IonQ Forte-E	N	0.1111	191.217	0.26
hidden shift	random 060 cx	hidden shift	24	9	4,266	2,376	900	IonQ Forte	N	0.1556	291.768	0.39
hidden shift	random 060 cx	hidden shift	24	9	4,266	2,376	900	IonQ Forte-E	N	0.09889	209.265	0.28
hidden shift	random 060 cx	hidden shift	28	9	4,902	2,412	900	IonQ Forte	N	0.09667	271.585	0.37
hidden shift	random 060 cx	hidden shift	28	9	4,902	2,412	900	IonQ Forte-E	N	0.06	231.558	0.32
hidden shift	random 060 cx	hidden shift	32	9	5,838	2,448	900	IonQ Forte	N	0.08889	266.22	0.36
hidden shift	random 060 cx	hidden shift	32	9	5,838	2,448	900	IonQ Forte-E	N	0.06889	221.917	0.30
hidden shift	random 060 cx	hidden shift	36	9	6,504	2,484	900	IonQ Forte	N	0.06333	282.348	0.38
hidden shift	random 060 cx	hidden shift	36	9	6,504	2,484	900	IonQ Forte-E	N	0.04111	241.767	0.33

Domain	Problem	Algorithm	#q	#qc	#1q	#2q	Shots	Backend	EM?	Score	Exec. Time (s)	Energy (kWh)
hidden shift	random 070 cx	hidden shift	8	9	1,392	2,592	900	IonQ Forte	N	0.5133	125.738	0.17
hidden shift	random 070 cx	hidden shift	8	9	1,392	2,592	900	IonQ Forte-E	N	0.4267	99.999	0.14
hidden shift	random 070 cx	hidden shift	12	9	2,178	2,628	900	IonQ Forte	N	0.2167	190.175	0.26
hidden shift	random 070 cx	hidden shift	12	9	2,178	2,628	900	IonQ Forte-E	N	0.2367	161.242	0.22
hidden shift	random 070 cx	hidden shift	16	9	2,844	2,664	900	IonQ Forte	N	0.1456	254.056	0.34
hidden shift	random 070 cx	hidden shift	16	9	2,844	2,664	900	IonQ Forte-E	N	0.1067	206.415	0.28
hidden shift	random 070 cx	hidden shift	20	9	3,510	2,700	900	IonQ Forte	N	0.09222	300.686	0.41
hidden shift	random 070 cx	hidden shift	20	9	3,510	2,700	900	IonQ Forte-E	N	0.06222	224.705	0.31
hidden shift	random 070 cx	hidden shift	24	9	4,266	2,736	900	IonQ Forte	N	0.08778	315.52	0.43
hidden shift	random 070 cx	hidden shift	24	9	4,266	2,736	900	IonQ Forte-E	N	0.06111	237.098	0.32
hidden shift	random 070 cx	hidden shift	28	9	5,052	2,772	900	IonQ Forte	N	0.09333	293.676	0.40
hidden shift	random 070 cx	hidden shift	28	9	5,052	2,772	900	IonQ Forte-E	N	0.05222	249.902	0.34
hidden shift	random 070 cx	hidden shift	32	9	5,778	2,808	900	IonQ Forte	N	0.06778	293.24	0.40
hidden shift	random 070 cx	hidden shift	32	9	5,778	2,808	900	IonQ Forte-E	N	0.03889	248.453	0.34
hidden shift	random 070 cx	hidden shift	36	9	6,444	2,844	900	IonQ Forte	N	0.05222	310.439	0.42
hidden shift	random 070 cx	hidden shift	36	9	6,444	2,844	900	IonQ Forte-E	N	0.03444	270.135	0.37
hidden shift	random 080 cx	hidden shift	8	9	1,452	2,952	900	IonQ Forte	N	0.4878	137.083	0.19
hidden shift	random 080 cx	hidden shift	8	9	1,452	2,952	900	IonQ Forte-E	N	0.3222	122.108	0.17
hidden shift	random 080 cx	hidden shift	12	9	2,148	2,988	900	IonQ Forte	N	0.2767	207.849	0.28
hidden shift	random 080 cx	hidden shift	12	9	2,148	2,988	900	IonQ Forte-E	N	0.1756	176.797	0.24
hidden shift	random 080 cx	hidden shift	16	9	2,844	3,024	900	IonQ Forte	N	0.1522	287.082	0.39
hidden shift	random 080 cx	hidden shift	16	9	2,844	3,024	900	IonQ Forte-E	N	0.08556	226.542	0.31
hidden shift	random 080 cx	hidden shift	20	9	3,510	3,060	900	IonQ Forte	N	0.07778	318.062	0.43
hidden shift	random 080 cx	hidden shift	20	9	3,510	3,060	900	IonQ Forte-E	N	0.06333	249.748	0.34
hidden shift	random 080 cx	hidden shift	24	9	4,206	3,096	900	IonQ Forte	N	0.05111	360.461	0.49
hidden shift	random 080 cx	hidden shift	24	9	4,206	3,096	900	IonQ Forte-E	N	0.02778	271.628	0.37
hidden shift	random 080 cx	hidden shift	28	9	4,932	3,132	900	IonQ Forte	N	0.06333	313.493	0.42
hidden shift	random 080 cx	hidden shift	28	9	4,932	3,132	900	IonQ Forte-E	N	0.03667	270.755	0.37
hidden shift	random 080 cx	hidden shift	32	9	5,688	3,168	900	IonQ Forte	N	0.03	342.783	0.46
hidden shift	random 080 cx	hidden shift	32	9	5,688	3,168	900	IonQ Forte-E	N	0.02111	300.371	0.41
hidden shift	random 080 cx	hidden shift	36	9	6,594	3,204	900	IonQ Forte	N	0.02778	357.02	0.48
hidden shift	random 080 cx	hidden shift	36	9	6,594	3,204	900	IonQ Forte-E	N	0.01778	303.665	0.41
hidden shift	random 090 cx	hidden shift	8	9	1,452	3,312	900	IonQ Forte	N	0.5644	98.59	0.13
hidden shift	random 090 cx	hidden shift	8	9	1,452	3,312	900	IonQ Forte-E	N	0.52	84.803	0.12
hidden shift	random 090 cx	hidden shift	12	9	2,058	3,348	900	IonQ Forte	N	0.1789	247.64	0.33
hidden shift	random 090 cx	hidden shift	12	9	2,058	3,348	900	IonQ Forte-E	N	0.08778	214.433	0.29
hidden shift	random 090 cx	hidden shift	16	9	2,904	3,384	900	IonQ Forte	N	0.1156	294.052	0.40
hidden shift	random 090 cx	hidden shift	16	9	2,904	3,384	900	IonQ Forte-E	N	0.08333	238.236	0.32
hidden shift	random 090 cx	hidden shift	20	9	3,540	3,420	900	IonQ Forte	N	0.05444	341.508	0.46
hidden shift	random 090 cx	hidden shift	20	9	3,540	3,420	900	IonQ Forte-E	N	0.03556	270.991	0.37
hidden shift	random 090 cx	hidden shift	24	9	4,236	3,456	900	IonQ Forte	N	0.04556	391.071	0.53
hidden shift	random 090 cx	hidden shift	24	9	4,236	3,456	900	IonQ Forte-E	N	0.02556	298.757	0.41
hidden shift	random 090 cx	hidden shift	28	9	4,902	3,492	900	IonQ Forte	N	0.04	358.3	0.48
hidden shift	random 090 cx	hidden shift	28	9	4,902	3,492	900	IonQ Forte-E	N	0.01778	310.097	0.42
hidden shift	random 090 cx	hidden shift	32	9	5,898	3,528	900	IonQ Forte	N	0.02222	371.237	0.50
hidden shift	random 090 cx	hidden shift	32	9	5,898	3,528	900	IonQ Forte-E	N	0.02333	312.573	0.43
hidden shift	random 090 cx	hidden shift	36	9	6,144	3,564	900	IonQ Forte	N	0.02222	365.548	0.49
hidden shift	random 090 cx	hidden shift	36	9	6,144	3,564	900	IonQ Forte-E	N	0.01667	308.711	0.42
hidden shift	random 100 cx	hidden shift	8	9	1,452	3,672	900	IonQ Forte	N	0.4967	127.539	0.17
hidden shift	random 100 cx	hidden shift	8	9	1,452	3,672	900	IonQ Forte-E	N	0.4411	105.144	0.14
hidden shift	random 100 cx	hidden shift	12	9	2,088	3,708	900	IonQ Forte	N	0.1478	267.451	0.36
hidden shift	random 100 cx	hidden shift	12	9	2,088	3,708	900	IonQ Forte-E	N	0.06889	234.253	0.32
hidden shift	random 100 cx	hidden shift	16	9	2,754	3,744	900	IonQ Forte	N	0.08667	332.502	0.45
hidden shift	random 100 cx	hidden shift	16	9	2,754	3,744	900	IonQ Forte-E	N	0.04111	282.994	0.39
hidden shift	random 100 cx	hidden shift	20	9	3,660	3,780	900	IonQ Forte	N	0.03778	381.747	0.52
hidden shift	random 100 cx	hidden shift	20	9	3,660	3,780	900	IonQ Forte-E	N	0.03556	286.84	0.39
hidden shift	random 100 cx	hidden shift	24	9	4,206	3,816	900	IonQ Forte	N	0.03222	361.195	0.49
hidden shift	random 100 cx	hidden shift	24	9	4,206	3,816	900	IonQ Forte-E	N	0.01222	309.506	0.42
hidden shift	random 100 cx	hidden shift	28	9	5,022	3,852	900	IonQ Forte	N	0.01889	387.744	0.52
hidden shift	random 100 cx	hidden shift	28	9	5,022	3,852	900	IonQ Forte-E	N	0.01667	332.271	0.45
hidden shift	random 100 cx	hidden shift	32	9	5,568	3,888	900	IonQ Forte	N	0.02444	398.406	0.54
hidden shift	random 100 cx	hidden shift	32	9	5,568	3,888	900	IonQ Forte-E	N	0.01222	340.251	0.46
hidden shift	random 100 cx	hidden shift	36	9	6,294	3,924	900	IonQ Forte	N	0.01556	417.336	0.56
hidden shift	random 100 cx	hidden shift	36	9	6,294	3,924	900	IonQ Forte-E	N	0.005556	355.231	0.48
machine learning	img loading	mps load	10	1	121	22	10,000	IonQ Forte	N	0.441	472.136	0.64
machine learning	img loading	mps load	10	1	121	22	10,000	IonQ Forte-E	N	0.466	463.735	0.63
machine learning	img loading	mps load	10	1	227	45	10,000	IonQ Forte	N	0.363	754.47	1.02
machine learning	img loading	mps load	10	1	227	45	10,000	IonQ Forte-E	N	0.462	812.766	1.11
machine learning	img loading	mps load	10	1	328	65	10,000	IonQ Forte	N	0.292	1,031.54	1.40
machine learning	img loading	mps load	10	1	328	65	10,000	IonQ Forte-E	N	0.471	1,075.05	1.46
machine learning	img loading	mps load	10	1	448	91	10,000	IonQ Forte	N	0.328	1,389.04	1.88
machine learning	img loading	mps load	10	1	448	91	10,000	IonQ Forte-E	N	0.649	1,455.72	1.98
machine learning	img loading	mps load	10	1	451	22	10,000	IonQ Aria	N	0.549	874.127	–
machine learning	img loading	mps load	10	1	544	113	10,000	IonQ Forte	N	0.39	1,627.65	2.20
machine learning	img loading	mps load	10	1	544	113	10,000	IonQ Forte-E	N	0.804	1,725.99	2.35
machine learning	img loading	mps load	10	1	653	136	10,000	IonQ Forte	N	0.473	1,941.48	2.63
machine learning	img loading	mps load	10	1	653	136	10,000	IonQ Forte-E	N	0.894	2,058.38	2.80
machine learning	img loading	mps load	10	1	861	45	10,000	IonQ Aria	N	0.475	1,603.14	–
machine learning	img loading	mps load	10	1	1,244	65	10,000	IonQ Aria	N	0.346	2,278.32	–
machine learning	img loading	mps load	10	1	1,708	91	10,000	IonQ Aria	N	0.452	3,115.04	–

Domain	Problem	Algorithm	#q	#qc	#1q	#2q	Shots	Backend	EM?	Score	Exec. Time (s)	Energy (kWh)
machine learning	img loading	mps load	10	1	2,084	113	10,000	IonQ Aria	N	0.497	3,774.4	–
machine learning	img loading	mps load	10	1	2,503	136	10,000	IonQ Aria	N	0.69	4,505.64	–
machine learning	img loading	mps load	10	25	1,718	550	10,000	IonQ Aria	Y	0.458	376.091	–
machine learning	img loading	mps load	10	25	2,275	550	10,000	IonQ Forte	Y	0.453	454.812	0.62
machine learning	img loading	mps load	10	25	2,821	1,125	10,000	IonQ Aria	Y	0.344	592.506	–
machine learning	img loading	mps load	10	25	3,935	1,625	10,000	IonQ Aria	Y	0.252	786.974	–
machine learning	img loading	mps load	10	25	4,225	1,125	10,000	IonQ Forte	Y	0.338	761.946	1.03
machine learning	img loading	mps load	10	25	5,157	2,275	10,000	IonQ Aria	Y	0.324	1,021.82	–
machine learning	img loading	mps load	10	25	5,575	1,625	10,000	IonQ Forte	Y	0.327	1,027.56	1.39
machine learning	img loading	mps load	10	25	6,413	2,800	10,000	IonQ Aria	Y	0.397	1,238.97	–
machine learning	img loading	mps load	10	25	7,444	3,350	10,000	IonQ Aria	Y	0.473	1,439.26	–
machine learning	img loading	mps load	10	25	7,600	2,275	10,000	IonQ Forte	Y	0.413	1,362.08	1.84
machine learning	img loading	mps load	10	25	8,700	2,800	10,000	IonQ Forte	Y	0.466	1,598.93	2.16
machine learning	img loading	mps load	10	25	10,675	3,350	10,000	IonQ Forte	Y	0.622	1,909.08	2.58
machine learning	img loading	mps load	12	1	401	81	10,000	IonQ Forte	N	0.728	1,252.22	1.69
machine learning	img loading	mps load	12	1	532	108	10,000	IonQ Forte	N	0.769	1,609.41	2.18
machine learning	img loading	mps load	12	1	649	135	10,000	IonQ Forte	N	0.788	1,949.7	2.64
machine learning	img loading	mps load	12	1	738	154	10,000	IonQ Forte	N	0.88	2,206.12	2.98
machine learning	img loading	mps load	12	1	904	188	10,000	IonQ Forte	N	0.918	2,626.9	3.55
machine learning	img loading	mps load	12	1	1,162	245	10,000	IonQ Forte	N	1.045	3,409.5	4.61
machine learning	img loading	mps load	12	1	1,425	300	10,000	IonQ Forte	N	1.18	4,184.66	5.66
machine learning	img loading	mps load	12	1	1,527	81	10,000	IonQ Aria	N	0.924	2,777	–
machine learning	img loading	mps load	12	1	1,691	355	10,000	IonQ Forte	N	1.232	4,975.03	6.73
machine learning	img loading	mps load	12	1	2,028	108	10,000	IonQ Aria	N	1.005	3,673.98	–
machine learning	img loading	mps load	12	1	2,487	135	10,000	IonQ Aria	N	1.055	4,488.71	–
machine learning	img loading	mps load	12	1	2,830	154	10,000	IonQ Aria	N	1.069	5,094.95	–
machine learning	img loading	mps load	12	1	3,464	188	10,000	IonQ Aria	N	1.163	6,184.9	–
machine learning	img loading	mps load	12	1	4,466	245	10,000	IonQ Aria	N	1.192	7,973.22	–
machine learning	img loading	mps load	12	25	4,724	2,000	10,000	IonQ Aria	Y	0.767	955.625	–
machine learning	img loading	mps load	12	25	6,250	2,700	10,000	IonQ Aria	Y	0.779	1,236.6	–
machine learning	img loading	mps load	12	25	7,651	3,350	10,000	IonQ Aria	Y	0.81	1,494.94	–
machine learning	img loading	mps load	12	25	9,062	3,850	10,000	IonQ Aria	Y	0.835	1,705.45	–
machine learning	img loading	mps load	12	25	10,407	4,675	10,000	IonQ Aria	Y	0.926	2,013.21	–
machine learning	img loading	mps load	12	25	13,489	6,125	10,000	IonQ Aria	Y	1.044	2,583.87	–
machine learning	img loading	mps load	12	25	16,128	7,475	10,000	IonQ Aria	Y	1.177	3,105.37	–
machine learning	img loading	mps load	12	25	19,088	8,850	10,000	IonQ Aria	Y	1.213	3,643.78	–
optimization	maxcut	qaoa	8	1	16	12	5,000	IonQ Aria	N	0.7878	155.439	–
optimization	maxcut	qaoa	8	1	16	12	5,000	IonQ Forte	N	0.7876	154.586	0.21
optimization	maxcut	qaoa	8	1	24	24	5,000	IonQ Aria	N	0.8609	206.149	–
optimization	maxcut	qaoa	8	1	24	24	5,000	IonQ Forte	N	0.8555	237.574	0.32
optimization	maxcut	qaoa	8	1	32	36	5,000	IonQ Aria	N	0.9024	261.216	–
optimization	maxcut	qaoa	8	1	32	36	5,000	IonQ Forte	N	0.8973	327.976	0.44
optimization	maxcut	qaoa	8	1	40	48	5,000	IonQ Aria	N	0.916	312.941	–
optimization	maxcut	qaoa	8	1	40	48	5,000	IonQ Forte	N	0.9125	412.534	0.56
optimization	maxcut	qaoa	8	1	48	60	5,000	IonQ Aria	N	0.9311	361.02	–
optimization	maxcut	qaoa	8	1	48	60	5,000	IonQ Forte	N	0.9218	497.328	0.67
optimization	maxcut	qaoa	8	1	56	72	5,000	IonQ Aria	N	0.934	413.33	–
optimization	maxcut	qaoa	8	1	56	72	5,000	IonQ Forte	N	0.9254	584.621	0.79
optimization	maxcut	qaoa	8	1	64	84	5,000	IonQ Aria	N	0.9285	469.584	–
optimization	maxcut	qaoa	8	1	64	84	5,000	IonQ Forte	N	0.9206	667.498	0.90
optimization	maxcut	qaoa	8	1	72	96	5,000	IonQ Aria	N	0.9259	520.808	–
optimization	maxcut	qaoa	8	1	72	96	5,000	IonQ Forte	N	0.9136	756.022	1.02
optimization	maxcut	qaoa	8	1	80	108	5,000	IonQ Aria	N	0.9222	576.632	–
optimization	maxcut	qaoa	8	1	80	108	5,000	IonQ Forte	N	0.908	842.626	1.14
optimization	maxcut	qaoa	8	1	88	120	5,000	IonQ Aria	N	0.922	632.544	–
optimization	maxcut	qaoa	8	1	88	120	5,000	IonQ Forte	N	0.8985	925.142	1.25
optimization	maxcut	qaoa	8	1	96	132	5,000	IonQ Aria	N	0.9126	680.373	–
optimization	maxcut	qaoa	8	1	96	132	5,000	IonQ Forte	N	0.8932	1,011.02	1.37
optimization	maxcut	qaoa	8	25	651	300	5,000	IonQ Aria	Y	0.8335	154.62	–
optimization	maxcut	qaoa	8	25	700	300	5,000	IonQ Forte	Y	0.8765	174.583	0.24
optimization	maxcut	qaoa	8	25	962	600	5,000	IonQ Aria	Y	0.8973	211.339	–
optimization	maxcut	qaoa	8	25	1,100	600	5,000	IonQ Forte	Y	0.9401	233.999	0.32
optimization	maxcut	qaoa	8	25	1,271	900	5,000	IonQ Aria	Y	0.9396	265.312	–
optimization	maxcut	qaoa	8	25	1,500	900	5,000	IonQ Forte	Y	0.9607	319.828	0.43
optimization	maxcut	qaoa	8	25	1,576	1,200	5,000	IonQ Aria	Y	0.9595	309.307	–
optimization	maxcut	qaoa	8	25	1,873	1,500	5,000	IonQ Aria	Y	0.9716	368.575	–
optimization	maxcut	qaoa	8	25	1,900	1,200	5,000	IonQ Forte	Y	0.9759	406.705	0.55
optimization	maxcut	qaoa	8	25	2,186	1,800	5,000	IonQ Aria	Y	0.9708	417.71	–
optimization	maxcut	qaoa	8	25	2,300	1,500	5,000	IonQ Forte	Y	0.984	492.308	0.67
optimization	maxcut	qaoa	8	25	2,461	2,100	5,000	IonQ Aria	Y	0.9665	470.509	–
optimization	maxcut	qaoa	8	25	2,700	1,800	5,000	IonQ Forte	Y	0.9861	576.998	0.78
optimization	maxcut	qaoa	8	25	2,770	2,400	5,000	IonQ Aria	Y	0.9619	519.381	–
optimization	maxcut	qaoa	8	25	3,044	2,700	5,000	IonQ Aria	Y	0.9587	572.582	–
optimization	maxcut	qaoa	8	25	3,100	2,100	5,000	IonQ Forte	Y	0.9859	668.184	0.90
optimization	maxcut	qaoa	8	25	3,352	3,000	5,000	IonQ Aria	Y	0.9551	624.848	–
optimization	maxcut	qaoa	8	25	3,500	2,400	5,000	IonQ Forte	Y	0.9845	758.111	1.03
optimization	maxcut	qaoa	8	25	3,625	3,300	5,000	IonQ Aria	Y	0.9498	674.818	–
optimization	maxcut	qaoa	8	25	3,900	2,700	5,000	IonQ Forte	Y	0.981	848.14	1.15
optimization	maxcut	qaoa	8	25	4,300	3,000	5,000	IonQ Forte	Y	0.973	928.292	1.26
optimization	maxcut	qaoa	8	25	4,700	3,300	5,000	IonQ Forte	Y	0.9668	1,013.59	1.37

Domain	Problem	Algorithm	#q	#qc	#1q	#2q	Shots	Backend	EM?	Score	Exec. Time (s)	Energy (kWh)
optimization	maxcut	qaoa	12	1	24	66	5,000	IonQ Aria	N	0.563	399.837	–
optimization	maxcut	qaoa	12	1	24	66	5,000	IonQ Forte	N	0.5599	616.513	0.83
optimization	maxcut	qaoa	12	1	24	66	5,000	IonQ Forte-E	N	0.6213	519.795	0.71
optimization	maxcut	qaoa	12	1	48	198	5,000	IonQ Aria	N	0.7748	991.815	–
optimization	maxcut	qaoa	12	1	48	198	5,000	IonQ Forte	N	0.7598	1,291.07	1.75
optimization	maxcut	qaoa	12	1	48	198	5,000	IonQ Forte-E	N	0.7855	1,426.39	1.94
optimization	maxcut	qaoa	12	1	72	330	5,000	IonQ Aria	N	0.821	1,570.59	–
optimization	maxcut	qaoa	12	1	72	330	5,000	IonQ Forte	N	0.8081	2,097.38	2.84
optimization	maxcut	qaoa	12	1	72	330	5,000	IonQ Forte-E	N	0.8001	2,203.71	3.00
optimization	maxcut	qaoa	12	1	96	462	5,000	IonQ Aria	N	0.8539	2,122.49	–
optimization	maxcut	qaoa	12	1	96	462	5,000	IonQ Forte	N	0.8522	2,872.06	3.89
optimization	maxcut	qaoa	12	1	96	462	5,000	IonQ Forte-E	N	0.8234	3,000.56	4.08
optimization	maxcut	qaoa	12	1	120	594	5,000	IonQ Aria	N	0.8644	2,730.34	–
optimization	maxcut	qaoa	12	1	120	594	5,000	IonQ Forte	N	0.8636	3,675.14	4.97
optimization	maxcut	qaoa	12	1	120	594	5,000	IonQ Forte-E	N	0.8204	3,874	5.27
optimization	maxcut	qaoa	12	1	144	726	5,000	IonQ Aria	N	0.8656	3,268.22	–
optimization	maxcut	qaoa	12	1	144	726	5,000	IonQ Forte	N	0.8675	5,154.44	6.97
optimization	maxcut	qaoa	12	1	144	726	5,000	IonQ Forte-E	N	0.8148	4,568.52	6.22
optimization	maxcut	qaoa	12	1	168	858	5,000	IonQ Aria	N	0.862	3,832.57	–
optimization	maxcut	qaoa	12	1	168	858	5,000	IonQ Forte	N	0.8662	6,025.95	8.15
optimization	maxcut	qaoa	12	1	168	858	5,000	IonQ Forte-E	N	0.8098	5,410.55	7.36
optimization	maxcut	qaoa	12	1	192	990	5,000	IonQ Aria	N	0.8561	4,432.02	–
optimization	maxcut	qaoa	12	1	192	990	5,000	IonQ Forte	N	0.8553	6,889.57	9.32
optimization	maxcut	qaoa	12	1	192	990	5,000	IonQ Forte-E	N	0.8041	6,294.98	8.57
optimization	maxcut	qaoa	12	1	216	1,122	5,000	IonQ Aria	N	0.8488	5,115.55	–
optimization	maxcut	qaoa	12	1	216	1,122	5,000	IonQ Forte	N	0.8535	7,865.47	10.64
optimization	maxcut	qaoa	12	1	216	1,122	5,000	IonQ Forte-E	N	0.7959	7,104.39	9.67
optimization	maxcut	qaoa	12	1	240	1,254	5,000	IonQ Aria	N	0.8407	10,756.6	–
optimization	maxcut	qaoa	12	1	240	1,254	5,000	IonQ Forte	N	0.8434	8,731.09	11.81
optimization	maxcut	qaoa	12	1	240	1,254	5,000	IonQ Forte-E	N	0.7938	7,894.1	10.74
optimization	maxcut	qaoa	12	1	264	1,386	5,000	IonQ Aria	N	0.8354	10,749.8	–
optimization	maxcut	qaoa	12	1	264	1,386	5,000	IonQ Forte	N	0.8351	8,430.36	11.40
optimization	maxcut	qaoa	12	1	264	1,386	5,000	IonQ Forte-E	N	0.7905	8,818.04	12.00
optimization	maxcut	qaoa	12	25	1,050	1,650	5,000	IonQ Forte	Y	0.5815	504.814	0.68
optimization	maxcut	qaoa	12	25	1,050	1,650	5,000	IonQ Forte-E	Y	0.5515	533.236	0.73
optimization	maxcut	qaoa	12	25	2,250	4,825	5,000	IonQ Forte	Y	0.7659	1,348.85	1.82
optimization	maxcut	qaoa	12	25	2,250	4,825	5,000	IonQ Forte-E	Y	0.7763	1,380.14	1.88
optimization	maxcut	qaoa	12	25	3,450	7,975	5,000	IonQ Forte	Y	0.8724	2,170.24	2.94
optimization	maxcut	qaoa	12	25	3,450	7,975	5,000	IonQ Forte-E	Y	0.7727	2,178.11	2.96
optimization	maxcut	qaoa	12	25	4,700	11,025	5,000	IonQ Forte	Y	0.9339	2,953.39	4.00
optimization	maxcut	qaoa	12	25	4,700	11,025	5,000	IonQ Forte-E	Y	0.8395	3,008.94	4.10
optimization	maxcut	qaoa	12	25	5,900	14,175	5,000	IonQ Forte	Y	0.9474	3,795.21	5.13
optimization	maxcut	qaoa	12	25	5,900	14,175	5,000	IonQ Forte-E	Y	0.8409	3,805.54	5.18
optimization	maxcut	qaoa	12	25	7,050	17,250	5,000	IonQ Forte	Y	0.9283	4,568.92	6.18
optimization	maxcut	qaoa	12	25	7,050	17,250	5,000	IonQ Forte-E	Y	0.8332	4,671.13	6.36
optimization	maxcut	qaoa	12	25	8,300	20,400	5,000	IonQ Forte	Y	0.9191	5,377.21	7.27
optimization	maxcut	qaoa	12	25	8,300	20,400	5,000	IonQ Forte-E	Y	0.8324	5,468.84	7.44
optimization	maxcut	qaoa	12	25	9,500	23,475	5,000	IonQ Forte	Y	0.9005	6,143.26	8.31
optimization	maxcut	qaoa	12	25	9,500	23,475	5,000	IonQ Forte-E	Y	0.8273	6,241.85	8.50
optimization	maxcut	qaoa	12	25	10,700	26,600	5,000	IonQ Forte	Y	0.8901	7,006.59	9.48
optimization	maxcut	qaoa	12	25	10,700	26,600	5,000	IonQ Forte-E	Y	0.8141	7,034.29	9.57
optimization	maxcut	qaoa	12	25	11,950	29,725	5,000	IonQ Forte	Y	0.8736	7,859.33	10.63
optimization	maxcut	qaoa	12	25	11,950	29,725	5,000	IonQ Forte-E	Y	0.8098	7,923.41	10.78
optimization	maxcut	qaoa	12	25	13,100	32,725	5,000	IonQ Forte	Y	0.8657	8,726.49	11.80
optimization	maxcut	qaoa	12	25	13,100	32,725	5,000	IonQ Forte-E	Y	0.8082	8,750.79	11.91
optimization	maxcut	qaoa	16	1	32	120	5,000	IonQ Forte	N	0.5858	808.992	1.09
optimization	maxcut	qaoa	16	1	32	120	5,000	IonQ Forte-E	N	0.6688	870.639	1.18
optimization	maxcut	qaoa	16	1	64	360	5,000	IonQ Forte	N	0.6962	2,238.44	3.03
optimization	maxcut	qaoa	16	1	64	360	5,000	IonQ Forte-E	N	0.7728	2,323.1	3.16
optimization	maxcut	qaoa	16	1	96	600	5,000	IonQ Forte	N	0.7789	3,659.82	4.95
optimization	maxcut	qaoa	16	1	96	600	5,000	IonQ Forte-E	N	0.7754	3,784.27	5.15
optimization	maxcut	qaoa	16	1	128	840	5,000	IonQ Forte	N	0.8253	5,069.16	6.86
optimization	maxcut	qaoa	16	1	128	840	5,000	IonQ Forte-E	N	0.8071	5,200.62	7.08
optimization	maxcut	qaoa	16	1	160	1,080	5,000	IonQ Forte	N	0.8428	6,418.03	8.68
optimization	maxcut	qaoa	16	1	160	1,080	5,000	IonQ Forte-E	N	0.8023	6,749.18	9.19
optimization	maxcut	qaoa	16	1	192	1,320	5,000	IonQ Forte	N	0.8451	7,817.81	10.58
optimization	maxcut	qaoa	16	1	192	1,320	5,000	IonQ Forte-E	N	0.7982	8,218.79	11.19
optimization	maxcut	qaoa	16	1	224	1,560	5,000	IonQ Forte	N	0.8154	9,303.38	12.59
optimization	maxcut	qaoa	16	1	224	1,560	5,000	IonQ Forte-E	N	0.7913	9,713.41	13.22
optimization	maxcut	qaoa	16	1	256	1,800	5,000	IonQ Forte	N	0.8047	10,639.7	14.39
optimization	maxcut	qaoa	16	1	256	1,800	5,000	IonQ Forte-E	N	0.7886	11,185.8	15.22
optimization	maxcut	qaoa	16	1	288	2,040	5,000	IonQ Forte	N	0.8219	11,985.9	16.21
optimization	maxcut	qaoa	16	1	288	2,040	5,000	IonQ Forte-E	N	0.7852	12,561.2	17.10
optimization	maxcut	qaoa	16	1	320	2,280	5,000	IonQ Forte	N	0.8129	13,443.3	18.19
optimization	maxcut	qaoa	16	1	320	2,280	5,000	IonQ Forte-E	N	0.7841	14,060.1	19.14
optimization	maxcut	qaoa	16	1	352	2,520	5,000	IonQ Forte	N	0.8036	14,755.1	19.96
optimization	maxcut	qaoa	16	1	352	2,520	5,000	IonQ Forte-E	N	0.781	15,242.8	20.75
optimization	maxcut	qaoa	16	25	1,400	2,900	5,000	IonQ Forte	Y	0.5951	849.99	1.15
optimization	maxcut	qaoa	16	25	1,400	2,900	5,000	IonQ Forte-E	Y	0.6624	863.053	1.17
optimization	maxcut	qaoa	16	25	3,000	8,550	5,000	IonQ Forte	Y	0.7022	2,342.65	3.17

Domain	Problem	Algorithm	#q	#qc	#1q	#2q	Shots	Backend	EM?	Score	Exec. Time (s)	Energy (kWh)
optimization	maxcut	qaoa	16	25	3,000	8,550	5,000	IonQ Forte-E	Y	0.7384	2,323.72	3.16
optimization	maxcut	qaoa	16	25	4,600	14,050	5,000	IonQ Forte	Y	0.7861	3,780.31	5.11
optimization	maxcut	qaoa	16	25	4,600	14,050	5,000	IonQ Forte-E	Y	0.7679	3,759.05	5.12
optimization	maxcut	qaoa	16	25	6,250	19,525	5,000	IonQ Forte	Y	0.8575	5,228.57	7.07
optimization	maxcut	qaoa	16	25	6,250	19,525	5,000	IonQ Forte-E	Y	0.8079	5,206.85	7.09
optimization	maxcut	qaoa	16	25	7,800	25,050	5,000	IonQ Forte	Y	0.8778	6,713.93	9.08
optimization	maxcut	qaoa	16	25	7,800	25,050	5,000	IonQ Forte-E	Y	0.8092	6,668.29	9.08
optimization	maxcut	qaoa	16	25	9,450	30,550	5,000	IonQ Forte	Y	0.8572	7,964.77	10.77
optimization	maxcut	qaoa	16	25	9,450	30,550	5,000	IonQ Forte-E	Y	0.8051	8,094.51	11.02
optimization	maxcut	qaoa	16	25	11,100	36,050	5,000	IonQ Forte	Y	0.8549	9,441.27	12.77
optimization	maxcut	qaoa	16	25	11,100	36,050	5,000	IonQ Forte-E	Y	0.8003	9,561.96	13.01
optimization	maxcut	qaoa	16	25	12,700	41,575	5,000	IonQ Forte	Y	0.8496	10,912.3	14.76
optimization	maxcut	qaoa	16	25	12,700	41,575	5,000	IonQ Forte-E	Y	0.798	10,943.8	14.90
optimization	maxcut	qaoa	16	25	14,250	46,975	5,000	IonQ Forte	Y	0.8292	12,388.3	16.76
optimization	maxcut	qaoa	16	25	14,250	46,975	5,000	IonQ Forte-E	Y	0.7963	12,207.5	16.62
optimization	maxcut	qaoa	16	25	16,050	52,500	5,000	IonQ Forte	Y	0.8134	13,813.9	18.69
optimization	maxcut	qaoa	16	25	16,050	52,500	5,000	IonQ Forte-E	Y	0.7927	13,769.2	18.74
optimization	maxcut	qaoa	16	25	17,550	57,950	5,000	IonQ Forte	Y	0.8022	15,200.4	20.56
optimization	maxcut	qaoa	16	25	17,550	57,950	5,000	IonQ Forte-E	Y	0.7932	15,512.4	21.11
optimization	maxcut	qaoa	24	1	48	36	5,000	IonQ Aria	N	0.77	274.95	–
optimization	maxcut	qaoa	24	1	48	36	5,000	IonQ Forte	N	0.3686	389.496	0.53
optimization	maxcut	qaoa	24	1	48	36	5,000	IonQ Forte-E	N	0.7748	367.226	0.50
optimization	maxcut	qaoa	24	1	48	48	5,000	IonQ Forte	N	0.4446	442.653	0.60
optimization	maxcut	qaoa	24	1	72	72	5,000	IonQ Aria	N	0.8236	448.056	–
optimization	maxcut	qaoa	24	1	72	72	5,000	IonQ Forte-E	N	0.8228	645.153	0.88
optimization	maxcut	qaoa	24	1	96	108	5,000	IonQ Aria	N	0.844	618.14	–
optimization	maxcut	qaoa	24	1	96	108	5,000	IonQ Forte-E	N	0.8442	908.805	1.24
optimization	maxcut	qaoa	24	1	120	144	5,000	IonQ Aria	N	0.8473	800.795	–
optimization	maxcut	qaoa	24	1	120	144	5,000	IonQ Forte-E	N	0.8546	1,183.87	1.61
optimization	maxcut	qaoa	24	1	144	180	5,000	IonQ Aria	N	0.8497	973.081	–
optimization	maxcut	qaoa	24	1	144	180	5,000	IonQ Forte	N	0.8048	1,406.06	1.90
optimization	maxcut	qaoa	24	1	144	180	5,000	IonQ Forte-E	N	0.8519	1,510.27	2.06
optimization	maxcut	qaoa	24	1	144	240	5,000	IonQ Forte	N	0.8311	1,712.32	2.32
optimization	maxcut	qaoa	24	1	168	216	5,000	IonQ Aria	N	0.8499	1,148.29	–
optimization	maxcut	qaoa	24	1	168	216	5,000	IonQ Forte-E	N	0.8469	1,708.41	2.33
optimization	maxcut	qaoa	24	1	192	252	5,000	IonQ Aria	N	0.8416	1,322.12	–
optimization	maxcut	qaoa	24	1	192	252	5,000	IonQ Forte-E	N	0.8401	1,977.51	2.69
optimization	maxcut	qaoa	24	1	216	288	5,000	IonQ Aria	N	0.8332	1,499.42	–
optimization	maxcut	qaoa	24	1	216	288	5,000	IonQ Forte-E	N	0.8276	2,256.67	3.07
optimization	maxcut	qaoa	24	1	240	324	5,000	IonQ Aria	N	0.8209	1,667.45	–
optimization	maxcut	qaoa	24	1	240	324	5,000	IonQ Forte	N	0.8302	2,473.5	3.35
optimization	maxcut	qaoa	24	1	240	324	5,000	IonQ Forte-E	N	0.8138	2,518.71	3.43
optimization	maxcut	qaoa	24	1	240	432	5,000	IonQ Forte	N	0.8668	3,016.76	4.08
optimization	maxcut	qaoa	24	1	264	360	5,000	IonQ Aria	N	0.8145	1,833.79	–
optimization	maxcut	qaoa	24	1	264	360	5,000	IonQ Forte-E	N	0.8153	2,769.21	3.77
optimization	maxcut	qaoa	24	1	288	396	5,000	IonQ Aria	N	0.8018	2,003.76	–
optimization	maxcut	qaoa	24	1	288	396	5,000	IonQ Forte-E	N	0.7963	3,006.89	4.09
optimization	maxcut	qaoa	24	1	336	468	5,000	IonQ Forte	N	0.839	3,557.17	4.81
optimization	maxcut	qaoa	24	1	336	624	5,000	IonQ Forte	N	0.8557	4,423.17	5.98
optimization	maxcut	qaoa	24	1	432	612	5,000	IonQ Forte	N	0.8294	4,615.3	6.24
optimization	maxcut	qaoa	24	1	432	816	5,000	IonQ Forte	N	0.8349	5,738.09	7.76
optimization	maxcut	qaoa	24	1	528	756	5,000	IonQ Forte	N	0.8091	5,657.35	7.65
optimization	maxcut	qaoa	24	1	528	1,008	5,000	IonQ Forte	N	0.8095	7,095.1	9.60
optimization	maxcut	qaoa	24	1	624	900	5,000	IonQ Forte	N	0.7836	6,617.38	8.95
optimization	maxcut	qaoa	24	1	624	1,200	5,000	IonQ Forte	N	0.7879	8,421.12	11.39
optimization	maxcut	qaoa	24	1	720	1,044	5,000	IonQ Forte	N	0.7698	7,594.6	10.27
optimization	maxcut	qaoa	24	1	720	1,392	5,000	IonQ Forte	N	0.7655	9,786.27	13.24
optimization	maxcut	qaoa	24	1	816	1,188	5,000	IonQ Forte	N	0.7616	8,527.44	11.54
optimization	maxcut	qaoa	24	1	816	1,584	5,000	IonQ Forte	N	0.7444	11,156.2	15.09
optimization	maxcut	qaoa	24	1	912	1,332	5,000	IonQ Forte	N	0.7416	9,683.28	13.10
optimization	maxcut	qaoa	24	1	912	1,776	5,000	IonQ Forte	N	0.7244	12,504.7	16.92
optimization	maxcut	qaoa	24	1	1,008	1,476	5,000	IonQ Forte	N	0.716	10,848.9	14.68
optimization	maxcut	qaoa	24	1	1,008	1,968	5,000	IonQ Forte	N	0.7108	13,749.5	18.60
optimization	maxcut	qaoa	24	1	1,104	1,620	5,000	IonQ Forte	N	0.6991	11,896.2	16.09
optimization	maxcut	qaoa	24	1	1,104	2,160	5,000	IonQ Forte	N	0.6938	14,871	20.12
optimization	maxcut	qaoa	24	1	1,200	1,764	5,000	IonQ Forte	N	0.6703	12,954.7	17.52
optimization	maxcut	qaoa	24	1	1,200	2,352	5,000	IonQ Forte	N	0.6778	16,223.7	21.95
optimization	maxcut	qaoa	24	1	1,296	1,908	5,000	IonQ Forte	N	0.6524	14,060.1	19.02
optimization	maxcut	qaoa	24	1	1,296	2,544	5,000	IonQ Forte	N	0.6642	17,484.6	23.65
optimization	maxcut	qaoa	24	25	1,791	900	5,000	IonQ Aria	Y	0.7675	291.042	–
optimization	maxcut	qaoa	24	25	2,100	900	5,000	IonQ Forte	Y	0.3733	374.355	0.51
optimization	maxcut	qaoa	24	25	2,100	900	5,000	IonQ Forte	Y	0.7925	366.565	0.50
optimization	maxcut	qaoa	24	25	2,100	1,200	5,000	IonQ Forte	Y	0.4475	460.501	0.62
optimization	maxcut	qaoa	24	25	2,751	1,800	5,000	IonQ Aria	Y	0.8329	463.271	–
optimization	maxcut	qaoa	24	25	3,300	1,800	5,000	IonQ Forte	Y	0.9098	634.61	0.86
optimization	maxcut	qaoa	24	25	3,686	2,700	5,000	IonQ Aria	Y	0.8691	637.218	–
optimization	maxcut	qaoa	24	25	4,500	2,700	5,000	IonQ Forte	Y	0.982	864.027	1.17
optimization	maxcut	qaoa	24	25	4,537	3,600	5,000	IonQ Aria	Y	0.8869	805.48	–
optimization	maxcut	qaoa	24	25	5,510	4,500	5,000	IonQ Aria	Y	0.9042	983.248	–
optimization	maxcut	qaoa	24	25	5,700	3,600	5,000	IonQ Forte	Y	0.9861	1,096.84	1.48

Domain	Problem	Algorithm	#q	#qc	#1q	#2q	Shots	Backend	EM?	Score	Exec. Time (s)	Energy (kWh)
optimization	maxcut	qaoa	24	25	6,348	5,400	5,000	IonQ Aria	Y	0.9087	1,155.15	–
optimization	maxcut	qaoa	24	25	6,900	4,500	5,000	IonQ Forte	Y	0.8321	1,365.86	1.85
optimization	maxcut	qaoa	24	25	6,900	4,500	5,000	IonQ Forte	Y	0.9935	1,349.54	1.83
optimization	maxcut	qaoa	24	25	6,900	6,000	5,000	IonQ Forte	Y	0.8467	1,769.74	2.39
optimization	maxcut	qaoa	24	25	7,381	6,300	5,000	IonQ Aria	Y	0.9075	1,335.37	–
optimization	maxcut	qaoa	24	25	8,100	5,400	5,000	IonQ Forte	Y	0.9947	1,600.98	2.17
optimization	maxcut	qaoa	24	25	8,235	7,200	5,000	IonQ Aria	Y	0.8902	1,505.79	–
optimization	maxcut	qaoa	24	25	9,245	8,100	5,000	IonQ Aria	Y	0.8867	1,680.09	–
optimization	maxcut	qaoa	24	25	9,300	6,300	5,000	IonQ Forte	Y	0.9964	1,850.04	2.50
optimization	maxcut	qaoa	24	25	10,194	9,000	5,000	IonQ Aria	Y	0.8781	1,850.65	–
optimization	maxcut	qaoa	24	25	10,500	7,200	5,000	IonQ Forte	Y	0.9954	2,104.61	2.85
optimization	maxcut	qaoa	24	25	11,132	9,900	5,000	IonQ Aria	Y	0.8606	2,027.26	–
optimization	maxcut	qaoa	24	25	11,700	8,100	5,000	IonQ Forte	Y	0.9533	2,466.74	3.34
optimization	maxcut	qaoa	24	25	11,700	8,100	5,000	IonQ Forte	Y	0.996	2,347.11	3.18
optimization	maxcut	qaoa	24	25	11,700	10,800	5,000	IonQ Forte	Y	0.9524	3,051.6	4.13
optimization	maxcut	qaoa	24	25	12,900	9,000	5,000	IonQ Forte	Y	0.9972	2,688.66	3.64
optimization	maxcut	qaoa	24	25	14,100	9,900	5,000	IonQ Forte	Y	0.9977	2,958.61	4.00
optimization	maxcut	qaoa	24	25	16,500	11,700	5,000	IonQ Forte	Y	0.7908	3,463.04	4.68
optimization	maxcut	qaoa	24	25	16,500	11,700	5,000	IonQ Forte	Y	0.9295	3,444.01	4.66
optimization	maxcut	qaoa	24	25	16,500	15,600	5,000	IonQ Forte	Y	0.9558	4,542.47	6.14
optimization	maxcut	qaoa	24	25	21,300	15,300	5,000	IonQ Forte	Y	0.6556	4,484.49	6.07
optimization	maxcut	qaoa	24	25	21,300	15,300	5,000	IonQ Forte	Y	0.8995	4,423.36	5.98
optimization	maxcut	qaoa	24	25	21,300	20,400	5,000	IonQ Forte	Y	0.7684	5,942.48	8.04
optimization	maxcut	qaoa	24	25	21,300	20,400	5,000	IonQ Forte	Y	0.8828	5,817.15	7.87
optimization	maxcut	qaoa	24	25	26,100	18,900	5,000	IonQ Forte	Y	0.7308	5,443.04	7.36
optimization	maxcut	qaoa	24	25	26,100	18,900	5,000	IonQ Forte	Y	0.9046	5,479.26	7.41
optimization	maxcut	qaoa	24	25	26,100	25,200	5,000	IonQ Forte	Y	0.7214	7,143.66	9.66
optimization	maxcut	qaoa	24	25	26,100	25,200	5,000	IonQ Forte	Y	0.8441	7,100.45	9.61
optimization	maxcut	qaoa	24	25	30,900	22,500	5,000	IonQ Forte	Y	0.879	6,383.41	8.64
optimization	maxcut	qaoa	24	25	30,900	30,000	5,000	IonQ Forte	Y	0.8292	8,255.41	11.17
optimization	maxcut	qaoa	24	25	35,700	26,100	5,000	IonQ Forte	Y	0.8386	7,420.61	10.04
optimization	maxcut	qaoa	24	25	35,700	34,800	5,000	IonQ Forte	Y	0.8084	9,612.66	13.00
optimization	maxcut	qaoa	24	25	40,500	29,700	5,000	IonQ Forte	Y	0.7732	8,426.37	11.40
optimization	maxcut	qaoa	24	25	40,500	39,600	5,000	IonQ Forte	Y	0.7697	10,969.1	14.84
optimization	maxcut	qaoa	24	25	45,300	33,300	5,000	IonQ Forte	Y	0.7523	9,443.6	12.77
optimization	maxcut	qaoa	24	25	45,300	44,400	5,000	IonQ Forte	Y	0.7526	12,357.3	16.72
optimization	maxcut	qaoa	24	25	50,100	36,900	5,000	IonQ Forte	Y	0.7266	10,501.1	14.21
optimization	maxcut	qaoa	24	25	50,100	49,200	5,000	IonQ Forte	Y	0.7366	13,700.6	18.53
optimization	maxcut	qaoa	24	25	54,900	40,500	5,000	IonQ Forte	Y	0.7309	11,657	15.77
optimization	maxcut	qaoa	24	25	54,900	54,000	5,000	IonQ Forte	Y	0.6986	15,038.6	20.34
optimization	maxcut	qaoa	24	25	59,700	44,100	5,000	IonQ Forte	Y	0.7152	12,778.8	17.29
optimization	maxcut	qaoa	24	25	59,700	58,800	5,000	IonQ Forte	Y	0.6929	16,245.4	21.98
optimization	maxcut	qaoa	24	25	64,500	47,700	5,000	IonQ Forte	Y	0.697	13,722.2	18.56
optimization	maxcut	qaoa	24	25	64,500	63,600	5,000	IonQ Forte	Y	0.6863	17,594.3	23.80
optimization	maxcut	qaoa	36	1	72	54	5,000	IonQ Forte	N	0.3509	539.713	0.73
optimization	maxcut	qaoa	36	1	72	54	5,000	IonQ Forte	N	0.7369	602.598	0.82
optimization	maxcut	qaoa	36	1	72	54	5,000	IonQ Forte-E	N	0.7307	765.334	1.04
optimization	maxcut	qaoa	36	1	72	72	5,000	IonQ Forte	N	0.4194	613.157	0.83
optimization	maxcut	qaoa	36	1	108	108	5,000	IonQ Forte	N	0.7951	981.602	1.33
optimization	maxcut	qaoa	36	1	108	108	5,000	IonQ Forte-E	N	0.775	929.893	1.27
optimization	maxcut	qaoa	36	1	144	162	5,000	IonQ Forte	N	0.8257	1,422.65	1.92
optimization	maxcut	qaoa	36	1	144	162	5,000	IonQ Forte-E	N	0.7954	1,327.08	1.81
optimization	maxcut	qaoa	36	1	180	216	5,000	IonQ Forte	N	0.8364	1,796.74	2.43
optimization	maxcut	qaoa	36	1	180	216	5,000	IonQ Forte-E	N	0.7981	1,710.48	2.33
optimization	maxcut	qaoa	36	1	216	270	5,000	IonQ Forte	N	0.7546	2,051.88	2.78
optimization	maxcut	qaoa	36	1	216	270	5,000	IonQ Forte	N	0.8476	2,256.54	3.05
optimization	maxcut	qaoa	36	1	216	270	5,000	IonQ Forte-E	N	0.7993	2,138.46	2.91
optimization	maxcut	qaoa	36	1	216	360	5,000	IonQ Forte	N	0.7698	2,587.89	3.50
optimization	maxcut	qaoa	36	1	252	324	5,000	IonQ Forte	N	0.8469	2,660.51	3.60
optimization	maxcut	qaoa	36	1	252	324	5,000	IonQ Forte-E	N	0.7917	2,511.31	3.42
optimization	maxcut	qaoa	36	1	288	378	5,000	IonQ Forte	N	0.8444	3,104.22	4.20
optimization	maxcut	qaoa	36	1	288	378	5,000	IonQ Forte-E	N	0.7784	2,879.44	3.92
optimization	maxcut	qaoa	36	1	324	432	5,000	IonQ Forte	N	0.8387	3,481.3	4.71
optimization	maxcut	qaoa	36	1	324	432	5,000	IonQ Forte-E	N	0.7635	3,323.28	4.52
optimization	maxcut	qaoa	36	1	360	486	5,000	IonQ Forte	N	0.7844	3,611.55	4.89
optimization	maxcut	qaoa	36	1	360	486	5,000	IonQ Forte	N	0.8349	3,908.3	5.29
optimization	maxcut	qaoa	36	1	360	486	5,000	IonQ Forte-E	N	0.7539	3,740.15	5.09
optimization	maxcut	qaoa	36	1	360	648	5,000	IonQ Forte	N	0.8009	4,612.71	6.24
optimization	maxcut	qaoa	36	1	396	540	5,000	IonQ Forte	N	0.8351	4,326.37	5.85
optimization	maxcut	qaoa	36	1	396	540	5,000	IonQ Forte-E	N	0.7521	4,113.14	5.60
optimization	maxcut	qaoa	36	1	432	594	5,000	IonQ Forte	N	0.8261	4,819.84	6.52
optimization	maxcut	qaoa	36	1	432	594	5,000	IonQ Forte-E	N	0.7287	4,483.64	6.10
optimization	maxcut	qaoa	36	1	504	702	5,000	IonQ Forte	N	0.77	5,300.15	7.17
optimization	maxcut	qaoa	36	1	504	936	5,000	IonQ Forte	N	0.7895	6,487.98	8.78
optimization	maxcut	qaoa	36	1	648	918	5,000	IonQ Forte	N	0.7532	6,965.83	9.42
optimization	maxcut	qaoa	36	1	648	1,224	5,000	IonQ Forte	N	0.7673	8,449.9	11.43
optimization	maxcut	qaoa	36	1	792	1,134	5,000	IonQ Forte	N	0.728	8,209.41	11.11
optimization	maxcut	qaoa	36	1	792	1,512	5,000	IonQ Forte	N	0.7395	10,447.2	14.13
optimization	maxcut	qaoa	36	1	936	1,350	5,000	IonQ Forte	N	0.7172	9,869.58	13.35
optimization	maxcut	qaoa	36	1	936	1,800	5,000	IonQ Forte	N	0.7119	12,427.5	16.81

Domain	Problem	Algorithm	#q	#qc	#1q	#2q	Shots	Backend	EM?	Score	Exec. Time (s)	Energy (kWh)
optimization	maxcut	qaoa	36	1	1,080	1,566	5,000	IonQ Forte	N	0.6889	11,479.8	15.53
optimization	maxcut	qaoa	36	1	1,080	2,088	5,000	IonQ Forte	N	0.6886	14,450.6	19.55
optimization	maxcut	qaoa	36	1	1,224	1,782	5,000	IonQ Forte	N	0.666	13,019.7	17.61
optimization	maxcut	qaoa	36	1	1,224	2,376	5,000	IonQ Forte	N	0.6627	16,519.7	22.35
optimization	maxcut	qaoa	36	1	1,368	1,998	5,000	IonQ Forte	N	0.6501	14,502.3	19.62
optimization	maxcut	qaoa	36	1	1,368	2,664	5,000	IonQ Forte	N	0.6455	18,737.4	25.35
optimization	maxcut	qaoa	36	1	1,512	2,214	5,000	IonQ Forte	N	0.6357	15,784.6	21.35
optimization	maxcut	qaoa	36	1	1,512	2,952	5,000	IonQ Forte	N	0.6294	20,825.3	28.17
optimization	maxcut	qaoa	36	1	1,656	2,430	5,000	IonQ Forte	N	0.6223	17,411.4	23.55
optimization	maxcut	qaoa	36	1	1,656	3,240	5,000	IonQ Forte	N	0.6129	22,496.2	30.43
optimization	maxcut	qaoa	36	1	1,800	2,646	5,000	IonQ Forte	N	0.6064	18,988.5	25.69
optimization	maxcut	qaoa	36	1	1,800	3,528	5,000	IonQ Forte	N	0.6024	24,482.1	33.12
optimization	maxcut	qaoa	36	1	1,944	2,862	5,000	IonQ Forte	N	0.5966	20,659.7	27.95
optimization	maxcut	qaoa	36	1	1,944	3,816	5,000	IonQ Forte	N	0.5925	26,522.1	35.88
optimization	maxcut	qaoa	36	25	3,150	1,350	5,000	IonQ Forte	Y	0.3515	524.819	0.71
optimization	maxcut	qaoa	36	25	3,150	1,350	5,000	IonQ Forte	Y	0.7346	586.295	0.79
optimization	maxcut	qaoa	36	25	3,150	1,800	5,000	IonQ Forte	Y	0.4248	636.173	0.86
optimization	maxcut	qaoa	36	25	4,950	2,700	5,000	IonQ Forte	Y	0.7938	880.019	1.19
optimization	maxcut	qaoa	36	25	6,750	4,050	5,000	IonQ Forte	Y	0.829	1,257.67	1.70
optimization	maxcut	qaoa	36	25	8,550	5,400	5,000	IonQ Forte	Y	0.8741	1,632.39	2.21
optimization	maxcut	qaoa	36	25	10,350	6,750	5,000	IonQ Forte	Y	0.7451	2,046.96	2.77
optimization	maxcut	qaoa	36	25	10,350	6,750	5,000	IonQ Forte	Y	0.9207	2,004.53	2.71
optimization	maxcut	qaoa	36	25	10,350	9,000	5,000	IonQ Forte	Y	0.7608	2,566.85	3.47
optimization	maxcut	qaoa	36	25	12,150	8,100	5,000	IonQ Forte	Y	0.9334	2,380.8	3.22
optimization	maxcut	qaoa	36	25	13,950	9,450	5,000	IonQ Forte	Y	0.9504	2,758.35	3.73
optimization	maxcut	qaoa	36	25	15,750	10,800	5,000	IonQ Forte	Y	0.9599	3,144.71	4.25
optimization	maxcut	qaoa	36	25	17,550	12,150	5,000	IonQ Forte	Y	0.7678	3,595.39	4.86
optimization	maxcut	qaoa	36	25	17,550	12,150	5,000	IonQ Forte	Y	0.9743	3,511.08	4.75
optimization	maxcut	qaoa	36	25	17,550	16,200	5,000	IonQ Forte	Y	0.7867	4,488.39	6.07
optimization	maxcut	qaoa	36	25	19,350	13,500	5,000	IonQ Forte	Y	0.9537	3,884.71	5.26
optimization	maxcut	qaoa	36	25	21,150	14,850	5,000	IonQ Forte	Y	0.977	4,264.18	5.77
optimization	maxcut	qaoa	36	25	24,750	17,550	5,000	IonQ Forte	Y	0.7847	5,099.39	6.90
optimization	maxcut	qaoa	36	25	24,750	23,400	5,000	IonQ Forte	Y	0.7625	6,576.25	8.90
optimization	maxcut	qaoa	36	25	31,950	22,950	5,000	IonQ Forte	Y	0.7654	6,678.32	9.03
optimization	maxcut	qaoa	36	25	31,950	30,600	5,000	IonQ Forte	Y	0.7435	8,590.43	11.62
optimization	maxcut	qaoa	36	25	39,150	28,350	5,000	IonQ Forte	Y	0.7457	8,316.91	11.25
optimization	maxcut	qaoa	36	25	39,150	37,800	5,000	IonQ Forte	Y	0.7397	10,647.2	14.40
optimization	maxcut	qaoa	36	25	46,350	33,750	5,000	IonQ Forte	Y	0.6838	10,006.6	13.54
optimization	maxcut	qaoa	36	25	46,350	45,000	5,000	IonQ Forte	Y	0.7162	12,490.3	16.90
optimization	maxcut	qaoa	36	25	53,550	39,150	5,000	IonQ Forte	Y	0.6613	11,332.7	15.33
optimization	maxcut	qaoa	36	25	53,550	52,200	5,000	IonQ Forte	Y	0.6944	14,375.4	19.45
optimization	maxcut	qaoa	36	25	60,750	44,550	5,000	IonQ Forte	Y	0.6413	12,629	17.08
optimization	maxcut	qaoa	36	25	60,750	59,400	5,000	IonQ Forte	Y	0.6701	16,542.5	22.38
optimization	maxcut	qaoa	36	25	67,950	49,950	5,000	IonQ Forte	Y	0.6397	14,614.1	19.77
optimization	maxcut	qaoa	36	25	67,950	66,600	5,000	IonQ Forte	Y	0.6686	18,675.8	25.26
optimization	maxcut	qaoa	36	25	75,150	55,350	5,000	IonQ Forte	Y	0.6353	15,753.9	21.31
optimization	maxcut	qaoa	36	25	75,150	73,800	5,000	IonQ Forte	Y	0.6579	20,725.2	28.04
optimization	maxcut	qaoa	36	25	82,350	60,750	5,000	IonQ Forte	Y	0.6252	17,161	23.21
optimization	maxcut	qaoa	36	25	82,350	81,000	5,000	IonQ Forte	Y	0.6334	22,207.1	30.04
optimization	maxcut	qaoa	36	25	89,550	66,150	5,000	IonQ Forte	Y	0.5991	18,806.7	25.44
optimization	maxcut	qaoa	36	25	96,750	71,550	5,000	IonQ Forte	Y	0.5941	20,423.8	27.63
optimization	maxcut	varqite	8	1	48	80	1,000	IonQ Forte	N	0.965	119.231	0.16
optimization	maxcut	varqite	12	1	84	144	1,000	IonQ Forte	N	0.9422	209.849	0.28
optimization	maxcut	varqite	16	1	112	192	1,000	IonQ Forte	N	0.9456	265.448	0.36
optimization	maxcut	varqite	20	1	140	240	1,000	IonQ Forte	N	0.8359	341.083	0.46
optimization	maxcut	varqite	24	1	168	288	1,000	IonQ Forte	N	0.9053	398.838	0.54
optimization	risk assessment	quantum copula	15	1	33	22	5,000	IonQ Aria	N	0.9617	167.243	–
optimization	risk assessment	quantum copula	15	1	33	22	5,000	IonQ Aria	N	0.9872	154.714	–
optimization	risk assessment	quantum copula	15	1	33	22	5,000	IonQ Forte	N	0.978	202.355	0.27
optimization	risk assessment	quantum copula	15	1	33	22	5,000	IonQ Forte-E	N	0.9621	200.977	0.27
optimization	risk assessment	quantum copula	15	1	60	105	5,000	IonQ Aria	N	0.9597	364.566	–
optimization	risk assessment	quantum copula	15	1	60	105	5,000	IonQ Aria	N	0.9699	369.757	–
optimization	risk assessment	quantum copula	15	1	60	105	5,000	IonQ Aria	N	0.9772	376.872	–
optimization	risk assessment	quantum copula	15	1	60	105	5,000	IonQ Forte	N	0.9992	519.268	0.70
optimization	risk assessment	quantum copula	15	1	60	105	5,000	IonQ Forte-E	N	0.8864	475.608	0.65
optimization	risk assessment	quantum copula	15	25	1,262	300	5,000	IonQ Forte	Y	0.982	215.151	0.29
optimization	risk assessment	quantum copula	15	25	1,262	300	5,000	IonQ Forte-E	Y	0.9618	205.286	0.28
optimization	risk assessment	quantum copula	15	25	1,303	300	5,000	IonQ Aria	Y	0.9886	174.87	–
optimization	risk assessment	quantum copula	15	25	1,303	300	5,000	IonQ Aria	Y	0.9931	172.72	–
optimization	risk assessment	quantum copula	15	25	1,303	300	5,000	IonQ Aria	Y	0.9984	163.287	–
optimization	risk assessment	quantum copula	15	25	1,637	1,400	5,000	IonQ Forte	Y	0.9976	535.362	0.72
optimization	risk assessment	quantum copula	15	25	1,637	1,400	5,000	IonQ Forte-E	Y	0.8814	487.56	0.66
optimization	risk assessment	quantum copula	15	25	2,184	1,225	5,000	IonQ Aria	Y	0.8675	346.735	–
optimization	risk assessment	quantum copula	15	25	2,487	1,400	5,000	IonQ Aria	Y	0.8484	387.811	–
optimization	risk assessment	quantum copula	15	25	2,487	1,400	5,000	IonQ Aria	Y	0.9938	372.689	–
optimization	risk assessment	quantum copula	15	25	2,487	1,400	5,000	IonQ Aria	Y	0.9944	372.541	–
optimization	risk assessment	quantum copula	15	25	2,487	1,400	5,000	IonQ Aria	Y	0.9993	373.768	–
optimization	risk assessment	quantum copula	15	25	2,487	1,400	5,000	IonQ Aria	Y	0.9998	373.558	–
optimization	risk assessment	quantum copula	18	1	39	27	5,000	IonQ Aria	N	0.9899	189.705	–
optimization	risk assessment	quantum copula	18	1	39	27	5,000	IonQ Aria	N	0.9941	182.149	–

Domain	Problem	Algorithm	#q	#qc	#1q	#2q	Shots	Backend	EM?	Score	Exec. Time (s)	Energy (kWh)
optimization	risk assessment	quantum copula	18	1	39	27	5,000	IonQ Forte	N	0.9886	230.987	0.31
optimization	risk assessment	quantum copula	18	1	39	27	5,000	IonQ Forte-E	N	0.958	247.427	0.34
optimization	risk assessment	quantum copula	18	1	72	153	5,000	IonQ Aria	N	0.9591	481.078	–
optimization	risk assessment	quantum copula	18	1	72	153	5,000	IonQ Aria	N	0.9681	477.073	–
optimization	risk assessment	quantum copula	18	1	72	153	5,000	IonQ Aria	N	0.9971	474.985	–
optimization	risk assessment	quantum copula	18	1	72	153	5,000	IonQ Forte	N	0.9569	671.726	0.91
optimization	risk assessment	quantum copula	18	1	72	153	5,000	IonQ Forte-E	N	0.8695	757.515	1.03
optimization	risk assessment	quantum copula	18	25	1,575	375	5,000	IonQ Forte	Y	0.9725	245.716	0.33
optimization	risk assessment	quantum copula	18	25	1,575	375	5,000	IonQ Forte-E	Y	0.928	219.161	0.30
optimization	risk assessment	quantum copula	18	25	1,587	375	5,000	IonQ Aria	Y	0.9608	193.881	–
optimization	risk assessment	quantum copula	18	25	1,587	375	5,000	IonQ Aria	Y	0.9853	198.327	–
optimization	risk assessment	quantum copula	18	25	1,587	375	5,000	IonQ Aria	Y	0.9892	192.033	–
optimization	risk assessment	quantum copula	18	25	1,850	1,925	5,000	IonQ Forte	Y	0.9513	676.589	0.92
optimization	risk assessment	quantum copula	18	25	1,850	1,925	5,000	IonQ Forte-E	Y	0.8708	648.717	0.88
optimization	risk assessment	quantum copula	18	25	2,968	1,750	5,000	IonQ Aria	Y	0.8061	441.056	–
optimization	risk assessment	quantum copula	18	25	3,191	1,875	5,000	IonQ Aria	Y	0.6155	466.605	–
optimization	risk assessment	quantum copula	18	25	3,191	1,875	5,000	IonQ Aria	Y	0.9669	492.97	–
optimization	risk assessment	quantum copula	18	25	3,191	1,875	5,000	IonQ Aria	Y	0.9731	477.468	–
optimization	risk assessment	quantum copula	18	25	3,191	1,875	5,000	IonQ Aria	Y	0.9747	476.707	–
optimization	risk assessment	quantum copula	18	25	3,191	1,875	5,000	IonQ Aria	Y	0.9997	477.164	–
optimization	risk assessment	quantum copula	21	1	45	32	5,000	IonQ Aria	N	0.9235	209.233	–
optimization	risk assessment	quantum copula	21	1	45	32	5,000	IonQ Aria	N	0.9923	196.473	–
optimization	risk assessment	quantum copula	21	1	45	32	5,000	IonQ Forte	N	0.9669	263.598	0.36
optimization	risk assessment	quantum copula	21	1	45	32	5,000	IonQ Forte-E	N	0.9592	235.231	0.32
optimization	risk assessment	quantum copula	21	1	84	210	5,000	IonQ Aria	N	0.9667	741.416	–
optimization	risk assessment	quantum copula	21	1	84	210	5,000	IonQ Aria	N	0.9953	739.529	–
optimization	risk assessment	quantum copula	21	1	84	210	5,000	IonQ Aria	N	0.9988	735.61	–
optimization	risk assessment	quantum copula	21	1	84	210	5,000	IonQ Forte	N	0.9812	998.354	1.35
optimization	risk assessment	quantum copula	21	1	84	210	5,000	IonQ Forte-E	N	0.8668	975.129	1.33
optimization	risk assessment	quantum copula	21	25	1,837	450	5,000	IonQ Forte	Y	0.9909	282.176	0.38
optimization	risk assessment	quantum copula	21	25	1,837	450	5,000	IonQ Forte-E	Y	0.9409	251.342	0.34
optimization	risk assessment	quantum copula	21	25	1,869	450	5,000	IonQ Aria	Y	0.934	211.57	–
optimization	risk assessment	quantum copula	21	25	1,869	450	5,000	IonQ Aria	Y	0.9683	216.334	–
optimization	risk assessment	quantum copula	21	25	1,869	450	5,000	IonQ Aria	Y	0.9891	211.05	–
optimization	risk assessment	quantum copula	21	25	2,012	3,225	5,000	IonQ Forte	Y	0.9352	1,002.23	1.36
optimization	risk assessment	quantum copula	21	25	2,012	3,225	5,000	IonQ Forte-E	Y	0.8486	1,006.74	1.37
optimization	risk assessment	quantum copula	21	25	3,811	2,275	5,000	IonQ Aria	Y	0.7883	552.505	–
optimization	risk assessment	quantum copula	21	25	4,922	3,250	5,000	IonQ Aria	Y	0.8069	740.53	–
optimization	risk assessment	quantum copula	21	25	4,922	3,250	5,000	IonQ Aria	Y	0.9469	749.643	–
optimization	risk assessment	quantum copula	21	25	4,922	3,250	5,000	IonQ Aria	Y	0.9583	744.426	–
optimization	risk assessment	quantum copula	21	25	4,922	3,250	5,000	IonQ Aria	Y	0.9838	752.603	–
optimization	risk assessment	quantum copula	21	25	4,922	3,250	5,000	IonQ Aria	Y	0.9886	749.976	–
optimization	risk assessment	quantum copula	24	1	51	37	5,000	IonQ Aria	N	0.9738	233.141	–
optimization	risk assessment	quantum copula	24	1	51	37	5,000	IonQ Aria	N	0.986	217.937	–
optimization	risk assessment	quantum copula	24	1	51	37	5,000	IonQ Forte	N	0.9869	296.797	0.40
optimization	risk assessment	quantum copula	24	1	51	37	5,000	IonQ Forte-E	N	0.9195	265.033	0.36
optimization	risk assessment	quantum copula	24	1	96	276	5,000	IonQ Aria	N	0.8165	720.862	–
optimization	risk assessment	quantum copula	24	1	96	276	5,000	IonQ Aria	N	0.8436	671.591	–
optimization	risk assessment	quantum copula	24	1	96	276	5,000	IonQ Aria	N	0.8529	677.993	–
optimization	risk assessment	quantum copula	24	1	96	276	5,000	IonQ Aria	N	0.8764	667.889	–
optimization	risk assessment	quantum copula	24	1	96	276	5,000	IonQ Forte	N	0.8768	881.769	1.19
optimization	risk assessment	quantum copula	24	1	96	276	5,000	IonQ Forte-E	N	0.7266	908.724	1.24
optimization	risk assessment	quantum copula	24	25	2,100	525	5,000	IonQ Forte	Y	0.9868	308.081	0.42
optimization	risk assessment	quantum copula	24	25	2,100	525	5,000	IonQ Forte-E	Y	0.9572	283.047	0.39
optimization	risk assessment	quantum copula	24	25	2,150	525	5,000	IonQ Aria	Y	0.9429	233.923	–
optimization	risk assessment	quantum copula	24	25	2,150	525	5,000	IonQ Aria	Y	0.963	242.076	–
optimization	risk assessment	quantum copula	24	25	2,150	525	5,000	IonQ Aria	Y	0.9965	234.363	–
optimization	risk assessment	quantum copula	24	25	2,775	2,750	5,000	IonQ Forte	Y	0.9024	900.91	1.22
optimization	risk assessment	quantum copula	24	25	2,775	2,750	5,000	IonQ Forte-E	Y	0.72	903.859	1.23
optimization	risk assessment	quantum copula	24	25	4,471	2,800	5,000	IonQ Aria	Y	0.625	666.322	–
optimization	risk assessment	quantum copula	24	25	4,471	2,800	5,000	IonQ Aria	Y	0.8516	676.33	–
optimization	risk assessment	quantum copula	24	25	4,471	2,800	5,000	IonQ Aria	Y	0.8538	679.047	–
optimization	risk assessment	quantum copula	24	25	4,471	2,800	5,000	IonQ Aria	Y	0.86	683.806	–
optimization	risk assessment	quantum copula	24	25	4,471	2,800	5,000	IonQ Aria	Y	0.8809	675.04	–
optimization	risk assessment	quantum copula	24	25	4,742	3,075	5,000	IonQ Aria	Y	0.9687	728.79	–
optimization	risk assessment	quantum copula	27	1	57	42	5,000	IonQ Forte	N	0.9885	319.052	0.43
optimization	risk assessment	quantum copula	27	1	57	42	5,000	IonQ Forte-E	N	0.9356	301.007	0.41
optimization	risk assessment	quantum copula	27	1	108	351	5,000	IonQ Forte	N	0.926	1,142.66	1.55
optimization	risk assessment	quantum copula	27	1	108	351	5,000	IonQ Forte-E	N	0.7668	1,336.46	1.82
optimization	risk assessment	quantum copula	27	25	2,337	600	5,000	IonQ Forte	Y	0.9214	341.995	0.46
optimization	risk assessment	quantum copula	27	25	2,337	600	5,000	IonQ Forte-E	Y	0.9678	315.491	0.43
optimization	risk assessment	quantum copula	27	25	3,712	3,525	5,000	IonQ Forte	Y	0.9225	1,170.33	1.58
optimization	risk assessment	quantum copula	27	25	3,712	3,525	5,000	IonQ Forte-E	Y	0.721	1,232.08	1.68
optimization	risk assessment	quantum copula	30	1	63	47	5,000	IonQ Forte	N	0.9988	372.798	0.50
optimization	risk assessment	quantum copula	30	1	63	47	5,000	IonQ Forte-E	N	0.9469	348.676	0.47
optimization	risk assessment	quantum copula	30	1	120	435	5,000	IonQ Forte	N	0.9152	1,152.97	1.56
optimization	risk assessment	quantum copula	30	1	120	435	5,000	IonQ Forte-E	N	0.6767	1,248.83	1.70
optimization	risk assessment	quantum copula	30	25	2,600	675	5,000	IonQ Forte	Y	0.9804	395.442	0.53
optimization	risk assessment	quantum copula	30	25	2,600	675	5,000	IonQ Forte-E	Y	0.9081	362.307	0.49
optimization	risk assessment	quantum copula	30	25	4,250	3,525	5,000	IonQ Forte	Y	0.9158	1,184.61	1.60

Domain	Problem	Algorithm	#q	#qc	#1q	#2q	Shots	Backend	EM?	Score	Exec. Time (s)	Energy (kWh)
optimization	risk assessment	quantum copula	30	25	4,250	3,525	5,000	IonQ Forte-E	Y	0.6466	1,183.8	1.61
qft	qft	cosine qft	12	1	99	156	1,000	IonQ Forte	N	0.591	263.161	0.36
qft	qft	cosine qft	12	1	99	156	1,000	IonQ Forte-E	N	0.5836	162.122	0.22
qft	qft	cosine qft	16	1	127	272	1,000	IonQ Forte	N	0.4529	401.26	0.54
qft	qft	cosine qft	16	1	127	272	1,000	IonQ Forte-E	N	0.3595	264.218	0.36
qft	qft	cosine qft	20	1	155	420	1,000	IonQ Forte	N	0.3241	522.35	0.71
qft	qft	cosine qft	20	1	155	420	1,000	IonQ Forte-E	N	0.2072	353.329	0.48
qft	qft	cosine qft	24	1	183	600	1,000	IonQ Forte	N	0.1589	636.525	0.86
qft	qft	cosine qft	24	1	183	600	1,000	IonQ Forte-E	N	0.109	440.939	0.60
qft	qft	cosine qft	28	1	211	812	1,000	IonQ Forte	N	0.1176	767.818	1.04
qft	qft	cosine qft	28	1	211	812	1,000	IonQ Forte-E	N	0.08999	555.718	0.76
qft	qft	cosine qft	32	1	239	1,056	1,000	IonQ Forte	N	0.06232	900.155	1.22
qft	qft	cosine qft	32	1	239	1,056	1,000	IonQ Forte-E	N	0.02699	593.987	0.81
qft	qft	cosine qft	36	1	267	1,332	1,000	IonQ Forte	N	0.03397	1,027.35	1.39
qft	qft	cosine qft	36	1	267	1,332	1,000	IonQ Forte-E	N	0.01371	680.11	0.93
qft	qft	modulated qft	12	1	23	82	1,000	IonQ Aria	N	0.9885	130.762	–
qft	qft	modulated qft	12	1	23	82	1,000	IonQ Forte	N	0.9945	232.996	0.32
qft	qft	modulated qft	12	1	23	82	1,000	IonQ Forte-E	N	0.9854	205.252	0.28
qft	qft	modulated qft	12	25	825	1,475	1,000	IonQ Forte	Y	0.8763	242.389	0.33
qft	qft	modulated qft	12	25	825	1,475	1,000	IonQ Forte-E	Y	0.8764	215.424	0.29
qft	qft	modulated qft	12	25	1,765	1,475	1,000	IonQ Aria	Y	0.7329	117.201	–
qft	qft	modulated qft	16	1	31	142	1,000	IonQ Aria	N	0.9886	167.821	–
qft	qft	modulated qft	16	1	31	142	1,000	IonQ Forte	N	0.9929	331.805	0.45
qft	qft	modulated qft	16	1	31	142	1,000	IonQ Forte-E	N	0.9824	304.376	0.41
qft	qft	modulated qft	16	25	1,275	2,275	1,000	IonQ Forte	Y	0.7756	339.182	0.46
qft	qft	modulated qft	16	25	1,275	2,275	1,000	IonQ Forte-E	Y	0.828	323.914	0.44
qft	qft	modulated qft	16	25	2,508	2,275	1,000	IonQ Aria	Y	0.7914	153.415	–
qft	qft	modulated qft	20	1	39	218	1,000	IonQ Aria	N	0.9792	214.079	–
qft	qft	modulated qft	20	1	39	218	1,000	IonQ Forte	N	0.9938	448.244	0.61
qft	qft	modulated qft	20	1	39	218	1,000	IonQ Forte-E	N	0.9735	401.515	0.55
qft	qft	modulated qft	20	25	1,725	3,075	1,000	IonQ Forte	Y	0.762	458.509	0.62
qft	qft	modulated qft	20	25	1,725	3,075	1,000	IonQ Forte-E	Y	0.8123	412.884	0.56
qft	qft	modulated qft	20	25	3,246	3,075	1,000	IonQ Aria	Y	0.8539	192.76	–
qft	qft	modulated qft	24	1	47	310	1,000	IonQ Aria	N	0.9627	251.607	–
qft	qft	modulated qft	24	1	47	310	1,000	IonQ Forte	N	0.9907	538.062	0.73
qft	qft	modulated qft	24	1	47	310	1,000	IonQ Forte-E	N	0.9634	496.094	0.68
qft	qft	modulated qft	24	25	2,175	3,875	1,000	IonQ Forte	Y	0.73	548.926	0.74
qft	qft	modulated qft	24	25	2,175	3,875	1,000	IonQ Forte-E	Y	0.7392	513.161	0.70
qft	qft	modulated qft	24	25	4,019	3,875	1,000	IonQ Aria	Y	0.839	234.033	–
qft	qft	modulated qft	28	1	55	418	1,000	IonQ Forte	N	0.9882	633.081	0.86
qft	qft	modulated qft	28	1	55	418	1,000	IonQ Forte-E	N	0.9642	594.454	0.81
qft	qft	modulated qft	28	25	2,625	4,675	1,000	IonQ Forte	Y	0.7455	648.972	0.88
qft	qft	modulated qft	28	25	2,625	4,675	1,000	IonQ Forte-E	Y	0.7813	608.127	0.83
qft	qft	modulated qft	32	1	63	542	1,000	IonQ Forte	N	0.9855	733.943	0.99
qft	qft	modulated qft	32	1	63	542	1,000	IonQ Forte-E	N	0.9615	674.432	0.92
qft	qft	modulated qft	32	25	3,075	5,475	1,000	IonQ Forte	Y	0.7348	745.482	1.01
qft	qft	modulated qft	32	25	3,075	5,475	1,000	IonQ Forte-E	Y	0.7348	702.848	0.96
qft	qft	modulated qft	36	1	71	682	1,000	IonQ Forte	N	0.986	846.981	1.15
qft	qft	modulated qft	36	1	71	682	1,000	IonQ Forte-E	N	0.9591	827.455	1.13
qft	qft	modulated qft	36	25	3,525	6,275	1,000	IonQ Forte	Y	0.7365	853.969	1.16
qft	qft	modulated qft	36	25	3,525	6,275	1,000	IonQ Forte-E	Y	0.7365	819.684	1.12
qft	qft	qft hidden phase	12	1	212	192	1,000	IonQ Forte	N	0.8658	104.93	0.14
qft	qft	qft hidden phase	16	1	276	320	1,000	IonQ Forte	N	0.5338	275.951	0.37
qft	qft	qft hidden phase	20	1	340	480	1,000	IonQ Forte	N	0.3945	403.189	0.55
qft	qft	qft hidden phase	24	1	404	672	1,000	IonQ Forte	N	0.365	528.46	0.71
qft	qft	qft hidden phase	28	1	468	896	1,000	IonQ Forte	N	0.2233	662.631	0.90
qft	qft	qft hidden phase	32	1	532	1,152	1,000	IonQ Forte	N	0.1656	774.863	1.05
unstructured search	faa	faa	4	1	510	216	1,000	IonQ Aria	N	0.366	212.555	–
unstructured search	faa	faa	4	1	510	216	1,000	IonQ Forte	N	0.23	324.702	0.44
unstructured search	faa	faa	4	1	510	216	1,000	IonQ Forte-E	N	0.139	333.737	0.45
unstructured search	faa	faa	4	1	520	216	1,000	IonQ Aria	N	0.341	212.175	–
unstructured search	faa	faa	4	1	520	216	1,000	IonQ Aria	N	0.365	210.17	–
unstructured search	faa	faa	4	1	520	216	1,000	IonQ Aria	N	0.39	210.457	–
unstructured search	faa	faa	4	1	520	216	1,000	IonQ Aria	N	0.394	214.66	–
unstructured search	faa	faa	4	1	520	216	1,000	IonQ Forte	N	0.211	320.402	0.43
unstructured search	faa	faa	4	1	520	216	1,000	IonQ Forte	N	0.212	323.759	0.44
unstructured search	faa	faa	4	1	520	216	1,000	IonQ Forte	N	0.226	324.254	0.44
unstructured search	faa	faa	4	1	520	216	1,000	IonQ Forte	N	0.23	327.132	0.44
unstructured search	faa	faa	4	1	520	216	1,000	IonQ Forte-E	N	0.139	331.612	0.45
unstructured search	faa	faa	4	1	520	216	1,000	IonQ Forte-E	N	0.158	319.05	0.43
unstructured search	faa	faa	4	1	520	216	1,000	IonQ Forte-E	N	0.172	326.721	0.44
unstructured search	faa	faa	4	1	520	216	1,000	IonQ Forte-E	N	0.174	327.12	0.45
unstructured search	faa	faa	4	1	530	216	1,000	IonQ Aria	N	0.33	211.398	–
unstructured search	faa	faa	4	1	530	216	1,000	IonQ Aria	N	0.334	214.473	–
unstructured search	faa	faa	4	1	530	216	1,000	IonQ Aria	N	0.343	211.601	–
unstructured search	faa	faa	4	1	530	216	1,000	IonQ Aria	N	0.346	214.012	–
unstructured search	faa	faa	4	1	530	216	1,000	IonQ Aria	N	0.351	209.609	–
unstructured search	faa	faa	4	1	530	216	1,000	IonQ Aria	N	0.368	211.336	–
unstructured search	faa	faa	4	1	530	216	1,000	IonQ Forte	N	0.204	321.227	0.43
unstructured search	faa	faa	4	1	530	216	1,000	IonQ Forte	N	0.205	321.733	0.44

Domain	Problem	Algorithm	#q	#qc	#1q	#2q	Shots	Backend	EM?	Score	Exec. Time (s)	Energy (kWh)
unstructured search	faa	faa	4	1	530	216	1,000	IonQ Forte	N	0.209	319.8	0.43
unstructured search	faa	faa	4	1	530	216	1,000	IonQ Forte	N	0.227	318.093	0.43
unstructured search	faa	faa	4	1	530	216	1,000	IonQ Forte	N	0.231	322.206	0.44
unstructured search	faa	faa	4	1	530	216	1,000	IonQ Forte	N	0.242	322.098	0.44
unstructured search	faa	faa	4	1	530	216	1,000	IonQ Forte-E	N	0.144	325.697	0.44
unstructured search	faa	faa	4	1	530	216	1,000	IonQ Forte-E	N	0.146	326.171	0.44
unstructured search	faa	faa	4	1	530	216	1,000	IonQ Forte-E	N	0.165	329.334	0.45
unstructured search	faa	faa	4	1	530	216	1,000	IonQ Forte-E	N	0.181	323.262	0.44
unstructured search	faa	faa	4	1	530	216	1,000	IonQ Forte-E	N	0.184	329.539	0.45
unstructured search	faa	faa	4	1	530	216	1,000	IonQ Forte-E	N	0.195	326.933	0.44
unstructured search	faa	faa	4	1	540	216	1,000	IonQ Aria	N	0.332	214.436	–
unstructured search	faa	faa	4	1	540	216	1,000	IonQ Aria	N	0.369	212.669	–
unstructured search	faa	faa	4	1	540	216	1,000	IonQ Aria	N	0.381	214.833	–
unstructured search	faa	faa	4	1	540	216	1,000	IonQ Aria	N	0.39	213.985	–
unstructured search	faa	faa	4	1	540	216	1,000	IonQ Forte	N	0.184	320.956	0.43
unstructured search	faa	faa	4	1	540	216	1,000	IonQ Forte	N	0.201	325.074	0.44
unstructured search	faa	faa	4	1	540	216	1,000	IonQ Forte	N	0.221	323.16	0.44
unstructured search	faa	faa	4	1	540	216	1,000	IonQ Forte	N	0.223	321.713	0.44
unstructured search	faa	faa	4	1	540	216	1,000	IonQ Forte-E	N	0.165	332.168	0.45
unstructured search	faa	faa	4	1	540	216	1,000	IonQ Forte-E	N	0.192	333.889	0.45
unstructured search	faa	faa	4	1	540	216	1,000	IonQ Forte-E	N	0.195	323.606	0.44
unstructured search	faa	faa	4	1	540	216	1,000	IonQ Forte-E	N	0.198	335.655	0.46
unstructured search	faa	faa	4	1	550	216	1,000	IonQ Aria	N	0.388	211.17	–
unstructured search	faa	faa	4	1	550	216	1,000	IonQ Forte	N	0.206	321.469	0.43
unstructured search	faa	faa	4	1	550	216	1,000	IonQ Forte-E	N	0.188	322.397	0.44
unstructured search	faa	faa	4	1	1,080	456	1,000	IonQ Aria	N	0.169	382.48	–
unstructured search	faa	faa	4	1	1,080	456	1,000	IonQ Forte	N	0.127	600.779	0.81
unstructured search	faa	faa	4	1	1,080	456	1,000	IonQ Forte-E	N	0.085	623.704	0.85
unstructured search	faa	faa	4	1	1,100	456	1,000	IonQ Aria	N	0.147	384.259	–
unstructured search	faa	faa	4	1	1,100	456	1,000	IonQ Aria	N	0.149	381.091	–
unstructured search	faa	faa	4	1	1,100	456	1,000	IonQ Aria	N	0.151	383.05	–
unstructured search	faa	faa	4	1	1,100	456	1,000	IonQ Aria	N	0.176	384.948	–
unstructured search	faa	faa	4	1	1,100	456	1,000	IonQ Forte	N	0.094	591.347	0.80
unstructured search	faa	faa	4	1	1,100	456	1,000	IonQ Forte	N	0.098	604.029	0.82
unstructured search	faa	faa	4	1	1,100	456	1,000	IonQ Forte	N	0.103	601.88	0.81
unstructured search	faa	faa	4	1	1,100	456	1,000	IonQ Forte	N	0.108	600.519	0.81
unstructured search	faa	faa	4	1	1,100	456	1,000	IonQ Forte-E	N	0.078	621.522	0.85
unstructured search	faa	faa	4	1	1,100	456	1,000	IonQ Forte-E	N	0.079	612.957	0.83
unstructured search	faa	faa	4	1	1,100	456	1,000	IonQ Forte-E	N	0.088	618.742	0.84
unstructured search	faa	faa	4	1	1,100	456	1,000	IonQ Forte-E	N	0.092	620.017	0.84
unstructured search	faa	faa	4	1	1,120	456	1,000	IonQ Aria	N	0.139	379.762	–
unstructured search	faa	faa	4	1	1,120	456	1,000	IonQ Aria	N	0.144	385.54	–
unstructured search	faa	faa	4	1	1,120	456	1,000	IonQ Aria	N	0.155	382.731	–
unstructured search	faa	faa	4	1	1,120	456	1,000	IonQ Aria	N	0.16	383.062	–
unstructured search	faa	faa	4	1	1,120	456	1,000	IonQ Aria	N	0.163	383.295	–
unstructured search	faa	faa	4	1	1,120	456	1,000	IonQ Aria	N	0.204	384.074	–
unstructured search	faa	faa	4	1	1,120	456	1,000	IonQ Forte	N	0.076	605.858	0.82
unstructured search	faa	faa	4	1	1,120	456	1,000	IonQ Forte	N	0.094	599.79	0.81
unstructured search	faa	faa	4	1	1,120	456	1,000	IonQ Forte	N	0.097	602.117	0.81
unstructured search	faa	faa	4	1	1,120	456	1,000	IonQ Forte	N	0.1	592.225	0.80
unstructured search	faa	faa	4	1	1,120	456	1,000	IonQ Forte	N	0.107	597.951	0.81
unstructured search	faa	faa	4	1	1,120	456	1,000	IonQ Forte	N	0.139	600.539	0.81
unstructured search	faa	faa	4	1	1,120	456	1,000	IonQ Forte-E	N	0.064	618.49	0.84
unstructured search	faa	faa	4	1	1,120	456	1,000	IonQ Forte-E	N	0.07	625.513	0.85
unstructured search	faa	faa	4	1	1,120	456	1,000	IonQ Forte-E	N	0.076	616.369	0.84
unstructured search	faa	faa	4	1	1,120	456	1,000	IonQ Forte-E	N	0.082	618.02	0.84
unstructured search	faa	faa	4	1	1,120	456	1,000	IonQ Forte-E	N	0.083	617.603	0.84
unstructured search	faa	faa	4	1	1,120	456	1,000	IonQ Forte-E	N	0.097	614.541	0.84
unstructured search	faa	faa	4	1	1,140	456	1,000	IonQ Aria	N	0.129	380.794	–
unstructured search	faa	faa	4	1	1,140	456	1,000	IonQ Aria	N	0.151	382.975	–
unstructured search	faa	faa	4	1	1,140	456	1,000	IonQ Aria	N	0.172	384.12	–
unstructured search	faa	faa	4	1	1,140	456	1,000	IonQ Aria	N	0.173	380.667	–
unstructured search	faa	faa	4	1	1,140	456	1,000	IonQ Forte	N	0.072	598.692	0.81
unstructured search	faa	faa	4	1	1,140	456	1,000	IonQ Forte	N	0.08	592.278	0.80
unstructured search	faa	faa	4	1	1,140	456	1,000	IonQ Forte	N	0.086	596.869	0.81
unstructured search	faa	faa	4	1	1,140	456	1,000	IonQ Forte	N	0.087	601.915	0.81
unstructured search	faa	faa	4	1	1,140	456	1,000	IonQ Forte-E	N	0.059	628.768	0.86
unstructured search	faa	faa	4	1	1,140	456	1,000	IonQ Forte-E	N	0.074	625.095	0.85
unstructured search	faa	faa	4	1	1,140	456	1,000	IonQ Forte-E	N	0.083	623.809	0.85
unstructured search	faa	faa	4	1	1,140	456	1,000	IonQ Forte-E	N	0.099	620.689	0.84
unstructured search	faa	faa	4	1	1,160	456	1,000	IonQ Aria	N	0.166	381.502	–
unstructured search	faa	faa	4	1	1,160	456	1,000	IonQ Forte	N	0.076	595.638	0.81
unstructured search	faa	faa	4	1	1,160	456	1,000	IonQ Forte-E	N	0.076	614.93	0.84
unstructured search	faa	faa	5	1	743	324	1,000	IonQ Aria	N	0.186	265.423	–
unstructured search	faa	faa	5	1	743	324	1,000	IonQ Forte-E	N	0.071	400.168	0.54
unstructured search	faa	faa	5	1	753	324	1,000	IonQ Aria	N	0.125	262.436	–
unstructured search	faa	faa	5	1	753	324	1,000	IonQ Aria	N	0.197	266.054	–
unstructured search	faa	faa	5	1	753	324	1,000	IonQ Aria	N	0.197	268.219	–
unstructured search	faa	faa	5	1	753	324	1,000	IonQ Aria	N	0.212	266.196	–
unstructured search	faa	faa	5	1	753	324	1,000	IonQ Aria	N	0.214	264.226	–

Domain	Problem	Algorithm	#q	#qc	#1q	#2q	Shots	Backend	EM?	Score	Exec. Time (s)	Energy (kWh)
unstructured search	faa	faa	5	1	753	324	1,000	IonQ Forte-E	N	0.057	392.828	0.53
unstructured search	faa	faa	5	1	753	324	1,000	IonQ Forte-E	N	0.058	398.24	0.54
unstructured search	faa	faa	5	1	753	324	1,000	IonQ Forte-E	N	0.059	397.937	0.54
unstructured search	faa	faa	5	1	753	324	1,000	IonQ Forte-E	N	0.063	396.782	0.54
unstructured search	faa	faa	5	1	753	324	1,000	IonQ Forte-E	N	0.064	432.084	0.59
unstructured search	faa	faa	5	1	763	324	1,000	IonQ Aria	N	0.109	265.77	–
unstructured search	faa	faa	5	1	763	324	1,000	IonQ Aria	N	0.134	272.212	–
unstructured search	faa	faa	5	1	763	324	1,000	IonQ Aria	N	0.153	266.879	–
unstructured search	faa	faa	5	1	763	324	1,000	IonQ Aria	N	0.153	271.333	–
unstructured search	faa	faa	5	1	763	324	1,000	IonQ Aria	N	0.179	270.362	–
unstructured search	faa	faa	5	1	763	324	1,000	IonQ Aria	N	0.18	266.625	–
unstructured search	faa	faa	5	1	763	324	1,000	IonQ Aria	N	0.181	266.663	–
unstructured search	faa	faa	5	1	763	324	1,000	IonQ Aria	N	0.2	265.031	–
unstructured search	faa	faa	5	1	763	324	1,000	IonQ Aria	N	0.209	266.683	–
unstructured search	faa	faa	5	1	763	324	1,000	IonQ Aria	N	0.211	268.929	–
unstructured search	faa	faa	5	1	763	324	1,000	IonQ Forte-E	N	0.049	397.274	0.54
unstructured search	faa	faa	5	1	763	324	1,000	IonQ Forte-E	N	0.053	385.108	0.52
unstructured search	faa	faa	5	1	763	324	1,000	IonQ Forte-E	N	0.055	401.346	0.55
unstructured search	faa	faa	5	1	763	324	1,000	IonQ Forte-E	N	0.059	433.268	0.59
unstructured search	faa	faa	5	1	763	324	1,000	IonQ Forte-E	N	0.064	398.84	0.54
unstructured search	faa	faa	5	1	763	324	1,000	IonQ Forte-E	N	0.069	391.585	0.53
unstructured search	faa	faa	5	1	763	324	1,000	IonQ Forte-E	N	0.071	398.003	0.54
unstructured search	faa	faa	5	1	763	324	1,000	IonQ Forte-E	N	0.074	414.543	0.56
unstructured search	faa	faa	5	1	763	324	1,000	IonQ Forte-E	N	0.076	402.892	0.55
unstructured search	faa	faa	5	1	763	324	1,000	IonQ Forte-E	N	0.085	397.799	0.54
unstructured search	faa	faa	5	1	773	324	1,000	IonQ Aria	N	0.118	263.629	–
unstructured search	faa	faa	5	1	773	324	1,000	IonQ Aria	N	0.131	264.124	–
unstructured search	faa	faa	5	1	773	324	1,000	IonQ Aria	N	0.143	265.945	–
unstructured search	faa	faa	5	1	773	324	1,000	IonQ Aria	N	0.156	270.985	–
unstructured search	faa	faa	5	1	773	324	1,000	IonQ Aria	N	0.164	272.961	–
unstructured search	faa	faa	5	1	773	324	1,000	IonQ Aria	N	0.168	269.012	–
unstructured search	faa	faa	5	1	773	324	1,000	IonQ Aria	N	0.177	271.526	–
unstructured search	faa	faa	5	1	773	324	1,000	IonQ Aria	N	0.197	270.609	–
unstructured search	faa	faa	5	1	773	324	1,000	IonQ Aria	N	0.201	266.1	–
unstructured search	faa	faa	5	1	773	324	1,000	IonQ Aria	N	0.227	266.383	–
unstructured search	faa	faa	5	1	773	324	1,000	IonQ Forte-E	N	0.066	398.758	0.54
unstructured search	faa	faa	5	1	773	324	1,000	IonQ Forte-E	N	0.066	440.089	0.60
unstructured search	faa	faa	5	1	773	324	1,000	IonQ Forte-E	N	0.069	432.49	0.59
unstructured search	faa	faa	5	1	773	324	1,000	IonQ Forte-E	N	0.069	433.4	0.59
unstructured search	faa	faa	5	1	773	324	1,000	IonQ Forte-E	N	0.071	427.149	0.58
unstructured search	faa	faa	5	1	773	324	1,000	IonQ Forte-E	N	0.078	408.518	0.56
unstructured search	faa	faa	5	1	773	324	1,000	IonQ Forte-E	N	0.078	433.242	0.59
unstructured search	faa	faa	5	1	773	324	1,000	IonQ Forte-E	N	0.079	387.32	0.53
unstructured search	faa	faa	5	1	773	324	1,000	IonQ Forte-E	N	0.088	392.303	0.53
unstructured search	faa	faa	5	1	773	324	1,000	IonQ Forte-E	N	0.091	390.99	0.53
unstructured search	faa	faa	5	1	783	324	1,000	IonQ Aria	N	0.125	264.074	–
unstructured search	faa	faa	5	1	783	324	1,000	IonQ Aria	N	0.129	264.636	–
unstructured search	faa	faa	5	1	783	324	1,000	IonQ Aria	N	0.21	269.354	–
unstructured search	faa	faa	5	1	783	324	1,000	IonQ Aria	N	0.216	268.345	–
unstructured search	faa	faa	5	1	783	324	1,000	IonQ Aria	N	0.223	270.473	–
unstructured search	faa	faa	5	1	783	324	1,000	IonQ Forte-E	N	0.071	394.114	0.54
unstructured search	faa	faa	5	1	783	324	1,000	IonQ Forte-E	N	0.088	401.461	0.55
unstructured search	faa	faa	5	1	783	324	1,000	IonQ Forte-E	N	0.09	411.014	0.56
unstructured search	faa	faa	5	1	783	324	1,000	IonQ Forte-E	N	0.101	433.22	0.59
unstructured search	faa	faa	5	1	783	324	1,000	IonQ Forte-E	N	0.111	436.248	0.59
unstructured search	faa	faa	5	1	793	324	1,000	IonQ Aria	N	0.207	271.17	–
unstructured search	faa	faa	5	1	793	324	1,000	IonQ Forte-E	N	0.118	425.253	0.58
unstructured search	faa	faa	5	1	1,573	684	1,000	IonQ Aria	N	0.066	495.537	–
unstructured search	faa	faa	5	1	1,573	684	1,000	IonQ Forte-E	N	0.046	787.172	1.07
unstructured search	faa	faa	5	1	1,593	684	1,000	IonQ Aria	N	0.062	491.753	–
unstructured search	faa	faa	5	1	1,593	684	1,000	IonQ Aria	N	0.072	495.781	–
unstructured search	faa	faa	5	1	1,593	684	1,000	IonQ Aria	N	0.078	494.616	–
unstructured search	faa	faa	5	1	1,593	684	1,000	IonQ Aria	N	0.079	491.968	–
unstructured search	faa	faa	5	1	1,593	684	1,000	IonQ Aria	N	0.085	495.65	–
unstructured search	faa	faa	5	1	1,593	684	1,000	IonQ Forte-E	N	0.032	770.119	1.05
unstructured search	faa	faa	5	1	1,593	684	1,000	IonQ Forte-E	N	0.037	843.673	1.15
unstructured search	faa	faa	5	1	1,593	684	1,000	IonQ Forte-E	N	0.041	783.139	1.07
unstructured search	faa	faa	5	1	1,593	684	1,000	IonQ Forte-E	N	0.043	774.756	1.05
unstructured search	faa	faa	5	1	1,593	684	1,000	IonQ Forte-E	N	0.043	780.117	1.06
unstructured search	faa	faa	5	1	1,613	684	1,000	IonQ Aria	N	0.055	490.14	–
unstructured search	faa	faa	5	1	1,613	684	1,000	IonQ Aria	N	0.061	498.824	–
unstructured search	faa	faa	5	1	1,613	684	1,000	IonQ Aria	N	0.068	491.875	–
unstructured search	faa	faa	5	1	1,613	684	1,000	IonQ Aria	N	0.068	493.056	–
unstructured search	faa	faa	5	1	1,613	684	1,000	IonQ Aria	N	0.07	499.071	–
unstructured search	faa	faa	5	1	1,613	684	1,000	IonQ Aria	N	0.075	495.917	–
unstructured search	faa	faa	5	1	1,613	684	1,000	IonQ Aria	N	0.08	494.495	–
unstructured search	faa	faa	5	1	1,613	684	1,000	IonQ Aria	N	0.082	504.292	–
unstructured search	faa	faa	5	1	1,613	684	1,000	IonQ Aria	N	0.085	493.968	–
unstructured search	faa	faa	5	1	1,613	684	1,000	IonQ Aria	N	0.086	497.125	–
unstructured search	faa	faa	5	1	1,613	684	1,000	IonQ Forte-E	N	0.03	766.855	1.04

Domain	Problem	Algorithm	#q	#qc	#1q	#2q	Shots	Backend	EM?	Score	Exec. Time (s)	Energy (kWh)
unstructured search	faa	faa	5	1	1,613	684	1,000	IonQ Forte-E	N	0.033	777.817	1.06
unstructured search	faa	faa	5	1	1,613	684	1,000	IonQ Forte-E	N	0.034	756.06	1.03
unstructured search	faa	faa	5	1	1,613	684	1,000	IonQ Forte-E	N	0.034	779.631	1.06
unstructured search	faa	faa	5	1	1,613	684	1,000	IonQ Forte-E	N	0.034	818.611	1.11
unstructured search	faa	faa	5	1	1,613	684	1,000	IonQ Forte-E	N	0.035	772.356	1.05
unstructured search	faa	faa	5	1	1,613	684	1,000	IonQ Forte-E	N	0.035	794.754	1.08
unstructured search	faa	faa	5	1	1,613	684	1,000	IonQ Forte-E	N	0.04	801.446	1.09
unstructured search	faa	faa	5	1	1,613	684	1,000	IonQ Forte-E	N	0.043	813.788	1.11
unstructured search	faa	faa	5	1	1,613	684	1,000	IonQ Forte-E	N	0.049	773.566	1.05
unstructured search	faa	faa	5	1	1,633	684	1,000	IonQ Aria	N	0.055	489.484	–
unstructured search	faa	faa	5	1	1,633	684	1,000	IonQ Aria	N	0.055	500.254	–
unstructured search	faa	faa	5	1	1,633	684	1,000	IonQ Aria	N	0.057	493.039	–
unstructured search	faa	faa	5	1	1,633	684	1,000	IonQ Aria	N	0.06	499.084	–
unstructured search	faa	faa	5	1	1,633	684	1,000	IonQ Aria	N	0.064	497.269	–
unstructured search	faa	faa	5	1	1,633	684	1,000	IonQ Aria	N	0.066	490.574	–
unstructured search	faa	faa	5	1	1,633	684	1,000	IonQ Aria	N	0.066	500.282	–
unstructured search	faa	faa	5	1	1,633	684	1,000	IonQ Aria	N	0.068	498.839	–
unstructured search	faa	faa	5	1	1,633	684	1,000	IonQ Aria	N	0.07	493.623	–
unstructured search	faa	faa	5	1	1,633	684	1,000	IonQ Aria	N	0.07	494.022	–
unstructured search	faa	faa	5	1	1,633	684	1,000	IonQ Forte-E	N	0.026	844.295	1.15
unstructured search	faa	faa	5	1	1,633	684	1,000	IonQ Forte-E	N	0.037	757.684	1.03
unstructured search	faa	faa	5	1	1,633	684	1,000	IonQ Forte-E	N	0.037	771.094	1.05
unstructured search	faa	faa	5	1	1,633	684	1,000	IonQ Forte-E	N	0.037	835.661	1.14
unstructured search	faa	faa	5	1	1,633	684	1,000	IonQ Forte-E	N	0.038	794.456	1.08
unstructured search	faa	faa	5	1	1,633	684	1,000	IonQ Forte-E	N	0.041	813.298	1.11
unstructured search	faa	faa	5	1	1,633	684	1,000	IonQ Forte-E	N	0.042	823.838	1.12
unstructured search	faa	faa	5	1	1,633	684	1,000	IonQ Forte-E	N	0.044	824.091	1.12
unstructured search	faa	faa	5	1	1,633	684	1,000	IonQ Forte-E	N	0.045	774.386	1.05
unstructured search	faa	faa	5	1	1,633	684	1,000	IonQ Forte-E	N	0.045	790.548	1.08
unstructured search	faa	faa	5	1	1,653	684	1,000	IonQ Aria	N	0.047	489.85	–
unstructured search	faa	faa	5	1	1,653	684	1,000	IonQ Aria	N	0.055	490.508	–
unstructured search	faa	faa	5	1	1,653	684	1,000	IonQ Aria	N	0.064	494.247	–
unstructured search	faa	faa	5	1	1,653	684	1,000	IonQ Aria	N	0.068	503.957	–
unstructured search	faa	faa	5	1	1,653	684	1,000	IonQ Aria	N	0.07	502.255	–
unstructured search	faa	faa	5	1	1,653	684	1,000	IonQ Forte-E	N	0.033	774.328	1.05
unstructured search	faa	faa	5	1	1,653	684	1,000	IonQ Forte-E	N	0.034	758.062	1.03
unstructured search	faa	faa	5	1	1,653	684	1,000	IonQ Forte-E	N	0.035	834.768	1.14
unstructured search	faa	faa	5	1	1,653	684	1,000	IonQ Forte-E	N	0.04	873.133	1.19
unstructured search	faa	faa	5	1	1,653	684	1,000	IonQ Forte-E	N	0.042	833.645	1.13
unstructured search	faa	faa	5	1	1,673	684	1,000	IonQ Aria	N	0.06	499.748	–
unstructured search	faa	faa	5	1	1,673	684	1,000	IonQ Forte-E	N	0.034	827.179	1.13
unstructured search	faa	faa	6	1	976	432	1,000	IonQ Aria	N	0.099	316.269	–
unstructured search	faa	faa	6	1	986	432	1,000	IonQ Aria	N	0.081	309.556	–
unstructured search	faa	faa	6	1	986	432	1,000	IonQ Aria	N	0.085	317.475	–
unstructured search	faa	faa	6	1	986	432	1,000	IonQ Aria	N	0.097	317.27	–
unstructured search	faa	faa	6	1	986	432	1,000	IonQ Aria	N	0.1	314.807	–
unstructured search	faa	faa	6	1	986	432	1,000	IonQ Aria	N	0.102	316.828	–
unstructured search	faa	faa	6	1	986	432	1,000	IonQ Aria	N	0.104	318.485	–
unstructured search	faa	faa	6	1	996	432	1,000	IonQ Aria	N	0.072	317.116	–
unstructured search	faa	faa	6	1	996	432	1,000	IonQ Aria	N	0.078	316.072	–
unstructured search	faa	faa	6	1	996	432	1,000	IonQ Aria	N	0.08	317.22	–
unstructured search	faa	faa	6	1	996	432	1,000	IonQ Aria	N	0.081	314.509	–
unstructured search	faa	faa	6	1	996	432	1,000	IonQ Aria	N	0.082	317.165	–
unstructured search	faa	faa	6	1	996	432	1,000	IonQ Aria	N	0.086	309.959	–
unstructured search	faa	faa	6	1	996	432	1,000	IonQ Aria	N	0.092	316.989	–
unstructured search	faa	faa	6	1	996	432	1,000	IonQ Aria	N	0.095	317.885	–
unstructured search	faa	faa	6	1	996	432	1,000	IonQ Aria	N	0.097	314.681	–
unstructured search	faa	faa	6	1	996	432	1,000	IonQ Aria	N	0.1	315.368	–
unstructured search	faa	faa	6	1	996	432	1,000	IonQ Aria	N	0.1	320.297	–
unstructured search	faa	faa	6	1	996	432	1,000	IonQ Aria	N	0.1	320.684	–
unstructured search	faa	faa	6	1	996	432	1,000	IonQ Aria	N	0.101	315.579	–
unstructured search	faa	faa	6	1	996	432	1,000	IonQ Aria	N	0.105	314.669	–
unstructured search	faa	faa	6	1	996	432	1,000	IonQ Aria	N	0.107	316.443	–
unstructured search	faa	faa	6	1	1,006	432	1,000	IonQ Aria	N	0.067	316.728	–
unstructured search	faa	faa	6	1	1,006	432	1,000	IonQ Aria	N	0.071	313.012	–
unstructured search	faa	faa	6	1	1,006	432	1,000	IonQ Aria	N	0.078	313.039	–
unstructured search	faa	faa	6	1	1,006	432	1,000	IonQ Aria	N	0.079	316.126	–
unstructured search	faa	faa	6	1	1,006	432	1,000	IonQ Aria	N	0.079	318.574	–
unstructured search	faa	faa	6	1	1,006	432	1,000	IonQ Aria	N	0.08	318.404	–
unstructured search	faa	faa	6	1	1,006	432	1,000	IonQ Aria	N	0.082	314.59	–
unstructured search	faa	faa	6	1	1,006	432	1,000	IonQ Aria	N	0.082	317.948	–
unstructured search	faa	faa	6	1	1,006	432	1,000	IonQ Aria	N	0.083	316.014	–
unstructured search	faa	faa	6	1	1,006	432	1,000	IonQ Aria	N	0.084	316.749	–
unstructured search	faa	faa	6	1	1,006	432	1,000	IonQ Aria	N	0.085	317.795	–
unstructured search	faa	faa	6	1	1,006	432	1,000	IonQ Aria	N	0.085	318.875	–
unstructured search	faa	faa	6	1	1,006	432	1,000	IonQ Aria	N	0.086	319.802	–
unstructured search	faa	faa	6	1	1,006	432	1,000	IonQ Aria	N	0.088	317.874	–
unstructured search	faa	faa	6	1	1,006	432	1,000	IonQ Aria	N	0.097	314.185	–
unstructured search	faa	faa	6	1	1,006	432	1,000	IonQ Aria	N	0.101	317.616	–
unstructured search	faa	faa	6	1	1,006	432	1,000	IonQ Aria	N	0.102	318.653	–

Domain	Problem	Algorithm	#q	#qc	#1q	#2q	Shots	Backend	EM?	Score	Exec. Time (s)	Energy (kWh)
unstructured search	faa	faa	6	1	1,006	432	1,000	IonQ Aria	N	0.106	316.995	–
unstructured search	faa	faa	6	1	1,006	432	1,000	IonQ Aria	N	0.107	316.918	–
unstructured search	faa	faa	6	1	1,006	432	1,000	IonQ Aria	N	0.118	318.627	–
unstructured search	faa	faa	6	1	1,016	432	1,000	IonQ Aria	N	0.07	319.763	–
unstructured search	faa	faa	6	1	1,016	432	1,000	IonQ Aria	N	0.071	319.819	–
unstructured search	faa	faa	6	1	1,016	432	1,000	IonQ Aria	N	0.075	313.121	–
unstructured search	faa	faa	6	1	1,016	432	1,000	IonQ Aria	N	0.084	309.759	–
unstructured search	faa	faa	6	1	1,016	432	1,000	IonQ Aria	N	0.085	319.232	–
unstructured search	faa	faa	6	1	1,016	432	1,000	IonQ Aria	N	0.086	317.506	–
unstructured search	faa	faa	6	1	1,016	432	1,000	IonQ Aria	N	0.086	319.096	–
unstructured search	faa	faa	6	1	1,016	432	1,000	IonQ Aria	N	0.088	318.042	–
unstructured search	faa	faa	6	1	1,016	432	1,000	IonQ Aria	N	0.095	319.243	–
unstructured search	faa	faa	6	1	1,016	432	1,000	IonQ Aria	N	0.096	317.516	–
unstructured search	faa	faa	6	1	1,016	432	1,000	IonQ Aria	N	0.097	316.137	–
unstructured search	faa	faa	6	1	1,016	432	1,000	IonQ Aria	N	0.101	320.823	–
unstructured search	faa	faa	6	1	1,016	432	1,000	IonQ Aria	N	0.103	317.61	–
unstructured search	faa	faa	6	1	1,016	432	1,000	IonQ Aria	N	0.116	317.448	–
unstructured search	faa	faa	6	1	1,016	432	1,000	IonQ Aria	N	0.12	317.929	–
unstructured search	faa	faa	6	1	1,026	432	1,000	IonQ Aria	N	0.074	318.196	–
unstructured search	faa	faa	6	1	1,026	432	1,000	IonQ Aria	N	0.08	314.676	–
unstructured search	faa	faa	6	1	1,026	432	1,000	IonQ Aria	N	0.081	318.427	–
unstructured search	faa	faa	6	1	1,026	432	1,000	IonQ Aria	N	0.092	316.512	–
unstructured search	faa	faa	6	1	1,026	432	1,000	IonQ Aria	N	0.1	319.067	–
unstructured search	faa	faa	6	1	1,026	432	1,000	IonQ Aria	N	0.101	314.138	–
unstructured search	faa	faa	6	1	1,036	432	1,000	IonQ Aria	N	0.096	318.306	–
unstructured search	faa	faa	6	1	2,066	912	1,000	IonQ Aria	N	0.045	596.294	–
unstructured search	faa	faa	6	1	2,086	912	1,000	IonQ Aria	N	0.029	590.846	–
unstructured search	faa	faa	6	1	2,086	912	1,000	IonQ Aria	N	0.052	599.976	–
unstructured search	faa	faa	6	1	2,086	912	1,000	IonQ Aria	N	0.053	594.24	–
unstructured search	faa	faa	6	1	2,086	912	1,000	IonQ Aria	N	0.056	595.721	–
unstructured search	faa	faa	6	1	2,086	912	1,000	IonQ Aria	N	0.056	600.404	–
unstructured search	faa	faa	6	1	2,086	912	1,000	IonQ Aria	N	0.058	598.241	–
unstructured search	faa	faa	6	1	2,106	912	1,000	IonQ Aria	N	0.04	598.533	–
unstructured search	faa	faa	6	1	2,106	912	1,000	IonQ Aria	N	0.043	591.967	–
unstructured search	faa	faa	6	1	2,106	912	1,000	IonQ Aria	N	0.044	596.055	–
unstructured search	faa	faa	6	1	2,106	912	1,000	IonQ Aria	N	0.044	596.697	–
unstructured search	faa	faa	6	1	2,106	912	1,000	IonQ Aria	N	0.044	598.804	–
unstructured search	faa	faa	6	1	2,106	912	1,000	IonQ Aria	N	0.048	586.151	–
unstructured search	faa	faa	6	1	2,106	912	1,000	IonQ Aria	N	0.048	597.45	–
unstructured search	faa	faa	6	1	2,106	912	1,000	IonQ Aria	N	0.048	599.57	–
unstructured search	faa	faa	6	1	2,106	912	1,000	IonQ Aria	N	0.05	605.99	–
unstructured search	faa	faa	6	1	2,106	912	1,000	IonQ Aria	N	0.051	593.978	–
unstructured search	faa	faa	6	1	2,106	912	1,000	IonQ Aria	N	0.053	594.94	–
unstructured search	faa	faa	6	1	2,106	912	1,000	IonQ Aria	N	0.055	601.315	–
unstructured search	faa	faa	6	1	2,106	912	1,000	IonQ Aria	N	0.056	598.074	–
unstructured search	faa	faa	6	1	2,106	912	1,000	IonQ Aria	N	0.058	598.756	–
unstructured search	faa	faa	6	1	2,106	912	1,000	IonQ Aria	N	0.063	599.004	–
unstructured search	faa	faa	6	1	2,126	912	1,000	IonQ Aria	N	0.026	600.357	–
unstructured search	faa	faa	6	1	2,126	912	1,000	IonQ Aria	N	0.035	595.171	–
unstructured search	faa	faa	6	1	2,126	912	1,000	IonQ Aria	N	0.038	597.222	–
unstructured search	faa	faa	6	1	2,126	912	1,000	IonQ Aria	N	0.04	594.633	–
unstructured search	faa	faa	6	1	2,126	912	1,000	IonQ Aria	N	0.04	601.733	–
unstructured search	faa	faa	6	1	2,126	912	1,000	IonQ Aria	N	0.041	599.379	–
unstructured search	faa	faa	6	1	2,126	912	1,000	IonQ Aria	N	0.042	594.301	–
unstructured search	faa	faa	6	1	2,126	912	1,000	IonQ Aria	N	0.044	604.134	–
unstructured search	faa	faa	6	1	2,126	912	1,000	IonQ Aria	N	0.046	594.335	–
unstructured search	faa	faa	6	1	2,126	912	1,000	IonQ Aria	N	0.047	594.987	–
unstructured search	faa	faa	6	1	2,126	912	1,000	IonQ Aria	N	0.047	599.61	–
unstructured search	faa	faa	6	1	2,126	912	1,000	IonQ Aria	N	0.047	602.643	–
unstructured search	faa	faa	6	1	2,126	912	1,000	IonQ Aria	N	0.048	595.279	–
unstructured search	faa	faa	6	1	2,126	912	1,000	IonQ Aria	N	0.049	602.18	–
unstructured search	faa	faa	6	1	2,126	912	1,000	IonQ Aria	N	0.052	597.513	–
unstructured search	faa	faa	6	1	2,126	912	1,000	IonQ Aria	N	0.053	596.227	–
unstructured search	faa	faa	6	1	2,126	912	1,000	IonQ Aria	N	0.053	601.927	–
unstructured search	faa	faa	6	1	2,126	912	1,000	IonQ Aria	N	0.055	600.405	–
unstructured search	faa	faa	6	1	2,126	912	1,000	IonQ Aria	N	0.057	599.632	–
unstructured search	faa	faa	6	1	2,126	912	1,000	IonQ Aria	N	0.057	600.603	–
unstructured search	faa	faa	6	1	2,146	912	1,000	IonQ Aria	N	0.035	596.134	–
unstructured search	faa	faa	6	1	2,146	912	1,000	IonQ Aria	N	0.037	595.144	–
unstructured search	faa	faa	6	1	2,146	912	1,000	IonQ Aria	N	0.038	592.483	–
unstructured search	faa	faa	6	1	2,146	912	1,000	IonQ Aria	N	0.043	597.672	–
unstructured search	faa	faa	6	1	2,146	912	1,000	IonQ Aria	N	0.045	598.867	–
unstructured search	faa	faa	6	1	2,146	912	1,000	IonQ Aria	N	0.045	600.738	–
unstructured search	faa	faa	6	1	2,146	912	1,000	IonQ Aria	N	0.046	598.723	–
unstructured search	faa	faa	6	1	2,146	912	1,000	IonQ Aria	N	0.047	595.052	–
unstructured search	faa	faa	6	1	2,146	912	1,000	IonQ Aria	N	0.049	594.945	–
unstructured search	faa	faa	6	1	2,146	912	1,000	IonQ Aria	N	0.051	600.086	–
unstructured search	faa	faa	6	1	2,146	912	1,000	IonQ Aria	N	0.052	599.37	–
unstructured search	faa	faa	6	1	2,146	912	1,000	IonQ Aria	N	0.055	595.918	–
unstructured search	faa	faa	6	1	2,146	912	1,000	IonQ Aria	N	0.056	603.546	–

Domain	Problem	Algorithm	#q	#qc	#1q	#2q	Shots	Backend	EM?	Score	Exec. Time (s)	Energy (kWh)
unstructured search	faa	faa	6	1	2,146	912	1,000	IonQ Aria	N	0.058	599.946	–
unstructured search	faa	faa	6	1	2,146	912	1,000	IonQ Aria	N	0.059	600.803	–
unstructured search	faa	faa	6	1	2,166	912	1,000	IonQ Aria	N	0.038	600.81	–
unstructured search	faa	faa	6	1	2,166	912	1,000	IonQ Aria	N	0.042	594.795	–
unstructured search	faa	faa	6	1	2,166	912	1,000	IonQ Aria	N	0.045	599.712	–
unstructured search	faa	faa	6	1	2,166	912	1,000	IonQ Aria	N	0.047	607.013	–
unstructured search	faa	faa	6	1	2,166	912	1,000	IonQ Aria	N	0.052	597.946	–
unstructured search	faa	faa	6	1	2,166	912	1,000	IonQ Aria	N	0.068	598.842	–
unstructured search	faa	faa	6	1	2,186	912	1,000	IonQ Aria	N	0.043	602.782	–