Quantum search on noisy intermediate-scale quantum devices

K. Zhang\(^1\)(a), K. Yu\(^2\) and V. Korepin\(^3\)

\(^1\)Department of Chemistry, State University of New York at Stony Brook - Stony Brook, NY 11794, USA
\(^2\)Computational Science Initiative, Brookhaven National Laboratory - Upton, NY 11973, USA
\(^3\)C.N. Yang Institute for Theoretical Physics, Stony Brook University - Stony Brook, NY 11794-3840, USA

received 12 June 2022; accepted in final form 9 September 2022
published online 27 September 2022

Abstract – Quantum search algorithm (also known as Grover’s algorithm) lays the foundation for many other quantum algorithms. Although it is very simple, its implementation is limited on noisy intermediate-scale quantum (NISQ) processors. Grover’s algorithm was designed without considering the physical resources, such as depth, in the real implementations. Therefore, Grover’s algorithm can be improved for NISQ devices. In this paper, we demonstrate how to implement quantum search algorithms better on NISQ devices. We present detailed benchmarks of the five-qubit quantum search algorithm on different quantum processors, including IBMQ, IonQ, and Honeywell quantum devices. We report the highest success probability of the five-qubit search algorithm compared to previous works. Our results show that designing the error-aware quantum search algorithms is possible, which can maximally harness the power of NISQ computers.

Copyright © 2022 EPLA

Introduction. – The ultimate goal of quantum computers is to implement quantum algorithms that are superior to their classical counterpart. During the last twenty years, quantum computers have vastly developed. Quantum processors with hundreds of qubits have been delicately designed and built. We have entered the noisy-intermediate-quantum (NISQ) era [1]. Quantum processors with hundreds of qubits have the ability to tackle problems far beyond the reach of classical computers. However, errors limit the number of consecutive operations that can be applied. The number of consecutive operations is also called depth.

To harness the power of NISQ processors, various quantum algorithms with shallow depths have been designed [2]. The practical power of the shallow depth algorithm is under extensive study now. Nevertheless, the promise of quantum computers largely relies on the application of functional quantum algorithms, such as Shor’s algorithm [3] and Grover’s algorithm [4]. Recently, more researches begin to benchmark the quantum computers based on those applications (application-oriented benchmarking) [5,6].

Because of its simplicity, Grover’s algorithm is usually the first quantum algorithm taught in the course of quantum computation. Grover’s algorithm solves the unstructured search problem [4,7]. It finds the target item, also called the marked item, in an unstructured database. Classically, the unstructured search problem can be solved by querying each item in the database. The target item can be recognized by the black-box function (oracle). Therefore, the oracular complexity of the classical search is \(O(N)\), assuming that the number of items in the database is \(N\). Grover’s algorithm has the oracular complexity \(O(\sqrt{N})\). The quadratic speedup is due to the superposition of quantum states.

Grover’s algorithm is optimal in the number of queries to the oracle, because of the linearity of quantum mechanics [8,9]. The idea of Grover’s algorithm is not limited to the unstructured search problem. The amplitude of wanted states can be amplified with a similar quadratic speedup [10,11]. Therefore, it makes the generalized quantum search algorithm applicable to a wide range of problems, such as quantum machine learning [12].

However, there is a gap between the oracular resources and the physical resources of real quantum computation, such as the number of qubits and circuit depths. During the last twenty years, many variants of Grover’s algorithm have been proposed [13–24]. Few of them have focused on the practical performance of quantum search algorithms on NISQ devices. It was Grover himself who first realized the possible trade-off between the number of oracles and the physical resource for real implementations (such
as depths) in the quantum search algorithm [15]. Following the same spirit, one can design the quantum search algorithm, which requires fewer computational resources compared to Grover’s algorithm [21]. The depth optimization of quantum search algorithms can also benefit their implementations in the post-NISQ era. In other words, reducing the circuit depth also reduces the error correction resources (and potentially the running time).

The implementation of Grover’s algorithm (with the three-qubit search domain) was firstly reported in 2017 [25]. Since then, more researches have studied the performance of Grover’s algorithm on real quantum processors [26–30]. Most of the realizations are up to four qubits ($N = 2^4$).

In this work, we study the performances of quantum search algorithms on the state-of-art quantum processors. The significance of our work is three-fold. First, we aim to benchmark different processors via the same search problem. Although the general framework of the application-oriented benchmarks was proposed recently [5, 6], improvements on the benchmark results are possible. For example, being aware of the connectivity of physical qubits can reduce unnecessary SWAP gates in the circuit. One can also take the advantage of the relative-phase Toffoli gates to reduce the circuit depth [31,32]. Second, Grover’s algorithm is not optimal in real implementations. Different quantum processors have different qubit connectivities and error rates. We aim to benchmark the same search problem solved by different search circuits including Grover’s original search algorithm. Third, cloud quantum computations, based on different types of physical qubits, are available to researchers, such as IBMQ [33], IonQ [34] and Honeywell quantum systems (Honeywell Quantum Solution combines Cambridge Quantum called Quantumm) [35]. We aim to benchmark the same algorithms across the different quantum processors. For the same search problem, different processors show different optimal search algorithms, because of their different qubit connectivities and error rates.

Quantum search algorithms. – Suppose that the number of items in the database, denoted as $N$, is a power of 2. Then there requires $n = \log_2 N$ number of qubits to represent all the items. The index of each item corresponds to one basis vector $|j\rangle$ in $H_2^\otimes n$. For convenience, the basis vector is chosen as the computational basis (bit strings of zeros and ones). The initial state is set as the uniform superpositions of all the basis vectors, denoted as $|s_n\rangle$. Although such a quantum state is highly nontrivial, it can be easily prepared with a one-depth circuit, given by

$$|s_n\rangle = H^\otimes n |0\rangle^\otimes n. \quad (1)$$

Here $H$ is the single-qubit Hadamard gate [36].

The information of the target state or the target item is encoded in the oracle. The oracle, also called the black box, can distinguish the target and non-target states. In quantum search algorithms, querying the oracle (implementing the oracle) would reflect the sign of the target state amplitude. The corresponding mathematical expression is

$$O_t = |2\rangle\langle t|,$$  \quad (2)

with the target state denoted as $|t\rangle$. Here we assume the uniqueness of the target state for simplicity. Grover’s algorithm also works if there are multiple target states [8].

The oracle $O_t$ is not enough to pick up the target state from $|s_n\rangle$. Define the operator $D_n$ as

$$D_n = \mathbb{1}_{2^n} - 2|s_n\rangle\langle s_n|,$$ \quad (3)

which is also called the diffusion operator. The operator $D_n$ reflects all amplitudes in terms of their average. Intuitively, the operator $D_n$ “diffuses” the amplitudes of target state with the uniformed amplitudes of non-target states.

Grover’s algorithm is realized by repeatedly applying the operator (also called the Grover operator)

$$G_n = D_n O_t \quad \quad (4)$$
on the initial state $|s_n\rangle$. Then the amplitude of the target state increases in a nonlinear way. It gives

$$|\langle t| G_n^j |s_n\rangle|^2 = \sin^2((2j + 1)\theta),$$ \quad (5)

with integer $j$ and $\theta = \arcsin(1/\sqrt{N})$. When the number of iteration $j$ approaches $\pi\sqrt{N}/4$, the success probability of finding the target state approaches 1. We can conclude that the oracular complexity of Grover’s algorithm is $O(\sqrt{N})$. It is a quadratic speedup compared to the classical complexity $O(N)$.

The increasing speed of target amplitude is nonlinear in Grover’s algorithm. When the number of iterations $j$ approaches $\pi\sqrt{N}/4$, the algorithm becomes less efficient. It has been argued in [8,37] that Grover’s algorithm stopping at $0.5829\sqrt{N}$ is the most efficient way. The corresponding success probability is around 0.8446.

Depth-optimized quantum search algorithms. Different unstructured search problems have different oracles $O_t$. For example, see [38] for the Advanced Encryption Standard decryption using Grover’s algorithm. On the other hand, the implementation of diffusion operator $D_n$ is unambiguous. The $n$-qubit diffusion operator $D_n$ is equivalent to the $n$-qubit Toffoli gate (up to single-qubit gates) [36]. The realization of $n$-qubit Toffoli gate on quantum computers is highly nontrivial. Quantum computers can only perform a set of basic single- and two-qubit gates called the universal gate set [39]. The specific number of the depth of $n$-qubit Toffoli gate depends on the connectivity and the type of universal gate set of quantum processors.

Grover firstly realized that the highly nonlocal diffusion operator can be replaced by the local diffusion operator [15]. Here local diffusion operator means reflecting the
part of amplitudes (divide the database into blocks). The local diffusion operator is a renormalized version of $D_n$, namely

$$D_m = (I_{2m} - 2|s_m⟩⟨s_m|) \otimes I_{2n-m}, \quad (6)$$

with integer $m \leq n$. The operator $D_m$ is equivalent to $m$-qubit Toffoli gate. If $m < n$, then realization of $D_m$ requires fewer gates (also fewer depths) than $D_n$. We can define the corresponding Grover operator as

$$G_m = D_m O_t. \quad (7)$$

To distinguish between $G_n$ and $G_m$, we call $G_n$ as the global Grover operator; $G_m$ as the local Grover operator. To be clarified, the local Grover operator still aims to solve the search problem with $N = 2^n$. The oracle $O_t$ cannot be renormalized for unstructured search problems.

The simplest application of local Grover operator $G_m$ is to renormalize the database. Then we can prepare the state $|s_m⟩\otimes|l⟩$ as initial. Here the bit string $l \in \{0, 1\}^{n-m}$ is randomly chosen. Correspondingly, the target state $|t⟩$ is partitioned into $|t⟩ = |t_2⟩ \otimes |t_1⟩$. Applying $G_m$ on $|s_m⟩\otimes|l⟩$ can find the target state $|t_2⟩$ if $l = t_1$. Randomly choosing $l$ gives the probability $P_{t=t_1} = 1/2^{n-m}$.

Then the total success probability of finding the target state is

$$P_{t=t_1} = Tr \left(|t_2⟩⟨t_2| G_m^* |s_m⟩⟨s_m| |G_m^† \right) \sin^2((2j + 1)\theta_b) \quad (8)$$

with the short notation $|s_m,l⟩ = |s_m⟩\otimes|l⟩$. Here $\sin \theta_b = 1/\sqrt{b}$ and $b = 2^m$. The algorithm combines the classical randomly guessing and the quantum search. Therefore, it is called the hybrid classical-quantum search algorithm [30]. The optimal $j$ reaching the maximal success probability is much smaller than $\pi \sqrt{N}/4$. Therefore the circuit depth is dramatically reduced. The drawback is the success probability never approaches 1. The circuit may need many rounds to find the target state. When $n - m$ is large, it is less efficient than the classical algorithm.

A more sophisticated way to apply the local Grover operator is to replace some of $G_n$ with $G_m$ [15,21–23]. Since $G_n$ does not commute with $G_m$, the order of $G_n$ and $G_m$ is important. It provides flexibility to design quantum search algorithms that use the least physical resource, such as depth. The proper metric characterizing the physical resource is the expected depth finding the target state. It is the average number of depths needed to find the target state. There is always a trade-off between the circuit depth and width (number of qubits). For example, see [40].

Consider a sequence of operators

$$L_{n,m}(γ) = G_{m}^{j_1}G_{m}^{j_2} \cdots G_{m}^{j_n}, \quad (9)$$

with $γ = \{j_1, j_2, \ldots, j_q\}$. Each local Grover operator $G_m$ acts on the same subspace of database. We denote the depth of $L_{n,m}(γ)$ as $d(L_{n,m}(γ))$. Different quantum processors (with different connectivities and universal gate sets) would have different $d(L_{n,m}(γ))$. The expected depth of finding the target state (using the operator $L_{n,m}(γ)$) is

$$\langle L_{n,m}(γ) \rangle = \frac{d(L_{n,m}(γ))}{\|⟨t|L_{n,m}(γ)|s_n⟩|^2}. \quad (10)$$

The optimal choice is to find the minimum of $\langle L_{n,m}(γ) \rangle$. The optimizations should go through all possible orders of $G_n$ and $G_m$. Besides, we can also choose different $m$ (the width of the local diffusion operator). Although replacing $D_n$ with $D_m$ will decrease the success probability, the saved depth (from $D_m$) may still reduce the expected depth $\langle L_{n,m}(γ) \rangle$. Theoretical study shows that Grover’s algorithm is not optimal in depth as long as the depth of $D_n$ is not negligible compared to the depth of $O_t$. We will show that implementations of $\langle L_{n,m}(γ) \rangle$ on NISQ devices not only reduce the depth but also increase the success probability (compared to Grover’s algorithm).

Grover’s algorithm becomes inefficient when $j$ approaches $\pi \sqrt{N}/4$. If we consider the expected depth of Grover’s algorithm, $L_{n,m}(γ)$ with $m = n$ and $γ = \{j\}$, the minimum of the expected depth is obtained when $j$ approaches $0.5829\pi \sqrt{N}$. Correspondingly, the minimal expected depth is around $0.6901\pi \sqrt{N}$ times the depth of $G_n$. It is certainly smaller than $\pi \sqrt{N}/4$ times the depth of $G_n$.

**Divide-and-conquer quantum search algorithms.** Another application of local Grover operator $G_m$ is the quantum partial search algorithm [41,42]. The quantum partial search algorithm trades accuracy for speed. It only finds partial bits of the target state, but the number of queries to oracle is smaller than Grover’s algorithm. The original quantum partial search algorithm is optimal in the number of queries to oracle [43,44]. However, it is not optimal in depth [21]. In fact we can optimize the depth over all possible configurations. Assume that we want to design a two-stage algorithm. The target state is partitioned into $|t⟩ = |t_2⟩ \otimes |t_1⟩$. Consider the operator $L_{n,m}^{(1)}(γ_1)$, defined in eq. (9), for the first stage. The probability that the operator $L_{n,m}^{(1)}(γ_1)$ finds the bit string $t_1$ is

$$P_{n,m}^{(1)}(γ_1) = Tr \left(|t_1⟩⟨t_1| L_{n,m}^{(1)}(γ_1) |s_n⟩⟨s_n| L_{n,m}^{(1)}(γ_1) \right). \quad (11)$$

Here the partial diffusion operator $D_{m}$ is acting on the first $m$ qubits. And we measure the last $n-m$ qubits. Note that the quantum partial search algorithm has the convention of measuring the qubits without acting by the partial diffusion operator. Only in special cases, measuring on qubits acted by the partial diffusion operator has a higher success probability. For more examples, see [21].

The second-stage circuit is a renormalized quantum search algorithm. Consider the operator $L_{n,m}^{(2)}(γ_2)$ with $k \leq m \leq n$. Assume that we find $t_2$ in the first stage.
Then the probability finding $t_2$ is
\[ P_{n,k}(\gamma_2) = \text{Tr} \left( |t_2\rangle \langle t_2| L_{n,k}^{(2)}(\gamma_2) |s_m,t_1\rangle \langle s_m,t_1| L_{n,k}^{(2)*} (\gamma_2) \right), \]
with the short notation $|s_m,t_1\rangle = |s_m\rangle \otimes |t_1\rangle$. Then we can find the expected depth of the above two-stage search algorithm, given by
\[ (L_{n,m}^{(1)}(\gamma_1) + L_{n,k}^{(2)}(\gamma_2)) \frac{d}{P_{n,m}(\gamma_1)} + \frac{d}{P_{n,k}(\gamma_2)} \] 
\[ \text{(13)} \]

The optimal realization is to choose $m,k,\gamma_1,\gamma_2$ which gives the minimum $(L_{n,m}^{(1)}(\gamma_1) + L_{n,k}^{(2)}(\gamma_2))$. If the oracle cannot verify the partial bits, we cannot separately optimize the expected depth of $L_{n,m}^{(1)}(\gamma_1)$ or $L_{n,k}^{(2)}(\gamma_2)$.

Surprisingly, when $d(O_1) < (1 + \sqrt{3})d(D_n)$, the minimal expected depth of two-stage algorithm can still be smaller than Grover’s algorithm (assuming no errors in quantum processors) [21]. The previous study has shown the advantage of the two-stage four-qubit quantum search algorithm on IBMQ devices [30].

**Quantum-search–oriented benchmarks.** – Now we present the results of the five-qubit quantum search algorithms on IBMQ, IonQ, and Honeywell quantum devices. The circuits include the original Grover’s algorithm, the depth-optimized, and the divide-and-conquer quantum search algorithms. First, we introduce the benchmark metrics, followed by our designed search circuits. Then we present the benchmark results.

**Benchmark metrics.** We choose the following metrics to benchmark the quantum search algorithm.

- **Success probability.** The success probability of finding the target state is the most important metric for quantum search algorithms.

- **Expected depth.** It is defined as the depth over the success probability, such as shown in eqs. (10) and (13). It characterizes the average depth for finding the target state.

- **Selectivity.** Selectivity was introduced in [45,46]. It is defined as
\[ S = \ln \left( \frac{P_t}{\max P_{nt}} \right), \]
assuming $\max P_{nt} \neq 0$. Here $P_t$ is the probability finding the target state; $P_{nt}$ is the probability finding the non-target state. The logarithmic function is added for convenience. A negative selectivity in quantum search algorithms means that it is worse than the random guess. The two-qubit search is the only case where Grover’s algorithm finds the target state with success probability 1 [47]. Then we define the corresponding selectivity as infinity.

- **Circuit fidelity.** The circuit fidelity is a universal benchmark metric introduced in [5]. It quantifies the distance between the output probabilistic distribution $P_{out}$ with the ideal $P_{ideal}$. Besides, it is normalized in terms of the uniform distribution $P_{uni}$. The definition is
\[ F(P_{out},P_{ideal}) = \frac{\min f(P_{out},P_{ideal})}{1 - f(P_{uni},P_{ideal})}. \]
\[ \text{(15)} \]

Obviously, we have $F(P_{out},P_{ideal}) \leq 1$ and $F(P_{ideal},P_{ideal}) = 1$. If the output is uniform, then $F(P_{uni},P_{ideal}) = 0$. Note that high fidelity does not necessarily mean a high success probability. The circuit fidelity quantifies the degree of degradation due to the noises. As argued in [5], the metric success probability and circuit fidelity are complementary for benchmarking.

**Benchmark circuits.** Previous studies on the implementation of Grover’s algorithm focus on the four-qubit search domain [26–30]. In our study, we march onto the five-qubit quantum search. We design the following circuits. Their corresponding quantum circuits can be found in the Supplementary Material Supplementarymaterial.pdf (SM).

- **G5M5:** apply one $G_5$ operator, and then measure all the five qubits. Here G and M stand for the Grover operator and the measurement respectively. G5M5 is the one-iteration five-qubit Grover’s algorithm.

- **G5G5M5:** apply two $G_5$ operators, and then measure all the five qubits. G5G5M5 is the two-iteration five-qubit Grover’s algorithm.

- **R2G3M3:** randomly initialize the values of two qubits then apply the $G_3$ operator and measure the other three qubits.

- **R3G2M2:** randomly initialize the values of three qubits then apply the $G_2$ operator and measure the other two qubits.

- **G2M2(G3M3):** in the first stage, apply the $G_2$ operator then measure two qubits; in the second stage, initialize the two qubits based on the first-stage measurement results, then apply the $G_3$ operator and measure the other three qubits.

- **G3M3(G2M2):** in the first stage, apply the $G_3$ operator then measure three qubits; in the second stage, initialize the three qubits based on the first-stage measurement results, then apply the $G_2$ operator and measure the other two qubits.
Quantssearno inntertmeda-scale quantum devicen

Fig. 1: Success probabilities of the quantum search circuits G5M5, G5G5M5, R2G3M3, R3G2M2, G2M2|G3M3, and G3M3|G2M2. The black dashed lines represent the theoretical success probabilities. The red dash-dotted line represents the highest success probability of the five-qubit search reported in [29]. The variance is based on $3 \times 400$ shots.

As a toy oracle, we set the oracle as a five-qubit controlled phase gate. It is also single-qubit-gate equivalent to the five-qubit Toffoli gate. See the SM for its implementation (with the help of the relative-phase Toffoli gate [32]). To exclude any bias related to the target state, we randomly choose the target state as $|01011\rangle$. For more detailed setups on the IBMQ, IonQ, and Honeywell quantum devices, see the SM.

Benchmark results. We implement the quantum search circuits designed above on four different quantum processors: IBMQ Lagos, IBMQ Mumbai, IonQ, and Honeywell System Model H1. The success probabilities of all circuits are shown in fig. 1. The circuit R3G2M2 gives the maximal success probability on both IBMQ Lagos and Mumbai. For IonQ, the maximum is given by the two-stage circuit G3M3|G2M2. Honeywell quantum device gives the success probability of around 0.49 for the two-iteration Grover’s algorithm G5G5M5. It is the highest among all the implementations. To the best of the authors’ knowledge, it is also higher than the success probabilities reported before [29,30]. See fig. 1 for the comparison. Due to the shallow depth of R3G2M2, its outputs are always very close to the theoretical values. Note that circuit R3G2M2 is still superior to the classical search with one query to oracle.

The circuit depths on Lagos and Mumbai are the same since we choose the physical qubits which have the same connectivity on those two devices. We estimate the circuit depths implemented on IonQ and Honeywell quantum devices via the universal gate set including CNOT and arbitrary single-qubit gates. The native two-qubit gates provided by IonQ and Honeywell are both single-qubit-gate equivalent to CNOT gate. Since both the qubits of IonQ and Honeywell are fully connected, the same circuits have the same estimated depth on those two devices. More details can be found in the SM.

We report the expected depth in fig. 2. The theoretically expected depth for IBMQ Lagos and Mumbai is larger than the ones for IonQ and Honeywell, because of the extra SWAP gates required for IBMQ Lagos and Mumbai. From fig. 2, we can obviously see that R3G2M2 has the lowest expected depth (both in theory and practice). Moreover, the expected depth of R3G2M2 is smaller than the one- and two-iteration Grover’s algorithm. Then we confirm that Grover’s algorithm is not optimal in depth. Therefore it is practical to design quantum search circuits which trade the success probability for circuit depths, either for NISQ or post-NISQ devices.

The results on selectivity metric are shown in fig. 3. Negative selectivity means that there is a non-target state having a higher probability than the target state. It means the failure of the algorithm. The theoretical selectivity of R3G2M2 is infinitely large since it is a renormalized two-qubit search circuit (with success probability one). The five-qubit search algorithm is the border for IBMQ devices. The IonQ device can afford one-iteration Grover’s algorithm. The Honeywell quantum device can successfully find the target state in all six different search circuits.

18002-p5
Circuit fidelity defined in eq. (15) characterizes the distance between the output and ideal distributions. Since the second stage circuit of G2M2|G3M3 is equivalent to R2G3M3, here we only benchmark the fidelity of the first stage circuit G2M2. For Honeywell quantum devices, relative deep circuits, such as G5G5M5, do not decrease much fidelity compared to other circuits. Honeywell quantum devices can handle well on circuits with over two hundred depths (around a hundred depths contain two-qubit gates). Circuits R3G2M2 and G2M2 have only one two-qubit diffusion operator. All processors can give fidelity over 0.5 on those two circuits. Circuits with negative fidelities are consistent with the negative selectivities shown in fig. 3.

Based on the benchmark results, one can compare them to the noisy simulation. Then the effective error rates can be estimated. One can conclude that the depth-optimized and the divide-and-conquer search circuits are more robust to errors, compared to Grover’s algorithm. More details can be found in the SM.

Conclusions and outlooks. – In this work, we have studied the performances of NISQ devices, including quantum processors from IBMQ, IonQ, and Honeywell, on the quantum search algorithms. Besides Grover’s algorithm, we also implement the depth-optimized and divide-and-conquer search algorithms [21]. We compare the different implementations of the quantum search algorithm and show that Grover’s algorithm does not guarantee the highest success probability as well as the lowest expected depth. Our five-qubit search implementations achieve the highest success probability compared to the previous works [29,30]. Compared to the search circuits given by the local diffusion operator, we show that the expected depth of five-qubit Grover’s algorithm is not optimal. In practice, both the IBMQ, IonQ, and Honeywell quantum computers show a lower expected depth of finding the target state by our optimized quantum search algorithms.

There is still a long way to run classical-impossible search tasks on quantum computers, such as the AES decryption. But mid-size quantum search algorithm, such as \( N = 2^{40} \), may serve as a subroutine, to provide a quantum speedup in the near future. How to optimize the quantum search algorithms (finding the minimal expected depth or the minimal T-depth [48]) running on fifty qubits is a highly nontrivial problem. How to determine the optimal divide-and-conquer strategy for the quantum search algorithms is also an open question, especially on different quantum processors with different connectivities and error rates. Note that the divide-and-conquer strategy also means that we can parallel run the algorithm [37]. The local diffusion operator also has the advantage of reducing circuit errors. It is also interesting to explore how the local diffusion operator influences the coherence and the success probability [49]. We leave those questions for future study.

The authors thank Tzu-Chieh Wei, Yusheng Zhao, and Hongye Yu for helpful discussions, especially Yusheng Zhao who pinpointed the bug in using Qiskit for IonQ devices. This research used quantum computing resources of the Oak Ridge Leadership Computing Facility, which is a DOE Office of Science User Facility supported under Contract DE-AC05-00OR22725. This research used resources of the National Energy Research Scientific Computing Center, a DOE Office of Science User Facility supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231 using NERSC award DDR-ERCAP0022229. We acknowledge the use of an IonQ quantum computer,
REFERENCES

[1] Preskill J., *Quantum*, 2 (2018) 79.
[2] Bharti K., Cervera-Lierta A., Kyaw T. H., Haug T., Alperin-Lea S., Anand A., Degroote M., Heimonen H., Kottmann J. S., Menke T., Mok W.-K., Sim S., Kwek L.-C. and Aspuru-Guzik A., *Rev. Mod. Phys.*, 94 (2022) 015004.
[3] Shi P., *Algorithms for quantum computation: Discrete logarithms and factoring*, in Proceedings 35th Annual Symposium on Foundations of Computer Science (IEEE) 1994, pp. 124–134.
[4] Grover L. K., *Phys. Rev. Lett.*, 79 (1997) 325.
[5] Lubinski T., Johri S., Varosy P., Coleman J., Zhao L., Necaise J., Baldwin C. H., Mayer K. and Proctor T., arXiv:2110.03137 [quant-ph] (2021).
[6] Georgopoulos K., Emary C. and Zuliani P., arXiv:2112.09457 [quant-ph] (2021).
[7] Giri P. R. and Korepin V. E., *Quantum Inf. Process.*, 16 (2017) 315.
[8] Boyer M., Brassard G., Hoyer P. and Tapp A., *Fortschr. Phys.*, 46 (1998) 493.
[9] Zalka C., *Phys. Rev. A*, 60 (1999) 2746.
[10] Brassard G., Hoyer P., Mosca M. and Tapp A., *Contemp. Math.*, 305 (2002) 53.
[11] Grover L. K., *Phys. Rev. Lett.*, 80 (1998) 4329.
[12] Diamante J., Wittek P., Pancotti N., Rebentrost P., Wiebe N. and Lloyd S., *Nature*, 545 (2017) 195.
[13] Galindo A. and Martín-Delgado M. A., *Phys. Rev. A*, 62 (2000) 062303.
[14] Long G.-L., *Phys. Rev. A*, 64 (2001) 022307.
[15] Grover L. K., *Phys. Rev. A*, 66 (2002) 052314.
[16] Kato G., *Phys. Rev. A*, 72 (2005) 032319.
[17] Tuls A., *Phys. Rev. A*, 86 (2012) 042331.
[18] Tuls A., *Phys. Rev. A*, 91 (2015) 052307.
[19] Jiang Z., Rieffel E. G. and Wang Z., *Phys. Rev. A*, 95 (2017) 062317.
[20] Gilliam A., Pistoia M. and Gonciulea C., arXiv:2007.10894 [quant-ph] (2020).
[21] Zhang K. and Korepin V. E., *Phys. Rev. A*, 101 (2020) 032346.
[22] Briánsk M., Gwinner J., Hlembotski V., Jarnicki W., Plis S. and Szady A., *Phys. Rev. A*, 103 (2021) 062425.
[23] Liu J. and Zhou H., arXiv:2103.14196 [quant-ph] (2021).
[24] Kwon H. and Bae J., *Phys. Rev. A*, 104 (2021) 062438.
[25] Figgatt C., Maslov D., Landsman K. A., Linke N. M., Debnath S. and Monroe C., *Nat. Commun.*, 8 (2017) 1918.
[26] Mandviwalla A., Onishi K. and Ji B., *Implementing Grover’s Algorithm on the IBM Quantum Computers*, in Proceedings of the 2018 IEEE International Conference on Big Data (Big Data) (IEEE) 2018, pp. 2531–2537.
[27] Sator T., Ohkura Y. and Van Meter R., *IEEE Trans. Quantum Eng.*, 1 (2020) 1.
[28] Gwinner J., Briánsk M., Burkel W., Czerwiński L. and Hlembotski V., arXiv:2007.06539 [quant-ph] (2021).
[29] Hlembotski V., Burczyński R., Jarnicki W., Szady A. and Tólowiecki J., arXiv:2010.03841 [quant-ph] (2020).
[30] Zhang K., Rao P., Yu K., Lim H. and Korepin V., *Quantum Inf. Process.*, 20 (2021) 233.
[31] Song G. and Klappenecker A., arXiv:quant-ph/0312225 (2003).
[32] Maslov D., *Phys. Rev. A*, 93 (2016) 022311.
[33] https://quantum-computing.ibm.com/ (2021).
[34] https://console.cloud.google.com/marketplace/product/ionq-public/ionq?pli=1 (2021).
[35] https://www.quantinuum.com/ (2021).
[36] Nielsen M. A. and Chuang I. L., *Quantum Computation and Quantum Information* (Cambridge University Press) 2010.
[37] Ying R. and Huggins R., *Quantum Inf. Procr.*, 21 (2017) 062303.
[38] Gilliam A., Pistoia M. and Gonciulea C., arXiv:2007.10894 [quant-ph] (2020).
[39] Zhang K. and Korepin V. E., *Phys. Rev. A*, 101 (2020) 032346.
[40] Briánsk M., Gwinner J., Hlembotski V., Jarnicki W., Plis S. and Szady A., *Phys. Rev. A*, 103 (2021) 062425.
[41] Liu J. and Zhou H., arXiv:2103.14196 [quant-ph] (2021).
[42] Kwon H. and Bae J., *Phys. Rev. A*, 104 (2021) 062438.
[43] Figgatt C., Maslov D., Landsman K. A., Linke N. M., Debnath S. and Monroe C., *Nat. Commun.*, 8 (2017) 1918.
[44] Mandviwalla A., Onishi K. and Ji B., *Implementing Grover’s Algorithm on the IBM Quantum Computers*, in Proceedings of the 2018 IEEE International Conference on Big Data (Big Data) (IEEE) 2018, pp. 2531–2537.
[45] Sator T., Ohkura Y. and Van Meter R., *IEEE Trans. Quantum Eng.*, 1 (2020) 1.
[46] Gwinner J., Briánsk M., Burkel W., Czerwiński L. and Hlembotski V., arXiv:2007.06539 [quant-ph] (2021).
[47] Hlembotski V., Burczyński R., Jarnicki W., Szady A. and Tólowiecki J., arXiv:2010.03841 [quant-ph] (2020).
[48] Zhang K., Rao P., Yu K., Lim H. and Korepin V., *Quantum Inf. Process.*, 20 (2021) 233.
[49] Song G. and Klappenecker A., arXiv:quant-ph/0312225 (2003).
[50] Maslov D., *Phys. Rev. A*, 93 (2016) 022311.
[51] https://quantum-computing.ibm.com/ (2021).
[52] https://console.cloud.google.com/marketplace/product/ionq-public/ionq?pli=1 (2021).
[53] https://www.quantinuum.com/ (2021).
[54] Nielsen M. A. and Chuang I. L., *Quantum Computation and Quantum Information* (Cambridge University Press) 2010.