Genetic Algorithm Based Quantum Circuits Optimization for Quantum Computing Simulation

Lu Wei
Xi'an Microelectronics Technology Institute
Xi'an, China
13991210602@163.com

Zhong Ma*
Xi'an Microelectronics Technology Institute
Xi'an, China
mazhong@mail.com

Yuqing Cheng
Xi'an Microelectronics Technology Institute
Xi'an, China
296521961@qq.com

Qianyu Liu
Xi'an Microelectronics Technology Institute
Xi'an, China
luxsie@163.com

2021 12th International Conference on Information, Intelligence, Systems & Applications (IISA) | 978-1-6654-0032-9/21/$31.00 ©2021 IEEE | DOI: 10.1109/IISA52424.2021.9555575

Abstract—**Quantum computing simulation platform can simulate the computation results of the quantum computer based on traditional computers, which is an effective way to promote the development of quantum computing software, algorithms and hardware at the current immature stage of the real quantum computer. Since quantum computers have exponential calculation acceleration compared with traditional computers, the main problems in implementing quantum computing simulation on traditional computers are low computational efficiency and long time-consuming. A quantum circuit which is a sequence of quantum gates acting on a collection of qubits is the general quantum computing model. So by the means of quantum circuit optimization, the calculation speed can be significantly increased while keeping the calculation result unchanged. The existing empirical rules of quantum circuit optimization methods have limitations and there is no common and automatic quantum circuit optimization method. In this paper, a general and automatic quantum circuit optimization method based on the genetic algorithm is proposed, by which the equivalent optimal quantum circuit is obtained through a finite number of searching in a large searching space. This method is not limited by the hardware of the quantum computing simulation and the composition of the quantum circuit. The experimental results show that for the QFT algorithm of 29 qubits, the running time can be shortened by 41.4% and for the variational circuit of 6 qubits, the running time can be shortened by 18.8% compared with the state-of-the-art quantum circuit optimization method. So this method can improve the quantum computing simulation capability and operating efficiency and provide a rapid development way for quantum algorithms and applications.**

Keywords—Quantum simulation , quantum circuits, genetic algorithm, automatic optimization

I. INTRODUCTION

Quantum computers use quantum mechanics to calculate [1]. Compared with traditional computers, the basic unit is qubits. Different from traditional digital bits, qubits can exist in the superposition of 0 and 1, which can take advantage of parallel computing to solve difficult tasks. At present, quantum computers are facing problems such as high R&D costs, susceptibility to noise, and difficulty in leaving the experimental environment.

Quantum computing simulation platforms can run quantum circuits on traditional computers, and get the results just like the real quantum computers. Before quantum computing hardware can reach sufficient scale and quality, quantum simulation platforms based on traditional computers will play an important role in the simulation and error correction of quantum circuits, the research and innovation of quantum algorithms, the development and verification of quantum software.

A quantum circuit which is a sequence of quantum gates acting on a collection of qubits, is the general computing model for quantum computing. It consists of a set of quantum gates, and can be used to describe arbitrary complex calculations. When running a quantum algorithm on a quantum simulation platform, when the number of qubits is large or the quantum circuit is complicated, the execution time of the quantum program is too long. So it is necessary to simplify the quantum circuit through optimization methods before running it on simulation platforms.

The optimization of quantum circuits is also an active research area, many methods have been proposed. [2] proposed the implementation of hardware parallel acceleration methods, such as vectorization, SIMD , Cache blocking, which are the basic acceleration parallel methods. [3] proposed a gate fusion method in which all one-qubit quantum gates are combined with the closer two-qubit quantum gate to reduce the number of quantum gates and simplify the calculation of quantum circuits. Similarly, [6] proposed a simple method to fuse several gates by multiplying their respective unitary matrices and multiply the resulting matrix to the state vector, instead of applying the original gates one-by-one. This rule is not useful for all quantum circuits to reduce computing time. The method proposed in [4] based on quantum gate mapping optimization is only effective for quantum circuits with certain characteristics. [5] proposed an optimization method only for rotation gates, which is not general. [10] proposed the method by means of changing the order and distribution of quantum states to reduce communication overhead and shorten running time, which is a specific implementation scheme limited to distributed simulation platforms. [12, 13] proposed similar methods to reduce the communication overhead for distributed simulation platforms. [14] proposed two strategies to quickly merge several quantum gates in quantum circuits: light and heavy

Funding: The Qian Xuesen Youth Innovation Foundation from China Aerospace Science and Technology Corporation (grant number 2019JY39).

optimizations to merge gates while the performance varies depending on the structure of quantum circuits and computing devices. The existing quantum circuit optimization methods independent of the simulation hardware mainly use quantum gate equivalence rules to simplify quantum circuits, or combine two-qubit gates with all surrounding one-qubit gates to reduce the number of quantum gates. At present, there is no common and automatic quantum circuit optimization method.

There are two challenges lies in automatically optimizing the quantum circuits, One challenge is the combination of available optimizing operation—such as the substitution of the equivalent gates—is huge, there is no way to evaluate every optimizing scheme. The second challenge is the lack of reliable evaluation methods for the computing time-consuming of a quantum circuit. In this paper, we propose an automatic and general quantum circuit optimization method based on a genetic algorithm. The optimal combination of quantum gates can be obtained through a limited search in a large searching space, thereby simplifying the calculation of quantum circuits and shortening the running time. There are two contributions in this paper, one is the general quantum circuit optimization method based on a genetic algorithm, the other is the method for evaluating the computing time-consuming of a quantum circuit.

II. Design

A. Overall Design Scheme

In this paper, a common and automatic quantum circuit optimization method based on the genetic algorithm is proposed. The genetic algorithm is used to guide the search for the available optimizing operations of quantum gates and complete the optimization of the quantum circuit.

The optimization method includes the following steps :

- Input the initial quantized circuit diagram.
- The primary chromosome representation of quantum gate based on a tree structure, including equivalent substitution coding set and fusion coding set, as shown in Section Ⅱ.C.
- Fitness evaluation of the primary chromosome is carried out according to the calculation quantity evaluation method in Section Ⅱ.C.
- Construction of genetic operators, including the selection of father chromosome based on the gambling wheel method in Section Ⅱ.C, the hybridization method based on random multi-point crossover tree structure in Section Ⅱ.C, and the mutation method based on non-uniform tree structure in Section Ⅱ.C.

The whole process of the method is shown in Fig. 1.

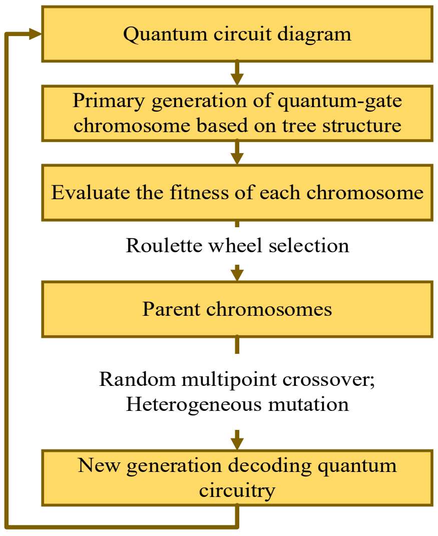
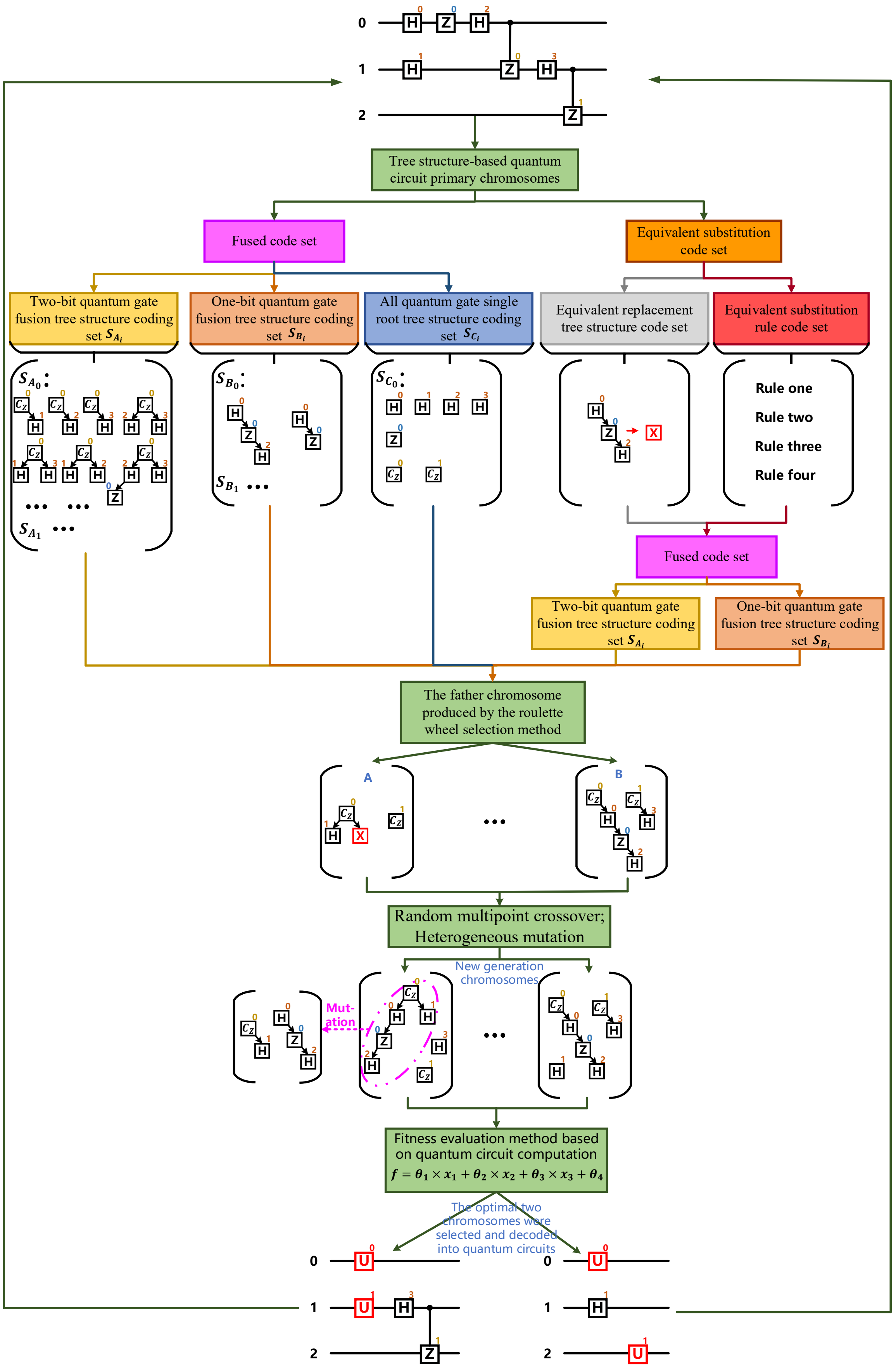

Quantum circuit diagram
Primary generation of quantum-gate chromosome based on tree structure
Evaluate the fitness of each chromosome
Roulette wheel selection
Parent chromosomes
Random multipoint crossover; Heterogeneous mutation
New generation decoding quantum circuitry

Fig. 1. The whole process of the proposed genetic algorithm based quantum circuit optimization method.

The specific implementation process is shown in Fig. 2. Firstly, the primary generation chromosome of the quantum gate based on tree structure is generated from the original quantum circuit, and the father chromosome is generated by the gambling wheel method. Then, hybridization based on random multi-point crossover tree and mutation based on non-uniform tree structure are adopted to generate a new generation of chromosomes. Finally, by section Ⅱ.C based on the data fitting of quantum computation evaluation method, select the best two quantum circuit as a new cycle of input, and while the generated quantum circuit computation remains unchanged for three generations, that is the optimal output quantum circuit.



Tree structure-based quantum circuit primary chromosomes
Fused code set
Equivalent substitution code set
Two-bit quantum gate fusion tree structure coding set S_{A_i}
One-bit quantum gate fusion tree structure coding set S_{B_i}
All quantum gate single root tree structure coding set S_{C_i}
Equivalent replacement tree structure code set
Equivalent substitution rule code set
Rule one
Rule two
Rule three
Rule four
Fused code set
Two-bit quantum gate fusion tree structure coding set S_{A_i}
One-bit quantum gate fusion tree structure coding set S_{B_i}
The father chromosome produced by the roulette wheel selection method
A
B
Random multipoint crossover; Heterogeneous mutation
New generation chromosomes
Mut-ation
Fitness evaluation method based on quantum circuit computation
$f = \theta_1 \times x_1 + \theta_2 \times x_2 + \theta_3 \times x_3 + \theta_4$
The optimal two chromosomes were selected and decoded into quantum circuits

Fig. 2. The specific process of searching and optimizing the structure of quantum circuit tree based on genetic algorithm.

B. *Quantum circuit evaluation based on data fitting*

It is difficult to predict the running time of the quantum circuits. To solve this problem, this paper proposes a method for evaluating quantum circuits based on data fitting. Firstly, analyzes the number of multiplication operations, the number of addition operations, the number of assignment operations, the number of ground state probability amplitude updates of the quantum gate commonly used in quantum circuits, as shown in Table 1. Secondly, get the total calculation amount of the quantum circuits based on the previous analysis. Finally, a prediction model is obtained by using least square based data fitting on the data consists of the actual running time of the quantum circuits and their corresponding calculation amount. That means, find function $f(x,\theta)$ to minimize the sum of the squares of the deviations of the function value f at the point x_i and the true value y_i:

$$min\sum_{i=1}^{n}(f(x_i,\theta)-y_i)^2=\sum_{i=1}^{n}(f(x_i,\hat{\theta})-y_i)^2 \qquad (1)$$

θ is the parameter to be determined, and $\hat{\theta}$ is the best parameter determined by the least square method. The fitting accuracy is:

$$e=\mathrm{mean}(\frac{\mathrm{abs}(y_i-\mathrm{f}(x_i,\hat{\theta}))}{y_i}\times 100\%) \qquad (2)$$

e is the fitting error, y_i is the running time of quantum circuits on the quantum simulation platform, and $f(x_i,\hat{\theta})$ is the predicted time obtained by the fitting formula.

The relationship between the number of multiplication operations, the number of addition operations, the number of assignment operations, and the running time of the quantum circuit can be fitted with a function, that function could be linear, quadratic or cubic. The simplest function is linear function, which is:

$$f=\theta_1\times x_1+\theta_2\times x_2+\theta_3\times x_3+\theta_4 \qquad (3)$$

The actual predicting function is chosen based on the fitting error, which is described in subsequent chapters.

In this way, the calculation time of a quantum circuit can be predicted without actual running, so as to provide a judgment basis for the optimization of the quantum circuit. It should be pointed out that the number of multiplication operations, addition operations, and assignment operations of quantum gates are related to the implementation of quantum computing simulation platforms. Table I analyzes the quantum simulation platform in [11] as an example and Pauli-Z, S, T and R_θ in this table require the same number of multiplication, addition, and assignment operations, resulting from the same interface used for these quantum gates in [11].

TABLE I. ANALYSIS OF THE AMOUNT OF COMPUTATION REQUIRED BY COMMON QUANTUM GATES

Quantum Gate	**Matrix Representation**	**Number of Multiplic-ation Operations**	**Number of Addition Operations**	**Number of Assignm-ent Operations**
H	$\frac{1}{\sqrt{2}}\begin{bmatrix}1&1\\1&-1\end{bmatrix}$	$2^n\times 2$	$2^n\times 2$	$2^n\times 2$
Pauli-X	$\begin{bmatrix}0&1\\1&0\end{bmatrix}$	0	0	$2^n\times 3$
Pauli-Y	$\begin{bmatrix}0&-i\\i&0\end{bmatrix}$	0	0	$2^n\times 3$
Pauli-Z	$\begin{bmatrix}1&0\\0&-1\end{bmatrix}$	$2^{n-1}\times 4$	$2^{n-1}\times 2$	$2^{n-1}\times 2$
S	$\begin{bmatrix}1&0\\0&i\end{bmatrix}$	$2^{n-1}\times 4$	$2^{n-1}\times 2$	$2^{n-1}\times 2$
T	$\begin{bmatrix}1&0\\0&e^{i\pi/4}\end{bmatrix}$	$2^{n-1}\times 4$	$2^{n-1}\times 2$	$2^{n-1}\times 2$
R_θ	$\begin{bmatrix}1&0\\0&e^{i\theta}\end{bmatrix}$	$2^{n-1}\times 4$	$2^{n-1}\times 2$	$2^{n-1}\times 2$
one-qubit U	$\begin{bmatrix}q_{11}&q_{12}\\q_{21}&q_{22}\end{bmatrix}$	$2^n\times 8$	$2^n\times 6$	$2^n\times 2$
$R_x(\theta)$	$\begin{bmatrix}\cos\theta/2&-i\sin\theta/2\\-i\sin\theta/2&\cos\theta/2\end{bmatrix}$	$2^n\times 8$	$2^n\times 6$	$2^n\times 2$
$R_y(\theta)$	$\begin{bmatrix}\cos\theta/2&-\sin\theta/2\\\sin\theta/2&\cos\theta/2\end{bmatrix}$	$2^n\times 8$	$2^n\times 6$	$2^n\times 2$
$R_z(\theta)$	$\begin{bmatrix}e^{-i\theta/2}&0\\0&e^{i\theta/2}\end{bmatrix}$	$2^n\times 8$	$2^n\times 6$	$2^n\times 2$
CNOT	$\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{bmatrix}$	0	0	$2^{n-1}\times 3$
CZ	$\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&0&-1\end{bmatrix}$	0	0	$2^{n-2}\times 2$
Controlled Phase Shift	$\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&0&e^{i\theta}\end{bmatrix}$	2^n	2^{n-1}	2^{n-1}
two-qubit U	$\begin{bmatrix}q_{11}&q_{12}&q_{13}&q_{14}\\q_{21}&q_{22}&q_{23}&q_{24}\\q_{31}&q_{32}&q_{33}&q_{34}\\q_{41}&q_{42}&q_{43}&q_{44}\end{bmatrix}$	$2^n\times 16$	$2^n\times 7$	$2^n\times 2$

C. *A search and optimization method for quantum circuit tree structure based on genetic algorithm*

1) Tree structure based quantum gate chromosome primary generation representation

a) String representation of a quantum gate

In this paper, only 15 basic common one-qubit and two-qubit quantum gates are considered, including: H, Pauli-X, Pauli-Y, Pauli-Z, S, T, R_θ, one-qubit U, $R_x(\theta)$, $R_y(\theta)$, $R_z(\theta)$, CNOT, CZ, two-qubit U and controlled Phase Shift. Assuming that the maximum quantum number supported in this paper is 34bit, the quantum circuit can support up to 20 time series and 40 two-qubit quantum gates. The hexadecimal string definition for a one-qubit quantum gate is shown in Fig .3.

LSB	Time series bit	Door type bit	Equivalent marker bit
2 bit	2 bit	1 bit	1 bit

Fig. 3. one-qubit quantum gate hexadecimal string representation.

b) Tree structure encoding

The tree structure encoding is divided into two types: fusion encoding and equivalent substitution encoding. An example diagram of a quantum circuit is shown in Fig .4, where a red dotted circle represents equivalent substitution and a green dotted rectangle represents gate fusion.

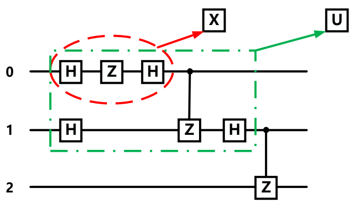


Fig. 4. Example diagram of a quantum circuit.

- Equivalent replacement tree structure encoding

There are 10 encoding methods of equivalent replacement tree structure, as shown in Fig. 5:

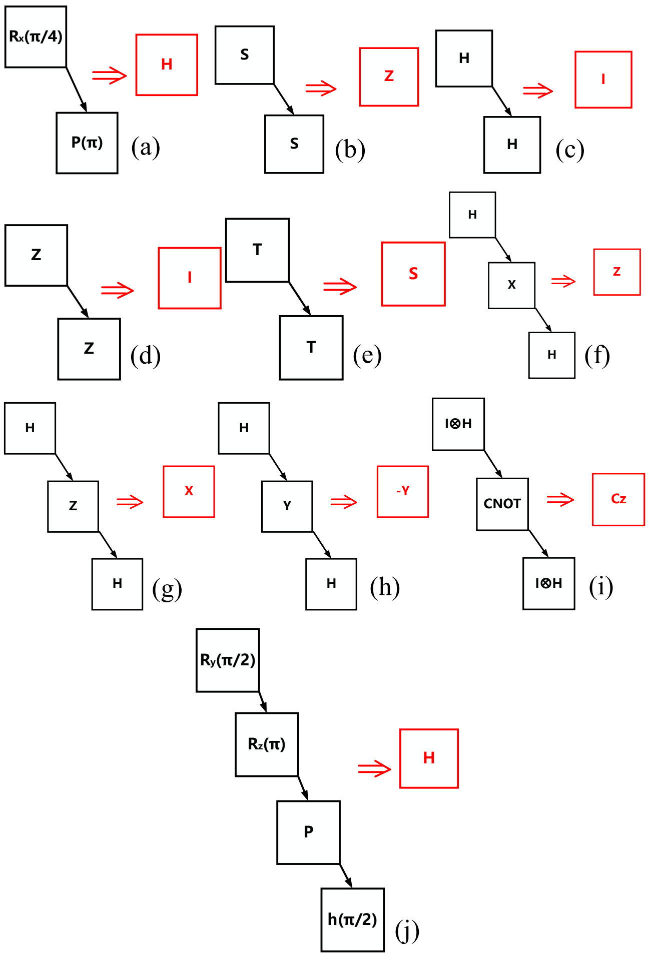


Fig .5. Quantum gate equivalent replacement tree structure.

- Equivalent substitution rule

In addition to the equivalent replacement formula of quantum gates, there are also equivalent replacement rules for quantum gates. The specific rules are as follows:

Rule 1: The cancellation-normalization rule: When the two adjacent compounds controlled non-gate control bits are the same as the target bit, the quantum gate can be canceled into the unitary matrix I.

Rule 2: Merging rule: For the adjacent gates, If the target bit is the same and there is only one different control bit , the different control terminals can be removed and merged into a single controlled gate.

Rule 3: CNOT gate decomposition rule: When the control bits and target bits of two continuously acting CNOT gate overlap, the quantum gate can be decomposed equivalently.

Rule 4: Quantum gate sequence transformation rule: If there are two or more adjacent quantum gates, if the target quantum bit and the control quantum bit are not the same, the sequence of these gates can be changed.

After quantum circuits are encoded according to the above equivalent substitution tree structure or equivalent substitution rules, fusion coding is carried out for quantum circuits to further generate fusion coding sets.

- Fusion tree structure encoding method

There are two encoding methods of the fusion tree structure. One is a two-qubit quantum gate, whose leaf node is a one-qubit quantum gate in the same two quantum bits and adjacent time series or two two-qubit quantum gates in the same time series. The other is the one-qubit quantum gate as the root and the neighboring one-qubit quantum gate or the same time series as the leaf node.

Specifically, the quantum circuit is regarded as a two-dimensional grid, where the row index N represents the number of quantum bits and the column index T represents the length of the time series.

Then the encoding mode of the fusion tree structure with the two-bit quantum gate as the root is as follows:

```
Input: All quantum gates contained in a quantum circuit represented as hexadecimal strings.
Output: All the subsets of the fusion tree structure are encoded with the two-bit quantum gate as the root.
    1. The two-bit quantum gate $A_i$ of each row is checked according to the length $T_i$ of each time series;
    2. The search pauses when the first two-bit quantum gate $A_0$ is found. And define an empty set $S_{A_0}$, add $A_0$to $S_{A_0}$ as the first element to keep the same;
    3. Assume that $A_0$ operates on the row qubits $N_a$ and $N_b$, and is located in the time series $T_t$. First, a one-qubit quantum gate is searched from the time series $T_0$ to $T_{t-1}$ on the $N_a$ row and added to the set $S_{A_0}$;
    4. Then, from the time series $T_{t+1}$, the one-qubit quantum gate is searched along $N_a$ and added to the set $S_{A_0}$ until another two-qubit quantum gate $A_a$ is encountered at the time $T_m$, at which point the $N_a$ row search stops;
    5. Then repeat the step 4 search on the $N_b$ row and add all the searched one-qubit quantum gates to the set $S_{A_0}$;
    6. If the two more two-bit quantum gates $A_a$ and $A_b$ searched by $N_a$ and $N_b$ are at $T_m$, then $A_a$ and $A_b$ can also be added to the set $S_{A_0}$;
    7. If the fusion is carried out according to Step 6, it starts again from time series $T_{t+1}$ and continues until the two more two−bit quantum gates that $N_a$ and $N_b$ meet in different time series, or reaches the end of the line. At this time, the coding set of the $A_0$ fusion tree structure of the two-bit quantum gate has been completed;
```

```
8. Starting from the time series $T_{t+1}$, continue to search for unprocessed two-bit quantum gates, and jump to Step 2 until all the two-bit quantum gates are processed;
9. For the $n(n < i)$ fusion tree structure coding sets $S_{A_0}, S_{A_1}, \dots, S_{A_n}$ generated after the completion of all two-bit quantum gate processing, the first element of the coding set is taken as the root node, and m subtrees are generated by sequential arrangement and combination starting from the minimum time series distance between the first element and the first element. The height of each subtree is required to be more than two layers. It should be noted that the time series in the quantum gate 16-bit string in the generated subtree coding set are arranged in order and cannot be combined across order.
```

The encoding mode of fusion tree structure rooted by the one-qubit quantum gate is as follows:

```
Input: All quantum gates contained in a quantum circuit represented as hexadecimal strings.
Output: A code set of all tree structures rooted by a one-qubit quantum gate.
  1. The one-qubit quantum gate $B_i$ of each row is checked according to the length $T_i$ of each time series;
  2. The search pauses when the first one-qubit quantum gate $B_0$ is found. And define an empty set $S_{B_0}$, add $B_0$ to $S_{B_0}$;
  3. Suppose $B_0$ operates on the row qubit $N_a$ and is located in the time series $T_t$. On the $N_a$ line, the one-qubit quantum gate is searched from the time series $T_t$ and added to the set $S_{B_0}$, until a two-qubit quantum gate $A_a$ is encountered at the moment $T_m$, or the end of the circuit is reached. At this time, the search on the $N_a$ line stops;
  4. Starting with the time series $T_t$, the search continues for unprocessed one-qubit quantum gates. If so, jump to Step 2; If not, skip to Step 5;
  5. Starting from the time series $T_{t+1}$, continue to search for unprocessed one-qubit quantum gates, and jump to Step 2 until all one-qubit quantum gates have been processed;
  6. For the $n(n < i)$ fusion tree structure coding sets $S_{B_0}, S_{B_1}, \dots, S_{B_n}$, and $m$ subtrees are generated by permutation and combination respectively. The height of each subtree is required to be more than two layers.
```

2) Method of generating father chromosome based on wheel selection method

Wheel selection is a method of selecting a number of chromosome tree structures from the primary population of chromosomes, $S_{A_i}{}'$, $S_{B_m}{}'$ and S_{C_n}, in proportion to the probability of being selected according to their fitness scores. The higher the fitness score, the higher the probability of the chromosome tree structure being selected, but this does not guarantee that the chromosome tree structure with the highest fitness score can be selected into the next generation, only means that it has the highest probability of being selected. The adaptability score evaluation index is generated according to the calculation formula of the quantum circuit in Section Ⅱ.C.

To be specific, first of all, the wheel selection method is successively used in the coding set $S_{A_i}{}'$ to select a total of $i$trees and form them into the parent chromosome I_0. Then, a total of m trees are selected in the coding set $S_{B_m}{}'$ using the roulet selection method successively, and they are added to the parent chromosome I_0. Finally, in S_{C_n}, a total of n trees are selected by the method of roulet selection and added to the parent chromosome I_0. The above steps are repeated to generate the parent staining population I.

3) A hybridization method based on random multipoint crossover tree structure

The hybridization method based on random multipoint crossover tree structure search is to randomly set multiple crossover points in individual chromosomes and then exchange gene blocks. Specifically, suppose that A and B are selected for hybridization of paternal individuals, where the father chromosome Acontains $i$subtrees and the father chromosome B contains m subtrees. Multiple crossover points are randomly selected on chromosome A and chromosome B respectively, and then the subtrees at the crossover points are exchanged to generate A new generation of chromosome $\tilde{A}$ and chromosome $\tilde{B}$.

4) Mutation method based on non-uniform tree structure

Non-uniform tree structure-based mutation method refers to random perturbation of original chromosomes, which can be divided into four forms, respectively:

- Split a subtree into two subtrees;
- merge two single subtrees into a single subtree;
- Equivalent substitution of a leaf structure without equivalent substitution in a subtree;
- Return the leaf nodes replaced by a subtree equivalent to the original tree structure.

The result of perturbation is then used as the new chromosome after mutation.

III. Implementation And Experimental Results

The environment tested in this paper is shown in the following table:

TABLE II. Test Hardware and software environment

Operation mode	**Test Environment**	**Software for quantum simulation**
CPU single thread	The operating system: Ubuntu 18.04 64bit The processor: Intel Core i7-6700 CPU@3.40GHz × 8 Memory: 15.6G	The Quest quantum simulation software in [11]
CPU multi-thread		
GPU	GPU model: NVIDIA GeForce GTX 1080 Number of stream processors: 2560 The core frequency: 1607MHz	

	CUDA Version:10.2 Memory capacity: 8G Memory bits wide: 256bit	

A. Quantum circuit evaluation based on data fitting

We collected 25 groups of quantum circuits running data to fitting the evaluation model, the data is generated on the CPU multi-thread mode and the GPU mode, including the number of multiplication operations, the number of addition operations, of the number of assignment operations and the actual running time, and use the linear function, the quadratic function, and the cubic function to fit the data. The linear function fits the data best, of which the fitting parameters $\theta_1, \theta_2, \theta_3, \theta_4$ are $6.5386 \times 10^{-7}, -1.7260 \times 10^{-6}, 5.9541 \times 10^{-6}$ and 0.5664. Using the fitting model to test data in Table II. The average error is 1.7301%.

TABLE III. FITTING MODEL TEST RESULTS

/	**Number of multiplication operations**	**Number of addition operations**	**Number of assignment operations**	**Actual running time(ms)**	**The prediction time by Fitting model (ms)**
1	16777216	7340032	2097152	11.707	11.354
2	12582912	5505024	1966080	10.959	10.998
3	16777216	7340032	2097152	11.5876	11.354
4	13107200	6029312	2490368	13.547	13.558
5	8912896	4194304	2359296	13.7262	13.202
6	13107200	6029312	2490368	13.711	13.558
…	…	…	…	…	…

The relationship between the actual running time and the number of multiplication, addition, and assignment operations is shown in Fig .6. The relationship between the linear fitting time and the number of multiplication, addition, and assignment operations is shown in Fig .7. The relationship between the linear fitting error and the number of multiplication, addition and assignment operations are shown in Figure .8. The color of the point in the figure represents the value . The more the color tends to blue, the smaller the value is. Comparing Fig .6 with Fig .7, the color of each point is very close, which means that the actual running time is very similar to the linear fitting time. It can be seen from Fig .8 that the color of most points is blue, which means that the fitting error is quite small. So we use the linear function as the fitting model to predict the running time of quantum circuits.

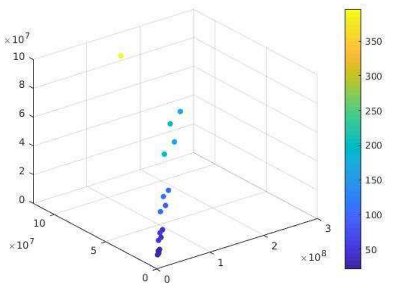


Fig .6. The relationship between the actual running time and the number of multiplication, addition, and assignment operations.

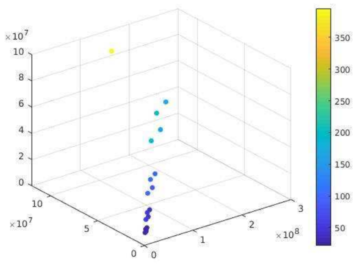


Fig .7. The relationship between the linear fitting time and the number of multiplication, addition, and assignment operations.

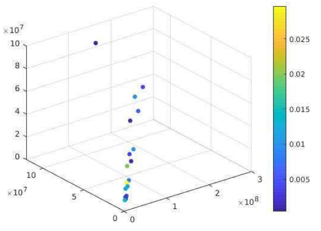


Fig .8. The relationship between the linear fitting error and the number of multiplication, addition, and assignment operations.

Using the data of quantum circuits running on the GPU mode, we can get the same conclusion that the linear function fits the data best, of which the fitting parameters $\theta_1, \theta_2, \theta_3, \theta_4$ are $1.3118 \times 10^{-6}, -3.3805 \times 10^{-6}, 7.4162 \times 10^{-6}$ and 0.7243 and the average fitting error is 1.3230%. So the quantum circuit evaluation based on data fitting proposed in this paper is a general scheme no matter what the computing hardware is.

B. The optimization method for quantum circuit tree structure based on genetic algorithm

This paper uses the best gate fusion method in [3] as the benchmarks. As shown in Fig .9, Fig .10 and Fig .11, the compared quantum circuits referred to [6] are the Quantum Fourier Transform(QFT)[7], the quantum teleportation circuit [9] , which are commonly used and have practical application value, and the variational circuit[8] which is very useful in quantum machine learning. A random quantum circuit is also selected.

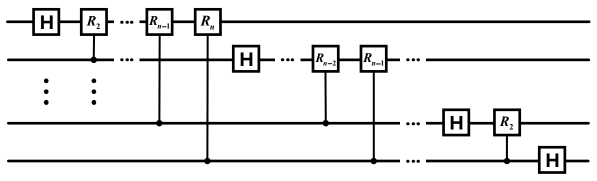


Fig .9. Structure of QFT circuit used in the benchmarks.

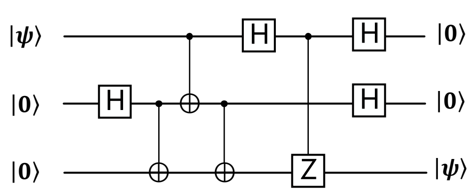


Fig .10. Structure of the quantum teleportation circuit used in the benchmarks.

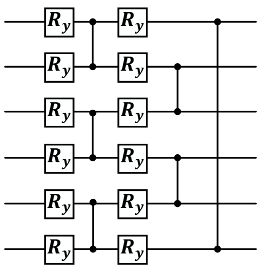


Fig .11. Structure of the variational circuit used in the benchmarks. All gates shown in this figure are repeated five times to give the full circuit.

The experimental results are shown in the table below.

TABLE IV. THE RESULTS OF PROPOSED OPTIMIZATION METHOD COMPARED WITH [3]

/	**The circuits used in the benchmarks**	**running time by method in [3]**	**running time by the genetic algorithm in this paper**	**improvement percentage**
1	6 qubits QFT	0.0210 ms	0.0181 ms	13.8%
2	29 qubits QFT	2306.804s	1351.548s	41.4%
3	the quantum teleportation circuit	0.001181 ms	0.000471 ms	60.1%
4	the variational circuit	0.03321ms	0.02697 ms	18.8%
5	a random quantum circuit	88.060ms	83.946ms	4.7%

The results show that the proposed genetic algorithm method outperforms the gate fusion method in[3] because the method in [3] has limitations and is only useful for quantum circuits with matching characteristics. The method proposed in this paper is general, and the optimal solution can be automatically found through the genetic algorithm for quantum circuits.

IV. CONCLUSION AND FUTURE DIRECTIONS

Quantum computing simulation systems play an important role in quantum computing research. The method of optimizing quantum circuits can be used to improve the quantum simulation efficiency. The existing methods optimize a quantum circuit according to its composition by replacement or fusion quantum gates, which have limited effects and lack a general and automatic optimization method. In this paper, a general and automatic quantum circuit optimization method based on the genetic algorithm is proposed, including a quantum circuit evaluation method based on data fitting and a quantum circuit tree searching optimization method based on the genetic algorithm. The quantum circuit evaluation method based on data fitting can estimate the running time of a quantum circuit without actual operation. The quantum circuit tree searching method can find the equivalent optimal quantum circuit through a finite number of searching in a large searching space. This method is not limited by the hardware operation mode of the quantum computing simulation platform and the composition of the quantum circuit. The experimental results show that compared with the state-of-the-art quantum circuit optimization method, this method can achieve a better speed improvement effect. For the QFT algorithm of 29 qubits, the running time can be shortened by 41.4% and for the variational circuit of 6 qubits, the running time can be shortened by 18.8%.

For further work, incorporating partitioning scheme into the proposed quantum circuit optimization methods is a promising research topic, which will provide the quantum simulator with the opportunity of running ultra-large-size quantum circuits.

REFERENCES

[1] Nielsen, Michael A., and Isaac Chuang. "Quantum computation and quantum information.", 2002, pp.558-559.

[2] Smelyanskiy, Mikhail, Nicolas PD Sawaya, and Alán Aspuru-Guzik. "qHiPSTER: The quantum high performance software testing environment." arXiv preprint arXiv:1601.07195, 2016.

[3] Broughton, Michael, et al. "Tensorflow quantum: A software framework for quantum machine learning." arXiv preprint arXiv:2003.02989, 2020.

[4] Häner, Thomas, and Damian S. Steiger. "0.5 petabyte simulation of a 45-qubit quantum circuit." In Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis on – SC 17. ACM Press, 2017.

[5] Nam, Yunseong, et al. "Automated optimization of large quantum circuits with continuous parameters." npj Quantum Information 4.1, 2018, pp.1-12.

[6] Efthymiou S, Ramos-Calderer S, Bravo-Prieto C, et al. Qibo: a framework for quantum simulation with hardware acceleration[J]. arXiv preprint arXiv:2009.01845, 2020.

[7] Coppersmith, Don. "An approximate Fourier transform useful in quantum factoring." arXiv preprint quant-ph/0201067, 2002.

[8] Moll, Nikolaj, et al. "Quantum optimization using variational algorithms on near-term quantum devices." Quantum Science and Technology 3.3, 2018, pp. 030503.

[9] Prokopenya, A. N. "Mathematica package "QuantumCircuit" for simulation of quantum computation." International Mathematica Symposium. 2015.

[10] Guerreschi, Gian Giacomo. "Fast simulation of quantum algorithms using circuit optimization." arXiv preprint arXiv:2010.09746, 2020.

[11] Jones, Tyson, et al. "QuEST and high performance simulation of quantum computers." Scientific reports 9.1, 2019, pp.1-11.

[12] The Huawei HiQ Team. "Huawei hiq: A high-performance quantum computing simulator and programming framework. "http://hiq.huaweicloud.com.

[13] K. De Raedt, K. Michielsen, H. De Raedt, B. Trieu,G. Arnold, M. Richter, Th. Lippert, H. Watanabe, and N. Ito. "Massively parallel quantum computer simulator. "Computer Physics Communications, 176(2):121–136, jan 2007.

[14] Suzuki, Yasunari, et al. "Qulacs: a fast and versatile quantum circuit simulator for research purpose." arXiv preprint arXiv:2011.13524, 2020.