Advances in Engineering Research, volume 113

International Forum on Mechanical, Control and Automation (IFMCA 2016)

Quantum Circuit Synthesis for Linear Nearest Neighbor Based on the Vector Transformation

Yu Lu[1,a] , Zhijin Guan[2,b] , Xueyun Cheng[3,c] , Keren Yu[4,d]

[1]College of Electronics and Information, Nantong University, Nantong 226019, China

[2,3]College of Computer Science and Technology, Nantong University, Nantong 226019, China

[4]College of Computer Science and Software Engineering, East China Normal University, Shanghai 200062, China

[a] luyu102591@163.com, [b] guan_g617@163.com, [c] 20081917@qq.com, [d] 1198762933@qq.com

Keywords: Quantum reversible circuit ; Circuit synthesis; vector transformation; Linear nearest neighbor

Abstract. In this paper, a synthesis method is proposed based on vector transformation for linear nearest neighbor quantum circuits. Its aim is to construct a linear nearest neighbor quantum circuit, and to reduce the quantum cost of the linear reversible circuit. The vector representation of the sequence of the circuit lines and qubits of quantum gate is given in this method. It realizes the nearest neighbor of the quantum circuit by moving the position of the qubits vector elements, and it does not cause confusion due to comparison and transformation of two vector elements in the circuit. The number of SWAP gates needed to make the quantum circuit nearest neighbors is given, and its correctness is proven. Compared with the current quantum circuit synthesis algorithms, the average optimization rate of the quantum cost is 39.69% for typical benchmark circuits. The algorithm can be applied to all quantum circuits of 2-qubit quantum gates, and can be used in large quantum circuits.

Introduction

Quantum computing is a computational paradigm with many advantages, such as large amount of information storage, fast operation and low energy consumption. So, great attention has been given to the field of quantum computation in many countries.

Quantum circuit synthesis originates from the research on quantum computer and reversible computer. Each technology to realize quantum reversible logic circuit needs a reasonable cost. The quantum cost of a quantum circuit depends heavily on techniques, we need to find effective methods to optimize.

Physical realization of quantum-computing logic gates involves geometrically adjacent qubits. We can put forward different projects for quantum computer construction under different architecture according to the way that each qubit interacts with other qubits in a physical system. In most cases, some commonly used quantum technologies required in LNN (Linear Nearest Neighbor) [1] architecture that qubits are arranged in a line and only adjacent qubits can interact. In order to achieve the LNN constraint of quantum technology and construct quantum circuits for LNN, so far, many related synthesis algorithms for LNN architecture have been put forward [2-6]. To realize the LNN quantum circuit synthesis method, M Saeedi [2] constructed interaction map, and the algorithm transforms optimization for the qubits interact distance into the problem of Minimum Linear Arrangement (MINLA). In addition, a lookahead technique is applied to improve the cost in quantum circuits. However, it becomes very expensive particularly if large quantum gates are considered. B Schaeffer [3] introduced rules and LNN criterions for synthesizing neighbor circuits. The main feature is taking circuit output as the goal, using reversible combined logic method to

design circuits and using the current synthesis technology to construct the circuits. This method frequently produces high quantum cost, and is thus not practical. VA kalmychkov [4] marked different quantum circuits and designed quantum circuits automation based on different symbols. The rules and methods to get sequence of original quantum gates under LNN structure are also proposed. However, this method is not suitable for quantum circuits composed of multiple-qubits quantum gates. Very recently, A Kole [5] showed that an optimization approach for circuit neighbor based on the global ordering. Its applicability is limited although it is suited to NCV library. The algorithm illustrated in reference [6] have been widely used in the LNN mode. The quantum error-correction mechanism and good LNN standards for circuits are proposed by MM Rahman.

For the sake of constructing optimized quantum circuits for LNN based on vector transformation, following work will be finished in this paper: (1) The vector expression of quantum qubits is given in order to realize quantum gates neighbor, and the implementation of the qubits neighbor is achieved by moving elements positions. (2) To solve the problem of circuit line confusion caused by the qubits move, we give line vector expression of quantum gates before and after neighbor operations. We also make the quantum circuits to LNN form through comparing and moving vector elements between original line and target line. (3) The rules for calculating the number of SWAP gates which in the process of method realization are summarized. The quantum cost in circuits is reduced and none of the SWAP gate can be removed.

The remainder of this paper is structured as follows. The next section briefly reviews the current quantum circuit synthesis methods, including 2-qubit gate function and related algorithms. The paper proposes some definitions and theorems which are used in new algorithms, mainly including the vector representations of qubits locations and line order. The optimized rules of adding SWAP gates should be finished based on local ordering and line regression algorithm for LNN quantum circuits synthesis. The detailed operations and examples for each method are attached in this paper. Finally, we give better results compared to those by the earlier ones including [9].

Basic concepts

Reversible quantum gate

The element of quantum circuit is reversible quantum gates. In the computation models of quantum circuits, a reversible quantum gate is a basic operation. A reversible circuit is one that implements a bijective function, or loosely, a circuit where the inputs can be recovered from outputs and all output values are achievable. Each input assignment of a reversible quantum gates maps to a unique output assignment.

In this paper, all quantum gates are 2-qubit reversible quantum gates. Common 2-qubit reversible quantum gates contain controlled-not gate, controlled-V gate, controlled-V^+ gate, and SWAP gate.

The controlled-not gate and its classification

Controlled-not gate, or CNOT gate [7], shown in Fig.1, has a control bit and a target bit. The value of target bit is inverted iff all control bits are assigned to1and this value is calculated as the XOR sum of input variables.

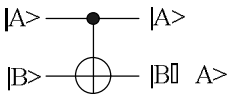


Fig.1 CNOT gate

We can divide CNOT gate into adjacent-CNOT gate and non-adjacent CNOT gate according to its neighbor qualities (the other 2-qubit quantum gates can use the same classification approach). If the NNC[7] (Nearest Neighbor Cost) value of a CNOT gate is 0, which is called adjacent-CNOT gate, otherwise known as the non-adjacent CNOT gate. There may be two variations of a CNOT gate as shown in Fig.2(a) and 2(b).

(a) Adjacent-CNOT gate (b) Non-adjacent CNOT gate

Fig.2 Two types of CNOT gate

The controlled-V gate and controlled- V^+ gate

Controlled-V gate and Controlled-V^+ gate are conjugate [8]. Their multiplication is a unit vector.T

Two controlled-V gates continuously are used to equal to a NOT gate (the function of NOT gate is to invert the output value). The symbol of controlled-V gate and controlled-V^+ gate are shown as Fig.3.

(a) Controlled-V gate (b) Controlled-V^+ gate

Fig.3 Controlled-V gate and Controlled-V^+ gate

SWAP gate

SWAP gate [7], shown in Fig.4, has two target bits and no control bits. The gate interchanges the values of target bits for a quantum gate.

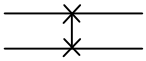


Fig.4 SWAP gate

Quantum logic synthesis

Reversible logic synthesis [7] is what we use to achieve corresponding reversible network with given logic gates according to constraint condition and limitations of no fan-out, no feedback and meet technological requirements to achieve quantum circuits, and also optimize quantum circuits under a certain cost model, with quantum cost being as little as possible.

Quantum cost

Quantum cost [8] or qc, is an important parameter to evaluate the circuit performance. It means the cost for realizing a given quantum circuit, and it depends on the number and type of gates in the circuit. In general, the qc of a CNOT gate is 1.The qc of a SWAP gate [8] is 3.

Linear nearest neighbor

Linear nearest neighbor [8] (LNN): Most of synthesis methods are built for LNN. Linear means the relationship between the input and output can be expressed in a linear function. Nearest neighbor means qubits are adjacent for a quantum gate.

Nearest Neighbor Cost

Nearest Neighbor Cost [7] (NNC): Consider a 2-qubit quantum gate *g* with control bit *c* and target bit *t*. The NNC of g is termed as $|c - t|-1$ (i.e., distance between control and target bit). So the NNC of a quantum circuit is defined as the sum of NNCs of its quantum gates. Optimal NNC for a circuit is 0 where all 2-qubit quantum gates are performed on adjacent qubits.

The naive method of adding SWAP gates

Any non-LNN circuit can be converted to LNN one by introducing additional SWAP gates. But the line ordering will make the difference in number of SWAP gates required in making the LNN circuit without any change on the input/output of quantum circuit.

Simple adding SWAP gate algorithm [10] is a naive method for the LNN architectures. It uses same number SWAP gates at the same position in front and after the non-adjacent quantum gates. More precisely, this algorithm doesn't consider SWAP gates remove and SWAP gates are added in front of each gate with non-adjacent control bit and target bit to "move" the control (target) bit towards the target (control) bit until they become adjacent. In Fig.5, as can be seen, the original gate is non-adjacent. Thus, to achieve NNC-optimality, SWAP gates in front and after the gate are inserted. The added SWAP gates shows a stair-step structure which the top SWAP gates on the outside and the rest SWAP gates are in form of downward and inward. The number of additive SWAP gates is twice over NNC value of quantum circuit [10]. Since the qc of each SWAP is 3, this method increases the total qc, but leads to an NNC value of 0.

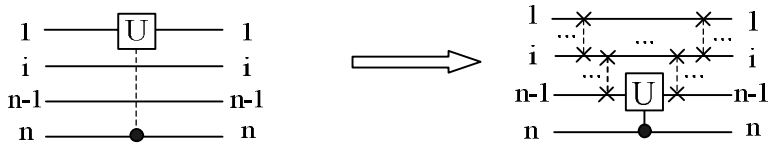


Fig.5 Simple adding SWAP gates diagram for arbitrarily 2-qubit quantum gate

The relationship between the number of additive SWAP gates and NNC value of arbitrary quantum circuit is as follows

$$N_{swap} = 2 * \sum_{1}^{n} NNC \quad (1)$$

Where

— N_{swap} is number of SWAP gates,

— NNC is the nearest neighbor cost of each quantum gate,

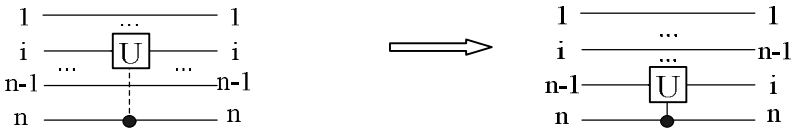
— n is the number of quantum gates.

However, as can easily be seen, synthesizing quantum circuits for LNN architectures using this method often leads to a significant increase in the quantum cost. In contrast, often smaller realizations (with NNC of 0) are possible. Hence, better quantum circuit realizations for the LNN architecture are described in Sect.4.

Local ordering

In order to save SWAP gates, line ordering can also be applied according to a local schema as follows. Consider a 2-qubit quantum gate where its above qubit *a* and below qubit *b* are placed at i^{th}

line and at the n^{th} line respectively. Two qubits are set in the adjacent position by moving the qubit a to the n-1^{th} line, which is called local ordering. Its aims at minimizing nearest neighbor cost to achieve NNC optimality. Local ordering can influence all the quantum gates in quantum circuits, which will close the distance between target and control qubit of quantum gates, that is, the NNC value of quantum gate will be decreased. In contrast to the naive method, the number of SWAP gates are reduced. Local ordering diagram is shown as Fig.6.



(a) (b)

Fig.6 Local ordering

Quantum circuit based on vector representation

Original line and the target line

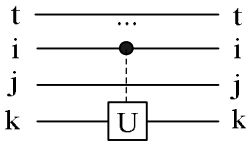
An n-qubit circuit can be numbered from 1 to n in a top-down way. To combat the problem of local ordering, we define the initial circuit line order of circuit as the original line. The circuit line order of final quantum gate output after local ordering is called the target line. The process of changing original line into target line is called line regression algorithm.

Qubits position vector Q_i and the line order vector L_i representation

As we know, nearest neighbor operation for quantum circuit only involves the movements of qubits positions and circuit lines. In order to express the qubits position and circuit lines of quantum gate, two vector representations are defined respectively. The vector operations are used to obtain quantum circuit for LNN. Thus, the problem of quantum circuits synthesis is reduced to vector operations.

Let n be the circuit line and number of the circuit from 1 to n in a top-down way. We use two n*1 dimension vectors to express the qubits position and the circuit lines location respectively. If a qubit is in the line, label the corresponding circuit line number as the element value in the vector, otherwise label as 0.We named qubits position vector Q_i which is used to record qubits position.

We use vectors to record circuit lines because local ordering will change the initial order of circuit line. Each element in the vector corresponds to the labeled number of each circuit line. This vector is called line order vector L_i.



□ Fig.7 Arbitrarily 2-qubit quantum gate diagram

An arbitrarily 2-qubit quantum gate with two qubits positions named i and k are shown in Fig.7, the qubits position vector is:

$$Q_i = (0, ..., i, ..., 0, ..., k)^T \tag{2}$$

The corresponding circuit line order vector is:

$$L_i = (t, ..., i, ..., j, ..., k)^T$$

(3)

Reconsider an arbitrarily 2-qubit quantum gate as depicted in Fig.6(a), the circuit line order vector of the original line is:

$$L_i = (1, ..., i, ..., j, ..., n-1, n)^T$$

(4)

In Fig.6(a), two qubits of the quantum gate are located in the i^{th} line and the n^{th} line, so qubits position vector is:

$$Q_i = (0, ..., i, ..., 0, ..., n)^T$$

(5)

In the process of local ordering, we need to fix the qubit in the n^{th} line, then move the qubit in the i^{th} line to the n-1^{th} line. Two qubits of quantum gate are adjacent at this point. Thus, qubits position vector and circuit line order vector respectively as followed after local ordering operation:

$$Q_i' = (0, ..., 0, ..., i, ..., n-1, n)^T$$

(6)

$$L_i' = (1, ..., n-1, ..., i, ..., n)^T \quad (7)$$

We can conclude from (6) that two non-zero elements are in non-adjacent rows at the qubits position vector of a non-adjacent quantum gate, vice versa. By definition of NNC, the NNC of each quantum gate can also be obtained through Q_i. Supposing two non-zero elements x, y of the vector Q_i, they correspond to two qubits positions one by one. As a result, the NNC of each quantum gate can be expressed as $|x\text{-}y|\text{-}1$. As for all the quantum gates in quantum circuits, if $|x\text{-}y|\text{-}1=0$, the quantum gate is an adjacent quantum gate, otherwise it is a non-adjacent quantum gate.

Theorem 1 In the process of converting original line L into the target line U, the number of additive SWAP gates is equal to the sum of the distances d_i which the same elements in the vector L, U on corresponding positions in line regression algorithm. The increased quantum cost is 3 times for the sum of d_i. The relationship between the number of SWAP gates and d_i is:

$$N_{swap}^{'} = \sum_{1}^{k} d_i$$

(8)

Where

— $N_{swap}^{'}$ represents the added SWAP gates number,

— k is the number of total d_i,

— d_i represents distance of same elements in the vector L, U on corresponding position,

— i ranges from 1 to k.

The relationship between increased quantum cost and d_i is:

$$Q_{swap} = N'_{swap} * 3 = \sum_{1}^{k} d_i * 3 \quad (9)$$

Where

— Q_{swap} represents increased quantum cost in quantum circuit.

Proof

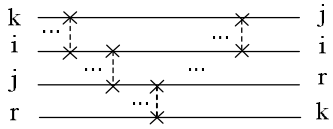


Fig.8 Line regression

Consider the circuit shown in Fig.8, if there is an original line, with *k, i, j, r* being horizontal circuit lines number, and $k<i<j<r$. The original line order vector can be expressed as:

$$L = (k, ..., i, ..., j, ..., r)^T \quad (10)$$

So the target line order vector is:

$$U = (j, ..., i, ..., r, ..., k)^T \quad (11)$$

To perform a line regression algorithm, element is initially processed from the first row of the original line order vector. The element *k* of vector L is in the 1^{th} row, and the corresponding position of element *k* can be found in the vector *U*. From above contents, we can see that element *k* is located in the r^{th} row of the vector *U* and the distance *d* between two elements *k* is *r*-1 • .To move element *k* in vector *L* into the corresponding position in vector *U* needs to move *k* down the *r*-1 times. The added SWAP gates are *r*-1, i.e. SWAP=*d* **,** . The algorithm should update the elements of the vector *L* and *U*, and repeat the above process for the rest of elements of vector *L*, until the *L* and *U* are the same.

From • and **,** , we know that the minimal number of additive SWAP gates is equal to the sum distances of d_i when a single arbitrary quantum gate is converted to an adjacent quantum gate in the quantum circuits constructed by 2-qubit gate. Since the quantum cost of a SWAP gate is 3, increased quantum cost by adding SWAP gates can be expressed as $Q_{swap} = N'_{swap} * 3$. And $N'_{swap} = \sum_{1}^{k} d_i$, therefore $Q_{swap} = \sum_{1}^{k} d_i * 3$. Theorem 1 is proven.

□

Example 1: Line regression algorithm

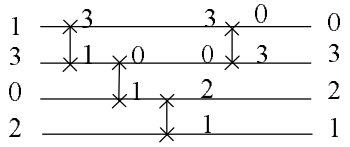


Fig.9 Example of line regression

Consider a circuit as an example of line regression algorithm shown in Fig.9. The original line vector L (1, 3, 0, 2) ^{T} should be converted to the target line vector U (0, 3, 2, 1) ^{T}. From the first row of vector L, the element 1is in the fourth row of vector U and the distance d_1 between two elements 1 in this two vectors is calculated as 3. Line regression operations needs three times, so the number of additive SWAP gates are three and the original line order vector turn into L_1 (3, 0, 2, 1) ^{T}. Similarly, the distance d_2 between element 3 in the first row of L_1 and in the second row of U is 1. A SWAP gate must be added to realize line regression. The circuit line order vector turns into L_2 (0, 3, 2, 1) ^{T} at this point. Vector L_2 and U achieve the same and operations are stopped. Four SWAP gates are added by implementing line regression algorithm, and it equals the sum of d_1 and d_2. The total increased quantum cost is calculated as 4*3 =12.

The linear nearest neighbor algorithm for quantum circuit based on vector

In this section, we propose a new synthesis approach which is called linear nearest neighbor algorithm and based on vector. More precisely, local ordering operation is introduced to avoid the existence of non-adjacent quantum gates after global ordering [11]. Furthermore, the line regression algorithm is proposed that determines NNC-optimal circuits with minimal quantum cost. The resulting circuits can later be exploited to optimize large circuits. This algorithm is aimed at using as few quantum gates as possible to realize quantum circuit for LNN with NNC of 0.

The algorithm converts each non-adjacent quantum gate to reach neighbor by vector operations. Line regression algorithm solves the circuit chaos problem caused by local ordering. This algorithm adds SWAP gates through comparison of elements between original line vector and target line vector. In addition, no unnecessary SWAP gates can be removed. The minimal number of SWAP gates can obtained by Thorem1.

Detailed algorithm is as follows：

Step1: Scan. Search the non-adjacent quantum gates in quantum circuit. If the value of NNC is 0, we carry out Step2. If not, we carry out Step4.

Step2: Local ordering. Give qubits position vector expression for each quantum gate, and implement local ordering operation for non-adjacent quantum gates.

Step3: Line regression. Give the vector expressions of original line order and target line order of each quantum gate after local ordering operations, and take line regression algorithm for circuits.

Step4: End.

The detailed steps of Step2 in algorithm are as follows:

2.1: Define an array A[i] to record qubits position for each quantum gate. Search the positions of two non-zero elements in A[i].

2.2: Fix the non-zero element b which is in later row j in the A[i]. Move non-zero element a from the i^{th} row to the j-1th row.

2.3: Assign 0 to the element which is in the i^{th} row, change the value of element in the j-1th row to b-1. Display qubits position vectors after this operation.

The detailed steps of Step3 in algorithm are as follows:

3.1: Define an array B[n] to record circuit line order of each quantum gate after local ordering,

and define array U[i] to record target line order.

3.2: Traverse each element in B[n] and define d_i as the distance between B[n] and its map U[n] according to theorem 1. Calculate each d_i.

3.3: Add the number of SWAP gates which is the same as d_i for each quantum gate, and update the elements of the B[n] and U[n].

3.4: Repeat Step3 for remaining elements in the B[n], until B[n] equals to U[n].

Example 2：The linear nearest neighbor algorithm based on vector

As an example, consider a non-neighbor quantum circuit shown in Fig.10. By applying local ordering for each non-adjacent quantum gate is introduced in Fig.11. Then, by applying line regression algorithm introduced in Fig.12, we get the final quantum circuit in Fig.13. This algorithm results in NNC optimality and reduces quantum cost.

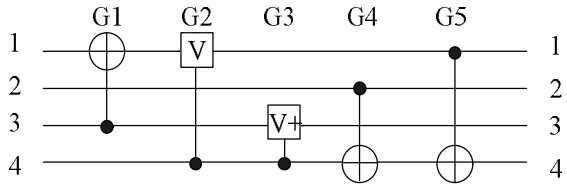


Fig.10 Initial quantum circuit

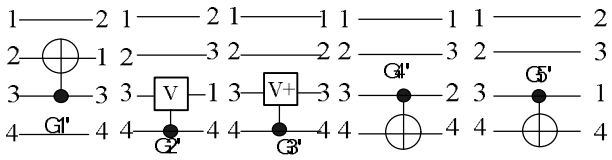


Fig.11 Local ordering

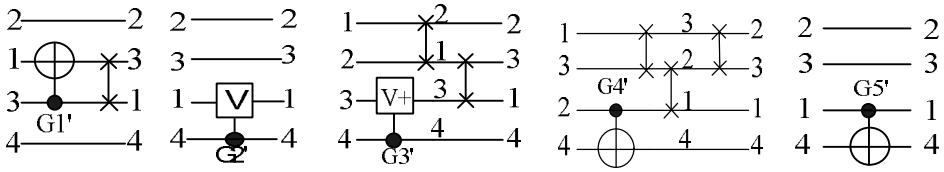


Fig.12 Implement line regression algorithm for each quantum gate

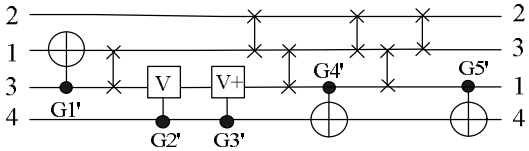


Fig.13 Integrated circuit

According to definition of qubits position vector, the qubits position vector of each quantum gate in Fig.10 is:

$$Q1=(1,0,3,0)^T,\ Q2=(1,0,0,4)^T,\ Q3=(0,0,3,4)^T,\ Q4=(0,2,0,4)^T,\ Q5=(1,0,0,4)^T,$$

We take local ordering operation for each quantum gate. For Q_1, there are two non-zero elements. The element 3 is in the bottom of vector which is located in the third row. Then we move another non-zero element 1 at the top of the vector which is located in the first row. The algorithm should change the value of element 1 into 0 and set the value of element which is in the second row into 2. So qubits position vector of G1 can be expressed as followed after local ordering:

$$Q1'=(0,2,3,0)^T$$

We take the same operations for the remaining quantum gates. The qubits position vector of each quantum gate vector after local ordering operation is:

$$Q1' = (0, 2, 3, 0)^T \ , \ Q2' = (0, 0, 3, 4)^T \ , \ Q3' = (0, 0, 3, 4)^T \ ,$$

$$Q4' = (0, 0, 3, 4)^T \ , \ Q5' = (0, 0, 3, 4)^T$$

We can conclude from the qubits position vectors of each quantum gate after local ordering operation that each vector has two non-zero elements which are located in two adjacent rows. In other words, each quantum gate gets to adjacent.

Due to local ordering operation, the quantum qubits of non-adjacent quantum circuit have reached adjacent. This operation also can change the output value of quantum circuit and circuit is no longer a whole. We need to integrate output of each quantum gate. It can be realized by line regression algorithm.

By definition of L_i, we take circuit line order of each quantum gate after local ordering as original line respectively and the output of last quantum gate as the target line. The target line vector can be expressed as:

$$out = (2, 3, 1, 4)^T$$

The output of circuit line vector of each quantum gate after local ordering are respectively shown in Fig.10:

$$L1 = (2, 1, 3, 4)^T \ , \ L2 = (2, 3, 1, 4)^T \ , \ L3 = (1, 2, 3, 4)^T \ ,$$

$$L4 = (1, 3, 2, 4)^T \ , \ L5 = (2, 3, 1, 4)^T$$

In Fig.11, for G1, we compare element in each row between vector L_1 and out. Element 2 both in the first row, so we don't do any operation; the element 1 of vector L_1 is in the second row, and it is in the third line in vector out. The algorithm moves this element in L_1 to the third line. It needs to add a SWAP gate between the second line and the third line. The output is $(2, 3, 1, 4)^T$ at this point which is same as vector *out*, so we stop operating. We take the same operations for the rest of the line vectors of quantum gates. The number of additive SWAP gates in quantum circuit for LNN is only six. However, if we use simple adding SWAP gate algorithm, the number of additive SWAP gates is NNC=2*6=12. In conclusion, this algorithm can reduce the number of SWAP gates and the quantum cost for circuit.

Experimental results and analysis

All approaches proposed in this paper have been implemented in standard C++. The testing hardware was an Intel (R) Core (TM) i5-2450M CPU@2.50 GHz with 4GB RAM and 64-bit OS of Windows 7. The typical benchmark circuits are used for our experiment. Since synthesis algorithms may use several multi-qubit gates during synthesis, all multi-qubit gates should be decomposed into a set of 2-qubit quantum gates for physical implementation. Decomposition tool proposed in [12] are extensively used for this purpose. Its specific implementation method can refer to [13]. Experiments include 3 to 8 lines of quantum circuits and the number of quantum gates from test data is 0 to 400.

Experimental results are compared with algorithm in reference [9], which is shown in Table 5.1. The average optimized quantum cost of quantum circuits is 39.69% with broad application. Except for a few of cases, the number of additive SWAP gates is less than the method in reference [9].

In order to verify the applicability of the algorithm in general circuits as well as large-scale circuits, this paper uses 3 to 10 circuit lines of quantum circuits composed of 2-qubit quantum gates to complete another set of experiments. The number of quantum gates in test circuits is 3 to 100. Comparing with the simple adding SWAP gate algorithm [10], the optimization rate of SWAP gates reaches 39.03% to 73.68% and the average optimization rate is 54.52%. With the increase of the quantum circuits, the optimization rate of SWAP gates has obvious advantages. Therefore, the algorithm also achieves quantum circuits for LNN on a large scale.

Table 1 Experimental Results

Benchmark	*n*	Original_qc	Old_qc	Vector_qc	Qc_optimization
3_17_13	3	14	26	16	38.46%
hwb4_52	4	23	63	43	31.75%
decod24_v3_46	4	9	21	18	14.29%
4_49_17	4	32	92	44	52.17%
4gt5_75	5	21	70	48	31.43%
4gt11_84	5	7	14	14	0
4gt10_v1_81	5	34	120	46	61.67%
4gt13_v1_93	5	16	74	42	43.24%
hwb5_55	5	104	335	265	20.90%
4mod5_v1_23	5	24	72	40	44.44%
4mod7_v0_95	5	38	121	80	33.88%
aj_e11_165	5	45	160	54	66.25%
alu_v4_36	5	31	98	70	28.57%
4gt4_v0_80	6	34	132	96	27.27%
4gt12_v1_89	6	42	141	150	0
hwb6_58	6	142	542	225	58.49%
mod5adder_128	6	83	330	102	69.09%
mod8_10_177	6	88	317	200	36.91%
ham7_104	7	83	327	196	40.06%
rd53_135	7	77	303	126	58.42%
hwb7_62	8	2325	12853	3049	76.28%
Ave_qc			39.69%		

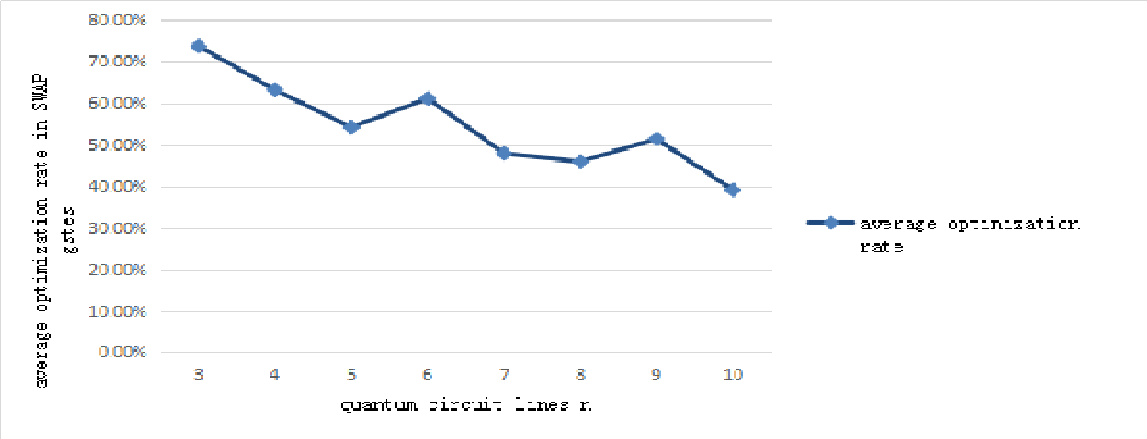


Fig.14 The average optimization rate of SWAP gates

Instructions: In table 1, the first row is the list of Benchmark circuits used in our experiment, next five rows are the number of quantum circuit lines (*n*), quantum cost (Original_qc) in the original circuit. Old_qc denotes the results reported by M Saeedi [9]. Vector_qc denotes quantum cost by using linear nearest neighbor algorithm based on vector. Then the percentage reduction of quantum (Qc_optimization) over [9] is shown in the last row. Ave_qc is the average percentage decrease in quantum cost.

The results of our algorithm on randomly generated quantum circuits are shown in Fig.14. The abscissa denotes the number of quantum circuit lines and the ordinate denotes the average optimization rate in SWAP gates of using our algorithm than simple adding SWAP gate algorithm. The data also indirectly reflect the algorithm can effectively reduce the quantum cost for quantum circuits.

Conclusion and discussions

This paper presents an algorithm of synthesizing quantum circuits through the research of relationship between vector and circuit, meanwhile, we use vector transformation to make control bit and target bit of each quantum gate into the form of adjacent. The proposed algorithm is suitable for a wide range of quantum circuits. Specially, the algorithm performs better in decreasing the number of SWAP gates. In order to solve the circuit lines confusion caused by local ordering, line regression algorithm is proposed. This algorithm doesn't change the function of original quantum circuit with the NNC value is 0. We can obtain the number of additive SWAP gates, that is quantum circuits of minimal qc.

In the next step of this work, we will extend the scope of the algorithm covers. Furthermore, we hope to reduce the number of SWAP gates. So that the algorithm can have greatly reduction on quantum cost for quantum circuits.

Acknowledgements

This work was financially supported by the National Natural Science Foundation (61402244),and Natural Science Foundation of the Colleges Jiangsu Province (14KJB520033).

References

[1] B Schaeffer, M Perkowski.: Linear reversible circuit synthesis in the linear nearest-neighbor model

[J]. IEEE International Symposium on Multiple-Valued Logic. 19:157-160 (2012)

[2] A Shafaei, M Saeedi, M Pedram.: Optimization of quantum circuits for interaction distance in linear
nearest neighbor architectures [J]. Design Automation Conference, 21(6):1-6 (2013)
[3] B Schaeffer, M Perkowski.: A cost minimization approach to synthesis of linear reversible circuits
[J]. Computer Science, 43(1):133-168 (2014)
[4] VA Kalmychkov, AV Krasilnikov, IV Matveeva.: Quantum circuits specifications design with lexical
verification [J]. XVIII International Conference on Soft Computing & Measurements (2015)
[5] A Kole, K Datta, I Sengupta.: A heuristic for linear nearest neighbor realization of quantum circuits
by SWAP gate insertion using N-Gate lookahead [J]. IEEE Journal on Emerging & Selected Topics
in Circuits & Systems, 6(1):1-11 (2016)
[6] MM Rahman, GW Dueck, A Chattopadhyay, R Wille.: Integrated synthesis of linear nearest
neighbor ancilla-Free MCT Circuits [J]. IEEE International Symposium on Multiple-valued Logic
(Ismvl) (2016)
[7] Y Hirata, M Nakanishi, S Yamashita, Y Nakashima.: An efficient conversion of quantum circuits to
a linear nearest neighbor architecture [J]. Quantum Information & Computation, 11(1):142-166 (2011)
[8] Z Sasanian, R Wille, DM Miller.: Realizing reversible circuits using a new class of quantum gate [J].
Design Automation Conference, 8(8):36-41 (2012)
[9] M Saeedi, R Wille, R Drechsler.: Synthesis of quantum circuits for linear nearest neighbor
architectures [J]. Computer Science, 10(3):355-377 (2012)
[10] K He, X. Tan, H Wang, G Shi.: GPU-accelerated parallel sparse LU factorization method for fast
circuit analysis [J]. IEEE Transactions on Very Large Scale Integration Systems, 24(3):1-1 (2015)
[11] S Lee, SJ Lee,T Kim, JS Lee,et al.: The cost of quantum gate primitives [J]. Journal of
multiple-valued logic and soft computing, 12(5):561-573 (2006)
[12] M Alfailakawi, L Alterkawi, I Ahmad, S Hamdan.: Line ordering of reversible circuits for linear
nearest neighbor realization [J]. Quantum information processing, 12(10): 3319-3339 (2013)
[13] A Lye, R Wille, R Drechsler.: Determining the minimal number of SWAP gates for
multi-dimensional nearest neighbor quantum circuits [C] //Design Automation Conference
(ASP-DAC), 2015 20th Asia and South Pacific. IEEE, 178-183 (2015)