On Variable Strength Quantum ECC

Salonik Resch and Ulya Karpuzcu

Abstract—Quantum error correcting codes (QECC) facilitate timely detection and correction of errors to increase the robustness of qubits. Higher expected error rates necessitate stronger (i.e., larger-distance) QECC to guarantee correct operation. With increasing strength, however, QECC overhead can easily become forbidding. Based on the observation that quantum algorithms exhibit varying spatio-temporal sensitivity to noise (hence, errors), this article explores challenges and opportunities of variable strength QECC where QECC strength gets adapted to the degree of noise tolerance, to minimize QECC overhead without compromising correctness.

1 Introduction

Quantum noise is hard to model accurately as it is highly complex and can have counter-intuitive impacts [4]. Additionally, the noise present in a physical system can change over time [16], making it difficult to properly characterize it via benchmarking procedures [17].

Quantum error correcting codes (QECC) group collections of physical qubits together to represent one logical qubit (each qubit in a quantum algorithm corresponds to a logical qubit). The logical qubit is much more resilient to noise than the individual physical qubits it is made of. A key property of QECC is that it translates the arbitrary noise on physical qubits into a discrete set of noise events on logical qubits. This allows us to accurately model the quantum noise acting on logical qubits with just a few types of noise events (i.e., X and Z errors [15]).

For example, physical quantum noise can lead to slight over- or under-rotations (considering individual qubit state representation in polar coordinates) [1]. This noise is analog in nature, as the angle of over- or under- rotation can be arbitrary. QEC involves measurement, which introduces *non-unitary* transformations to the physical quantum state (while leaving the computational quantum state intact): Effectively, each measurement forces a binary decision on the impact of noise – either the qubit snaps back to the uncorrupted state, removing the noise, or a complete noise event is the case, "flipping" the state of the qubit. This digitizes noise into a set of discrete errors, enabling detection and correction.

QECC can only detect and correct errors on physical qubits if the physical noise induced error rates are sufficiently low. An excessive number of noise events on multiple physical qubits, occurring simultaneously, may result in a *logical* error which is either undetectable or uncorrectable. QECC is typically designed with the target of removing logical noise events entirely. The maximum number of simultaneous physical errors QECC can tolerate is called the *code distance* which determines the *code strength*.

Unfortunately, the cost of creating large distance QECC is very high. The number of physical qubits required for each logical qubit grows quadratically with the code distance, potentially reaching thousands of physical qubits for each logical qubit [5] even for modest error rates. The resource requirement quickly grows beyond what currently available quantum hardware can realistically support.

• *The authors are with the Department of Electrical and Computer Engineering, University of Minnesota, Minneapolis, MN 55455 USA. E-mail: {resc0059, ukarpuzc}@umn.edu.*

Manuscript received 3 July 2022; accepted 4 August 2022. Date of publication 22 August 2022; date of current version 16 September 2022.
(Corresponding author: Salonik Resch.)
Digital Object Identifier no. 10.1109/LCA.2022.3200204

In this work we explore how opportunistically reducing the code distance and thereby risking rare logical errors can help mitigate the hardware resource overhead required to build a (nearly) fault-tolerant quantum computer. This strategy can prove effective to the extent the quantum application can tolerate rare errors – i.e., can still produce the correct output with high probability. Two basic methods span the entire design space:

1) Lowering the code distance overall (for all logical qubits at all times)
2) Selectively lowering the code distance for a subset of the qubits at specific times

Method 2) is especially suitable for Surface Codes [5] where changing the number of physical qubits dedicated to each logical qubit is possible. For both cases, we use statistical fault injection (i) to accurately estimate the probability of success in the presence of errors; (ii) to differentiate more noise-sensitive regions of the application from the less noise-sensitive to allocate scarce QECC resources based on need.

2 Background

Improving the reliability of quantum programs considering physical qubit characteristics is a well studied problem. Numerous papers such as [16] explored application mapping strategies considering variation in the reliability of physical qubits, while others focused on minimizing the number of gate evaluations to reduce the exposure time to noise [3], [9]. Statistical fault injection based studies of program sensitivity to physical noise, to optimize gate scheduling and qubit placement, also exist [11]. An important distinction of our study from this lineage of work is that we operate at the logical qubit level rather than the physical. Noise can be modeled more accurately at the logical level as QECC effectively forces noise to manifest as X and Z errors. This is in stark contrast to modeling noise at the physical level, which involves many different models. Worse, each noise model can result in a different outcome [12], and the noise in a physical experiment changes over time [17]. As a result, statistical fault injection at the physical level can produce contradicting conclusions depending on the noise model used [11]. At the same time, we primarily focus on fine-tuning QECC distance according to algorithmic needs, as opposed to common noise mitigation strategies at the physical level which focus on application mapping/scheduling.

2.1 Surface Codes

Surface codes are a promising form of QECC. They logically arrange qubits into a two-dimensional lattice, and only require interactions between nearest neighbors [5], [10], allowing the qubits to remain in place. This makes them easy to use with modern quantum computers, such as superconducting quantum computers, which have stationary qubits and only allow interactions between physically adjacent qubits. Surface codes can be conceptually visualized as "patches" of physical qubits, where patches can move and interact with each other by performing operations and measurements on the qubits [10]. This is referred to as *lattice surgery*, and represents the state of the art [6], [10]. A surface code of code distance d requires d^2 physical qubits per logical qubit. Logical gates on logical qubits require roughly d time cycles [10]. Changing d for each qubit also roughly takes d cycles. To increase d: initialize more physical qubits and involve them in the next round of error correction. To decrease d: measure a subset of the physical qubits in the logical qubit, and then exclude them in the next round [5]. For both, then perform d rounds of error correction to remove any errors which occurred during the process.

2.2 Gate Decomposition

A qubit can be logically represented as a point on the unit sphere (called Bloch sphere), where operations (gates) on it become rotations around different axis. $R_x(\theta)$, $R_y(\theta)$, and $R_z(\theta)$ gates correspond to rotations around the x, y, and z axes by an arbitrary angle θ. Such rotation gates are often used when operating quantum computers without QEC, where logical gates correspond directly to physical rotations on individual qubits. The specific gate set available depends on the specific machine, but generally precise rotations around at least two axes are typically available. However, such rotations do not work directly with surface codes (or most QECC). When the physical qubits are grouped together to form a logical qubit, only a specific set of gates are possible [5]. In this work we use the Clifford+T gate set, the most widely used universal gate set which can also be used on surface codes. It includes, among others, X, Y, Z, H, S, and T gates, which all correspond to rotations by π, $\pi/2$, or $\pi/4$ around various axes. Two-qubit variations of these gates exist, where a gate is performed on a *target* qubit only if the *control* qubit is in a specific (i.e., the $|1\rangle$) state. For example, the CNOT gate is a controlled X gate. A controlled phase gate is a controlled R_z gate.

Logical versions of $R_x(\theta)$, $R_y(\theta)$, and $R_z(\theta)$ gates are required for many algorithms, hence it is still necessary to implement them. To perform them on surface codes, they are *approximated* with sequences of gates. For this, we use the *gridsynth* algorithm [13], [14], which converts $R_z(\theta)$ into sequences of X, H, S, and T gates. The length of the sequence depends on the angle of rotation, θ, and the precision to which we need to approximate it. For example, a rotation by $\pi/3$ can be approximated with Equation (1)

$$R_z(\pi/3) \approx HSTHSHTHSHTST. \tag{1}$$

There is also a trade-off between the sequence length and the achieved accuracy. Shorter sequences take less time to perform but can lead to errors in the program (even in the absence of noise) due to the higher degree of approximation. We tested different sequence lengths when performing a noiseless simulation of a benchmark quantum algorithm (which we discuss in Section 3.2). As an example, based on these experiments, we observe that for an average sequence length of 34 gates per rotation, the correct output was produced only 50% of the time. Increasing the sequence length to an average of 44 gates increased the probability of correct output to 98.5%.

Luckily, it is sufficient to approximate R_z rotations, as any quantum operation U can be decomposed into a set of three R_z rotations (using angles β, γ, and δ) and H gates

$$U = R_z(\beta)\ R_x(\gamma)\ R_z(\delta) = R_z(\beta)\ H\ R_z(\gamma)\ H\ R_z(\delta). \tag{2}$$

In our case study we use standard Clifford+T gates, however, we note that further optimizations exist to tailor the operations specifically for surface codes [10].

3 The Case for Variable Strength QECC

3.1 Latency Versus Reliability Trade-Off

Reducing the QECC distance (i.e., making QECC weaker), by construction reduces the overhead, and hence, the time it takes to complete the program. On the other hand, a smaller code distance reduces the probability of success. Most quantum algorithms, even in the absence of noise, produce the correct result with some probability. Therefore, the algorithm has to be run multiple times before the correct answer is produced statistically. The number of repetitions depends on the probability of a successful trial (PST). $1/PST$ runs are required to get the correct result, on average. Hence, *mean time to success* is a key metric of interest. If the algorithm has a latency of L, the mean time to success becomes L/PST. Let L_{weak} and PST_{weak} denote the latency and probability of success of the target algorithm using a weaker (i.e., smaller distance) QECC; where L and PST denote the latency and probability of success of the target algorithm with default strength (i.e., distance) QECC

$$\begin{aligned} L_{weak} &< L \\ PST_{weak} &< PST, \end{aligned} \tag{3}$$

applies, and a weaker QECC would only work if

$$\frac{L_{weak}}{PST_{weak}} < \frac{L}{PST}. \tag{4}$$

In the following, we quantitatively analyze this trade-off.

3.2 Sensitivity to Noise in Time and Space

The impact of a logical error on the success of a quantum program depends on when (at which cycle in the execution) and where (in which qubit) the error occurs [11]. We use *statistical fault injection* to characterize this behavior, where we inject errors at different locations at different times and track the propagation to the output of the program. Since we are working with QEC, we can assume that errors are restricted to X and Z errors (Y errors are a combination of X and Z errors). As a representative case study, we profile the Quantum Phase Estimation (QPE) algorithm, which incorporates the inverse Quantum Fourier Transform (QFT) as the main computational kernel. QPE is representative of algorithms a fault-tolerant quantum computer (i.e., a quantum computer which features QEC) is likely to run, and has important applications in quantum chemistry [8]. The input to QPE is a quantum state which has a phase angle difference between its constituent qubits. QPE detects this difference and produces a bitstring representing the angle. To produce the input quantum state, we perform noiseless state preparation. A sequence of controlled-phase gates are performed with qubits 0-4 as control and qubit 5 as the target. A noiseless QPE implementation acting on this state produces a single output bitstring with high probability ($> 98\%$). This eases validation of the output.

For statistical significance, we simulate each fault with 1,000,000 shots on Qiskit [2]. Each fault is a logical error on a single qubit at a single moment in time, leaving all other qubits undisturbed. We test with X error, Z error, and XZ error (both). To model realistic noise, we assume the error is 50% X and and 50% Z, excluding the rare case where both occur. We use this model for the remainder of the paper.

This model is realistic because if the probability of X and Z errors is p, the probability of X (without Z) and Z (without X) is $p(1-p)$, but the probability of both is p^2. For small p, $p(1-p) >> p^2$. Thus, error is predominantly either X or Z.

We evaluate the quality of the output by comparing the probability of successful trial (PST) to that of the noiseless output. As we set the input state so that there is only one correct answer, this metric is Relative PST (Equation (5))

$$Relative\ PST = \frac{PST_{noisy}}{PST_{ideal}} \tag{5}$$

Fig. 1 depicts the relative PST as a heatmap, to help visually inspect the significance of the error at each location and at each point in time. The x-axis is time; the y-axis, space (i.e., different qubits). Thick, red lines indicate sensitive regions. Some regions are more tolerant to noise than others, enabling exploitation of variable distance codes.

4 Variable Strength QEC

Knowledge about the underlying noise sensitivity of a quantum program, both in time and space, makes variable strength (i.e.,

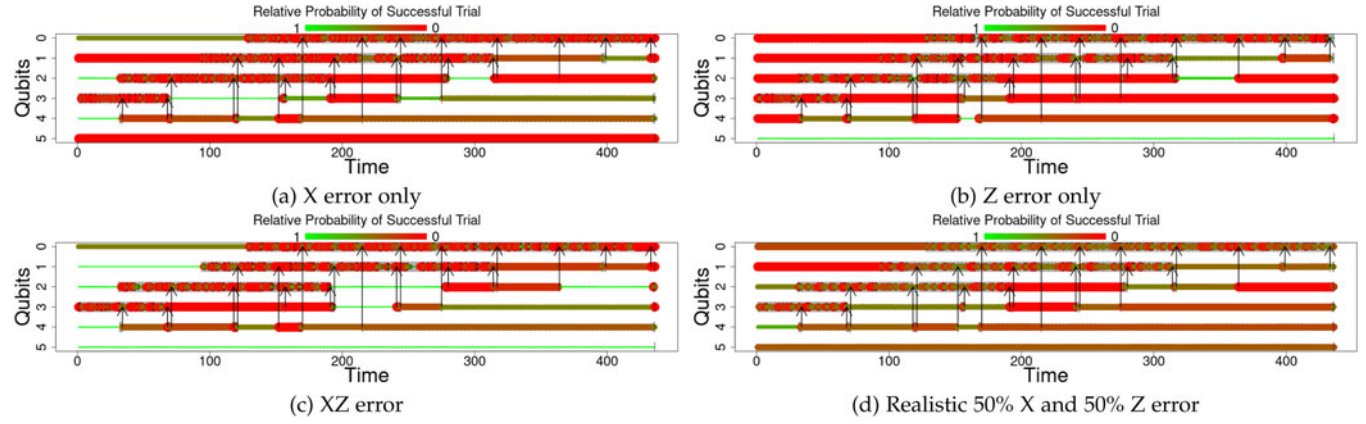

Fig. 1. Noise sensitivity heatmap of six-qubit QPE alrogithm to different logical errors. Thin green lines represent insensitive; thick red lines, sensitive regions. Vertical lines show single qubit gates. Arrows between qubits show two qubit gates. Qubit five has no operations on it after state preparation. In our noise model, an error on a two-qubit gate affects both qubits.

distance) QEC possible, where different logical qubits get protected by different levels (i.e., different code distances d) of QEC at different times. If a logical qubit is less susceptible to logical noise at a point in time, it can us lighter weight QEC than more susceptible qubits.

We start by acknowledging a fundamental limitation to this idea. The logical error rate decreases *exponentially* with d. The physical qubit count increases with d^2. The time overhead increases linearly with d. *Hence, we can save quadratic space and linear time, but risk exponential increases in error rates.* This suggests that variable-strength QEC can easily backfire if applied too aggressively.

4.1 Success Rate as a Function of Code Distance

We estimate the probability of logical error, P_L, from the physical error rate, p, for code distance d with the analytical formula provided by Fowler et al. for surface codes [5]

$$P_L \approx 0.03 \times (p/0.0057)^{(d+1)/2}. \quad (6)$$

The probability of successful trial (PST) is the probability of measuring the correct result at the end of the quantum program. With sufficiently high d, no logical errors would occur, and PST would be 1 (for quantum programs that have a single correct output). As d decreases, PST decreases exponentially until it hits nearly 0. The smallest d that is acceptable is a function of the physical error rate.

Using information from Fig. 1, we know which logical qubits are more sensitive to noise, which we leverage to designate a variable distance QECC. We experiment with single- and two-distance QECC: For example, 3,5 is a QECC which uses $d = 3$ on less susceptible qubits and $d = 5$ on more susceptible qubits. We consider qubits to be susceptible if the PST is below 40% in Fig. 1d.

We estimate PST from the probability of two events:

1) No logical error occurs
2) A single logical error occurs, but the output is the same as in the case of no error

These two probabilities combined provide a lower bound on PST. It is also possible that two or more logical errors occur and the output remains the same. However, counting these possibilities quickly becomes intractable because a total of $(N_{gates})^{N_{errors}}$ must be considered, where N_{gates} is the number of quantum gates performed in the algorithm and N_{errors} is the number of possible errors. Hence, in our analysis, two or more errors represent a failure.

The PST for all d are shown in Fig. 2. The dashed lines capture the variable (two-) distance QECC, which achieve resilience in-between their constituent code distances.

4.2 Time to Solution

We now combine the PST information from Section 3.2 with the latency of each logic operation for a given code distance, to estimate the time to solution. As noted in Section 3.1, the time to solution corresponds to L/PST. A gate on a logical qubit with distance d takes d cycles to complete. Since the distance of each logical qubit is known throughout the program, we can easily find the corresponding latency, i.e., L/PST which provides the time to solution as shown in Fig. 3. This is a best-case analysis for variable QEC, considering only the overhead for (logical) gate times and code distance conversion. There are additional overheads in lattice surgery or magic state distillation, see Section 4.3.

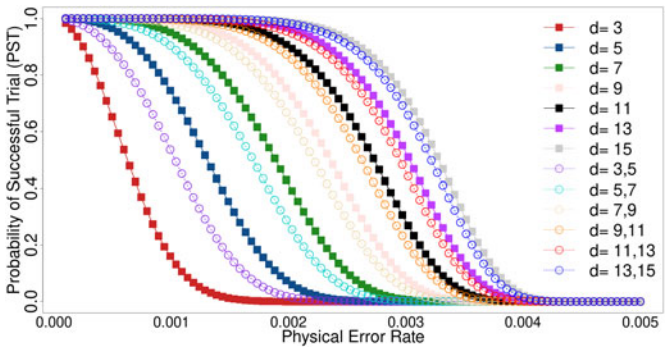

Fig. 2. PST of QPE for different (including variable) surface code distances, over a range of physical noise rates.

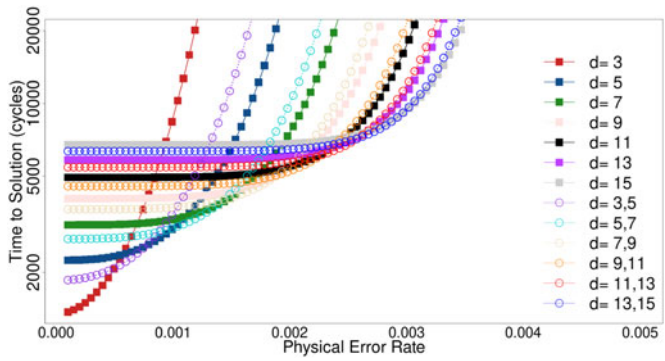

Fig. 3. Time to solution for different code distances.

TABLE 1
Physical Noise Ranges of Code Distance Optimality

Noise Range	Optimal Code Distance
$<=$ 5e-4	3
5.1e-4 - 7.6e-4	3,5
7.7e-4 - 1.0e-3	5
1.01e-3 - 1.38e3	5,7
1.39e-3 - 1.47e-3	7
1.48e-3 - 1.89e-3	7,9
1.90e-3 - 2.26e-3	9,11
2.27e-3 - 2.58e-3	11,13
2.59e-3 - 2.95e-3	13,15
$>$ 2.96e-3	15

It is noteworthy that the optimal code distance depends on the error rate. As intuition suggests, at low error rates lower code distances are preferable due to the lower overhead. However, as the error rate increases the codes begin to fail. Since error suppression is exponential, once the codes begin to fail, the reliability degrades dramatically and $P_{success}$ drops quickly. This necessitates a larger number of trials to obtain the correct solution, causing high latency.

Each variable distance code is optimal within a range of error rates. For example, the QECC $d = 3, 5$ is optimal at error rates where $d = 3$ begins to fail. $d = 3, 5$ can provide a faster solution than $d = 5$, until it breaks down and $d = 5$ becomes necessary to tolerate errors. The range where each code distance is optimal is listed in Table 1. It is possible to combine significantly different code distances. For example, using $d = 7, 15$ instead of $d = 7, 9$. However, this is sub-optimal. When $d = 7, 9$ is optimal, $d = 7$ is failing, but $d = 9$ remains strong. Hence, the additional protection provided by $d = 15$ is overkill, and consumes resources unnecessarily.

4.3 Practical Considerations

Variable distance QECC is promising, but practical limitations exist. Superconducting quantum architectures, for which surface codes are most appropriate, logical qubits are arranged next to each other in a two-dimensional lattice [7]. This enables trading physical qubits between neighboring logical qubits. In Fig. 1d, qubit 3 can use more physical qubits initially, but then transfer them to qubit 4 when qubit 4 becomes more sensitive. However, trading qubits may create non-uniform layouts which do not match well with the physical topology, possibly wasting qubits. Additionally, latency of such state-of-the-art quantum computers is not limited by logic gates, but by the preparation of special quantum states required to perform specific operations (such as magic state distillation for T gates [10]). Hence, improvements in the gate latency from variable strength QECC may not be significant. Also, quantum fault injection is tractable only for small circuits. Obtaining accurate sensitivity estimates for larger circuits poses a challenge, due to the inability to simulate such circuits. However, it may be possible to identify patterns in small circuits (Fig. 1), and use this to predict sensitive regions in larger circuits.

Finally, physical hardware noise rates will also impact optimal code distances. Our analysis exploits "software" sensitivity at the *logical* level. In practice, this information can be combined with *physical* hardware noise rates to find the optimal distance, a machine dependent optimization.

5 Conclusion

Our profiling analysis based on statistical fault injection shows that quantum programs can be relatively insensitive to isolated logical errors, and that variable distance QECC can reduce the time to solution by exploiting the spatio-temporal differences in noise sensitivity. However, any decrease in code distance due to variable strength QECC creates an exponential increase in logical failure rates, which may eliminate the benefits if not carefully administered.

Acknowledgments

The authors acknowledge the Minnesota Supercomputing Institute (MSI) at the University of Minnesota for providing resources that contributed to the research results reported within this paper. URL: http://www.msi.umn.edu.

References

[1] S. Bravyi et al., "Correcting coherent errors with surface codes," *NPJ Quantum Inf.*, vol. 4, no. 1, pp. 1–6, 2018.
[2] A. Cross, "The IBM Q experience and QISKit open-source quantum computing software," in *Proc. APS March Meeting Abstr.*, 2018, pp. L58–003.
[3] X. Dou and L. Liu, "A new qubits mapping mechanism for multi-programming quantum computing," in *Proc. ACM Int. Conf. Parallel Archit. Compilation Techn.*, 2020, pp. 349–350.
[4] A. Erhard et al., "Characterizing large-scale quantum computers via cycle benchmarking," *Nature Commun.*, vol. 10, no. 1, pp. 1–7, 2019.
[5] A. G. Fowler et al., "Surface codes: Towards practical large-scale quantum computation," *Phys. Rev. A*, vol. 86, no. 3, 2012, Art. no. 032324.
[6] C. Horsman et al., "Surface code quantum computing by lattice surgery," *New J. Phys.*, vol. 14, no. 12, 2012, Art. no. 123011.
[7] A. Javadi-Abhari et al., "Optimized surface code communication in superconducting quantum computers," in *Proc. 50th Annu. IEEE/ACM Int. Symp. Microarchit.*, 2017, pp. 692–705.
[8] B. P. Lanyon et al., "Towards quantum chemistry on a quantum computer," *Nature Chem.*, vol. 2, no. 2, pp. 106–111, 2010.
[9] G. Li et al., "Tackling the qubit mapping problem for NISQ-era quantum devices," in *Proc. 24th Int. Conf. Archit. Support Program. Lang. Oper. Syst.*, 2019, pp. 1001–1014.
[10] D. Litinski, "A game of surface codes: Large-scale quantum computing with lattice surgery," *Quantum*, vol. 3, 2019, Art. no. 128.
[11] S. Resch, S. Tannu, U. R. Karpuzcu, and M. Qureshi, "A day in the life of a quantum error," *IEEE Comput. Archit. Lett.*, vol. 20, no. 1, pp. 13–16, Jan.–Jun. 2020.
[12] S. Resch and U. R. Karpuzcu, "Benchmarking quantum computers and the impact of quantum noise," *ACM Comput. Surv.*, vol. 54, no. 7, pp. 1–35, 2021.
[13] P. Selinger, Newsynth: Exact and approximate synthesis of quantum circuits, Mar. 2014. Accessed: Jul. 3, 2022. [Online]. Available: https://www.mathstat.dal.ca/~selinger/newsynth/
[14] P. Selinger, "Efficient clifford+ T approximation of single-qubit operators," 2012, *arXiv:1212.6253*.
[15] A. M. Steane, "Introduction to quantum error correction," *Philos. Trans. Roy. Soc. London. Ser. A: Math., Phys. Eng. Sci.*, vol. 356, no. 1743, pp. 1739–1758, 1998.
[16] S. S. Tannu and M. K. Qureshi, "Not all qubits are created equal: A case for variability-aware policies for NISQ-era quantum computers," in *Proc. 24th Int. Conf. Archit. Support Program. Lang. Oper. Syst.*, 2019, pp. 987–999.
[17] E. Wilson, S. Singh, and F. Mueller, "Just-in-time quantum circuit transpilation reduces noise," in *Proc. IEEE Int. Conf. Quantum Comput. Eng.*, 2020, pp. 345–355.