IEICE TRANS. FUNDAMENTALS, VOL.E106–A, NO.2 FEBRUARY 2023

PAPER

An Efficient Method to Decompose and Map MPMCT Gates That Accounts for Qubit Placement

Atsushi MATSUO[†,††a)], **Wakaki HATTORI**[†b)], ***Nonmembers*, *and*** **Shigeru YAMASHITA**[†c)], ***Senior Member***

SUMMARY *Mixed-Polarity Multiple-Control Toffoli (MPMCT)* gates are generally used to implement large control logic functions for quantum computation. A logic circuit consisting of MPMCT gates needs to be mapped to a quantum computing device that invariably has a physical limitation, which means we need to (1) decompose the MPMCT gates into one- or two-qubit gates, and then (2) insert *SWAP* gates so that all the gates can be performed on *Nearest Neighbor Architectures (NNAs)*. Up to date, the above two processes have only been studied independently. In this work, we investigate that the total number of gates in a circuit can be decreased if the above two processes are considered *simultaneously* as a single step. We developed a method that inserts SWAP gates while decomposing MPMCT gates unlike most of the existing methods. Also, we consider the effect on the latter part of a circuit carefully by considering the qubit placement when decomposing an MPMCT gate. Experimental results demonstrate the effectiveness of our method.

key words: *quantum circuit, Mixed-Polarity Multiple-Control Toffoli (MPMCT) gate, nearest neighbor architecture (NNA)*

1. Introduction

Quantum algorithms consist of two parts: one is for quantum-specific computation and one for classical logic functions. In most cases, the former part does not change for the same quantum algorithm, but the latter one will need to change depending on the problem instance [1], [2]. It is thus necessary to design classical logic functions for each problem instance [3].

To achieve a large logic function with a quantum circuit, the most popular logic gate is called *Mixed-Polarity Multiple-Control Toffoli (MPMCT)* gates. After generating a quantum circuit including large MPMCT gates, two processes are required to implement the circuit on a real quantum computing device. First, we need to decompose MPMCT gates with many qubits into elementary gates, i.e., one- or two-qubit gates [4]–[7], because only one- or two-qubit operations are physically supported. In this paper, we call this process "the decomposition of MPMCT gates." Next, in the second process, we need to change the qubit placement by appropriately inserting *SWAP* gates [8]–[14] so that the resulting circuit can be implemented on a *Nearest Neighbor Architecture (NNA)* [15] that supports two-qubit operations *only* between adjacent two qubits. When a quantum circuit can be implemented on an NNA, we call it *NNA-compliant*. We also call this second process "mapping to an NNA."

To date, almost all previous works have considered the above processes separately. As such, when decomposing MPMCT gates, these methods do not consider the effect on the latter mapping to an NNA. They focus first on decomposing MPMCT gates into as small a number of elementary gates as possible, and then after decomposing all the MPMCT gates in a quantum circuit, they insert SWAP gates to map elementary gates to the NNA. However, the number of necessary SWAP gates to make a circuit NNA-compliant typically depends on how the MPMCT gates were decomposed in the first process. For example, even though the MPMCT gates are decomposed into the least number of elementary gates in the first process, there may be cases where many SWAP gates are required for the second process, which means the final circuit will be sub-optimal. While there is one previous work that considers the two processes as a single step by using templates [16], it does not consider the current qubit placement when decomposing the MPMCT gates. There is thus still room for improvement, especially when there are many MPMCT gates in a quantum circuit.

In this paper, we propose a method that implements the two processes as a single step by decomposing an MPMCT gate into elementary gates and mapping each elementary gate to an NNA at the same time. This is accomplished by the following techniques.

- Select the ancillary bits used in the recursive decompositions of an MPMCT gate carefully.
- Divide the control bits of an MPMCT gate by k-means clustering.
- Use the templates and the desired adjacent relation graphs of qubits for decomposing and mapping small MPMCT gates.
- Insert SWAP gates to determine a qubit placement that satisfies the adjacent relation graph of qubits.

After giving an overview of the previous methods in Sect. 2, we present our proposed techniques in Sect. 3. In Sect. 4, we report the results of experiments demonstrating the effectiveness of the proposed method. We conclude in Sect. 5 with a brief summary and mention of future work.

Manuscript received May 6, 2022.
Manuscript revised July 27, 2022.
Manuscript publicized August 10, 2022.

[†]The authors are with the Graduate School of Information Science and Engineering, Ritsumeikan University, Kusatsu-shi, 525-8577 Japan.
[††]The author is with the IBM Quantum, IBM Research - Tokyo, Tokyo, 103-8510 Japan.
a) E-mail: matsuoa@jp.ibm.com
b) E-mail: doyle@ngc.is.ritsumei.ac.jp
c) E-mail: ger@cs.ritsumei.ac.jp
DOI: 10.1587/transfun.2022EAP1050

2. Background

As discussed in Sect. 1, a popular logic primitive gate for quantum circuits is the Mixed-Polarity Multiple-Control Toffoli (MPMCT) gate, which is a quantum gate consisting of the target bit t, the positive control bits $\{x_{i_1}, x_{i_2}, \ldots x_{i_k}\}$, and the negative control bits $\{x_{i_k+1}, x_{i_k+2}, \ldots x_n\}$. It maps t to $(x_{i_1} x_{i_2} \ldots x_{i_k} \overline{x_{i_{k+1}}}\ \overline{x_{i_{k+2}}} \ldots \overline{x_{i_n}}) \oplus t$, where the notation $\oplus$ represents the exclusive-OR operation. Thus, the value of its target bit is inverted if all the positive and negative control bits are set to 1 and 0, respectively. An example is shown in Fig. 1, where positive and negative control bits are represented as black circles and white circles, respectively. In this example, the leftmost gate is an MPMCT gate with nine positive control bits.

It is almost impossible to implement a quantum gate acting on more than two qubits. Therefore, to create an MPMCT gate with many control bits, we need to decompose the MPMCT gate in a given circuit into several *elementary gates* consisting of one or two qubits each. Among the many studies on how to decompose MPMCT gates, the decomposition proposed by [5] is one of the best decomposition so far if we can use one ancillary bit. In the example shown in Fig. 1, an MPMCT gate with many control bits is decomposed into four MPMCT gates with fewer control bits (M_0, M_1, M_2, and M_3), and four two-qubit gates (G_0, G_1, G_2, and G_3). This decomposition can be recursively applied until the number of control bits is two or one. The MPMCT gate with one (positive) control bit is a CNOT gate, and the one with two (positive) control bits is a Toffoli gate. A Toffoli gate can be decomposed into five elementary gates.

When we decompose an MPMCT gate in this way, the control bits are divided into two groups: one used for the control bits of M_0 and M_2 and the other for M_1 and M_3. In our example, the control bits are divided into $\{q_0, q_{11}, q_{12}, q_{15}, q_{16}\}$ for M_0 and M_2, and $\{q_1, q_3, q_{14}, q_{17}\}$ for M_1 and M_3. We can also divide the control bits into two *arbitrary* groups, and select any one of the unused bits as an ancillary bit. Thus, we select the ancillary bit and the decomposition of the control bits *at random* when we decompose an MPMCT gate.

While this is the decomposition used in our proposed method, any other decomposition form can be easily adopted into our framework. Our method can therefore utilize new and better forms of decomposition as they are discovered.

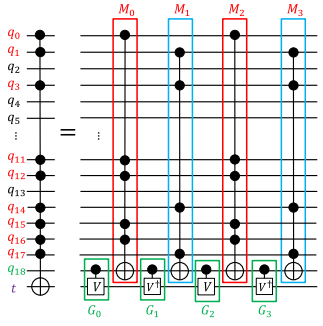

Fig. 1 Example of decomposing an MPMCT gate.

The current (or even the future) quantum devices are based on NNAs that supports two-qubit operations *only* between two adjacent qubits. Thus, after obtaining a circuit consisting of only elementary gates, we may need to map the circuit to an NNA.

To make a circuit consisting of only elementary gates NNA-compliant, we change the qubit placement by inserting SWAP gates repeatedly. For example, if there is a two-qubit gate acting on q_i and q_j that are not adjacent, we move the location of q_i so that it is next to q_j by repeatedly swapping the location of q_i and the adjacent qubit on a path between q_i and q_j. If there are many such gates (acting on distant qubits), it is not trivial to make a circuit NNA-compliant with few total SWAP gates. Indeed, there have been many studies on how to insert the SWAP gates more efficiently, e.g., [8]–[10].

In the following, we use the notation $S(q_i, q_j)$ to describe an operation to swap the quantum states of q_i and q_j by means of a SWAP gate that can be implemented with three CNOT gates.

3. Proposed Method

3.1 Overview of Proposed Method

We propose performing both processes (i.e., decomposing MPMCT gates and inserting SWAP gates for the decomposed gates) *simultaneously* as a single step that considers the current qubit placement to map a quantum circuit to an NNA. In the following, we explain the four key ideas that are beneficial to understanding our method.

Our Idea 1: Select the ancillary bit carefully when decomposing an MPMCT gate.

As we can see in Fig. 1, the previous methods select the ancillary bit and the division of the control bits *at random* when decomposing an MPMCT gate. Then, after decomposing all MPMCT gates into elementary (i.e., one- or two-qubit) gates, we need to consider how to insert SWAP gates to make the circuit NNA-compliant.

Our first observation is that the number of necessary SWAP gates to make a circuit NNA-compliant depends on how we divide the control bits into two groups and which qubit we choose for the ancillary bit when we decompose the MPMCT gates. Therefore, in the proposed method, when we decompose MPMCT gates, we select the ancillary bit with the following strategy.

- We choose the closest qubit to the target bit of the MPMCT gate as the ancillary bit, and move them next to each other by inserting SWAP gates.
- After selecting the ancillary bit and making it adjacent to the target bit, we keep their positions and we continuously use either the ancillary bit or the target bit used in the first decomposition as an ancillary bit for the successive decompositions.

For example, if t and q_{18} in Fig. 1 are close to each other, we choose q_{18} as the ancillary bit for the decomposition. We then insert SWAP gates to make them adjacent. After that, we do not move them, and t is used as the ancillary bit to decompose M_0, M_1, M_2, and M_3. Specifically, when we decompose M_0, we have four MPMCT gates whose target bits are q_{18}. To decompose these four gates, we use t as the ancillary bit. In this way, the ancillary bit becomes the target bit of the four MPMCT gates, and we use the target bit of an MPMCT gate as the ancillary bit for the decomposition of the next level. In this example, we always use either t or q_{18} as the ancillary bit for all the recursive decompositions. Since we keep the location of t and q_{18}, we do not need to insert any SWAP gate for the two-qubit gates generated at all the decompositions (e.g., G_0, G_1, G_2, and $G3$ in Fig. 1). This strategy enable us to decrease the number of necessary SWAP gates dramatically. The details are explained at greater length in Sect. 3.2.

Our Idea 2: Divide the control bits carefully when decomposing an MPMCT gate.

We also consider how to divide the control bits when we decompose MPMCT gates. It seems to be obvious that the qubits in the same group should be placed close to each other near in the current qubit placement, and so this is the strategy we adopt. Our method uses *k-means clustering* [17] to divide the control bits of an MPMCT gate into two groups based on the current qubit placement. We explain how k-means clustering is utilized for our purpose in Sect. 3.3.

Our Idea 3: Perform the two processes simultaneously as a single step.

As discussed in Sect. 2, the prior methods first repeatedly decompose MPMCT gates into elementary gates until all the MPMCT gates in a quantum circuit become elementary gates, and then, in a second process, insert SWAP gates so that the circuit can be mapped to an NNA. However, by dividing the whole task into two independent optimization problems, it is obvious that the first process may produce a bad intermediate solution for the second process.

Thus, in our proposed method, we change the current qubit placement at the same time as we decompose each MPMCT gate. For example, when decomposing M_0 in Fig. 1, we divide the control bits into two groups by the k-means clustering based on the *current* qubit placement. As detailed later, when we map a small MPMCT gate to an NNA, we can change the qubit placement if necessary. Thus, when we finish decomposing and mapping M_0, the qubit placement may change from the one before we decomposed M_0. Therefore, when we divide the control bits of M_1, we consider the current qubit placement, which may be different from the one before we decomposed M_0.

In other words, in contrast to the previous methods, we do not decompose M_0 and M_1 at the same time. Instead, we first decompose M_0 and then map it to an NNA by inserting SWAP gates (if necessary), which may change the qubit placement. After that, we consider the decomposition of M_1 based on the changed qubit placement. This allows us to consider an appropriate qubit placement for each MPMCT gate when we decompose it.

Our concrete method to perform the above two processes can be summarized as follows.

- We decompose only the first gate from the beginning of the circuit until the gate becomes *small* enough to be mapped to an NNA.
- When the first gate becomes small enough, we map it to an NNA by using a *template* prepared in advance.

In our method, we assume an MPMCT gate is small if the number of control bits is less than six (as discussed later).

The previous methods decompose all the four gates appearing at the decomposition of an MPMCT gate recursively until they become elementary gates, and then insert SWAP gates to map the circuit to an NNA after all the decompositions have been completed.

In contrast, our method decomposes *only* the first gate into four gates, and then again decomposes only the first gate of the four gates generated in the previous decomposition. We continue this until the first gate is decomposed to a *small* gate, and then, we map the small gate to an NNA by using a template. When we map a template circuit, we may need to change the qubit placement by inserting SWAP gates (discussed later). After that, we move on to the next MPMCT gate, decompose it, and map it to an NNA.

For example, assume the first gate to be decomposed has 27 control bits. Our method decomposes the gate into two MPMCT gates with 12 control bits and two MPMCT gates with 15 control bits. (The division of the control bits is determined by the k-means clustering, as explained in Sect. 3.3.) Next, our method only decomposes the first MPMCT gate with 12 control bits into, say, two gates with four control bits and two gates with eight control bits. Then, since the first gate with four control bits is small enough, and thus, we map it to an NNA by using a template (explained in Sect. 3.4). We then decompose the next gate with eight control bits further and continue the process.

Our idea 4: Map an MPMCT gate whose control bits are less than six to an NNA by using a template.

As explained in detail in Sect. 3.4, if the number of the control bits of an MPMCT gate is two, three, four, or five, we can compute the *desired positional relationship* of qubits for the MPMCT gate such that we can make it NNA-compliant with the smallest number of inserted SWAP gates if the qubit placement satisfies the relationship. We thus precompute the desired positional relationship of qubits for each small MPMCT gate, and prepare the mapping result for the gate as a template with the desired positional relationship of qubits. (This means that if the qubit placement satisfies the relationship for a template, we can map the template circuit with the smallest number of inserted SWAP gates.)

Thus, to map a template circuit to an NNA, we need to change the qubit placement by inserting SWAP gates to satisfy the desired positional relationship of qubits for the

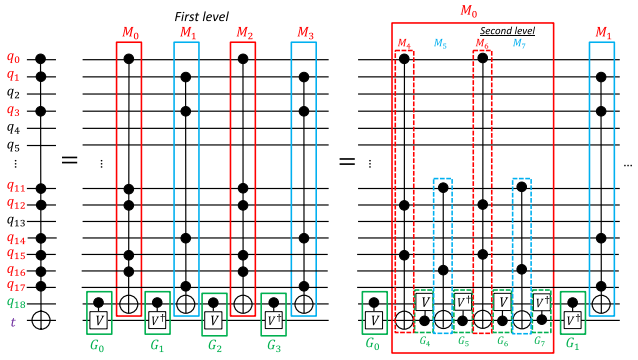

Fig. 2 Selecting the ancillary bits for decomposing MPMCT gates.

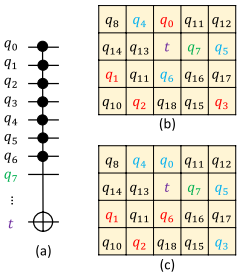

Fig. 3 (a) An MPMCT gate with six control bits. (b) We randomly divide the control bits from (a) into two groups and then (c) divide the control bits of each group using the k-means clustering.

template. We use the A* algorithm [18] to determine how best to insert SWAP gates to get to the desired positional relationship of qubits. We explain how to use the A* algorithm for that purpose in Sect. 3.5.

3.2 Selecting Ancillary Bits

Our method chooses the closest qubit to the target bit of an MPMCT gate as an ancillary bit when we decompose the MPMCT gate. We keep the ancillary bit and the target bit next to each other, and then alternately use one of them from the first decomposition as an ancillary bit for the successive decompositions.

Figure 2 shows how the proposed method selects ancillary bits when MPMCT gates are decomposed. In this example, we decompose the leftmost gate, and q_{18} is the adjacent qubit of the target bit t. At the first decomposition, we therefore use q_{18} as the ancillary bit, and then, the leftmost MPMCT gate is decomposed into four smaller MPMCT gates (M_0, M_1, M_2, and M_3), and four two-qubit gates (G_0, G_1, G_2, and G_3). In the decomposition of the second level, we use t as the ancillary bit, and decompose the M_0 generated in the first decomposition into four smaller gates (M_4, M_5, M_6, and M_7) and four two-qubit gates (G_4, G_5, G_6, and G_7). In this process, we always use either t or q_{18} as the ancillary bit for all the recursive decompositions. Furthermore, while inserting SWAP gates to map MPMCT gates M_0 through M_3 to an NNA, we always keep t and q_{18} next to each other. As a result, we do not need to insert any additional SWAP gate for the two-qubit gates generated when decomposing MPMCT gates, and we can decrease the number of necessary SWAP gates dramatically.

If all the adjacent qubits of the target bit t are used for the control bits of the MPMCT gates, they cannot be used as an ancillary bit. In such a case, we choose the closest bit to t that is not used for control bits as the ancillary bit. Then, we insert SWAP gates to make the ancillary and the target bits adjacent when necessary. We can still decrease the number of necessary SWAP gates substantially in such a case.

3.3 Dividing Control Bits by Clustering

Figure 3 shows how our proposed method divides the control bits of an MPMCT gate into two groups by k-means clustering [17]. In this example, we decompose the MPMCT gate in (a), and each qubit is placed on the 5×4 two-dimensional grid shown in (b), where the coordinates of the bottom left cell are (0, 0).

The control bits of the MPMCT gate are $\{q_0, q_1, q_2, q_3, q_4, q_5, q_6\}$, and we use q_7 as the ancillary bit. First, we randomly divide the control bits of the MPMCT gate into two groups, e.g., $C_1 = \{q_0, q_1, q_2, q_3\}$ and $C_2 = \{q_4, q_5, q_6\}$ in Fig. 3(b). We then calculate the center of the coordinates for each group, which is $(1.75, 1)$ for C_1 and $(2.33, 2)$ for C_2. We then reassign each control bit to the group whose center of the coordinates is closer with respect to the Manhattan distance. For example, in the case of q_0, the Manhattan distance from q_0 to the center of the coordinates of C_1 is $|2 - 1.75| + |3 - 1| = 2.25$. Similarly, that from q_0 to the center of coordinates of C_2 is $|2 - 2.33| + |3 - 2| = 1.33$. Thus, we reassign q_0 to C_2.

We repeatedly calculate the centers of each group and reassign the control bits to one of the groups until no change occurs in the reassigning process. As a result, we can eventually divide them into two groups $C_1 = \{q_1, q_2, q_6\}$ and $C_2 = \{q_0, q_3, q_4, q_5\}$, as shown in Fig. 3(c). The qubits in the same group here are close to each other on the grid, which means we can expect to decompose and map the MPMCT gate to an NNA with only a small number of SWAP gates.

3.4 Templates for Small MPMCT Gates

Since we can compute the best mapping of small MPMCT gates to an NNA in advance, we prepare decomposing templates for small MPMCT gates control bits numbering two, three, four, or five.

An MPMCT gate with three control bits can be decomposed into two CNOT gates, two Toffoli gates, and four two-qubit gates. For making the templates, we consider the following two points.

(1) A Toffoli gate has two types of decomposition. First, it can be decomposed into five two-qubit gates, as with the

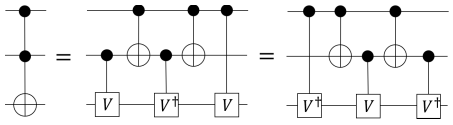

Fig. 4 Decompositions of a Toffoli gate.

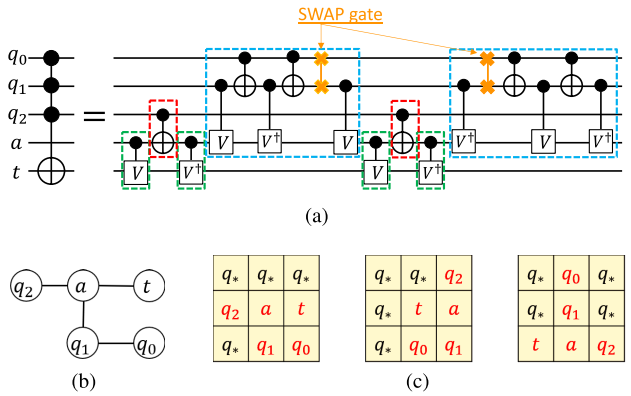

Fig. 5 (a) Template for an MPMCT gate with three control bits. (b) Desired adjacent relation graph for the gate in (a). (c) Qubit placements satisfying the desired adjacent relation graph in (b).

middle circuit in Fig. 4. Second, since a Toffoli gate is self-inverse, it can also be decomposed like the rightmost circuit in Fig. 4. As we will see later, we carefully choose one of the two decompositions to reduce the need for additional SWAP gates.

(2) After decomposing an MPMCT into elementary gates, we often obtain quantum gates that are not NNA-compliant, which means we need to insert additional SWAP gates to map them to an NNA. For that reason, we include those additional SWAP gates in our templates, and utilize the desired positional relationship (explained later).

We create the template for an MPMCT gate with three control bits as shown in Fig. 5(a). In this template, two Toffoli gates are decomposed into elementary gates while excluding the SWAP gates in the blue dashed boxes. Note that the first Toffoli gate is decomposed as the middle circuit in Fig. 4, while the second one is decomposed as the rightmost circuit in Fig. 4. This is done because if we decompose the second Toffoli gate in the same manner as the first, we need four additional SWAP gates in the template instead of two. We use this strategy in the other templates as well.

The graph in Fig. 5(b) describes the desired positional relationship of qubits for the template in Fig. 5(a). We call this graph a *desired adjacent relation graph*. Each node in the graph represents a qubit, and an edge represents a quantum gate operating between the connected nodes. From this graph, we can see there are quantum gates operating only between $(q_2, a), (a, t), (a, q_1)$, and (q_1, q_0) in the template. Thus, if we implement a qubit placement satisfying the desired adjacent relation graph, we can decompose an MPMCT gate with three control bits into elementary gates as the template, and then map them to an NNA with the qubit placement. Figure 5(c) shows an example of qubit placements satisfying

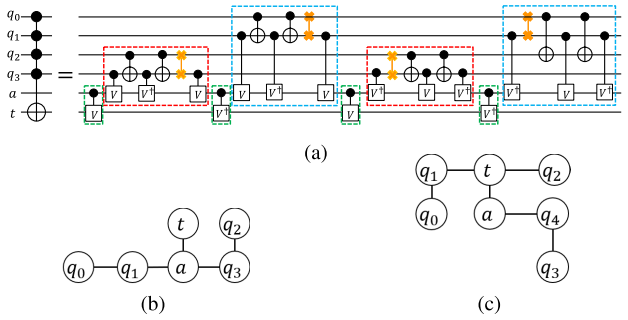

Fig. 6 (a) Template for an MPMCT gate with four control bits. (b) Desired adjacent relation graph of qubits for (a). (c) Desired adjacent relation graph for the template of an MPMCT gate with five control qubits.

the desired adjacent relation graph in Fig. 5(b). Note that q_* in Fig. 5(c) is a *don't care qubit*, which is not used in the MPMCT gate with three control bits. Each qubit placement in Fig. 5(c) seems completely different from the others, but as long as a qubit placement satisfies the desired adjacent relation graph in Fig. 5(b), we can perform the decomposition and then map an MPMCT gate with three control bits to an NNA with that qubit placement.

The case of an MPMCT gate with four control bits is similar to the case with three control bits discussed above. We create a template for an NNA including additional SWAP gates, and pre-compute the desired adjacent relation graph of qubits for the template. Figure 6(a) shows the template for an MPMCT gate with four control bits for an NNA including SWAP gates, and the graph in Fig. 6(b) shows the desired adjacent relation graph of qubits for the template in Fig. 6(a). We omit the details of the template for an MPMCT gate with five qubits due to space limitations, but the graph in Fig. 6(c) shows the desired adjacent relation graph of qubits for the template of an MPMCT gate with five control bits. Similar to the case of an MPMCT gate with three control bits, if qubit placements satisfy these desired adjacent relation graphs, we can decompose and map an MPMCT gate with four or five control bits to an NNA with the respective qubit placements. With these templates, we can decompose and map small MPMCT gates to an NNA efficiently.

3.5 Inserting SWAP Gates by the A* Algorithm

We use the A* algorithm to efficiently insert SWAP gates for determining the qubit placement that satisfies the desired adjacent relation graph for a template. The A* algorithm is a search algorithm that has been widely utilized for finding an efficient way to insert SWAP gates [19], [20].

In our case, a node corresponds to a qubit placement, and there is an edge between two nodes if the corresponding qubit placements can be interchanged with each other by a single SWAP gate. The start node corresponds to an initial qubit placement. Goal nodes correspond to qubit placements satisfying the input desired adjacent relation graph. Since there are usually multiple goal nodes in our scenario, we use the A* algorithm to find the path from the initial qubit place-

ment to one of the goal nodes with the least cost. Cumulative cost $g(n)$ is the number of SWAP gates used from the initial node to the current node, and estimation cost $h(n)$ is the total length of the Manhattan distance between the positions of the ancillary bit and the target bit, and that between each control bit and the ancillary bit. With the above information, we run the A* algorithm to find an efficient way of inserting SWAP gates to determine a qubit placement that satisfies the desired adjacent relation graph.

For the case of mapping multiple MPMCT gates to an NNA, we also calculate the costs of the next MPMCT gate while calculating the costs of the currently targeted MPMCT gate, and add them to the costs of the current MPMCT gate with some weight. By doing so, we can identify a qubit placement for decomposing and mapping the currently targeted gate. This qubit placement is also expected to be convenient for decomposing and mapping the next gate to an NNA.

3.6 Simultaneous Decomposition and Mapping

We combine all the methods mentioned in Sects. 3.2 to 3.5 and decompose and map MPMCT gates to an NNA simultaneously as a single step. Algorithm 1 shows the proposed method of this simultaneous decomposition and mapping. Also, the flow of simultaneous decomposition and mapping of an MPMCT gate with 27 control bits is shown in Fig. 7, where a node represents an MPMCT gate, and the number depicted in each node represents its number of its control bits.

In Fig. 7, r_0 represents an MPMCT gate with 27 control bits. First, our proposed method chooses an ancillary bit.

Algorithm 1: Decomposition of MPMCT gates in a quantum circuit and mapping of the quantum circuit to an NNA

Require: A quantum circuit including MPMCT gates.
Ensure: An NNA-compliant quantum circuit converted from the input circuit.

```
while If there are MPMCT gates in the quantum circuit do
    M ← the leftmost MPMCT gate in the quantum circuit.
    if the number of control bits of M > 8 then
        Choose an ancillary bit for M.
        Divide control bits of M into two groups by k-means
        clustering, and decompose M into four smaller MPMCT
        gates and four two-qubit gates.
    else if the number of control bits of M is 3, 4 or 5 then
        Create a qubit placement satisfying the corresponding
        desired adjacent relation graph by the A* algorithm.
        Decompose and map M to an NNA by using a template.
    else
        //the number of control bits < 8
        Calculate the total number of gates of mapping all the
        decomposed small MPMCT gates to an NNA for all the
        combinations.
        Decompose and map M to an NNA based on the best
        decomposition of the control bits.
    end if
end while
```

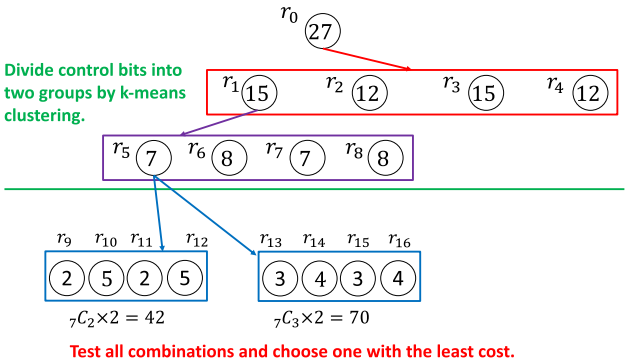

Fig. 7 Flow of simultaneous decomposition and mapping of an MPMCT gate with 27 control bits to an NNA.

Note that we alternately use either the ancillary bit or the target bit of the first decomposition as the ancillary bit for the successive decompositions. We then divide its control bits into two groups by k-means clustering. When they are divided into 15 and 12, r_0 is decomposed into two MPMCT gates with 15 control bits, r_1 and r_3, two MPMCT gates with 12 control bits, r_2 and r_4, and four two-qubit gates. Note that the two-qubit gate generated when decomposing large MPMCT gates is omitted in the figure. Similarly, r_1 is decomposed into four MPMCT gates, r_5, r_6, r_7, and r_8, and four two-qubit gates.

Let us consider the decomposition of r_5 that has seven control bits. As we mentioned, we divide the control bits into two groups by k-means clustering based on the current qubit placement. However, if the number of control bits is small enough, we can compare all the mapping results from all the possible divisions of the control bits. Thus, in our method, we try all the possible divisions for MPMCT gates with seven or fewer control bits as follows. We consider two types of decomposition for r_5. First, we can decompose it into two MPMCT gates with two control bits (r_9 and r_{11}) and two MPMCT gates with five control bits (r_{10} and r_{12}). Second, we can decompose r_5 into two MPMCT gates with three control bits (r_{13} and r_{15}) and two MPMCT gates with four control bits (r_{14} and r_{16}). For the former, there are ${}_7C_2 \times 2 = 42$ combinations for assigning the control bits. In the above, ${}_7C_2$ is doubled because for this decomposition, we have two options for ordering the MPMCT gates with two and five control bits: $(2,5,2,5)$ (as the figure) and $(5,2,5,2)$. We try all the 42 combinations to obtain the best decomposition and mapping by using the techniques discussed in Sects. 3.4 and 3.5. Similarly, for the latter, we try all the ${}_7C_3 \times 2 = 70$ combinations by using the same techniques. After that, we choose the one with the least number of gates, and then decompose and map r_5 to an NNA based on it. We then move on to the next MPMCT gate r_6, and decompose and map it to an NNA based on the qubit placement after decomposing and mapping r_5 to an NNA. Also, we divide the control bits of r_2 into two groups the k-means clustering based on the qubit placement after decomposing and mapping r_8 to an NNA.

In the proposed method, we decompose a large MPMCT

gate into smaller MPMCT gates by selecting an ancillary bit carefully, and then divide its control bits into two groups by k-means clustering considering the qubit placement. When the number of the control bits of an MPMCT gate is two, three, four, or five, we create a qubit placement satisfying the corresponding desired adjacent relation graph by the A* algorithm, and decompose and map it to an NNA by using the templates prepared in advance with the qubit placement. When the number of the control bits of an MPMCT gate is six or seven, we calculate the total number of gates for decomposing and mapping all the small decomposed small MPMCT gates to an NNA for every combination. We then choose the best one and decompose and map it to an NNA based on the chosen one. With the above process, we can decompose and map an MPMCT gate to an NNA simultaneously in a single step.

Lastly, we discuss the scalability of our method. Our method currently uses the A* algorithm to insert SWAP gates to achieve the desired adjacent relations for the templates of small MPMCT gates. Therefore, if a quantum circuit has a large number of qubits, the A* algorithm might not complete in a realistic time. In that case, we need another heuristic algorithm to insert SWAP gates, and/or we need to find an efficient method to divide a circuit into smaller ones to which our method can be applied. These improvement to treat large quantum circuits may be our future work.

4. Experimental Results

We implemented our proposed method (Sect. 3) and the previous method in C++ to compare the total number of two-qubit gates in a circuit after decomposition and mapping of MPMCT gates to a 2D grid NNA. The experiments were conducted on a machine with AMD Ryzen(TM) 7-3700X 3.60-GHz 8-Core 16-Thread, and two DDR4 8-GB memories. For the benchmark circuits, we selected only circuits that contain MPMCT gates with many control bits from the *Reversible Logic Synthesis Benchmarks Page* [21] along with some randomly created circuits consisting 20 MPMCT gates with the control bits of each MPMCT gate randomly determined.

The conventional methods do not perform two process in a single step, so we combined the best method in each field, the work by Miller et al. [5] and one by Zulehner et al. [20], for comparison. It decomposes the MPMCT gates using the best decomposition forms [5] from Fig. 1. However, unlike our proposed method, it randomly selects an ancillary bit and randomly divides the control bits of an MPMCT gate into two groups. For the SWAP gates, we implemented a method of inserting SWAP gates into a quantum circuit from left to right by using the A* algorithm based on the work by Zulehner et al. [20]. We think comparison with this combined method can demonstrate the effectiveness of our method.

Table 1 lists the average numbers of two-qubit gates and the average execution times of the proposed and conventional methods. *Circuit*, *Qubit*, *Gate*, and *n* indicate the name of the circuit, the number of qubits, the number of gates, and the maximum number of control bits of an MPMCT gate in the circuit, respectively.

Table 1 Comparison of results of proposed and conventional methods.

Circuit (Qubit, Gate, n)	*♯ of two-qubit gates*			*Execution time (ms)*	
	Conventional	*Proposed*	%	*Conventional*	*Proposed*
symmetric (10, 73, 9)	16906	10313	39	7843	145
symmetric (10, 74, 9)	15268	4642	70	6564	67
symmetric (10, 347, 7)	3273	1857	43	256	18
cycle (12, 19, 10)	2101	744	65	112	11
cycle (20, 48, 17)	12831	3167	75	5140	245
ham (15, 70, 4)	694	760	−10	14	25
ham (15, 109, 4)	335	205	39	5	4
ham (15, 132, 7)	3039	2876	5	245	87
mod-adder (20, 55, 10)	3089	1505	51	234	111
mod-adder (40, 210, 20)	38135	14817	61	38101	5710
hwb50ps (56, 589, 6)	3031	4526	49	10646	4526
hwb100ps (107, 1375, 7)	29420	21468	27	20053	18451
Random 1 (49, 20, 24)	10305	4114	60	3482	3244
Random 2 (64, 20, 32)	8912	3541	65	8912	6834
Random 3 (81, 20, 40)	25263	10941	57	18042	16102

From these results, we found that the proposed method can decompose and map circuits with MPMCT gates to an NNA with 47% fewer two-qubit gates on average. Because we can use few benchmark circuits containing large MPMCT gates, we also evaluated our method by using randomly generated circuits. From the experimental results, we consider that our method can decrease two-qubit gates especially when there are many large MPMCT gates. This is because considering qubit placements simultaneously is very effective in terms of decreasing the number of necessary SWAP gates when dividing the control bits of large MPMCT gates. In addition, when we decompose large MPMCT gates, we need to recursively decompose them into smaller MPMCT gates multiple times. In this case, the method of selecting ancillary bits (Sect. 3.2) significantly contributes to decreasing the number of SWAP gates required for the two-qubit gates generated by decomposing MPMCT gates. Thus, implementing both processes simultaneously is effective for achiving NNA-compliant circuits with a small number of two-qubit gates. Also, the execution time of the proposed method is sufficiently fast.

5. Conclusion

In this paper, we proposed a method that simultaneously decomposes and maps MPMCT gates to an NNA in a single step. We also proposed the following useful techniques.

- Select the ancillary bits used in the recursive decompositions of an MPMCT gate carefully.
- Divide the control bits of an MPMCT gate by k-means clustering.
- Use templates and the desired adjacent relation graphs of qubits for decomposing and mapping small MPMCT gates.
- Insert SWAP gates to determine a qubit placement that satisfies the adjacent relation graph of qubits.

The results of our experiments showed that, compared to a conventional method, our method can decrease the total number of two-qubit gates significantly.

In future work, we will investigate the use of relative

phase Toffoli gates [22], [23] for the decomposition. We can expect to decrease the total number of two-qubit gates even more, for two reasons. First, a relative phase Toffoli gate can be decomposed into fewer elementary gates than a conventional Toffoli gate. Second, all the two-qubit gates generated by the decomposition of a relative phase Toffoli gate interacts only with the control bits and the target bit of the relative phase Toffoli gate, unlike the decomposition of a conventional Toffoli gate. Obviously, the first advantage helps to improve our proposed method, while the second advantage helps to decrease the number of SWAP gates required in the templates. Utilizing the decomposition based on a relative phase Toffoli gate will be the focus of our future work.

Acknowledgments

This work was supported by JSPS KAKENHI Grant Numbers 20H05966 and 20H0057.

References

[1] L.K. Grover, "A fast quantum mechanical algorithm for database search," Proc. Twenty-eighth Annual ACM Symposium on Theory of Computing, STOC'96, New York, NY, USA, pp.212–219, ACM, 1996.
[2] P.W. Shor, "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer," SIAM J. Comput., vol.26, no.5, pp.1484–1509, Oct. 1997.
[3] S. Yamashita, S. Minato, and M.D. Michael, "DDMF: An efficient decision diagram structure for design verification of quantum circuits under a practical restriction," IEICE Trans. Fundamentals, vol.E91-A, no.12, pp.3793–3802, Dec. 2008.
[4] A. Barenco, C.H. Bennett, R. Cleve, D.P. DiVincenzo, N. Margolus, P. Shor, T. Sleator, J. Smolin, and H. Weinfurter, "Elementary gates for quantum computation," 1995. cite arxiv:quant-ph/9503016Comment: 31 pages, plain latex, no separate figures, submitted to Phys. Rev. A. Related information on http://vesta.physics.ucla.edu:7777/
[5] D.M. Miller, R. Wille, and Z. Sasanian, "Elementary quantum gate realizations for multiple-control toffoli gates," 2011 41st IEEE International Symposium on Multiple-Valued Logic, pp.288–293, May 2011.
[6] M. Soeken, Z. Sasanian, R. Wille, D.M. Miller, and R. Drechsler, "Optimizing the mapping of reversible circuits to four-valued quantum gate circuits," 2012 IEEE 42nd International Symposium on Multiple-Valued Logic, pp.173–178, May 2012.
[7] R. Wille, M. Soeken, C. Otterstedt, and R. Drechsler, "Improving the mapping of reversible circuits to quantum circuits using multiple target lines," 2013 18th Asia and South Pacific Design Automation Conference (ASP-DAC), pp.145–150, Jan. 2013.
[8] C. Lin, S. Sur-Kolay, and N.K. Jha, "PAQCS: Physical design-aware fault-tolerant quantum circuit synthesis," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol.23, no.7, pp.1221–1234, July 2015.
[9] M.Y. Siraichi, V.F.d. Santos, C. Collange, and F.M.Q.a. Pereira, "Qubit allocation as a combination of subgraph isomorphism and token swapping," Proc. ACM Program. Lang., vol.3, no.OOPSLA, pp.120:1–120:29, Oct. 2019.
[10] A. Zulehner, A. Paler, and R. Wille, "Efficient mapping of quantum circuits to the IBM QX architectures," 2018 Design, Automation Test in Europe Conference Exhibition (DATE), pp.1135–1138, March 2018.
[11] R. Wille, M. Saeedi, and R. Drechsler, "Synthesis of reversible functions beyond gate count and quantum cost," arXiv preprint arXiv:1004.4609, 2010.
[12] R. Wille, A. Lye, and R. Drechsler, "Optimal SWAP gate insertion for nearest neighbor quantum circuits," 2014 19th Asia and South Pacific Design Automation Conference (ASP-DAC), pp.489–494, 2014.
[13] T. Itoko, R. Raymond, T. Imamichi, and A. Matsuo, "Optimization of quantum circuit mapping using gate transformation and commutation," Integration, vol.70, pp.43–50, 2020.
[14] A. Matsuo, W. Hattori, and S. Yamashita, "Reducing the overhead of mapping quantum circuits to ibm q system," 2019 IEEE International Symposium on Circuits and Systems (ISCAS), pp.1–5, 2019.
[15] Y. Hirata, M. Nakanishi, S. Yamashita, and Y. Nakashima, "An efficient conversion of quantum circuits to a linear nearest neighbor architecture," Quantum Info. Comput., vol.11, no.1, pp.142–166, Jan. 2011.
[16] M. Saeedi, R. Wille, and R. Drechsler, "Synthesis of quantum circuits for linear nearest neighbor architectures," Quantum Inf. Process., vol.10, no.3, pp.355–377, 2011.
[17] J. MacQueen, "Some methods for classification and analysis of multivariate observations," Proc. 5th Berkeley symposium on mathematical statistics and probability, vol.1, pp.281–297, 1967.
[18] P.E. Hart, N.J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," IEEE Trans. Syst. Sci. Cybern., vol.4, no.2, pp.100–107, 1968.
[19] H. Wakaki and S. Yamashita, "Quantum circuit optimization by changing the gate order for 2D nearest neighbor architectures," 10th International Conference, RC 2018, Leicester, UK, Proceedings, pp.228–243, 2018.
[20] A. Zulehner, H. Bauer, and R. Wille, "Evaluating the flexibility of A* for mapping quantum circuits," Reversible Computation, M.K. Thomsen and M. Soeken, eds., Cham, pp.171–190, Springer International Publishing, 2019.
[21] D. Maslov, "Reversible Logic Synthesis Benchmarks Page," https://webhome.cs.uvic.ca/~dmaslov/, Accessed: 2019-12-26.
[22] S. Hu, D. Maslov, M. Pistoia, and J. Gambetta, "Efficient circuits for quantum search over 2D square lattice architecture," Proc. 56th Annual Design Automation Conference 2019, DAC'19, New York, NY, USA, Association for Computing Machinery, 2019.
[23] D. Maslov, "On the advantages of using relative phase toffolis with an application to multiple control toffoli optimization," arXiv:1508.03273, 2015.

Atsushi Matsuo received his B.E. and M.E. degrees in Information Science and Engineering from Ritsumeikan University, Shiga, Japan in 2011 and 2013, respectively. He is currently a Ph.D. candidate at the Graduate School of Information Science and Engineering, Ritsumeikan University, while working at IBM Research - Tokyo. His research interests include quantum circuit design, compilers for quantum circuits, and variational quantum algorithms.

Wakaki Hattori received his B.E. and M.E. degrees in Information Science and Engineering from Ritsumeikan University in 2018 and 2020, respectively. He is currently a graduate student of Graduate School of Information Science and Engineering, Ritsumeikan University, Shiga, Japan. His research interests include quantum circuit design, quantum cost reduction and synthesis of the nearest neighbor architecture.

Shigeru Yamashita is a professor at the Department of Computer Science, College of Information Science and Engineering, Ritsumeikan University. He received his B.E., M.E. and Ph.D. degrees in Information Science from Kyoto University, Kyoto, Japan, in 1993, 1995 and 2001, respectively. His research interests include new types of computation and logic synthesis for them. He received the 2000 IEEE Circuits and Systems Society Transactions on Computer-Aided Design of Integrated Circuits and Systems Best Paper Award, SASIMI 2010 Best Paper Award, 2010 IPSJ Yamashita SIG Research Award, and 2010 Marubun Academic Achievement Award of the Marubun Research Promotion Foundation. He is a senior member of IEEE, and a member of IPSJ.