electronics

MDPI

Article

Optimizing the Quantum Circuit for Solving Boolean Equations Based on Grover Search Algorithm

Hui Liu [1,2,*], Fukun Li [3] and Yilin Fan [3]

[1] College of Computer and Information Engineering, Henan Normal University, Xinxiang 453007, China
[2] Key Laboratory of Artificial Intelligence and Personalized Learning in Education of Henan Province, Xinxiang 453007, China
[3] College of Software, Henan Normal University, Xinxiang 453007, China
* Correspondence: liuhui@htu.edu.cn or liuhui806@126.com

Abstract: The solution of nonlinear Boolean equations in a binary field plays a crucial part in cryptanalysis and computational mathematics. To speed up the process of solving Boolean equations is an urgent task that needs to be addressed. In this paper, we propose a method for solving Boolean equations based on the Grover algorithm combined with preprocessing using classical algorithms, optimizing the quantum circuit for solving the equations, and implementing the automatic generation of quantum circuits. The method first converted Boolean equations into Boolean expressions to construct the oracle in the Grover algorithm. The quantum circuit was emulated based on the IBM Qiskit framework and then simulated the Grover algorithm on this basis. Finally, the solution of the Boolean equation was implemented. The experimental results proved the feasibility of using the Grover algorithm to solve nonlinear Boolean equations in a binary field, and the correct answer was successfully found under the conditions that the search space was 2^{21} and three G iterations were used. The method in this paper increases the solving scale and solving speed of Boolean equations and enlarges the application area of the Grover algorithm.

Keywords: algorithm optimization; Grover quantum algorithm; solving Boolean equations; oracle; IBM Qiskit

Citation: Liu, H.; Li, F.; Fan, Y. Optimizing the Quantum Circuit for Solving Boolean Equations Based on Grover Search Algorithm. *Electronics* **2022**, *11*, 2467. https://doi.org/10.3390/electronics11152467

Academic Editor: Gwanggil Jeon

Received: 30 June 2022
Accepted: 3 August 2022
Published: 8 August 2022

1. Introduction

A large number of scientific problems are often ascribed to the problem of solving Boolean equations. Research on quantum algorithms for solving nonlinear Boolean equations in binary fields is a critical problem in computational mathematics, cryptographic, design, and analysis.

Feynman first proposed the concept of a "quantum computer" in 1981, pointing out that only a quantum computer can effectively simulate a quantum system. In 1985, Deutsch proposed the Quantum Turing Machine for the first time and designed the Deutsch algorithm [1], which laid the basic idea for a quantum algorithm. In 1992, Deutsch and Jozsa jointly proposed the Deutsch–Jozsa algorithm [2], which was the first time exponential acceleration of a classical algorithm was achieved. The Shor algorithm proposed by Shor [3] in 1994 could solve the core large prime factorization problem in an RSA cryptosystem with polynomial time complexity and had better performance at an exponential level compared with classical algorithms. In 1996, Grover proposed the Grover quantum search algorithm, which could search for targets in an unsorted database with a time complexity of $O\left(\sqrt{N}\right)$ [4–7].

In recent years, more scholars have devoted themselves to quantum algorithm research with the development of scientific theories on quantum computing. They have obtained many valuable findings, such as the problem of solving polynomial systems with noise, integer factorization problems, and knapsack problems. The Grover algorithm can further

generalize the problem of solving Boolean equations and expand it to the search problem of finding a target in the solution space of equations. Younes et al. [8] proposed a hybrid quantum search engine. When the number of solutions in the search space exceeded N/3, the algorithm's time complexity was O(1). Zhou et al. [9] optimized the Grover algorithm by changing the phase rotation angle. When the number of solutions in the search space exceeded N/4, the target solution could be found with a success probability of no less than 98.01% in only one search in the search space. When the number of solutions in the search space equaled N/2, the probability of being found was 1. Zhu et al. [10] developed a robust quantum search algorithm based on the Grover–Long algorithm, with the same time complexity as the Grover algorithm and a high tolerance for M/N uncertainty. In [11], by adaptively changing the search space of different target component proportions, the Grover algorithm achieved overall improvement in the probability of success. Liu et al. [12] proposed a new method of integer factorization based on the Grover search algorithm. Zhang et al. [13] solved the problem of route planning for two manipulators based on the Grover algorithm. Xie et al. [14] proposed an improved quantum search algorithm and applied it to solve the kernel properties of rough sets, which improved the probability of the target component as a whole. Hou et al. [15] proposed an algorithm that leveraged the quadratic speed-up offered by the Grover search to achieve a quantum algorithm for the knapsack problem that showed improvement from classical algorithms. Ju et al. [16] showed how quantum Boolean circuits could be used to implement the oracle and the inversion-about-average function in Grover's search algorithm and proposed a template of quantum circuits that was capable of searching for the solution of a certain class of one-way functions. Harrow et al. [17] proved that any classical algorithm for solving linear systems of equations generically required exponentially more time than our quantum algorithm. At present, the main problems faced by using the Grover algorithm to solve Boolean equations are still the depth of the quantum circuit and the probability of quantum states.

IBM Qiskit [18–22] is an open-source quantum-computing framework that provides basic qubits, quantum logic gates, and quantum simulators on which quantum circuits and various quantum algorithms can be simulated. In this paper, we use the IBM Qiskit framework to simulate the quantum circuit of a Grover algorithm, solve Boolean equations of different variables, and prove the feasibility of the Grover algorithm in solving the equations. The experimental results show that the solutions of the equations can be found with a high probability when the number of variables is smaller than 21.

Section 2 presents the main steps of the Grover algorithm and the characteristics of nonlinear Boolean equations in a binary field. Section 3 introduces the principle of converting Boolean equations into the quantum circuit and uses the Grover algorithm to solve Boolean equations on this premise. Section 4 optimizes the quantum circuit for solving Boolean equations from three aspects: classical algorithm preprocessing, ancilla bit multiplexing, and the uses of a phase quantum gate. Section 5 introduces the experimental results of solving Boolean equations with different variables, as well as the relationship between the number of variables, the depth of the circuit, and the number of qubits, and analyzes the experimental results. Section 6 concludes this paper and gives an outlook on future work.

2. Grover Algorithm

2.1. The Main Steps of the Grover Algorithm

In 1996, Grover [23] proposed an unstructured quantum search algorithm, a typical quantum-computing algorithm. Its time complexity for searching an unordered database is $O\left(\sqrt{N}\right)$, which is a polynomial speed-up compared to the time complexity of $O(N)$ for the classical search algorithm. The Grover algorithm does not consider the properties of a problem's internal structure when searching for a solution but only focuses on the characteristics of the search items. Therefore, the algorithm has strong versatility and has a secondary speed-up effect on many classic search problems [3]. The principle of the Grover algorithm is mainly to exploit the powerful parallel ability of quantum to continuously

iterate the quantum state, change the probability range to maximize that of the target state, and finally, to obtain it through one measurement. The algorithm mainly includes three parts: the preparation of quantum superposition, the G operator, and measurement. The G operator includes two parts, namely, the oracle and Grover diffusion. The Grover algorithm's complete quantum circuit framework is shown in Figure 1.

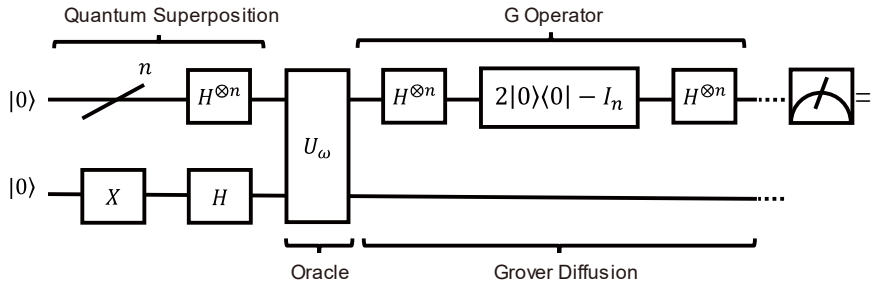

Figure 1. Grover search algorithm quantum circuit framework diagram.

1. Prepare the quantum superposition: The Hadamard Transform is applied to the initial state to obtain a uniform superposition state of equal probability amplitude $|\phi\rangle$, as shown in Equation (1):

$$|\phi\rangle = H^{\otimes n}|00\cdots 0\rangle = \frac{1}{\sqrt{N}}\sum_{x=0}^{N-1}|x\rangle \tag{1}$$

   where $|\phi\rangle$ is the uniform superposition of the initial state, $N = 2^n$, and N is the problem's solution space.
2. Construct the oracle: The oracle marks the target state and reverses its phase, while the phases of other states remain unchanged, that is, $O|x\rangle \to (-1)^{f(x)}|x\rangle$. If $|x\rangle$ is the target state, then $f(x) = 1$, and the phase of the state is reversed at this time; otherwise, $f(x) = 0$.
3. Grover diffusion: By constructing a unitary matrix U_ϕ, the superposition state is subjected to the averaged inversions of the probability amplitude. The probability amplitude of the target state is flipped, and the other states remain the same, thereby increasing the state's amplitude. U_ϕ is as in Equation (2):

$$U_\phi = 2|\phi\rangle\langle\phi| - I \tag{2}$$

4. Repeats of the iterations: Repeat steps 2 and 3. In theory, when the number of iterations reaches $\left\lceil \frac{\pi}{4}\sqrt{\frac{N}{M}} \right\rceil$ (N is the number of elements in the search space, and M is the number of targets to be searched), the target state is searched with a probability close to 1 [24].

2.2. Application of the Grover Algorithm in Solving Boolean Equations

The Boolean equations to be solved in this paper had the following characteristics:

1. The value range of the variable is the binary field {0, 1}.
2. Operations include addition, subtraction, multiplication, and division. This article only focuses on Boolean equations involving addition and multiplication operations, which can represent subtraction and division operations.
3. Each term of the equation is a single-variable or two-variable multiplication term.
4. The number of variables is the same as the number of equations.

Equation (3) shows three-variable Boolean equations, and its result is in Equation (4):

$$\begin{cases} x_1 + x_2 = 0 \\ x_1 * x_2 = 1 \\ x_1 + x_2 * x_3 = 1 \end{cases} \tag{3}$$

The result is:

$$\begin{cases} x_1 = 1 \\ x_2 = 1 \\ x_3 = 0 \end{cases} \tag{4}$$

We used two-dimensional arrays to store the coefficients of the Boolean equations and one-dimensional arrays to store the values of the Boolean equations. We traversed each equation in the code, added the corresponding quantum logic gate, and drew it to run on the quantum circuit. When the Grover search algorithm was applied to the Boolean equations, first the addition and multiplication operations in the Boolean equations were respectively equivalent to the XOR and AND operations in the Boolean logic relationship, so the equations were converted into a Boolean logic formula [25], and Boolean logic expressions were optimized using classical algorithms to simplify the Boolean equations. Second, we built a quantum circuit for solving Boolean equations and implemented the preparation of a uniform superposition state in the Grover algorithm, the oracle, and the Grover diffusion on the quantum circuit. The oracle and Grover diffusion operations were collectively called the G operator, and the target's probability was increased by repeatedly iterating the G operator. Finally, the quantum circuit was measured, and the correct result was obtained. The method proposed in this paper solved twenty-one-variable Boolean equations using the Grover algorithm. When the number of iterations of the quantum circuit reached three, the correct answer could be measured, and the probability of finding the answer was the greatest at this time.

3. Simulation of the Grover Algorithm Based on the Qiskit Framework

Qiskit is an open-source quantum-computing framework developed by IBM with essential quantum bits providing various quantum simulators, quantum algorithms, and quantum gates, such as the H gate, Pauli-X gate, CZ gate, CNOT gate, Toffoli gate, and Phase gate, as shown in Figure 2. Qiskit also provides a Python API that can program and simulate quantum algorithms on classical computers.

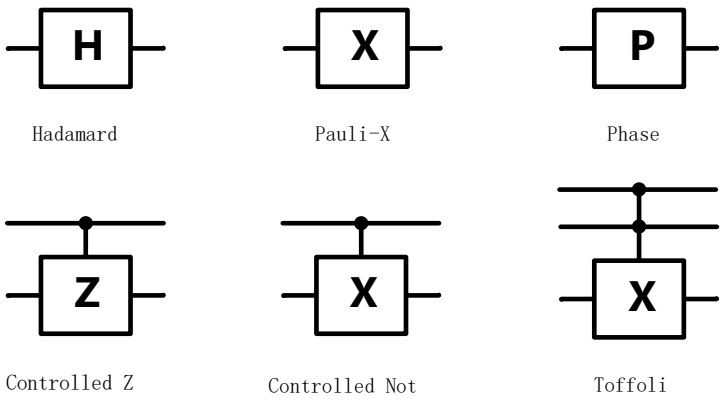

Figure 2. Partial quantum gate.

In this paper, we used Qiskit to simulate the Grover algorithm on the Jupyter Notebook, tried to discover the implementation of the algorithm on Boolean equations, and built a quantum circuit for the Grover algorithm to solve the equations. As shown in Figure 3, when building a quantum circuit, according to the characteristics of Boolean equations, the addition operations of the Boolean equations are equivalent to the XOR operation in

the Boolean logic relationship, which can be simulated with two CX quantum gates on the quantum circuit.

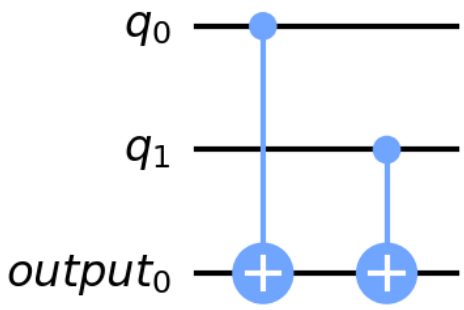

Figure 3. Quantum circuit logic XOR operation.

As shown in Figure 4, the multiplication operations of the Boolean equations are equivalent to the AND operation in the Boolean logic relationship, which can be simulated with a CCX quantum gate on the quantum circuit.

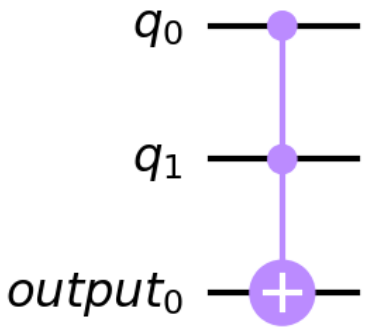

Figure 4. Quantum circuit logic AND operation.

In this paper, the combination of two CX quantum gates was called the XOR element, and one CCX quantum gate was called the AND element. The oracle part of Boolean equations could be composed of both XOR and AND elements. This way of expressing Boolean equations on quantum circuits provided feasibility for solving the equations. Figure 5 is a quantum circuit diagram of Equation (3).

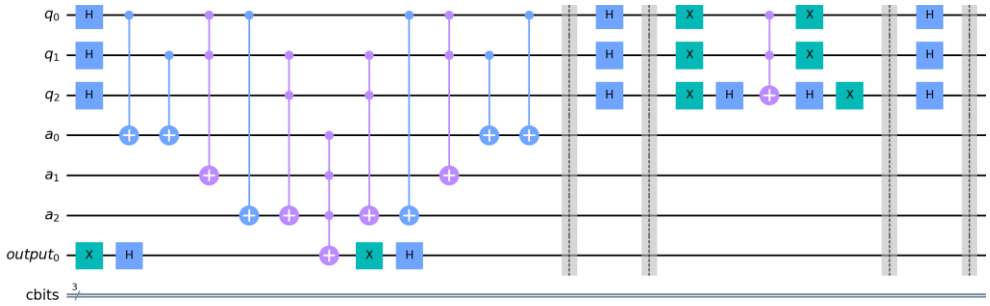

Figure 5. Quantum circuit diagram of three-variable Boolean equations.

The quantum circuit first used the XOR and AND elements to construct the oracle in the quantum circuit (the area before the first dashed line in Figure 5) and introduced four ancilla bits to form the quantum circuit of the Boolean equations. Then, the phase of the quantum state was flipped by constructing the U_ϕ operator (the region between the first and fourth dashed lines in Figure 5), increasing the probability of the target state. Finally, the quantum state was measured using a measurement gate to obtain the target state. The quantum circuit could search for the correct answer with a probability close to 1 after one iteration of the quantum circuit.

4. Quantum Circuit Optimization

This paper focused on optimizing quantum circuits for solving Boolean equations to increase the solving scale. The main factors affecting the solving scale were the depth of the quantum circuit and the number of qubits. Reducing the depth of quantum circuits could be achieved with classical algorithm preprocessing. The quantum circuit for solving Boolean equations contained two kinds of qubits: the variable bits representing the variables of the equation and the ancilla bits representing the equation. With the increase in the solving scale of Boolean equations, ancilla bits accounted for more than 50% of all qubits, which significantly impacted the program's execution. When building a quantum circuit, the variable bits could not be reduced, and the ancilla bits could be reduced by multiplexing the bits and using the phase quantum gate to replace the CX/CCX/MCX quantum gate.

The quantum circuit optimization for solving Boolean equations included three steps: classical algorithm preprocessing, multiplexing the ancilla bits, and replacing CX/CCX/MCX quantum gates with phase quantum gates. The framework of the quantum circuit optimization process is shown in Figure 6.

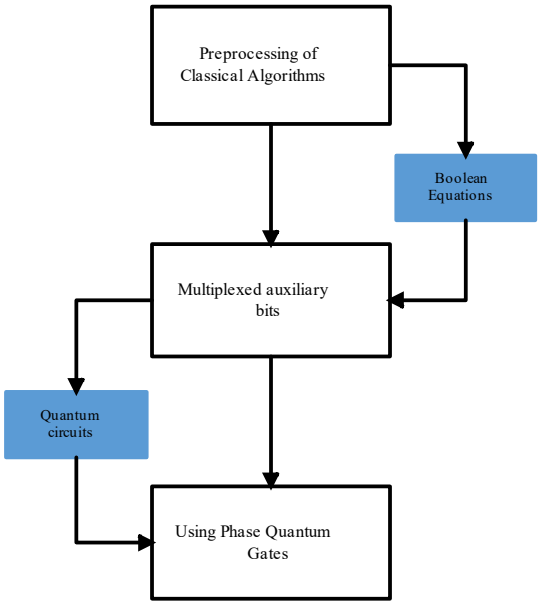

Figure 6. The main procedure for optimization of the quantum circuit.

4.1. Preprocessing of Classical Algorithms

When the number of Boolean equation variables is too large, there are many identical terms in different equations in the equation system. According to the operation rules of Boolean equations, these identical terms can be eliminated from each other, thereby reducing the degree of complexity of Boolean equations. Equation (5) is a five-variable Boolean equation:

$$\begin{cases} x_1 + x_2 = 1 \\ x_2 + x_1 * x_4 = 0 \\ x_3 + x_4 + x_5 = 0 \\ x_1 + x_3 + x_4 = 1 \\ x_1 * x_2 + x_3 + x_4 = 0 \end{cases} \tag{5}$$

From Boolean Equation (5), it can be seen that Equation (3) ($x_3 + x_4 + x_5 = 0$) and Equation (4) ($x_1 + x_3 + x_4 = 1$) have the same terms, and the addition of these two equations can eliminate the terms to obtain a new equation ($x_1 + x_5 = 1$). The degree of complexity of the new equation is less than those of Equations (3) and (4), so replacing Equation (3) or (4)

with the new equation can simplify the Boolean equations without changing the solutions of the equations. This leads to the following general formula:

$$x_a + x_b + x_c + \cdots + x_q = 1 \tag{6}$$

$$x_a + x_c + x_d + \cdots + x_p = 0 \tag{7}$$

Adding Equation (6) to Equation (7) results in Equation (8):

$$x_b + x_d + \cdots + x_p + x_q = 1 \tag{8}$$

Among them, *a*,*b*,*c*,..., *z* are discrete, increasing, random integers. If the number of terms of Equation (7) is less than the numbers of terms for Equations (5)–(7) can be added to the original Boolean equations to replace Equation (5) or (6) to optimize the degree of complexity of the Boolean equations.

4.2. Multiplexing Ancilla Bits

When the solving scale of Boolean equations reaches 14 variables, the total number of quantum qubits required is 29, of which the number of ancilla bits is 15, accounting for half of all the qubits. In building the quantum circuit for Boolean equation solving, an ancilla bit represents the state of an equation. The Boolean equations are grouped in an arithmetic-sequence-like method, and one ancilla bit can represent the states of multiple equations, which further reduces the number of ancilla bits and increases the solving scale. Formula (9) is used to find the minimum integer value of *t*:

$$s \leq \frac{t \cdot (t+1)}{2} \tag{9}$$

In (9), *s* is the number of equations, and *t* is the number of equations in the first group. On this basis, Algorithm 1 is used to find the grouping of Boolean equations.

Algorithm 1. Find the grouping of Boolean equations.

Require: g[100], T, i
Ensure: g[100] = {0}, T = 0, I = 0
1: while T $\leq$ s do
2: g[i] = t
3: T = T + t
4: i = i + 1
5: t = t $-$ 1
6: end while
7: if T $\equiv$ s then
8: return g[]
9: else T > s then
10: g[i] = (s + T)/2
11: return g[100]
12: end if

The value stored in g[100] is the number of equations in each group.

4.3. Using Phase Quantum Gates

According to the characteristics of the oracle in the Grover algorithm, the oracle marks the target state by inverting the phase of the quantum state [6]. In the traditional Grover algorithm, the oracle is usually constructed using CX/CCX/MCX quantum gates, resulting in the need to use an additional ancilla bit. In theory, building an oracle with phase quantum gates can reduce one ancilla bit, as well as the depth of the quantum circuits, compared to CX/CCX/MCX gates, thereby increasing the rate and scale of solving Boolean equations. The replacement of CX/CCX/MCX quantum gates with phase quantum gates is shown in

Figure 7. We designed several sets of experiments to compare them and found that using phase quantum gates instead of CX/CCX/MCX gates could effectively improve the speed and scale of solving Boolean equations.

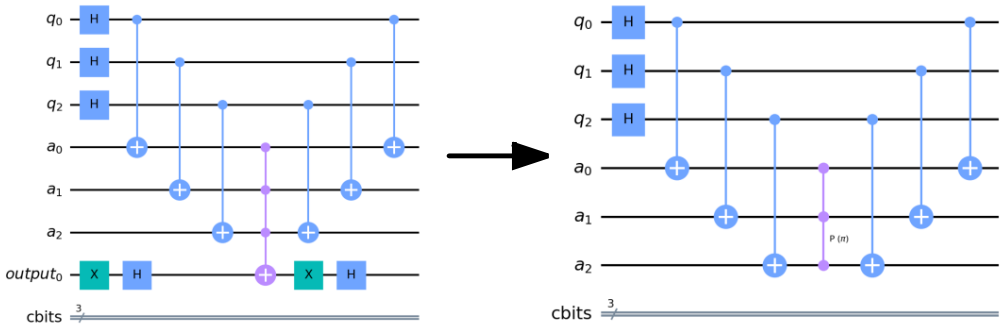

Figure 7. Using phase quantum gate.

Through these methods, we successfully optimized the quantum circuit. The solution scale and speed of solving Boolean equations were improved. However, these optimization methods were limited to Boolean equations in a binary field. The multiplexing ancilla bit method dramatically reduced the number of qubits but deepened the depth of quantum circuits. Using the phase quantum gate method increased the solution scale to twenty-one variables, but it could not be solved stably for Boolean equations with too much complexity.

5. Experimental Results

In this paper, six experiments were carried out: classical algorithm preprocessing, multiplexing the ancilla bits, using phase quantum gates, etc. The experimental results showed that the maximum solving scale of the Boolean equations based on the Grover search algorithm could reach twenty-one variables.

The experiments were carried out with the Sugon cloud- computing cluster, and the experimental environment is shown in Table 1. The dataset in the experiment was in two-dimensional array format, which contained the coefficients and values of the equations, from one variable to twenty-one variables. The code randomly generated the data and optimized to remove unqualified ones.

Table 1. Computing environment.

Device	Setup
OS	X86_64 GNU/Linux
CPU	32-core x86 processor with 2.5 GHz frequency
Network	HDR Infiniband
RAM	128 G
Virtual environment	128 G
Quantum computing framework	Anaconda3-5.2.0
Quantum simulator	qasm_simulator
Programming language	Python 3.6.5

The first three experiments were classical algorithm preprocessing, multiplexing the ancilla bits, and using phase quantum gates. Figure 8 shows the changes in the number of Boolean equations before and after preprocessing with a classical algorithm (starting with 10 variables). The horizontal axis represented the number of variables in the Boolean equations, and the vertical axis represented the number of terms. As seen from the figure, the classical algorithm preprocessing eliminated the same terms in the equation, which could significantly reduce the degree of complexity of the Boolean equations in a relatively short time, which was helpful for the following solving operations of larger-scale Boolean equations.

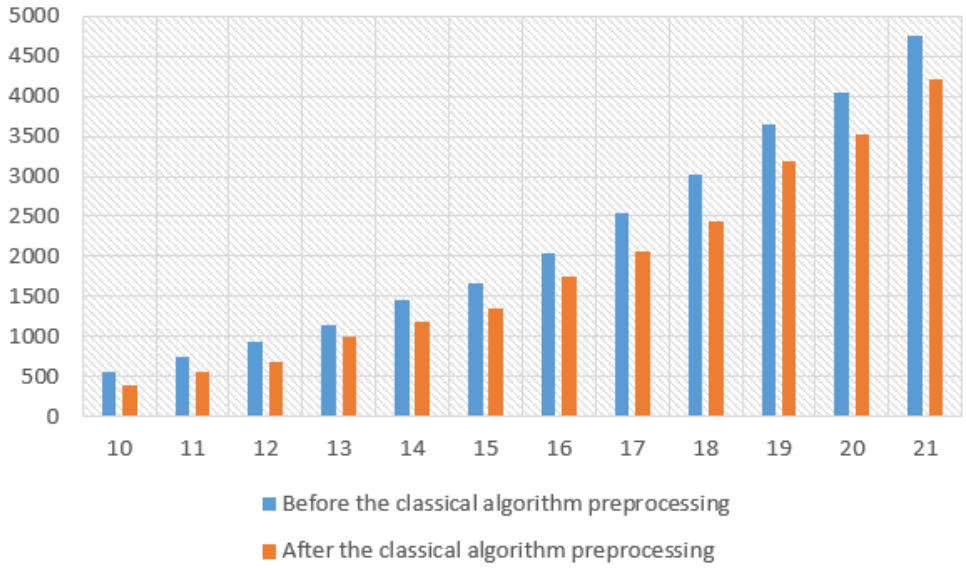

Figure 8. Changes in the number of Boolean equations before and after preprocessing with a classical algorithm.

Figure 9 shows the difference in the number of qubits before and after multiplexing the ancilla bits, where the horizontal axis represented the number of variables of the Boolean equations, and the vertical axis represented the number of qubits used by the quantum circuit. As seen from the figure, after the Boolean equations were grouped with the arithmetic sequence-like method, when the scale of the equation was small, the effect of multiplexing the ancilla bits was not obvious. However, with the increase in the solving scale of the equations, the number of quantum bits was significantly reduced, and the solving scale reached 19 variables.

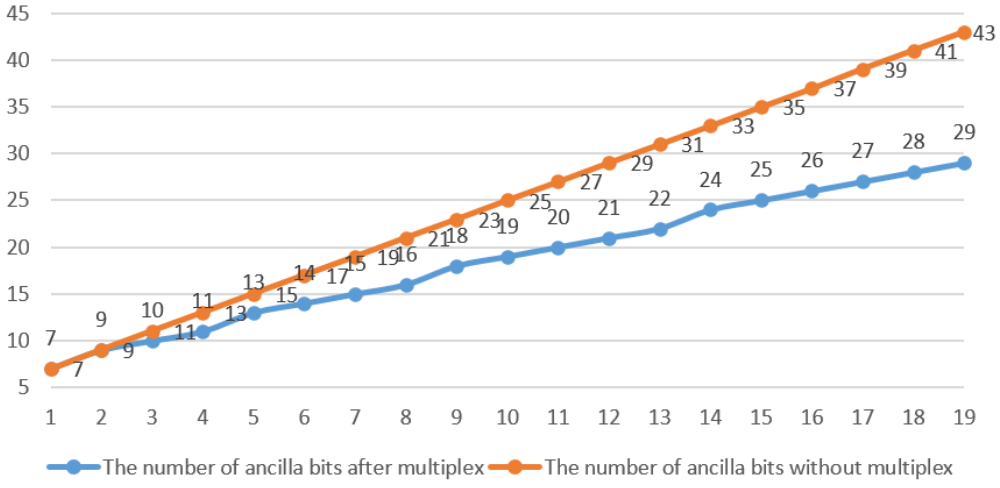

Figure 9. The difference in the number of qubits before and after multiplexing the ancilla bits.

Through the analysis of the oracle structure and experimental results, replacing the CX/CCX/MCX quantum gate with the phase quantum gate could achieve the same effect of marking the target state and reducing the number and depth of quantum circuits. As shown in Figure 10, the targets of the quantum circuit in Figure 7 were correctly marked. We compared the experimental results and found that, for a Boolean equation system with nineteen variables, the solution rate using the phase quantum gate was increased, and the solution scale was increased to twenty-one variables.

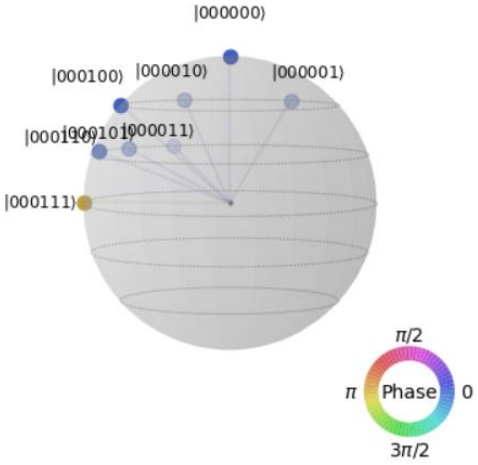

Figure 10. Oracle renderings.

Next were three Boolean-equation-solving experiments. The first experiment was compared with the second to analyze the results and their probabilities for the two-variable and twenty-one-variable Boolean equations. The second one used two twenty-one-variable Boolean equations for comparison and analyzed the quantum circuit and the change in probability amplitude by iterating for different times. The third set of experiments analyzed the relationship between qubits and quantum circuits using multiple Boolean equations of varying scales.

Figure 11 shows the results of the Grover algorithm for solving two-variable Boolean equations. In this experiment, the answer was searched in a solution space of 2^2, the average quantum circuit depth was 34, and the shots were 1024. The correct answer could be found through only one G iteration, and the highest probability was obtained in all the cases.

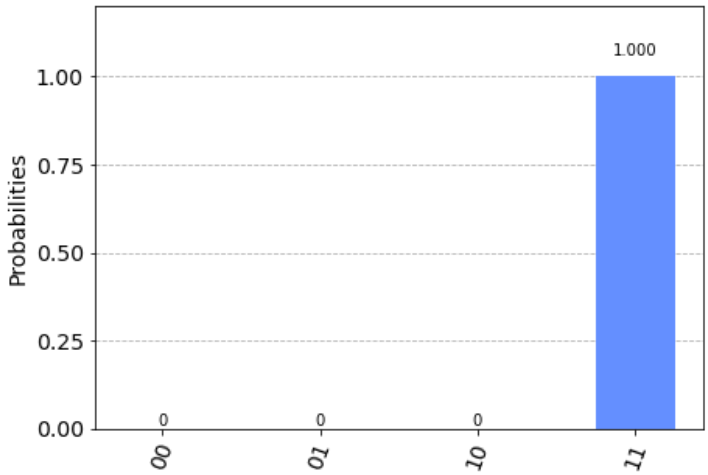

Figure 11. The results of the Grover algorithm for the two-variable Boolean equations.

Figures 12 and 13 show the results of the Grover algorithm for solving Boolean equations with twenty-one variables in a solution space of 2^{21} with one and three iterations, respectively. The shots were one million, and the respective average quantum circuit depths were 67,897 and 229,503 (because there was too much data in the solution space, only part of the data was selected for display). It can be seen from the experimental results that the algorithm did not obtain the correct answer when iterated one time and obtained the answer with a higher probability—100101011100000001111—when iterating three times. Through the relationship between the probability amplitude, the number of iterations, and the depth of the quantum circuit, we know that, when the number of iterations increased,

the probability of the Grover algorithm successfully finding the target increased. However, the quantum circuit was also deeper, which affected the solving scale and time. In this experiment, the algorithm performed three G iterations to search for the correct answer in the vast solution space with a high probability, and when iterated $\left\lceil \frac{\pi}{4}\sqrt{\frac{N}{M}} \right\rceil$ times, the algorithm searched for the answer with a probability close to 1.

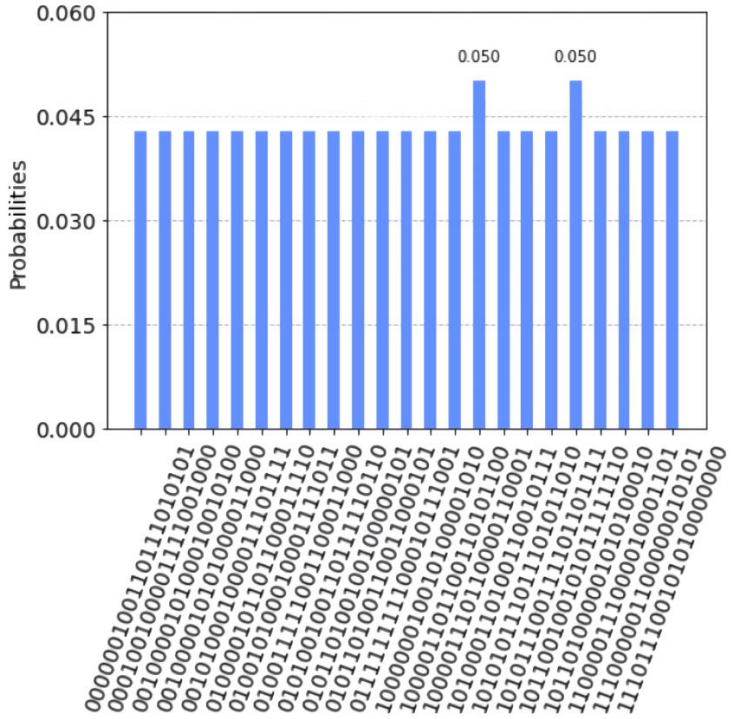

Figure 12. The result of one iteration of the twenty-one variable Boolean equation system.

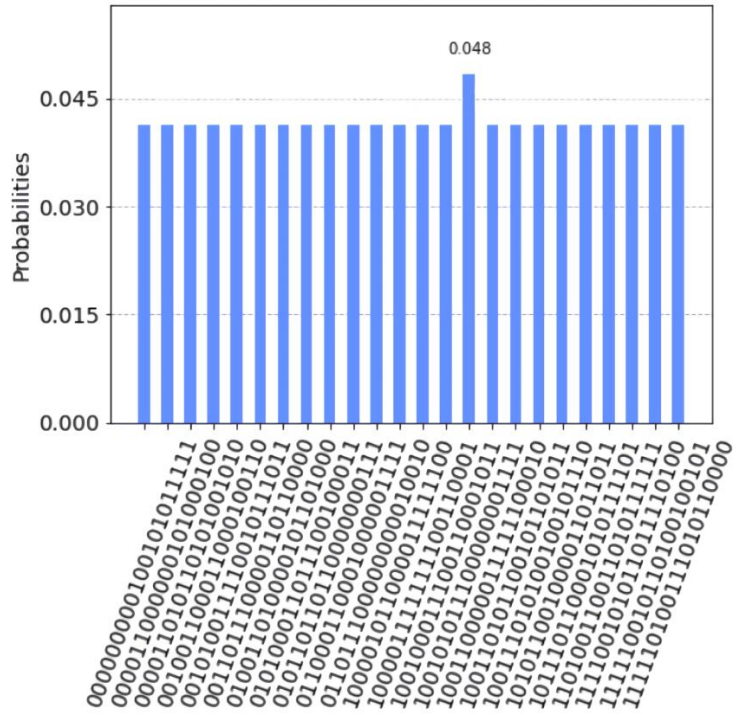

Figure 13. The results of three iterations of the twenty-one variable Boolean equation system.

To fully understand the relationship between the number of qubits and quantum circuits, Qiskit was used to solve Boolean equations on different scales many times, and the experimental results were recorded. Table 2 lists the number of qubits and the average quantum circuit depth (three iterations) used for Boolean equations on different scales (starting with ten variables).

Table 2. Number of qubits and circuit depth required for Boolean equations of different scales.

The Scale of Boolean Equations	The Number of Qubits	The Average Depth of the Quantum Circuit
10	16	20,727
11	18	25,613
12	19	36,206
13	20	54,044
14	21	62,507
15	22	80,926
16	24	95,188
17	25	114,292
18	26	141,799
19	27	163,339
20	28	173,713
21	29	223,899

It can be seen from Table 2 that, when the scale of the Boolean equations increased, the depth of the quantum circuit became increasingly deeper with the increase in the number of qubits and the number of iterations. When the solving scale reached 22 variables, because the quantum circuit was too deep and the number of qubits was too large, the simulator in Qiskit could not continue to run. Through these three experiments, the situation of the Grover search algorithm for solving large-scale Boolean equations was fully demonstrated.

6. Conclusions

Quantum computing is a rapidly emerging technology that follows the laws of quantum mechanics and uses qubits for computing. Compared with classical computers, quantum computers have obvious advantages in many aspects, such as accuracy and efficiency. Quantum algorithms can perform any computation exponentially more efficiently on quantum computers than on conventional computers.

In this paper, we used the quantum simulator provided by the Qiskit framework to run the Grover algorithm to solve nonlinear Boolean equations in a binary field and prove the feasibility of the Grover algorithm on this problem. By analyzing the structure of Boolean equations and quantum circuits, the quantum circuits were optimized, increasing the solving scale to twenty-one variables. When building a quantum circuit, using classical algorithm preprocessing, multiplexing the ancilla bits, and phase quantum gates to optimize the quantum circuit while increasing the solving scale of Boolean equations, the depth of the quantum circuit was significantly deepened. The experiments showed that the depth of the quantum circuit had a significant influence on the solution of Boolean equations. In theory, if the depth of the quantum circuit could be reduced, the solving scale could continue to grow. When using the Grover algorithm to solve Boolean equations, the main factor affecting the solving scale was the oracle, making the quantum circuit depth rise when distinguishing the target. It was affected by noise when searching for a target, which seriously reduced the probability amplitude of the target.

Our current work mainly was to verify the feasibility of the Grover algorithm in solving Boolean equations and to conduct preliminary experiments to explore the full scale of solving Boolean equations. In future work, we plan to study two aspects: (1) simplifying the terms of Boolean equations using data structures that store the equations efficiently and (2) continuing to optimize the quantum circuit and reduce the impact of noise on the target through oracle optimization.

Author Contributions: H.L. designed the analysis, designed the research experiment, wrote and revised the manuscript, and conducted data analysis and details of the work. F.L. verified the data and conducted the statistical analysis. H.L. and Y.F. collected the data and conducted the analysis. All authors have read and agreed to the published version of the manuscript.

Funding: This work was funded by "Research on innovation ecology and application of domestic advanced computing platform" under Grant 221100210600. This work was also supported by "The Doctoral Research Start-up Fee Support Project of Henan Normal University" under Grant 201911.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Deutsch, D. Quantum theory, the Church–Turing principle and the universal quantum computer. *Proc. R. Soc. Lond. A. Math. Phys. Sci.* **1985**, *400*, 97–117.
2. Deutsch, D.; Jozsa, R. Rapid solution of problems by quantum computation. *Proc. R. Soc. Lond. Ser. A Math. Phys. Sci.* **1992**, *439*, 553–558.
3. Shor, P.W. Algorithms for quantum computation: Discrete logarithms and factoring. In Proceedings of the 35th Annual Symposium on Foundations of Computer Science, Washington, DC, USA, 20–22 November 1994; pp. 124–134.
4. Grover, L.K. Quantum Mechanics Helps in Searching for a Needle in a Haystack. *Phys. Rev. Lett.* **1997**, *79*, 325–328. [CrossRef]
5. Farhi, E.; Gutmann, S. Analog analogue of a digital quantum computation. *Phys. Rev. A* **1998**, *57*, 2403–2406. [CrossRef]
6. Lloyd, S. Quantum search without entanglement. *Phys. Rev. A* **1999**, *61*, 010301. [CrossRef]
7. Liu, X.; Song, H.; Wang, H.; Jiang, D.; An, J. Survey on improvement and application of Grover algorithm. *Comput. Sci.* **2021**, *48*, 315–523.
8. Younes, A.; Rowe, J.E.; Miller, J.F. A Hybrid Quantum Search Engine: A fast Quantum algorithm for multiple matches. *Quantum Phys.* **2003**. [CrossRef]
9. Zhou, L.Z.; Fei, L.I.; Zheng, B.Y. An improved quantum grover algorithm. *J. Nanjing Univ. Posts Telecommun. (Nat. Sci.)* **2011**. [CrossRef]
10. Zhu, Y.; Wang, Z.; Yan, B.; Wei, S. Robust Quantum Search with Uncertain Number of Target States. *Entropy* **2021**, *23*, 1649. [CrossRef] [PubMed]
11. Xie, X.; Duan, L.; Qiu, T.; Kang, X. Search space self-adaptive quantum search algorithm. *J. Chin. Comput. Syst.* **2021**, *42*, 732–735.
12. Song, H.; Liu, X.; Wang, H.; Yin, M.; Jiang, D. Integer decomposition based on Grover search algorithm. *Comput. Sci.* **2021**, *48*, 20–25.
13. Wei, Z. Research on Quantum Circuit Algorithm Based on IBM q Quantum Cloud Platform. Master's Thesis, Yangzhou University, Yangzhou, China, 2020.
14. Xie, X.; Duan, L.; Qiu, T.; Yang, Y. Improved quantum search algorithm and its application on computation of core. *Comput. Eng. Appl.* **2020**, *56*, 57–61.
15. Hou, W.; Perkowski, M. Quantum-based algorithm and circuit design for bounded Knapsack optimization problem. *Quantum Inf. Comput.* **2020**, *20*, 766–786. [CrossRef]
16. Ju, Y.-L.; Tsai, I.-M.; Kuo, S.-Y. Quantum Circuit Design and Analysis for Database Search Applications. *IEEE Trans. Circuits Syst. I Regul. Pap.* **2007**, *54*, 2552–2563. [CrossRef]
17. Harrow, A.W.; Hassidim, A.; Lloyd, S. Quantum Algorithm for Linear Systems of Equations. *Phys. Rev. Lett.* **2009**, *103*, 150502. [CrossRef] [PubMed]
18. IBM. Qiskit: Elements for Building a Quantumfuture[EB/OL]. 2021. Available online: https://github.com/Qiskit/qiskit (accessed on 4 March 2022).
19. IBM: QDT (Qiskit Development Team). Qiskit Terra API Reference[EB/OL]. 2021. Available online: https://qiskit.org/documentation/apidoc/terra.html (accessed on 4 March 2022).
20. IBM: QDT (Qiskit Development Team). Qiskit AerAPI Referenc[EB/OL]. 2021. Available online: https://qiskit.org/documentation/apidoc/aer.html (accessed on 4 March 2022).
21. IBM: QDT (Qiskit Development Team). Qiskit Ignis API Reference[EB/OL]. 2021. Available online: https://qiskit.org/documentation/apidoc/ignis.html (accessed on 4 March 2022).
22. IBM: QDT (Qiskit Development Team). Qiskit Aqua API Reference[EB/OL]. 2021. Available online: https://qiskit.org/documentation/apidoc/aqua.html (accessed on 4 March 2022).
23. Grover, L.K. A fast quantum mechanical algorithm for database search. In Proceedings of the STOC'96 28th Annual ACM Symposium on the Theory of Computing, Philadelphia, PA, USA, 22–24 May 1996; pp. 1–8.
24. Chen, Y. The Optimization of Grover Quantum Search in Multiple Solutions Data Space of Larger Than Available Quantum Bits. Master's Thesis, Northwest University, Xi'an, China, 2021.
25. Boole, G. *The Mathematical Analysis of Logic: Being an Essay towards a Calculus of Deductive Reasoning*; Forgotten Books: London, UK, 2007.