Implications of Noise & Sustainability in Small-Scale Quantum Computing

Amy Bui
Computer Science
Tufts University
Somerville, MA, USA
amy.bui@tufts.edu

Jessica Nordlund
Electrical & Computer Engineering
Tufts University
Somerville, MA, USA
jessica.nordlund@tufts.edu

Matthew Steinborn
Electrical & Computer Engineering
Tufts University
Somerville, MA, USA
matthew.steinborn@tufts.edu

***Abstract*—This work examines the current state of Noisy Intermediate-Scale Quantum (NISQ) computation, specifically in regards to noise and sustainability. In this exercise, two simple circuits (Superdense Coding Protocol and Bit Flip Error Correction) were implemented in a high level quantum programming language and submitted onto available hardware target in Azure's quantum portal. These circuits' outputs were then used to profile the noise through the output probability densities and state fidelities in order to characterize how much the circuits intended operation was compromised due to noise intrinsic in quantum computing hardware.**

After capturing the noise characteristics, we created a model to estimate the overall power consumption required to create a algorithm that meets various error thresholds. Overall, we find that tighter error tolerances result in circuits which expend more energy due to the increased complexity required to introduce error correction methods.

I. Introduction & Motivations

With the incredibly shrinking scale of transistors to the nanoscale where quantum mechanical properties start interfering with functionality, quantum computers (QC) are poised to become the next great leap in computational devices. This technology looks to exploit the very quantum mechanical properties impeding the growth in Moore's Law. By isolating a quantum system, QC would transform states of quantum particles called qubits in order to store information and compute data. These methods theoretically allow QCs to solve problems not feasibly solved by classical computers more efficiently. While the idea of quantum computers was conceived in 1982 by physicist Richard Feynman (for the original purpose of simulating quantum mechanics), little advancements were made on actual QCs for a long-time due to limitations in hardware and research. However, with more recent advances in quantum hardware and availability of several quantum computing platforms, we are now in the what is called the Noisy Intermediate-Scale Quantum (NISQ) era of quantum computing. While the software development side of quantum research has outpaced hardware development for a while, that gap will soon start closing.

NISQ computers are still highly limited by their susceptibility to noise, which is defined as any unintended disturbances to the quantum system used for computation that can negatively impact results and correctness. Research into characterizing noise and how it impacts quantum systems is an active area of research; however, given how recent the new hardware developments are, there are still no particular guidelines or standard on how best to characterize and measure noise in a QC, just as there is no standard methodology to test and benchmark QCs themselves. Nonetheless, noise modeling is essential to the development of quantum hardware as it should be accounted for with any QC measurements taken.

II. Background & Related Works

A. Quantum Computing

Although quantum computing theory has been around for decades, it is only recently that advances in physics and chemistry have allowed physical quantum devices to be realized, with the first quantum computers being built at the beginning of the millennium [1]. Because of quantum computing's recent nascency, the field is still working towards proving quantum supremacy and implementing useful algorithms. More specifically, the current NISQ era is characterized by two large and conflicting problems: noise and scale [2].

Current quantum computers are too noisy, prone to error, and unreliable. Algorithms exist to help correct this behavior, but they require significantly more qubits than even the most advanced hardware can provide. The following subsections give insight into the different types of quantum hardware and common system representations.

1) Quantum System Representation: The Quantum Circuit Model describes the state of a quantum system ψ, an n-qubit system, as the following *probability distribution* [2]

$$|\psi\rangle = \sum_{b\in\{0,1\}^n} \alpha_b |b\rangle \tag{1}$$

where α_b is the amplitude of basis bit-string $|b\rangle$, and $|b\rangle$ is observed with probability $|\alpha_b|^2$. The probability distribution p_i after measuring one quantum state $|\psi_i\rangle = \sum_i \alpha_i |x_i\rangle$ is represented by $\{|x_i\rangle\}_i$.

The probability distribution of all the bit-strings is also called *superposition*. For example, the superposition state of one qubit is $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$. Here, when the outcomes are equal for states $|0\rangle$ and $|1\rangle$, then $\alpha = \beta = \frac{1}{\sqrt{2}}$.

A noisy quantum state is modeled by a probability distribution over pure quantum states [2]:

$$\{p_i, |\psi_i\rangle\}_i \tag{2}$$

and the *mixed state* a of a noisy quantum system ρ is given by the *density matrix representation* of a state [2]:

$$\rho = \sum_i p_i |\psi_i\rangle\langle\psi_i| \tag{3}$$

The following is an example of one gate operation that is used in the experiment; other transformations are similarly represented. The *Hadamard Transformation* is a single-qubit quantum gate and transforms states as the following:

$$H = \begin{cases} |0\rangle \mapsto \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle, \\ |1\rangle \mapsto \frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle \end{cases} \tag{4}$$

2) Quantum Hardware: There are many ways to make a qubit and define its state, with the two most popular methods today being the trapped ion qubit, in which the energy levels of atomic ions are used as the quantum states, and the superconducting qubit, in which the energy levels of a atom-like structure simulated on an IC are used as the quantum states [2]. In both cases, though, the quantum states of the qubits are very fragile and susceptible to unwanted change. In order for the quantum physics to work, these quantum computers need to be kept at just a few tens of mK using a *dilution refrigerator* [3]. Additionally, quantum circuits often have traditional CPUs attached, which are in charge of running the quantum circuit and interpreting the results [2].

B. Noise

A quantum system inherently always involves some incoherent processes in order for it to work. Useful perturbations in quantum computing are gate operations and measurement of the quantum system. However, a quantum system is also susceptible to random, unintended disturbances beyond the users control. Just as how measurements and gate operations cause quantum states to change and even cause decoherence, so do these other disturbances, or *noise*. No quantum system is completely isolated and pure, so quantum computing results must take into account the effects of noise. Noise itself is inherently difficult to characterize and model. It changes based on conditions, times, and temperatures; noise does not scale between large and small quantum computers because they are different systems, and often noise requires separate physical experimenting in order to model [2] [4].

Dedicated research for noise modeling is done in order to better estimate effects of noise in a system [2] [5] [6] [7]. However, these are currently only applicable for quantum computers because quantum computing simulators cannot efficiently and accurately characterize actual noise that would be encountered. In the following, different types of noises are described and we provide brief descriptions of noise models and detection schemes. These are different from noise mitigation and error correction techniques, which are not the primary focus of this work.

1) Types of Noise:

- **Environment** - A quantum system is susceptible to many environmental factors, including temperature, radiation, magnetic fields, and even Earth's rotation [2] [4]. Interactions with the environment can lead to unitary rotations of quantum states, decoherence, and collapse.
- **Entanglement** - When qubits interact in an unintended or unspecified way, i.e. *entanglement* or *cross-talk*, this can lead to a mixing of quantum states, qubit decoherence, and loss of state information required for final measurements. Such cross-talk can occur between logical qubits and logical qubits and ancilla qubits used for control and error correction. Unintended cross-talk can happen as a result of desired entanglements, such as during operations or measurements, as well as during the qubit preparation stage or while qubits are idle.
- **Imprecise Operations** - Gates that are imprecisely calibrated can rotate qubits in an undesired way. Such changes, again, lead to incorrect state measurements and ultimately influences how the applied quantum error correction (QEC) works.
- **Leakages & Seepages** - Leakage and seepage are occurrences when qubit states change in and out of computational subspace. Quantum computing assumes qubits can occupy only two states, $|0\rangle$ and $|1\rangle$, or computational subspace. But there are actually more than two states they could occupy [4]. Noise can cause qubits to leak out of subspace and seep back into subspace. This back and forth also creates noise seen in QC output.

2) Relevant Noise Models and Detection Schemes:

- **Pauli and Clifford Noise Models** - Pauli and Clifford channels refer to operators that act on quantum states, but themselves can be a source of error in the output, i.e. Pauli and Clifford noise. Pauli operators P include {identity (*I*), not (*X*), Y, and Z gates}, and Clifford operators C include {Hadamard (*H*), phase (*S*), and *CNOT* gates}. To model these noises, a gate from a set of Pauli and Clifford channels is randomly inserted into the circuit at some probability. These models are used as a way to characterize some interactions with the environment and unintended state measurements [2] [4] [8] [9].
- **Cross-Talk Detection** - *Harper, et al.* describe a method for detecting and characterizing cross-talk. Sequences of random gates of length m are selected, the probability distribution of measured outcomes are estimated, the Walsh-Hadamard transform of the estimates are used to learn the decay constants, and these values are transformed again using the Walsh-Hadamard transform to yield the list of effective error rates. Their procedure constructs a quantum noise correlation matrix for qubit-qubit interactions, which visualizes relationships between pairs of qubits and give an estimate of the probability distribution of the average noise in the system [10].

- **Leakage Detection** - *Bultink, et al.* and *Varbanov, et al.* each came up with similar procedures for detecting leakages and modifying outputs to account for such [11] [12]. Their protocols involve taking repeated measurements of the quantum system. Then, using a statistical model called the Hidden Markov model (HMM), they can output where leakages are detected. HMM is a tool that has applications in many areas, including computational biology, machine learning, language processing, etc. When computing the probability of some event, such as the processes taken in quantum computation, there are "hidden" or unobserved events or parameters (i.e. noise) that would influence observable ones (i.e. state preparation, gate operations, measurements, etc.). Since regular QEC methods don't handle qubit leakages, these leakage detection methods can point to leakages in the data, for which they may be corrected for or discarded in the output.

C. Fidelity

Fidelity is described as how close two distributions are to one another. In terms of quantum computing, since quantum systems are represented as probability distributions/density matrix, fidelity can be used to describe how similar a *noisy* system is to a pure, *noiseless* system. Fidelity between two quantum systems is defined slightly differently depending on the author, but we provide two here [6] [13]:

$$F(\rho,\sigma) = \mathbf{tr}\left[\sqrt{\sqrt{\rho}\sigma\sqrt{\rho}}\right]^2 \tag{5}$$

and

$$F(\rho,\sigma) = \mathbf{tr}\left[\sqrt{\sqrt{\rho}\sigma\sqrt{\rho}}\right] \tag{6}$$

where ρ is the "pure" target system and σ is the actual system measured. Fidelity is defined using the trace, **tr**, of a matrix, or sum of the diagonal elements. When $F = 1$, the two systems are said to be identical.

Stemming from fidelity, *process fidelity* is a measure of how close operations in a noise system are to those in the target system. The operation $\mathcal{G}$ maybe described using unitary matrix G, where the mapping of density matrix ρ is [7]:

$$\mathcal{G} : \rho \rightarrow G\rho G^{\dagger} \tag{7}$$

If $\mathcal{G}$ is the noiseless operation and $\tilde{\mathcal{G}}$ is the noisy counterpart, process fidelity is defined by this trace [4] [7]:

$$F(\mathcal{G}(\rho), \tilde{\mathcal{G}}(\rho)) = \mathbf{tr}\left[\mathcal{G}(\rho)\tilde{\mathcal{G}}(\rho)\right] \tag{8}$$

State fidelity, similarly defined for two quantum states, is given by equation 5. This is how the Qiskit SDK for QC computes state fidelity[1]. When one of the state is a pure state, fidelity is given by

$$F(\rho,\sigma) = \langle\psi_1|\sigma|\psi_1\rangle, \text{ where } \rho = |\psi_1\rangle\langle\psi_1| \tag{9}$$

Fidelities are not the only quantum benchmarking metrics, nor are they considered the best all around. For purposes of this work, fidelity will be a means to discuss and quantify how close two systems are to one another, as well as visualize their differences.

[1] https://qiskit.org/documentation/stubs/qiskit.quantum_info.state_fidelity.html

D. Sustainability

When quantifying the environmental impact of computing (traditional or quantum), there are two main categories of resource consumption: operational and hardware manufacturing and infrastructure [14]. For the current NISQ era, quantifying the impact of quantum hardware manufacturing is extremely difficult. As discussed before, there are many different ways to create a qubit, and these devices are not being mass produced, resulting in a lack of data for this particular step in the life-cycle. While it will be important to consider manufacturing costs of quantum devices in the future, it is out of scope for this project.

In terms of operational energy, commercial quantum computing has two stages:

1) **The Queue** - Commercial quantum computers require users to queue their jobs on a data server, and wait times can be on the order of seconds or weeks, depending on the popularity of the computer. Determining the amount of energy it takes for a job to be stored in the cloud requires knowledge of the job size (how many bytes are stored on the server) and the wait time. Using power metrics of data servers in terms of bytes and time (W/byte/year), the queue energy can be calculated using the following equation.

$$E_{queue} = P_{queue} * time_{queue} * size \tag{10}$$

2) **The Computation** - The energy required for an actual quantum computation includes the energy used by the quantum circuit itself and the energy required to keep the qubits stable. More specifically, the total energy consumption can be modeled by multiplying the time spent on the qunatum processing unit (QPU) by the sum of the power of the quantum circuit and the power of the dilution refrigerator used to keep the qubits near absolute zero.

$$E_{comp} = time_{QPU} * (P_{QPU} + P_{cooling}) \tag{11}$$

By combining the energy requirements of the queue and the computation, a model of total energy can be created.

$$E_{total} = E_{queue} + E_{QPU} \tag{12}$$

It is also important to note that a full life-cycle analysis of the sustainability of quantum computing would include facility requirements, in addition to the manufacturing requirements. These aspects are out of scope for this project, though.

III. Experimental Design

The experiments performed in this study sought to capture the noise characteristics when running on real hardware as well as characterize the resources required to run quantum algorithms – in order to forecast their impact on the systmems overall sustainability. In order to capture the noise profile of real quantum hardware, some simple circuits were created, again using a high level quantum programming language, and executed on real quantum hardware. To better understand the physical requirements for real quantum algorithms, Microsoft's Resource Estimator tool was used, which allows for compile time analysis of a quantum algorithm, written in Q# or another high level language

A. Noise Characterization on Real Quantum Hardware

As previously mentioned, the first step in the experiment performed was running simple quantum circuits on real hardware in order to characterize the impact of noise on quantum software performance.

Circuits Implemented: The circuits created must remain lightweight in terms of complexity (e.g. number of gates and qubits), in order to meet the limited resource availability on modern NISQ hardware. Therefore, we looked towards existing simple quantum circuits that have been proposed for legitimate purposes but do not have the overhead of an algorithm like Grover's or other more complex circuit implementations. Another requirement we had when generating these circuits was that they produce a determinate output; many quantum algorithms generate the correct response with a very high probability, however in order to separate the noise from the intrinsic probabilities featured in quantum circuit design, we elected to avoid these types of circuits.

With the above caveats in mind, the first circuit created was the Superdense Coding Protocol [15], which is a proposed quantum circuit to be used for communication. The motivation behind this circuit (Figure 1) is to provide a secure method of communication through quantum entanglement; eavesdroppers will require all three qubits in order to decode the message, even as quantum gates are applied in intermediary stages. More importantly for this exercise, however, is that the circuit only requires six quantum gates (we add a seventh in the middle to make the quantum communication slightly less arbitrary), three qubits, and results in a constant output. This implementation can be seen in Fig. 1.

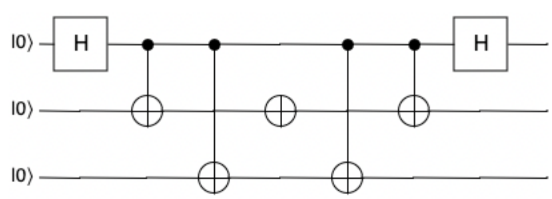


Fig. 1: Implementation of the Superdense Coding Protocol in Quirc

The next circuit created was the single bit-flip code circuit; this circuit is proposed as an error mitigation method, wherein the ancillary qubits can detect a bit flip in any of the three qubits above [16]. This is done through entanglement as the ancillary qubits will have their values flipped as well upon an incorrect bit flip. Based on the combination of possible outputs the original three bits can be reconstructed as well, making this circuit not just capable of error detection, but also error correction. Again this circuit produces only one output in a non-noisy environment ([0,0] which corresponds to no error in the circuits truth table). The implementation can be seen in Fig. 2.

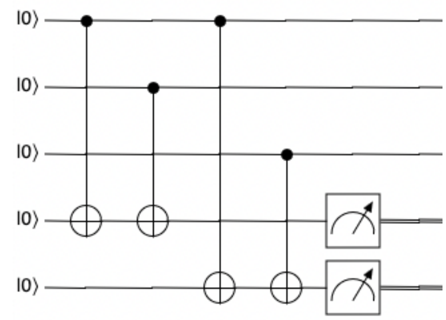


Fig. 2: Implementation of the Bit Flip Correction Circuit in Quirc

Available Targets: The intention in creating both of these quantum circuits in Q# is to submit them as jobs to different quantum targets. Azure's portal provides a number of quantum computers that can be targeted, and we selected two of these for this investigation. The primary selection criteria was queue time, which for some targets could be upwards of three weeks. In order to generate data in a timely manner (and allow for retrials should the operation not work as intended), we identified the IonQ Harmony and Rigetti Aspen M2 as ideal targets. Their qualities are as follows:

1) Rigetti Aspen M2: This quantum computer contains 80 qubits which are of the superconducting variety. Their online resources claim 40-240ns gate times, and 20uS coherence. Their listed worst case single gate error rate is 7.2% for a controlled Z operation.
2) IonQ Harmony: This QC hardware contains much fewer qubits – 11 in total. However, this hardware has much higher fidelity, which IonQ credits due to it's usage of trapped ion qubits which intrinsically have lower error rates; the largest gate error is 4% for the same CZ operation.

B. Resource Estimation

As previously mentioned, Azure's quantum computing portal provides a resource estimation tool that generates a number of predicted hardware requirements to run a provided quantum program. This tool actually provides estimates for fault tolerant implementations; however, fault tolerance is outside of the scope of this survey since they are not supported by the currently available NISQ hardware. These fault tolerant parameters are easily removed from the estimators output, so instead this survey looks at the resources required to implement the algorithm itself along with error correction methods. The program implemented on this hardware for this examination is Grover's Quantum Search Algorithm (Fig. 3). Simply put, this algorithm seeks to find the one input to a quantum oracle that will output a "one"; all other bit values will cause the quantum oracle to generate a "zero". This can be more easily understood as a password determining algorithm in which the quantum oracle implements the cryptographic hash used to obfuscate the password entered. For a more in depth survey of the implementation and implications of Grover's algorithm one should refer to the literature [17] [18].

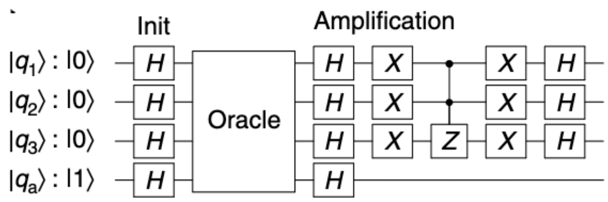


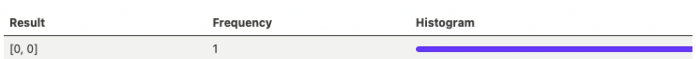
Fig. 3: Circuit Implementation of Grover's Search; from [18]

The circuit in Fig. 3 was implemented in Q# using a bitwise XOR as the quantum oracle. Using this circuit, and submitting it to the resource estimator, we are able to generate the hardware requirements for different types of quantum hardware (e.g. different qubit parameters), as well as varying error tolerances. These estimations provide valuable input to the sustainability models, and the results of this are analyzed and contextualize in Section IV.

IV. Results & Analysis

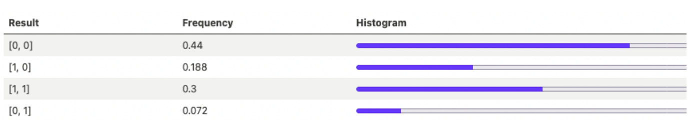
A. Noise Profiling and Fidelity

The outputs of the implemented quantum circuits – which were run on real quantum hardware 500 times – were captured. Histograms showcasing the results of each operations expected value and what we measured from the results of the quantum hardware were created. These results can be seen in Fig. 4 - 7.



Result	Frequency	Histogram
[0, 0]	1	

Fig. 4: Histogram of Simulated Results From Riggeti: Error Correction Circuit



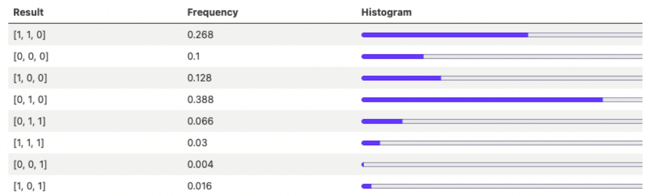
Result	Frequency	Histogram
[0, 0]	0.44	
[1, 0]	0.188	
[1, 1]	0.3	
[0, 1]	0.072	

Fig. 5: Histogram of Real Results from Rigetti Hardware: Error Correction Circuit

Result	Frequency	Histogram
[0, 1, 0]	1	

Fig. 6: Histogram of Simulated Results From Riggeti: Superdense Coding Circuit

What is immediately clear from the results above is that although the circuits were designed to have a completely deterministic outcome, the results showcase a broad range of measurement outcomes. The circuits themselves see slightly different output inconsistencies as well. The error correction circuit features a 44% chance of measuring the correct option while the super dense coding circuit shows a 38% chance of measuring the correct output. While this disparity might present the error correction circuit as "more reliable", with it returning the correct result more frequently, further analysis is required.



Result	Frequency	Histogram
[1, 1, 0]	0.268	
[0, 0, 0]	0.1	
[1, 0, 0]	0.128	
[0, 1, 0]	0.388	
[0, 1, 1]	0.066	
[1, 1, 1]	0.03	
[0, 0, 1]	0.004	
[1, 0, 1]	0.016	

Fig. 7: Histogram of Real Results from Rigetti Hardware: Superdense Coding Circuit

Table 1 showcases the state fidelities of the two created circuits, which was generated following the methodology discussed previously. From this table, it is clear that although the correct output probability is higher in the error correction circuit, the probability distribution for the superdense coding protocol is actually closer to it's simulated results.

Fidelity Results		
Circuit Implemented	Hardware Type	State Fidelity
Bit Flip Detection	Aspen M2	0.194
Superdense Coding	Aspen M2	0.292

TABLE 1: Histogram of Simulated Results From Riggeti: Superdense Coding Circuit

One can understand why the superdense coding fidelity is higher conceptually by considering the fact that the circuit relies on 3 qubit measurement, while the bit-flip code only requires two. This means that while the bit flip circuit has 4 possible states, the superdense circuit has 8. Therefore, although the overall probability of correct measurement is higher in the error correction circuit, the superdense probability is higher relative to the possible number of measurement states. Put more bluntly, the 38% for eight states represents a higher contribution than 44% into four.

B. Resource Estimation & Sustainability

As discussed in Section 2d, the sustainability of quantum computing can be analyzed in terms of energy consumption. While our original plan was to collect experimental data for queue time and QPU time, some of our IonQ runs were unsuccessful. In order to supplement the experimental data, resource estimation was used (and denoted by a *). Additionally, the runtime provided by the interface only reported to the nearest second, which is orders of magnitude larger than the actual expected time on the QPU. More specifically, the interface reported a runtime of 1s for the Bit Flip Detection, which is known to run hundreds of times within microseconds, so the energy consumption estimate should be taken with caution.

Looking at two examples of hardware runs, a better understanding of quantum energy consumption can be attained. Table 2 contains experimental (and estimated) data for both a Bit Flip Detection and a Grover's Search run. These values were then plugged into equations 10, 11, and 12, along with the power metrics in Table 3, to calculate the total energy of the quantum system.

Algorithm	Size	Time_{queue}	Time_{QPU}	Energy
Bit Flip	3KB	1min	1s	20 kJ
Grover's	32KB	436hrs*	600us*	12J

TABLE 2: Hardware Results with Energy Estimation. Data shown for two algorithms (Bit Flip Detection and Grover's Search). Size refers to the approximate number of bytes being stored in the cloud while the QPU is busy. Energy estimations are made using the equations 10, 11, and 12. *Non-experimental data collected from a resource estimator.

Cloud Server	31.6 kWh/TB/year
QPU	5 kW
Dilution Refrigerator	15 kW
Supercomputer	2500 kW

TABLE 3: Power metrics for cloud storage [19], a 50-bit quantum computer (including classic CPU interface), a dilution refrigerator [3], and a supercomputer [1]. These values are used to calculate the total energy consumption.

From Table 2, it is clear that the amount of time spent on the QPU is the driving factor for quantum energy consumption, and the dilution refrigerator ($P_{cooling}$) is the largest contributor. Moreover, quantum computing is known for its quick runtimes in comparison to classical computing, and since supercomputers consume more power and take longer to run, it seems that quantum computing has energy supremacy. However, it is important to consider the use cases of quantum hardware and supercomputers. Firstly, the goal of quantum computing is to run algorithms that are not feasible on classical hardware, so comparison between the power consumptions is useless. Secondly, supercomputers are built to run multiple tasks at one time, so the entire power consumption cannot necessarily be contributed to a smaller program.

It is also important to think about how quantum computing will scale in the future. Right now, in the NISQ era, the quantum hardware available is in the 50 to 500 qubit range, and so far, the power requirements of the QPU and dilution refrigerator have maintained the same order of magnitude [1]. However, leaving the noisy part of the NISQ era will require quantum hardware with less error. Using the resource estimator for Grover's Search, the effect of removing error can be analyzed.

From Figure 8, it is clear that the techniques proposed for removing noise require an exponential increase in the number of physical qubits. Additionally, Figure 9 shows the time spent on the QPU also increases exponentially, which could have large impacts on energy scaling. It is hard to know how the energy requirements of future quantum hardware will change,

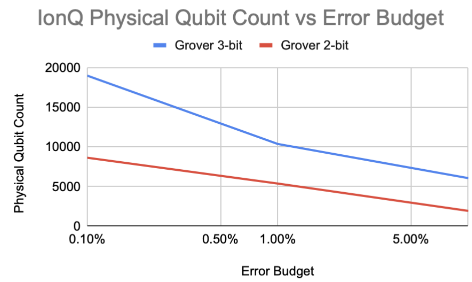


Fig. 8: Resource estimation of physical qubits required for the IonQ to complete a Grover's Search algorithm within the error budget. Error budget on a log scale.

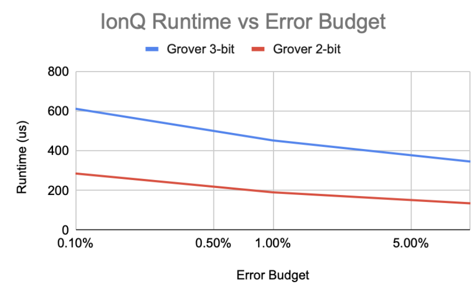


Fig. 9: Resource estimation of runtime required for the IonQ to complete a Grover's Search algorithm within the error budget. Error budget on a log scale.

especially since the hardware does not exist yet, but, moving forward, it will be extremely important to continuously factor in the energy costs of quantum computing and see how it scales.

V. Future Work

Quantum computing is still a relatively new field, and in order to benefit from the powers of quantum computing, it is essential to continue researching the noise in current quantum circuits. The excessive noise in this NISQ era makes it extremely difficult to create reliable quantum hardware. Additionally, since error correction and fault tolerant methods for implementing quantum algorithms require an exponential number of physical qubits, the future of quantum computing is uncertain. Researchers need to answer the question of how useful and larger algorithms requiring more logical qubits can be implemented, and can they be implemented sustainably. Next steps from this project would be to collect more experimental data for an array of quantum algorithms. While wait times for quantum computers can be limiting, this data set would provide more insight into the energy consumption and noise characterization of different benchmarks.

As far as direct corollaries to this work, researchers with more direct access to QC hardware (e.g. ones who have the hardware on hand or directly accessible) can attempt to profile how the physical representation of quantum circuits impact the noise characteristics. For example, if a program only uses a few qubits, designers can attempt to use different locations within the QC hardware to see if the noise is consistent throughout. Similarly researchers could look at other factors that may impact noise, such as time of day, number of jobs run concurrently (provided the hardware supports this). With this information, manufacturers could be able to create robust noise models that take into account all possible vectors for noise injection.

Another exercise could attempt to analyze the compiler's optimization in terms of how small of a binary created. This was an important part of our sustainability model, as the compiled quantum circuit was transferred via the cloud to a online job provider. Optimizing this stage could mean overall power reduction, and a more sustainable QC environment for developers. Compilation tools can also assist in error reduction, as if compilers are made aware of the underlying hardware's noise profile they can employ one of the methods discussed previously to mitigate the overall impact.

VI. Conclusion

Overall this work showcases how current NISQ hardware can introduce significant error on seemingly simple operations, which in turn has noticeable impact on energy consumption and sustainability. The circuits implemented in this survey contained only four and seven gates respectively, while measuring three or fewer qubits; despite their size, these deterministic circuits, ended up generating the correct results only 44% and 38% of the time for either circuit. Therefore, even these circuits, which are designed to generate only one possible output, must be performed a number of times in order to meet a specific error threshold.

This aforementioned error threshold then has non-negligible impacts on sustainability as the tighter tolerance requires quantum circuits that are more complicated – both in terms of gates and number of qubits – and have longer execution times. This work introduces a method to model the expected power consumption required to perform quantum execution, which is noticeably smaller than what is required for traditional methods of performing similar operations classically. This work also determined that quantum hardware may in fact be quite scalable as the current refrigeration methods can support many more qubits, and the rack energy required is a small component relative to the energy expense of cooling. However, future methods and noise removal techniques may have a large impact.

There is still much work to be done in profiling the energy and noise characteristics of quantum hardware, especially in the current NISQ environment; it is important that designers continue to try and understand the full implications of quantum hardware in the future, especially fault tolerant systems, as these will introduce a large number of additional complexity in order to create less noisy outputs.

References

[1] N. Elsayed, A. S. Maida and M. Bayoumi, "A Review of Quantum Computer Energy Efficiency," 2019 IEEE Green Technologies Conference(GreenTech), Lafayette, LA, USA, 2019, pp. 1-3, doi: 10.1109/GreenTech.2019.8767125.

[2] Y. Ding, F. Chong. Quantum Computer Systems: Research for Noisy Intermediate-Scale Quantum Computers. Springer Cham. Synthesis Lectures on Computer Architecture. 2020.

[3] Villalonga, B.; Lyakh, D.; Boixo, S.; Neven, H.; Humble, T.S.; Biswa, R.; Rieffel, E.G.; Ho, A.; Mandrà, S. Establishing the quantum supremacy frontier with a 281 pflop/s simulation. arXiv 2019, arXiv:1905.00444.

[4] S. Resch and U. R. Karpuzcu. 2021. Benchmarking Quantum Computers and the Impact of Quantum Noise. ACM Comput. Surv. 54, 7, Article 142. 2022.

[5] J. J. Wallman and Joseph Emerson. 2016. Noise tailoring for scalable quantum computation via randomized compiling. Phys. Rev. A 94, 5 (2016), 052325.

[6] M. A. Nielsen and I. L. Chuang, Quantum Computation and Quantum Information. Cambridge University Press, Cambridge. 2010.

[7] A. Erhard, J. J. Wallman, L. Postler, M. Meth, R. Stricker, E. A. Martinez, P. Schindler, T. Monz, J. Emerson, and R. Blatt. Characterizing large-scale quantum computers via cycle benchmarking. 2019.

[8] E. Magesan, D. Puzzuoli, C. E. Granade, and D. G. Cory. 2013. Modeling quantum noise for efficient testing of fault-tolerant circuits. Phys. Rev. A 87, 1 (2013), 012324.

[9] Chen, S., Liu, Y., Otten, M. et al. The learnability of Pauli noise. Springer Science and Business Media, LLC. 14. 2023.

[10] R. Harper, S. Flammia, J. Wallman. Efficient learning of quantum noise. Nature Physics. 16. 1-5. (2021).

[11] C. C. Bultink et al., Protecting quantum entanglement from leakage and qubit errors via repetitive parity measurements. Sci. Adv.6, (2020).

[12] B. M. Varbanov, F. Battistel, B. M. Tarasinski, et al. Leakage detection for a transmon-based surface code. npj Quantum Inf 6, 102 (2020).

[13] S. T. Flammia and Y. Liu. Direct fidelity estimation from few Pauli measurements. Phys. Rev. Lett. 106, 23. 2011.

[14] U. Gupta et al., "Chasing Carbon: The Elusive Environmental Footprint of Computing," in IEEE Micro, vol. 42, no. 4, pp. 37-47, 1 July-Aug. 2022, doi: 10.1109/MM.2022.3163226.

[15] Bennett CH, Wiesner SJ. Communication via one- and two-particle operators on Einstein-Podolsky-Rosen states. Phys Rev Lett. 1992 Nov 16;69(20):2881-2884. doi: 10.1103/PhysRevLett.69.2881. PMID: 10046665.

[16] Peres A. Reversible logic and quantum computers. Phys Rev A Gen Phys. 1985 Dec;32(6):3266-3276. doi: 10.1103/physreva.32.3266. PMID: 9896493.

[17] Lov K. Grover. 1996. A fast quantum mechanical algorithm for database search. In Proceedings of the twenty-eighth annual ACM symposium on Theory of Computing (STOC '96). Association for Computing Machinery, New York, NY, USA, 212–219. https://doi.org/10.1145/237814.237866

[18] Figgatt, C., Maslov, D., Landsman, K.A. et al. Complete 3-Qubit Grover search on a programmable quantum computer. Nat Commun 8, 1918 (2017). https://doi.org/10.1038/s41467-017-01904-7

[19] "Methodology Documentation," Cloud Carbon Footprint. [Online]. Available: https://www.cloudcarbonfootprint.org/docs/methodology#power-usage-effectiveness.