quantum reports

MDPI

Article

Shallow-Depth Quantum Circuit for an Unstructured Database Search

Junpeng Zhan

Department of Renewable Energy Engineering, Alfred University, Alfred, NY 14802, USA; zhanj@alfred.edu

Abstract: Grover's search algorithm (GSA) offers quadratic speedup in searching unstructured databases but suffers from exponential circuit depth complexity. Here, we present two quantum circuits called HX and Ry layers for the searching problem. Remarkably, both circuits maintain a fixed circuit depth of two and one, respectively, irrespective of the number of qubits used. When the target element's position index is known, we prove that either circuit, combined with a single multi-controlled X gate, effectively amplifies the target element's probability to over 0.99 for any qubit number greater than seven. To search unknown databases, we use the depth-1 Ry layer as the *ansatz* in the Variational Quantum Search (VQS), whose efficacy is validated through numerical experiments on databases with up to 26 qubits. The VQS with the Ry layer exhibits an exponential advantage, in circuit depth, over the GSA for databases of up to 26 qubits.

Keywords: Grover's search algorithm; search unstructured databases; variational quantum search

Citation: Zhan, J. Shallow-Depth Quantum Circuit for an Unstructured Database Search. *Quantum Rep.* **2024**, *6*, 550-563. https://doi.org/10.3390/quantum6040037

Received: 7 September 2024
Revised: 10 October 2024
Accepted: 21 October 2024
Published: 25 October 2024

1. Introduction

Quantum search algorithms are a significant area of research in quantum computing because of their potential to revolutionize various fields with (exponentially) faster solutions compared with classical algorithms [1,2]. Among these algorithms, Grover's search algorithm (GSA) [3,4] is one of the most well-known, offering a *quadratic* speedup [5] in searching unstructured databases. It has been applied to offer quadratic speedup in solving critical problems, including NP-complete problems [6–12], cryptography [13,14], quantum machine learning [15–19], quantum state preparation [20,21], collision problems [22], and more [1,2,23,24]. Moreover, GSA can serve as a sub-routine of many quantum algorithms.

Because of its importance, researchers have explored various aspects of GSA to enhance its performance. Generalized GSA [10] and Quantum Amplitude Amplification [25] were proposed to tackle GSA's limitation of handling only one target element. Refs. [26,27] revised GSA to ensure finding the target element with certainty. GSA was implemented on a real unstructured classical database [28], on a real quantum computer [29], and with fewer gates [30]. Researchers also realized GSA by adiabatic evolution [31,32]. Ref. [33] proposed a variational learning Grover's quantum search algorithm, which shows improvement over GSA for three- and four-qubit cases, but lacks improvement in larger qubit cases. However, despite the success, GSA faces a drawback in its circuit depth, which grows exponentially with qubit numbers, limiting its applicability to larger databases. While several variants have been explored, none have reduced the circuit depth complexity, and GSA's optimality remains established [5,6,34].

This paper addresses the *problem* of finding the target element in an unstructured database that has one good element and $(2^n - 1)$ bad elements (n denotes the number of data qubits). The "target element" and "good element" are used interchangeably in this paper. The *goal* of quantum search algorithms, like GSA, is to amplify the probability of the target element to nearly 1. To tackle the challenge of circuit depth, in this paper, we

propose the following algorithms to construct two shallow-depth quantum circuits: HX and Ry layers, with depths of 2 and 1, respectively. We prove that either circuit, together with a C^n (X) gate, achieves the goal by amplifying the probability of the sole target element from $1/2^n$ to over 0.99 for any n greater than seven. However, both algorithms rely on knowing the position index of the target element in advance, limiting their use to analysis purposes only, not for searching unknown databases.

To overcome this limitation, we use the Ry layer as the *ansatz* in the Variational Quantum Search (VQS) algorithm [35], designed for unstructured database searches without prior knowledge of the target's position. Our experiments validate the effectiveness of the VQS with the Ry layer for n up to 26. This shows that the use of shallow-depth parameterized quantum circuits, like the Ry layer, in variational quantum algorithms [36–39], such as the VQS, offers an exponential advantage over GSA in circuit depth for up to 26 qubits. This promising approach opens new avenues for improving the efficiency of quantum search algorithms, potentially leading to quantum supremacy in solving critical problems mentioned above.

The VQS algorithm presented in this paper offers practical value, as it can identify the desired element in an unstructured database without prior knowledge of its index, thereby serving a similar purpose to Grover's search algorithm. The VQS, utilizing a single layer of Ry gates, demonstrates the ability to efficiently amplify the probability of locating a good element without any prior information.

The second contribution of this work is a proof demonstrating that one layer of Ry gates is always sufficient to amplify the probability of a desired element from an extremely small initial value ($1/2^n$ where n is the number of qubits) to a probability close to 1. In this proof, the assumption of knowing the index of the good element (which could be any value between 1 and 2^n) is made to simplify the proof process. Given this assumption, the Ry layer or HX layer can be easily constructed, as shown in Algorithms 1 and 2 of this paper, to amplify the probability of the desired element. This proof is intended to validate the perfect reachability of the ansatz (i.e., the single layer of Ry gates) used by the VQS, and the proof itself is solid under the given assumption. Reachability means that using a single layer of Ry gates, there always exist parameters in Ry (θ) (where θ is the parameter) such that this layer can amplify the probability of any unknown element from an extremely small initial value ($1/2^n$) to a probability close to 1.

Note that the proof and the VQS are independent components of the work, each serving a distinct purpose. The proof relies on prior knowledge for the sake of simplicity in demonstrating the perfect reachability of the ansatz, while the VQS algorithm itself does not require prior knowledge and is applicable to practical database search scenarios. Therefore, each part holds its own value.

To provide a more complete context for the reader, we briefly discuss the physical implementation of qubits, which plays a crucial role in the realization of quantum algorithms. While this paper primarily focuses on the mathematical modeling of quantum circuits, it is important to acknowledge the physical aspects related to the construction of qubits, as they directly influence the practical feasibility of such models.

Various physical systems have been proposed and demonstrated for the construction of qubits, each with its own advantages and limitations. Charge qubits, spin qubits, light qubits, and quantum dots are among the most commonly explored structures. *Charge qubits* typically use superconducting materials to manipulate charge states, whereas *spin qubits* rely on the manipulation of the spin of a single electron, often in a semiconductor setting [40]. *Light qubits*, or *photonic qubits*, utilize photon polarization, where quantum information can be encoded in multiple ways, including the polarization of photons (horizontal or vertical) or their paths of travel (path encoding) [1,40,41]. *Quantum dots*, on the other hand, can trap single electrons and are used as a platform for both charge and spin qubits [41–43]. For further details on the different physical structures used to construct qubits, we refer the reader to Refs. [1,40,41].

2. Method

Vector Forms of Three Quantum States. We use n qubits and n Hadamard gates to create an equal superposition of all 2^n elements, as shown on the left-hand side of the leftmost dashed red line in Figure 1. Here, we provide the vector forms of three quantum states, namely, $|0,\psi_0\rangle$, $|\psi_1\rangle$, and $|\psi_2\rangle$, which are respectively indicated in the three dashed red lines in Figure 1a.

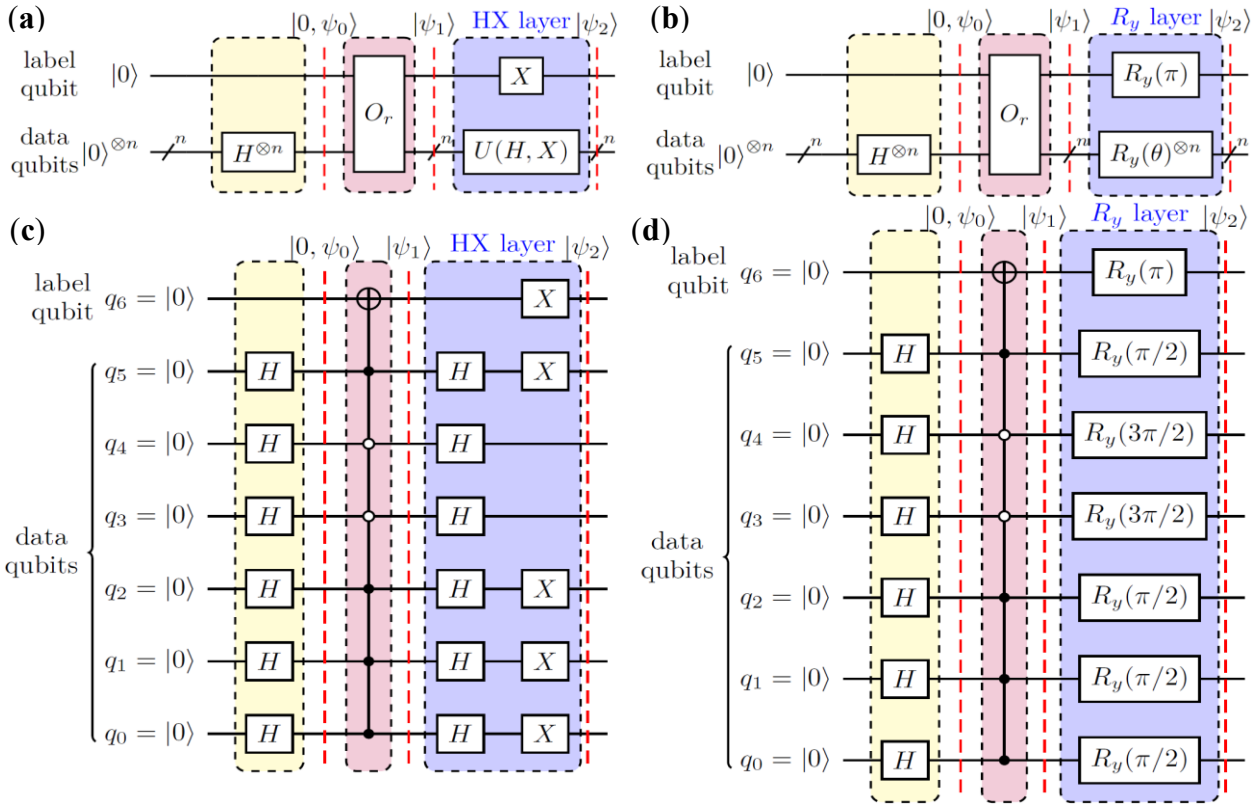

Figure 1. The quantum circuit used to generate an n-qubit database and amplify the probability of the only good element in it to nearly 1. (**a**,**b**) The quantum circuits in compact form for n-qubit data using an HX layer and an R_y layer (the blue blocks in (**a**,**b**)), respectively, where the ***HX layer*** consists of Hadamard and X gates and the R_y ***layer*** consists of $R_y(\theta)$ gates. (**c**,**d**) The detailed circuits for n = 6 using the HX and R_y layers, respectively. The yellow block, excluding the label qubit, generates a state that is an equal superposition of a single good element and (2^n-1) bad elements, i.e., all elements have the same initial probability. The red block (***Oracle***) labels the good element as $|1\rangle$ at the label qubit and assigns $|0\rangle$ to all bad elements. The blue block amplifies the probability of the good element to nearly 1. The label qubit is the highest (most significant) one. In panel (**a**), the circuit $U(H,X)$ consists of Hadamard and X gates. In panels (**c**,**d**), $n=6$ and the position index of the good element is 39 (its binary form is 100111).

The state $|0,\psi_0\rangle$ is the initial state where the label qubit is in the $|0\rangle$ state, and the data qubits are in an equal superposition of all possible basis states, as generated by the Hadamard gates on each data qubit. Mathematically, it is written as $|0\rangle\otimes(|0\rangle+|1\rangle)^{\otimes n}$, which is an equal superposition of all 2^n states and can be represented in the following vector form:

$$|0,\psi_0\rangle = [\underbrace{1_0^{\mathrm{b}},\cdots,1_{k-1}^{\mathrm{b}},1_k^{\mathrm{g}},1_{k+1}^{\mathrm{b}},\cdots,1_{N-1}^{\mathrm{b}}}_{\text{1st half: } N \text{ elements}},\underbrace{0,0,\cdots,0}_{N\text{ elements}}]^T/\sqrt{N} \quad (1)$$

where $N=2^n$, superscripts b and g indicate bad and good elements, respectively, and subscripts 0~N−1 represent the index of an element in the vector. Throughout this paper, the index always counts from 0. For example, 1_k^{g} denotes the k^{th} element as a good element.

The state $|\psi_1\rangle$ is the result of applying the oracle O_r to the initial superposition state $|0,\psi_0\rangle$. The oracle 'marks' the 'good' state by moving the good element from the first half of the vector to the corresponding position in the second half. The relationship between $|\psi_1\rangle$ and $|0,\psi_0\rangle$ can be represented as follows:

$$|\psi_1\rangle = O_r|0,\psi_0\rangle = [\underbrace{1_0^{\mathrm{b}},\cdots,1_{k-1}^{\mathrm{b}},0_k^{\mathrm{g}},1_{k+1}^{\mathrm{b}},\cdots,1_{N-1}^{\mathrm{b}}}_{\text{1st half: } N \text{ elements}},\underbrace{0,\cdots,0,1_{N+k}^{\mathrm{g}},0,\cdots,0}_{\text{2nd half: } N \text{ elements}}]^T/\sqrt{N} \quad (2)$$

where oracle O_r is implemented as $C^n(X)$, an n-qubit-controlled X gate, as shown in Figure 1b. As indicated in Equation (2), $C^n(X)$ changes the index of the good element from k to $N + k$.

After the oracle, the HX or R_y layers are applied to further amplify the probability of the "good" state. This results in the state $|\psi_2\rangle$, which has an increased amplitude for the "good" element, making it more likely to be measured. For ease of analysis, we express $|\psi_2\rangle$ in the following vector form:

$$|\psi_2\rangle = [\beta_0,\beta_1,\cdots,\beta_{N-1},\beta_N,\cdots,\beta_{2N-1}]^T \quad (3)$$

where

$$\sum_{i=0}^{2N-1}|\beta_i|^2 = 1 \quad (4)$$

Here, we propose Algorithm 1 to construct the HX layer, a two-layer circuit comprising exclusively Hadamard and X gates. The combination of the HX layer and the oracle (the red blocks in Figure 1) has the same purpose as GSA: amplifying the probability of the good element to nearly 1.

To better understand Algorithm 1, we provide three examples (k = 5, 8, and 39) of the HX layer generated by Algorithm 1 in the next three paragraphs.

For k = 5, its binary form is $j_2j_1j_0 = 101$. Then, we have $Y_2 \otimes Y_1 \otimes Y_0$. By replacing Y_2 and Y_0 with XH as $j_2 = j_0 = 1$, and replacing Y_1 with H as $j_1 = 0$, we can obtain the following:

$$\begin{aligned} XH \otimes H \otimes XH &= \frac{1}{\sqrt{2}}\begin{bmatrix}1 & -1\\ 1 & 1\end{bmatrix} \otimes \frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix} \otimes \frac{1}{\sqrt{2}}\begin{bmatrix}1 & -1\\ 1 & 1\end{bmatrix} \\ &= \frac{1}{\sqrt{8}}\begin{bmatrix} 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \end{bmatrix} \end{aligned} \quad (5)$$

which shows that row 5 is the only all-1 row (ignoring the coefficient $\frac{1}{\sqrt{8}}$). It is essential to note that throughout this paper, the row index starts from 0, i.e., the first row is row 0.

Algorithm 1. Pseudo code for generating the HX layer (the blue block in Figure 1a).

Input: the number of qubits n and the index of the good element in decimal form, $k, \forall k \in [0, 2^n - 1]$.
Output: quantum gates in the HX layer.

```
Convert k into the binary form b_{n-1}b_{n-2}···b_1b_0.
Add an X gate in the label qubit (the most significant qubit).
Let m = n
while m ≥ 1
    if b_{m-1} = 1
    Add a Hadamard gate followed by an X gate to qubit q_{m-1}.
    else
        Add a Hadamard gate to qubit q_{m-1}.
    m ← m−1
```

For k = 8, its binary form is 1000. Then we have

$$XH \otimes H \otimes H \otimes H = \frac{1}{\sqrt{2}}\begin{bmatrix}1 & -1\\ 1 & 1\end{bmatrix} \otimes \frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix} \otimes \frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix} \otimes \frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix} \quad (6)$$

We can easily confirm that the tensor product shown in Equation (6) has only one all-1 row (disregarding the coefficient 0.25), positioned in row 8.

For k = 39, its binary form is 100111. Then, we can use $XH \otimes H \otimes H \otimes XH \otimes XH \otimes XH$ to obtain a matrix whose only all-1 row is located at row 39. This is shown in Figure 1c.

Lemma 1. *The tensor product* $\begin{bmatrix}a_{n-1,0}\\ a_{n-1,1}\end{bmatrix} \otimes \cdots \otimes \begin{bmatrix}a_{2,0}\\ a_{2,1}\end{bmatrix} \otimes \begin{bmatrix}a_{1,0}\\ a_{1,1}\end{bmatrix} \otimes \begin{bmatrix}a_{0,0}\\ a_{0,1}\end{bmatrix}$ *results in a column vector with* 2^n *elements. The position index of the element* $a_{n-1,i_{n-1}} \cdots a_{2,i_2} a_{1,i_1} a_{0,i_0}$ *in the column vector is equal to the decimal value of the binary form* $i_{n-1} \cdots i_2 i_1 i_0$*, where* $i_r \in \{0,1\}, \forall\, r \in [0, n-1]$.

Proof of Lemma 1 is provided in the Appendix A. To better understand this, we provide two examples. Consider the column vector associated with $\begin{bmatrix}a_{3,0}\\ a_{3,1}\end{bmatrix} \otimes \begin{bmatrix}a_{2,0}\\ a_{2,1}\end{bmatrix} \otimes \begin{bmatrix}a_{1,0}\\ a_{1,1}\end{bmatrix} \otimes \begin{bmatrix}a_{0,0}\\ a_{0,1}\end{bmatrix}$. The position index of the element $a_{3,0}a_{2,0}a_{1,0}a_{0,1}$ in this vector is 1, represented by its binary form 0001. Similarly, the position index of $a_{3,1}a_{2,0}a_{1,0}a_{0,1}$ in the same vector is 9, with its binary form being 1001.

Theorem 1. *For a given integer* $k \in [0, 2^n - 1]$*, its binary form is* $b_{n-1}b_{n-2} \cdots b_1 b_0$*, where n is the smallest integer satisfying* $k \le 2^n - 1$*. For a tensor product* $Y_{n-1} \otimes Y_{n-2} \otimes \cdots \otimes Y_1 \otimes Y_0$*, where* Y_r *denotes* XH *if* $b_r = 1$ *and* H *if* $b_r = 0, \forall\, r \in [0, n-1]$*, we can express the product as* $M/\sqrt{2^n}$*, where* M *is a* 2^n *by* 2^n *matrix. Then, M has one and only one all-1 row (i.e., each element in the row is 1) located at the k*th *row, with the row index k starting from 0.*

Proof. Note that $X = \begin{bmatrix}0 & 1\\ 1 & 0\end{bmatrix}, H = \frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix}, XH = \frac{1}{\sqrt{2}}\begin{bmatrix}1 & -1\\ 1 & 1\end{bmatrix}$. Since each Y_r has exactly one row of [1 1]/$\sqrt{2}$, the all-1 row in M is formed from the tensor product of the [1 1]/$\sqrt{2}$ row of each Y_r. In other words, if the [1 −1]/$\sqrt{2}$ row is involved in the tensor product, the corresponding result will not yield all 1's. Since no Y_r has two rows of [1 1]/$\sqrt{2}$, M has only one all-1 row.

According to Lemma 1, considering Y_r as a two-row vector $\begin{bmatrix}a_{r,0}\\ a_{r,1}\end{bmatrix}$, the kth row of M (the binary form of k is $b_{n-1}b_{n-2} \cdots b_1 b_0$) is the tensor product of row b_r of all $Y_r, \forall\, r \in [0, n-1]$. When $b_r = 1$, $Y_r = XH$ and its row b_r is [1 1]/$\sqrt{2}$. When $b_r = 0$, $Y_r = H$ and its row b_r is also [1 1]/$\sqrt{2}$. Therefore, row k of M is the tensor product of n items of [1 1]/$\sqrt{2}$, i.e., $([1\ 1])^{\otimes n}/\sqrt{2^n}$. Thus, M's all-1 row is located at the kth row.

Theorem 2. *For the quantum circuit depicted in Figure 1a, where the HX layer is created by Algorithm 1, the probability of obtaining the good element by measuring* $|\psi_2\rangle$ *is* $(1 - 2^{-n})^2$*, where* n *is the number of data qubits.*

Proof. In Figure 1a, the relationship between $|\psi_2\rangle$ and $|\psi_1\rangle$ can be represented as follows:

$$|\psi_2\rangle = \begin{bmatrix}\mathbf{0} & U(H,X)\\ U(H,X) & \mathbf{0}\end{bmatrix} |\psi_1\rangle \quad (7)$$

Algorithm 1 assumes the kth element is the only good element in the vector form of $|\psi_0\rangle$, where the binary form of k is $b_{n-1}b_{n-2} \cdots b_1 b_0$. In the HX layer generated by Algorithm 1, the $U(H,X)$ in Equation (7) is the same as the $Y_{n-1} \otimes Y_{n-2} \otimes \cdots \otimes Y_1 \otimes Y_0$ described in

Theorem 1, where Y_r denotes XH if $b_r = 1$ and H if $b_r = 0, \forall\, r \in [0, n-1]$. According to Theorem 1, the all-1 row of matrix $U(H,X)$ generated by Algorithm 1 is located at row k, i.e., each element in row k of matrix $U(H,X)$ is equal to $1/\sqrt{N}$, where $N = 2^n$. Now, plugging Equation (2) into Equation (7), we can calculate the $(N+k)^{\text{th}}$ element of $|\psi_2\rangle$ as follows:

$$\beta_{N+k} = \left(1/\sqrt{N}\right)(N-1)/\sqrt{N} = 1 - 1/2^n \tag{8}$$

The probability of obtaining the good element is equal to $\beta_{N+k}^2 = (1-2^{-n})^2$. Note that the $(N+k)^{\text{th}}$ element of $|\psi_2\rangle$ corresponds to the k^{th} element of $|\psi_0\rangle$, which is the good element we are interested in measuring.

Comments: The ***goal*** of designing $U(H,X)$ is to let each element in its row k be $1/\sqrt{N}$, which leads to a probability of obtaining the good element equal to $\beta_{N+k}^2 = (1-1/2^n)^2$. This probability value is equal to 0.25, 0.5625, 0.7656, 0.8789, 0.9386, 0.9690, 0.9844, 0.9922, and 0.9961 for n = 1~9, respectively.

To summarize, by using a $C^n(X)$ gate and an HX layer, we successfully amplify the probability of the good element from $1/2^n$ to a value larger than 0.95 and 0.99 for n values greater than 5 and 7, respectively, where we have prior knowledge of the position index of the good element.

Here, Algorithm 2 details the procedure of generating the R_y layer (the blue block in Figure 1b).

Algorithm 2. Pseudo code for generating the R_y layer (the blue block in Figure 1b).

Input: the number of qubits n and the position index of the good element in decimal form, $k, \forall k \in [0, 2^n - 1]$.
Output: quantum gates in the R_y layer.

```
Convert k into the binary form b_{n-1} b_{n-2} ··· b_1 b_0.
Add an R_y(π) gate in the label qubit (the most significant qubit).
Let m = n
while m ≥ 1
    if b_{m-1} = 1
        Add an R_y(π/2) gate to qubit q_{m-1}.
    else
        Add an R_y(3π/2) gate to qubit q_{m-1}.
    m ← m−1
```

Vector Forms of Quantum State $|\psi_2\rangle$. The states $|0,\psi_0\rangle$ and $|\psi_1\rangle$ in Figure 1b are identical to those in Figure 1a. Hence, only state $|\psi_2\rangle$ in Figure 1b is presented here. The R_y layer obtained from Algorithm 2, shown in the blue blocks in Figure 1b, can be expressed as

$$R_y(\pi) \otimes R_y(\theta)^{\otimes n} = \begin{bmatrix} \mathbf{0} & -R_y(\theta)^{\otimes n} \\ R_y(\theta)^{\otimes n} & \mathbf{0} \end{bmatrix} \tag{9}$$

where $R_y(\theta)^{\otimes n}$ is an N by N matrix. For the sake of simplicity, we refer to the matrix given in Equation (9) as the R_y*-layer matrix*.

Similar to Equation (7), based on Equation (9), the states $|\psi_2\rangle$ and $|\psi_1\rangle$ in Figure 1b have the following relationship:

$$|\psi_2\rangle = \begin{bmatrix} \mathbf{0} & -R_y(\theta)^{\otimes n} \\ R_y(\theta)^{\otimes n} & \mathbf{0} \end{bmatrix} |\psi_1\rangle \tag{10}$$

Theorem 3. *For the quantum circuit given in Figure 1b, where the* R_y *layer is generated by Algorithm 2, the probability of obtaining the good element by measuring* $|\psi_2\rangle$ *is* $(1-2^{-n})^2$*, where* n *is the number of data qubits.*

Proof. We are going to establish that the R_y layer and the HX layer created by Algorithm 1 are equivalent, proving the theorem. Like Algorithm 1, Algorithm 2 assumes the k^{th} element is the only good element in $|\psi_0\rangle$.

Given $R_y(\theta) = \begin{bmatrix} \cos(\frac{\theta}{2}) & -\sin(\frac{\theta}{2}) \\ \sin(\frac{\theta}{2}) & \cos(\frac{\theta}{2}) \end{bmatrix}$, we have $R_y\left(\frac{\pi}{2}\right) = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix}$, which is identical to XH. Thus, step 6 of Algorithms 1 and 2 is equivalent.

We can express $R_y(\theta)^{\otimes n}$ as a tensor product $Y_{n-1} \otimes Y_{n-2} \otimes \cdots \otimes Y_1 \otimes Y_0$, where $Y_r, r \in [0, n-1]$, represents either $R_y(\pi/2)$ or $R_y(3\pi/2)$.

When the number of $R_y(3\pi/2)$ gates is *even*, each element in row k of $R_y(\theta)^{\otimes n}$ is 1, disregarding the coefficient $1/\sqrt{2^n}$. Consequently, β_{N+k} given in Equation (8) is also the solution of Equation (10), making it valid for Figure 1b.

When the number of $R_y(3\pi/2)$ gates is *odd*, each element in row k of $R_y(\theta)^{\otimes n}$ is –1, again disregarding the coefficient $1/\sqrt{2^n}$. In this case, we have

$$\beta_{N+k} = -\left(1/\sqrt{N}\right)(N-1)/\sqrt{N} = -1 + 1/2^n \tag{11}$$

As the probability is the square of magnitude β_{N+k}, Equations (8) and (11) result in the same probability. Therefore, regardless of whether the number of $R_y(3\pi/2)$ gates is even or odd, the probability of finding the good element from the output, $|\psi_2\rangle$, in either Figure 1a or Figure 1b is the same. Thus, step 8 of Algorithms 1 and 2 is equivalent. In summary, Algorithms 1 and 2 are equivalent. □

Scalability of Algorithms 1 and 2. The analyses presented for Figure 1a,b above do not impose any restrictions or assumptions regarding the number of qubits. Hence, the conclusions drawn are applicable for any number of qubits. Specifically, Algorithms 1 and 2 each can generate a quantum circuit with its corresponding matrix form having only one all-1 or all-negative-1 row, located at row k, $k \in [0, 2^n - 1]$, for any number of qubits n.

As the location index, k, of the good element in a database is known in advance, Algorithms 1 and 2 are not used to find location of the good element. Instead, they are utilized to prove that a quantum circuit with a depth of 2 or 3 (i.e., the R_y layer or the HX layer, together with a $C^n(X)$ gate, as shown in Figure 1) can significantly amplify the probability of the good element from $1/2^n$ to nearly 1 if n is larger than 5. Note that for the same task GSA requires a quantum circuit whose depth increases exponentially with the number of qubits. This observation highlights the significant *advantage* of the circuits generated by Algorithms 1 and 2, in terms of circuit depth.

Variational Quantum Search (VQS). The VQS [35] is a type of variational quantum algorithm that involves the interaction between classical and quantum computers [38,44]. For the completeness of this paper, we provided some basic information about the VQS. We employed an iterative process between classical and quantum computers to update the ansatz of the VQS, as depicted in Figure 1. We used Equation (12) as the objective function for the optimizer in the VQS, guaranteeing that the *global minimum* objective function is linked to both the optimal parameters for the ansatz and the total probability of good elements being amplified to 1.

In the classical part, an optimizer is used to update the parameter $\boldsymbol{\theta}$ of the ansatz based on the objective function $f(\boldsymbol{\theta})$

$$f(\boldsymbol{\theta}) = -0.5\langle\psi_1|\psi_2\rangle + 0.5\langle\psi_1|Z \otimes I^{\otimes n}|\psi_2\rangle \tag{12}$$

where $|\psi_1\rangle$ and $|\psi_2\rangle$ are the states before and after the ansatz, respectively, as shown in Figure 1 and Figure 2c, Z and I are Pauli Z and the identity matrix, respectively, and n is the number of data qubits. $\langle\psi_1|\psi_2\rangle$ is obtained by measuring the circuit given in Figure 2a, i.e., $\langle Z_1\rangle = \langle\psi_1|\psi_2\rangle$. $\langle\psi_1|Z \otimes I^{\otimes n}|\psi_2\rangle$ is obtained by measuring the circuit given in Figure 2b, i.e., $\langle Z_2\rangle = \langle\psi_1|Z \otimes I^{\otimes n}\psi_2\rangle$. More details are available in Ref. [35].

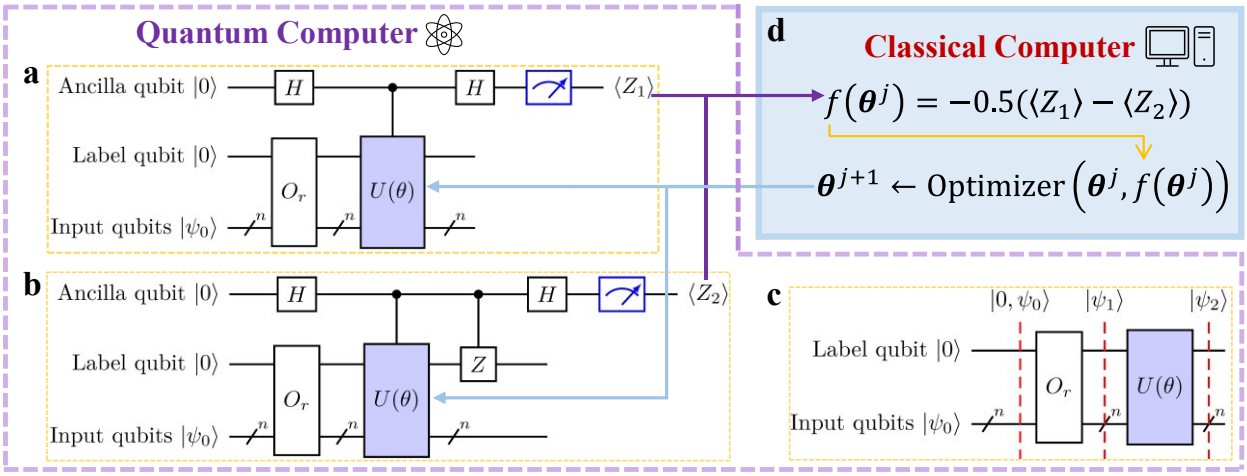

Figure 2. Schematic of the VQS. The VQS uses an iterative process between **a**, **b**, and **d** to find the optimal parameters of the ansatz. (**a**,**b**) Two quantum circuits used in the VQS, respectively. (**c**) A parameterized quantum circuit that is executed once following the final iteration of VQS, using the parameters determined by the last iteration. The ansatz, $U(\boldsymbol{\theta})$, uses the Ry layer given in Figure 1b. (**d**) The classical part of the VQS. The notation with a forward slash and 'n' in the upper right corner indicates n qubits. In the j^{th} iteration of the VQS, the measurement expectations $\langle Z_1\rangle$ and $\langle Z_2\rangle$ from **a** and **b**, respectively, are sent to a classical computer (**d**), which calculates the objective $f(\boldsymbol{\theta}^j)$ and new Ansatz parameters $\boldsymbol{\theta}^{j+1}$. Then, $\boldsymbol{\theta}^{j+1}$ is used in the $(j+1)^{\text{th}}$ iteration.

Notably, Equation (12) represents the inner product between the 2nd half of $|\psi_1\rangle$ and $|\psi_2\rangle$. That is, $f(\boldsymbol{\theta}) = -\langle[\underbrace{0,\cdots,0}_{N},1^{\text{g}}_{N+k},0,\cdots,0]^T/\sqrt{N},[\beta_N,\cdots,\beta_{2N-1}]^T\rangle$.

In the classical part, the optimization problem can be written as

$$\text{minimize } f(\boldsymbol{\theta}) = -\sum_{i=1}^{N_g} 1^{\text{g}}_{N+k}\beta_{N+k}/\sqrt{N} \tag{13}$$

subject to (4).

We assume that k represents the index of the good element in an unknown, unstructure database, though its value is not known a priori, and $k \in [0, N-1]$. The oracle within the quantum circuit can map the index of the good element from k to $N+k$. This process is analogous to Grover's search algorithm, where the oracle modifies the phase of the good element without requiring explicit knowledge of the index of the good element.

It becomes evident that when $\beta_{2N+k} = 1$, the objective function $f(\boldsymbol{\theta})$ achieves its global minimum. According to Equation (4), once $\beta_{2N+k} = 1$, all other β_j values, where $j \neq 2N+k$, become 0. This means that when we measure $|\psi_2\rangle$, we will observe only one state with probability 1. Specifically, the measurement yields

$$|\psi_2\rangle = [\underbrace{0,\cdots,0}_{2N},1^{\text{g}}_{N+k},0,\cdots,0]^T$$

with a probability of 1. Based on this result, the value of k can be determined with absolute certainty.

During the transition from $|\psi_1\rangle$ to $|\psi_2\rangle$, the amplitudes of all bad elements are reduced to zero, while the amplitude of the good element is amplified to 1. This results in a perfect quantum search. Initially, $|\psi_1\rangle$ is a superposition of both bad and good elements, but the final state, $|\psi_2\rangle$, becomes a pure state containing only the good element. As such, the probability of locating the good element becomes 1.

3. Experimental Results

In addition to the theoretical analysis given above, we numerically verify the efficacy of the VQS in identifying the good element in five different unstructured databases, with the results shown in Figure 3. The results related to 2-, 8-, and 14-qubit (20- and 26-qubit)

data states were obtained using Pennylane's default.qubit (lightning.gpu) device on an Intel i5-6500 CPU (NVIDIA A40 × 4 48-GB GPU) [45]. For the results given below, a single layer of $R_y(\theta)$ gates was used as the ansatz of the VQS without prior knowledge of the position index of the good elements.

The initial values of $\boldsymbol{\theta}$ in the ansatz are randomly sampled from a uniform distribution between 0 and 2π. Two termination criteria are employed in the VQS algorithm. (1) *Iteration Limit*: The algorithm concludes when the number of iterations reaches a specified threshold (set to 300 in this study). (2) *Consecutive Small-Change Event*: Alternatively, termination is triggered if a small-change event occurs consecutively for a defined count (set to five in this research). This small-change event is characterized as follows: the absolute value of the relative change in objective functions between two successive iterations is less than a predetermined small value (set to 1×10^{-4} in this investigation). The VQS process terminates upon satisfaction of either criterion, whichever happens first.

Figure 3 shows that the VQS indeed can find the good element as the amplified probability is very close to 1 in most runs out of 100 for $n = 8$, 14, 20, and 26. However, in some runs (22, 16, 16, and 16 for the respective cases) out of 100 runs, the amplified probability is close to 0. The VQS utilizing the R_y layer in the ansatz manages to amplify the probability to nearly 1 in most runs, except for the two-qubit case. The relatively poor performance in the two-qubit case aligns with our theoretical analysis, as explained below.

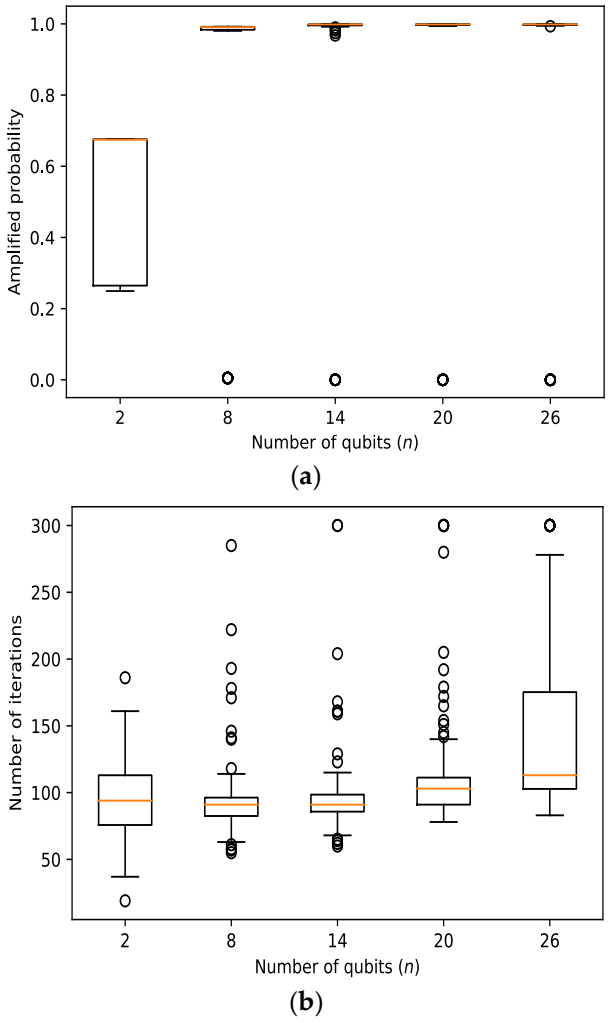

Figure 3. Box plot results from 100 runs of the VQS using the R_y layer as the ansatz for an n-qubit input state. (**a**) The amplified probability of the good element. (**b**) The number of iterations used when a termination criterion is met.

When n equals two and the R_y layer is used as the ansatz in the VQS, according to Equation (8), the probability of finding the good element is $(1 - 1/2^2)^2 = 0.5625$, which is roughly in the middle of the box result for the two-qubit case (the leftmost one in Figure 3a). In other words, the numerical results in Figure 3a validate the analysis given above.

Figure 3b shows the number of iterations required for the VQS to meet the termination criteria across different qubit sizes. The results suggest that while the number of iterations increases slightly as the qubit size grows, it remains within a reasonable range even for larger qubit states like n = 26. The median number of iterations for n = 26 is approximately 150, but the spread is larger, indicating that some runs require significantly more iterations up to the threshold of 300. For smaller qubit systems, such as n = two, the number of iterations is relatively low, with a median around 100, but with a tighter distribution compared with larger qubit states. These results indicate that the VQS is computationally efficient across different database sizes and scales, as the increase in required iterations does not grow exponentially with the number of qubits.

Although in some runs, the amplified probability is close to 0, the majority of runs successfully amplify the probability of the good element to very close to 1, as shown in Figure 3a for n = 8, 14, 20, and 26. This implies that if we run the VQS algorithm twice, it is highly likely that at least one of the runs will identify the good element. In other words, the probability of both runs failing to find the good element is very low. Therefore, we recommend running the VQS algorithm multiple times—twice or a few times—to ensure that the good element is almost always found, maximizing the algorithm's reliability.

The significance of using the VQS with the R_y layer is that it has been proved above that the R_y layer, together with $C^n(X)$, can effectively amplify the probability of the good element from $1/2^n$ to nearly 1 for *any number of qubits* being larger than five. This scalability feature allows us to apply the VQS to databases of any size, while still maintaining a circuit depth of only two (i.e., one R_y layer and one $C^n(X)$ layer). The results validate that the VQS with the R_y layer exhibits an exponential advantage over the GSA in terms of circuit depth for up to 26 qubits.

4. Discussion

The quantum resources used by the VQS primarily include an oracle (we used a multi-control CNOT gate as the oracle) to construct the unstructured database, followed by an ansatz. We emphasize that the VQS in this paper uses only a single layer of Ry gates in its ansatz, which is highly efficient compared with GSA. This is a key advantage of the proposed VQS. We need to run the VQS iteratively, and our experiments show that fewer than 300 iterations are required. While re-establishing a new quantum state is necessary for each iteration, we believe that requiring fewer than 300 runs is efficient and suitable for current Noisy Intermediate-Scale Quantum (NISQ) devices. NISQ systems can handle repeated runs effectively but struggle with circuits that contain too many layers; the VQS, with its shallow depth, is therefore better-suited to NISQ compared with GSA.

The key to the efficiency of VQS lies in finding the optimal parameters for each Ry gate in the ansatz. Regarding the concern about the extensive use of classical computational resources, we argue that for an n-qubit VQS, there are only n parameters to be optimized, which is manageable given that these parameters are used to find good elements in a 2^n-element database. Consequently, the classical resources required are minimal, with limited memory needed because of the small number of parameters. We utilized the ADAM optimizer to determine the optimal parameters, and there are other approaches in the literature to optimize parameters efficiently (though these are beyond the scope of this paper). This line of research will be crucial in enhancing the practical value of VQS for larger databases, and it will be an important focus of our future work.

5. Conclusion

This paper introduces two algorithms to construct the depth-2 HX layer and the depth-1 R_y layer. We prove that either layer, along with the $C^n(X)$ gate, can efficiently amplify the probability of the sole good element in any large unstructured databases from $1/2^n$ to nearly 1, exhibiting an exponential advantage in circuit depth compared with GSA. Both algorithms assume prior knowledge of the good element's position index.

To find the sole good element **without prior knowledge** of its position, we use the VQS with the R_y layer as the *ansatz*. Our experimental results on 8-, 14-, 20-, and 26-qubit unstructured databases show that the VQS successfully finds the good element with a probability close to 1 in 78 to 84 out of 100 independent runs. The results also validate that the VQS with the R_y layer has an exponential advantage over GSA, in terms of circuit depth, for up to 26 qubits. This validation highlights the potential of using low-depth parameterized quantum circuits, such as the VQS, for an unstructured database search and other critical problems, potentially leading to achieving quantum supremacy over classical computing.

Appendix A.: Proof of LEMMA 1

Proof. Here, we use mathematical induction to prove Lemma 1.

For the convenience of expression, we use $P(A,T)$ to denote the position index of element A in the column vector of T and use dec(b) to denote decimal value of a binary form b. As an example, dec(B_m) denotes the decimal value of binary form B_m.

***Base Case* (*n* = 1):** For n = 1, the tensor product results in $\begin{bmatrix} a_{0,0} \\ a_{0,1} \end{bmatrix}$, which is a column vector with 2^1 = 2 elements. The position index of the element a_{0,i_0} in the column vector is equal to the decimal value of the binary form i_0, where $i_0 \in \{0,1\}$.

Inductive Hypothesis (n = m): For the convenience of expression, let $T_m = \begin{bmatrix} a_{m-1,0} \\ a_{m-1,1} \end{bmatrix} \otimes \cdots \otimes \begin{bmatrix} a_{2,0} \\ a_{2,1} \end{bmatrix} \otimes \begin{bmatrix} a_{1,0} \\ a_{1,1} \end{bmatrix} \otimes \begin{bmatrix} a_{0,0} \\ a_{0,1} \end{bmatrix}$. Assume that for some positive integer m, the tensor product T_m results in a column vector with 2^m elements, and the position index of element $A_m = a_{m-1,i_{m-1}} \cdots a_{2,i_2} a_{1,i_1} a_{0,i_0}$ in the column vector of T_m is equal to the decimal value of the binary form $B_m = i_{m-1} \cdots i_2 i_1 i_0$, where $i_r \in \{0,1\}, \forall\, r \in [0, m-1]$, which can be expressed as

$$P(A_m, T_m) = \text{dec}(B_m) \tag{14}$$

***Inductive Step* (*n* = *m* + 1):** Consider the tensor product of m+1 binary vectors

$$T_{m+1} = \begin{bmatrix} a_{m,0} \\ a_{m,1} \end{bmatrix} \otimes \begin{bmatrix} a_{m-1,0} \\ a_{m-1,1} \end{bmatrix} \otimes \cdots \otimes \begin{bmatrix} a_{2,0} \\ a_{2,1} \end{bmatrix} \otimes \begin{bmatrix} a_{1,0} \\ a_{1,1} \end{bmatrix} \otimes \begin{bmatrix} a_{0,0} \\ a_{0,1} \end{bmatrix} \tag{15}$$

Considering that the tensor product T_m results in a column vector with 2^m elements, we know $T_{m+1} = \begin{bmatrix} a_{m,0} T_m \\ a_{m,1} T_m \end{bmatrix}$, which is a column vector with $2^m + 2^m = 2^{m+1}$ elements.

Part 1: When $i_m = 0$, $A_{m+1} = a_{m,0} A_m$. As $P(A_m, T_m)$ =dec(B_m), we have

$$P\left(A_{m+1}, a_{m,0} T_m\right) = P\left(a_{m,0} A_m, a_{m,0} T_m\right) = \text{dec}(B_m) \tag{16}$$

As $B_{m+1} = i_m i_{m-1} \cdots i_2 i_1 i_0 = 0 i_{m-1} \cdots i_2 i_1 i_0$, we know that dec($B_{m+1}$)=dec($B_m$). Then, considering that $a_{m,0} T_m$ is in the first half of T_{m+1} and according to Equation (16), we know

$$P(A_{m+1}, T_{m+1}) = P\left(A_{m+1}, a_{m,0} T_m\right) = \text{dec}(B_m) = \text{dec}(B_{m+1}), \quad i_m = 0 \tag{17}$$

Part 2: When $i_m = 1$, we have $A_{m+1} = a_{m,1} A_m$ and

$$B_{m+1} = i_m i_{m-1} \cdots i_2 i_1 i_0 = 1 i_{m-1} \cdots i_2 i_1 i_0 \tag{18}$$

Then, we know

$$\text{dec}(B_{m+1}) = \text{dec}(B_m) + 2^m \tag{19}$$

As $P(A_m, T_m) = \text{dec}(B_m)$, we have

$$P\left(a_{m,1}A_m, a_{m,1}T_m\right) = \text{dec}(B_m) \tag{20}$$

Then, considering $a_{m,1}T_m$ is in the second half of T_{m+1}, and each half of T_{m+1} has 2^m elements, we have

$$\begin{aligned} &P(A_{m+1}, T_{m+1}) = P\left(a_{m,1}A_m, T_{m+1}\right) = 2^m + \\ &P\left(a_{m,1}A_m, a_{m,1}T_m\right) = \text{dec}(B_m) + 2^m, \ \ i_m = 1 \end{aligned} \tag{21}$$

Considering Equation (19), we can reform Equation (21) as

$$P(A_{m+1}, T_{m+1}) = \ \text{dec}(B_{m+1}), i_m = 1 \tag{22}$$

In summary, Equations. (17) and (22) indicate that whether i_m is 0 or 1, we have $P(A_{m+1}, T_{m+1}) = \ \text{dec}(B_{m+1})$.

Hence, by mathematical induction, Lemma 1 is rigorously proven. □

Funding: This research was supported by the NSF ERI program, under award number 2138702.

Data Availability Statement: No new data were created or analyzed in this study.

Acknowledgments: This work used the Delta system at the National Center for Supercomputing Applications through allocation CIS220136 from the Advanced Cyberinfrastructure Coordination Ecosystem: Services and Support (ACCESS) program, which is supported by National Science Foundation grants #2138259, #2138286, #2138307, #2137603, and #2138296. We acknowledge the use of IBM Quantum services for this work. The views expressed are those of the authors and do not reflect the official policy or position of IBM or the IBM Quantum team.

Conflicts of Interest: The author declares no conflicts of interest in writing this manuscript or in the decision to publish the results.

Reference

1. Nielsen, M.A.; Chuang, I.L. *Quantum Computation and Quantum Information: 10th Anniversary Edition*; Cambridge University Press: Cambridge, UK, 2011.
2. Giri, P.R.; Korepin, V.E. A Review on Quantum Search Algorithms. *Quantum Inf. Process.* **2016**, *16*, 315.
3. Grover, L.K. Quantum mechanics helps in searching for a needle in a haystack. *Phys. Rev. Lett.* **1997**, *79*, 325.
4. Grover, L.K. A Fast Quantum Mechanical Algorithm for Database Search. In Proceedings of the Annual ACM Symposium on Theory of Computing, Philadelphia, PA, USA, 22–24 May 1996; Volume Part F129452.
5. Zalka, C. Grover's quantum searching algorithm is optimal. *Phys. Rev. A* **1999**, *60*, 2746.
6. Bennett, C.H.; Bernstein, E.; Brassard, G.; Vazirani, U. Strengths and Weaknesses of Quantum Computing. *SIAM J. Comput.* **2006**, *26*, 1510. https://doi.org/10.1137/S0097539796300933.
7. Fürer, M. Solving NP-Complete Problems with Quantum Search. In *LATIN 2008: Theoretical Informatics, Proceedings of the LATIN 2008, Búzios, Brazil, 7–11 April 2008*; Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics); Springer: Berlin/Heidelberg, Germany, 2008; Volume 4957, pp. 784–792.
8. Aaronson, S. Guest Column: NP-complete problems and physical reality. *ACM SIGACT News* **2005**, *36*, 30–52.
9. Farhi, E.; Goldstone, J.; Gutmann, S.; Lapan, J.; Lundgren, A.; Preda, D. A Quantum Adiabatic Evolution Algorithm Applied to Random Instances of an NP-Complete Problem. *Science* **2001**, *292*, 472.
10. Byrnes, T.; Forster, G.; Tessler, L. Generalized Grover's Algorithm for Multiple Phase Inversion States. *Phys. Rev. Lett.* **2018**, *120*, 060501.
11. Zhan, J. Quantum Feasibility Labeling for NP-complete Vertex Coloring Problem. *arXiv* **2023**, *arXiv:*2301.01589.
12. Cerf, N.J.; Grover, L.K.; Williams, C.P. Nested quantum search and NP-hard problems. *Appl. Algebra Eng. Commun. Comput.* **2000**, *10*, 311.
13. Shor, P.W. Algorithms for quantum computation: Discrete logarithms and factoring. In Proceedings of the Annual IEEE Symposium on Foundations of Computer Science, FOCS, Santa Fe, NM, USA, 20–22 November 1994; pp. 124–134.
14. Tutul, I.K.; Karimi, S.; Zhan, J. Shallow Depth Factoring Based on Quantum Feasibility Labeling and Variational Quantum Search. *arXiv* **2023**, *arXiv:*2305.19542.
15. Du, Y.; Hsieh, M.H.; Liu, T.; Tao, D. A Grover-search based quantum learning scheme for classification. *New J. Phys.* **2021**, *23*, 023020.
16. Khanal, B.; Rivas, P.; Orduz, J.; Zhakubayev, A. Quantum Machine Learning: A Case Study of Grover's Algorithm. In Proceedings of the 2021 International Conference on Computational Science and Computational Intelligence, CSCI, Las Vegas, NV, USA, 15–17 December 2021; pp. 79–84.

17. Lee, B.; Perkowski, M. Quantum Machine Learning Based on Minimizing Kronecker-Reed-Muller Forms and Grover Search Algorithm with Hybrid Oracles. In Proceedings of the 19th Euromicro Conference on Digital System Design, DSD, Limassol, Cyprus, 31 August–2 September 2016; pp. 413–422.
18. Aïmeur, E.; Brassard, G.; Gambs, S. Quantum speed-up for unsupervised learning. *Mach. Learn.* **2013**, *90*, 261.
19. Liao, Y.; Zhan, J. Expressibility-Enhancing Strategies for Quantum Neural Networks. *arXiv* **2022**, *arXiv:*2211.12670.
20. Zhang, X.M.; Yung, M.H.; Yuan, X. Low-depth quantum state preparation. *Phys. Rev. Res.* **2021**, *3*, 043200.
21. Matos, G.; Johri, S.; Papić, Z. Quantifying the Efficiency of State Preparation via Quantum Variational Eigensolvers. *PRX Quantum* **2021**, *2*, 010309.
22. Brassard, G.; Hoyer, P.; Tapp, A. Quantum cryptanalysis of hash and claw-free functions. In *LATIN'98: Theoretical Informatics, Proceedings of the Third Latin American Symposium, Campinas, Brazil, 20–24 April 1998*; Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics); Springer: Berlin/Heidelberg, Germany, 1997; Volume 1380, pp. 163–169.
23. Mermin, N.D. *Quantum Computer Science: An Introduction*; Cambridge University Press: Cambridge, UK, 2007; ISBN 9780521876582.
24. Preskill, J. Lecture Notes for Physics 229: Quantum Information and Computation. *Calif. Inst. Technol.* **1998**, *16*, 1–8.
25. Brassard, G.; Høyer, P.; Mosca, M.; Tapp, A. Quantum amplitude amplification and estimation. *Contemp. Math.* **2002**, *305*, 53.
26. Roy, T.; Jiang, L.; Schuster, D.I. Deterministic Grover search with a restricted oracle. *Phys. Rev. Res.* **2021**, *4*, L022013.
27. Long, G.L. Grover algorithm with zero theoretical failure rate. *Phys. Rev. A* **2001**, *64*, 022307.
28. Broda, B. Quantum search of a real unstructured database. *Eur. Phys. J. Plus* **2015**, *131*, 1–4.
29. Figgatt, C.; Maslov, D.; Landsman, K.A.; Linke, N.M.; Debnath, S.; Monroe, C. Complete 3-Qubit Grover search on a programmable quantum computer. *Nat. Commun.* **2017**, *8*, 1918.
30. Gilliam, A.; Pistoia, M.; Gonciulea, C.; Chase, J. Optimizing Quantum Search Using a Generalized Version of Grover's Algorithm. *arXiv* **2020**, *arXiv:*2005.06468.
31. Farhi, E.; Goldstone, J.; Gutmann, S.; Sipser, M. Quantum Computation by Adiabatic Evolution. *arXiv* **2000**, *arXiv:*quant-ph/0001106. https://doi.org/10.48550/arXiv.quant-ph/0001106.
32. Roland, J.; Cerf, N.J. Quantum search by local adiabatic evolution. *Phys. Rev. A* **2002**, *65*, 042308.
33. Morales, M.E.S.; Tlyachev, T.; Biamonte, J. Variational learning of Grover's quantum search algorithm. *Phys. Rev. A* **2018**, *98*, 062333.
34. Boyer, M.; Brassard, G.; Hoeyer, P.; Tapp, A. Tight bounds on quantum searching. *Fortschritte der Phys.* **1996**, *46*, 493.
35. Zhan, J. Variational Quantum Search with Shallow Depth for Unstructured Database Search. *arXiv* **2022**, *arXiv:*2212.09505. https://doi.org/10.48550/arXiv.2212.09505.
36. Cerezo, M.; Arrasmith, A.; Babbush, R.; Benjamin, S.C.; Endo, S.; Fujii, K.; McClean, J.R.; Mitarai, K.; Yuan, X.; Cincio, L.; et al., Variational quantum algorithms. *Nat. Rev. Phys.* **2021**, *3*, 625.
37. McClean, J.R.; Romero, J.; Babbush, R.; Aspuru-Guzik, A. The theory of variational hybrid quantum-classical algorithms. *New J. Phys.* **2016**, *18*, 023023.
38. Kübler, J.M.; Arrasmith, A.; Cincio, L.; Coles, P.J. An adaptive optimizer for measurement-frugal variational algorithms. *Quantum* **2020**, *4*, 263.
39. Peruzzo, A.; McClean, J.; Shadbolt, P.; Yung, M.H.; Zhou, X.Q.; Love, P.J.; Aspuru-Guzik, A.; O'Brien, J.L. A variational eigenvalue solver on a photonic quantum processor. *Nat. Commun.* **2014**, *5*, 4213.
40. Hidary, J.D. *Quantum Computing: An Applied Approach*; Springer: Cham, Switzerland, 2019.
41. Wong, T.G. *Introduction to Classical and Quantum Computing*; Rooted Grove: Omaha, Nebraska, 2022.
42. Bosco, S.; Benito, M.; Adelsberger, C.; Loss, D. Squeezed hole spin qubits in Ge quantum dots with ultrafast gates at low power. *Phys. Rev. B* **2021**, *104*, 115425.
43. Stavrou, V.N.; Veropoulos, G.P. Significance of an external magnetic field on two-phonon processes in gated lateral semiconductor quantum dots. *Solid State Commun.* **2014**, *191*, 10–13.
44. Huang, H.L.; Xu, X.-Y.; Guo, C.; Tian, G.; Wei, S.-J.; Sun, X.; Bao, W.-S.; Long, G.-L. Near-Term Quantum Computing Techniques: Variational Quantum Algorithms, Error Mitigation, Circuit Compilation, Benchmarking and Classical Simulation. *Sci. China Phys. Mech. Astron.* **2023**, *66*, 250302.
45. Bergholm, V.; Izaac, J.; Schuld, M.; Gogolin, C.; Ahmed, S.; Ajith, V.; Alam, M.S.; Alonso-Linaje, G.; AkashNarayanan, B.; Asadi, A.; et al. PennyLane: Automatic Differentiation of Hybrid Quantum-Classical Computations. *arXiv* **2018**, *arXiv:*1811.04968. https://doi.org/10.48550/arxiv.1811.04968.