FCM: A Fusion-aware Wire Cutting Approach for Measurement-based Quantum Computing

Zewei Mo, Yingheng Li, Aditya Pawar, Xulong Tang, Jun Yang, Youtao Zhang
University of Pittsburgh
Pittsburgh, Pennsylvania, USA
{zewei.mo,yil392,adp110,tax6,juy9}@pitt.edu,zhangyt@cs.pitt.edu

ABSTRACT

Measurement-based quantum computing (MBQC) is a promising quantum computing paradigm that carries out computation through *one-way* measurements on entangled photon qubits. Practical photonic hardware first generates a 2D mesh of resource states with each being a small number of entangled photon qubits and then exploits fusion operations to connect resource states to scale up the computation. Given that the fusion operation is highly error-prone, it is important to reduce the number of fusions for an MBQC circuit.

In this paper, we propose FCM, a fusion-aware scheme that exploits wire cutting to improve the fidelity of MBQC. By cutting a large MBQC circuit into several smaller subcircuits, FCM effectively reduces the number of fusions in each subcircuit and thus improves the computation fidelity. Given circuit cutting requires classical post-processing to combine the results of subcircuits, FCM strives to achieve the best cutting strategy under different settings. Evaluation of representative benchmarks demonstrates that, when cutting a large circuit to two subcircuits, FCM reduces the maximum number of fusions of all subcircuits by 59.6% on average (up to 69.1%).

CCS CONCEPTS

• **Computer systems organization → Quantum computing**.

KEYWORDS

Measurement-based Quantum Computing, Photonic, Fidelity

ACM Reference Format:
Zewei Mo, Yingheng Li, Aditya Pawar, Xulong Tang, Jun Yang, Youtao Zhang. 2024. FCM: A Fusion-aware Wire Cutting Approach for Measurement-based Quantum Computing. In *61st ACM/IEEE Design Automation Conference (DAC '24), June 23–27, 2024, San Francisco, CA, USA.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3649329.3657346

1 INTRODUCTION

Measurement-based quantum computing (MBQC) is a promising quantum computing paradigm that carries out computation through *one-way* measurements on entangled photon qubits [13]. Recent studies showed that practical photonic hardware first generates a 2D mesh of resource states with each being multiple entangled photon qubits and then exploits fusions to combine resource states to scale up to accomplish computation tasks [16]. Compared to mainstream quantum computing paradigms (e.g., superconducting [12]), MBQC exhibits good scalability and long coherence time. Photonic chips that integrate both quantum photonic and electronic components are being manufactured in the industry [6].

However, fusion is a highly error-prone, low-fidelity quantum operation. Assuming its failure probability is α, and mapping a quantum circuit to photonic hardware results in n fusions, the circuit success rate is $(1-\alpha)^n$, which deteriorates quickly with large n values. At $\alpha = 1\%$, an MBQC circuit having 69 or more fusions has a possibility lower than 50% to get the correct result. This clearly is a big concern nowadays as α is 10% [17] and a large quantum circuit may result in thousands of fusions. While the fusion failure is expected to reduce significantly with dramatic advancements in photonic technology in near-term [5], even at α=0.1%, we can only improve n to 693. Therefore, the number of fusion operations remains a major constraint in MBQC paradigm.

Zhang *et al.* propose a compilation framework *OneQ* [7] to map quantum circuits to physical resource states and take a pioneering step to reduce fusions from imperfect mapping. However, the number of fusions is mostly determined by the size and complexity of the quantum circuit. A large quantum circuit, even with *OneQ* optimization, may result in a physical mapping with more than an acceptable number of fusions. A simple yet effective approach is to cut a large quantum circuit into smaller subcircuits and exploit post-processing to combine the results from subcircuits, similar to that in wire cutting [18]. On the one hand, cutting improves fidelity as its fidelity is determined by the subcircuit with the most fusions. On the other hand, cutting introduces a post-processing overhead that increases exponentially with the number of cuts. Although previous studies have exploited wire cutting to decompose circuits [10, 18], they are fusion oblivious — since the number of fusions cannot be determined until the given circuit gets mapped to physical resource states. More importantly, existing wire-cutting-based solutions focus on generating subcircuits of limited width, which might result in an imbalanced number of fusions for each subcircuit, thereby suffering low overall fidelity.

In this paper, we propose FCM, a Fusion-aware Cutting approach for MBQC. FCM splits a large quantum circuit into multiple subcircuits such that each subcircuit has fewer fusion operations, which significantly improves the fidelity. We formulate the problem using mixed-integer programming (MIP) and exploit an MIP solver to optimize cutting decisions under different settings. When cutting a large circuit to m subcircuits, if our goal is to improve the fidelity, we strive to balance the number of fusion counts in all subcircuits; if our goal is to minimize the post-processing overhead, we prioritize the number of cuts in choosing the cutting positions. The balance

DAC '24, June 23–27, 2024, San Francisco, CA, USA

ACM ISBN 979-8-4007-0601-1/24/06.
https://doi.org/10.1145/3649329.3657346

of fusion count per subcircuit would be relaxed as long as each subcircuit has fewer than a threshold of fusions.

To summarize, we make the following contributions.

- We reveal the potential opportunity of decomposing a circuit for fusion reduction through wire cutting.
- We formulate the problem using MIP and optimize cutting decisions under different settings.
- We evaluate FCM on representative benchmarks. Our results show that FCM achieves significant fusion reductions and post-processing overhead reductions.

2 BACKGROUND AND RELATED WORK

2.1 Measurement-based Quantum Computing

Measurement-based quantum computing (MBQC) is a promising quantum computing paradigm that carries out computation through *one-way* measurements on entangled photonic qubits [14]. When deploying a quantum circuit for MBQC by the compiler, several steps of transformation [7] are required to map the circuit to photonic hardware, which is demonstrated by Figure 1. A quantum circuit (Figure 1(a)) is first transformed into the graph state (Figure 1(b)). It is then transformed into the fusion graph (Figure 1(c)) by leveraging fusion operations. Each node in the fusion graph represents one resource state, e.g., a three-qubit resource state [11] in the figure. In the end, the fusion graph is mapped to the photonic hardware using the routing algorithm [7] (Figure 1(d)).

Graph state in MBQC. The graph state represents a quantum circuit as a graph of entangled qubits, i.e., $G = (V, E)$, where qubits at vertices are initialized as the $|+\rangle$ state and controlled-Z gates at edges are applied to the qubits whose corresponding vertices in the graph are connected. Thus, the graph state can be defined as

$$|G\rangle = \prod_{(a,b)\in E} U^{\{a,b\}} |+\rangle^{\otimes V}$$

where $|+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ and operator $U^{\{a,b\}}$ stands for the controlled-Z gate between qubits a and b. With this graph state, MBQC can be carried by a pattern of Z-measurements and equatorial measurements with specific measurement angles. Then, we can obtain the measurement pattern for this graph state, which is generated by combining the graph state with the measurement basis of all qubits.

Figure 1 exploits an existing translation algorithm [2] to translate the sample circuit (Figure 1(a)) to the graph state (Figure 1(b)). The details of the algorithm can be found in [2]. In Figure 1(b), directed edges indicate the dependency between qubits, and qubits with 'in', 'out', or 'in/out' act as the input or output qubits in the given circuit. Also, qubits are marked by the different measurement bases in the graph state.

Fusion and fusion graph in MBQC. The fusion operation is a native measurement operation in linear optics for MBQC. The typical fusion with a success rate of 75% is called Bell fusion, which applies the two-qubit measurement in Bell state projection [19] to project a two-qubit state to an entangled state. The fusion used in this paper is in XZ- and ZX-basis, which can connect one graph state to the other one. Figure 1 demonstrates how one fusion connects two graph states. The fusion performed on qubits 2 and 3 eliminates these two qubits and attaches qubit 1 to qubit 4, merging two two-qubit graph states into one two-qubit graph state. Recent studies report that the success rate of single fusion can be improved further (e.g., improved from 75% to 90% in some cases [17]).

The fusion graph is to bridge the gap between the graph state and the photonic hardware — each node in the graph state is one qubit state while each node in the photonic hardware is one three-qubit resource state. For the generated fusion graph (Figure 1(c)), each node is a resource state that contains three entangled qubits; and each edge indicates the fusion operation that connects two resource states. During the transformation, when there exist no sufficient photonic qubits to be grouped into multiple resource states, the new qubits need to be introduced so that each photonic qubit belongs to one resource state.

Photonic hardware in MBQC. For Figure 1(d), nodes and edges in the photonic hardware represent resource states and fusions, respectively. All resource states of photonic hardware at one time slot are referred to as one physical layer [7]. A mapped quantum circuit may delay the photonic qubits in allocated resource states to future time slots, resulting in occupying multiple layers. In this paper, the number of physical layers occupied before and after circuit cutting is reported as the physical depth.

2.2 Wire Cutting in Quantum Circuit

Wire cutting is a powerful tool for decomposing an n-qubit quantum circuit into multiple smaller ones (referred to as subcircuits) so that each can fit on k-qubit quantum devices (k<n). For example, the sample quantum circuit in Figure 2 can be wire cut into two separated subcircuits c_1 and c_2.

Since the unitary matrix of any quantum gate can be decomposed into a set of orthonormal matrix bases, the result of the quantum circuit can be obtained from post-processing the results from two subcircuits. For qubit A in c_1, multiple measurements in different basis are performed on the upstream vertex of this cutting to generate three results. Thus, qubit A is called the measurement qubit in this cutting. On the other hand, in circuit c_2, four various initialized states are assigned to the initialization qubit A', the downstream vertex of the cutting. It leads to four different output values of c_2. In the end, all these results are combined to reconstruct the result of the original circuit. More details about wire cutting and reconstruction can be referred to [18].

It's important to note that the post-processing overhead increases exponentially with the number of cuts. Thus, finding a cutting solution with a small number of cuts is critical for mitigating the post-processing overhead.

3 MOTIVATION

Since fusion is a highly error-prone and low-fidelity operation in quantum computing, there exists a great necessity to reduce the number of fusions for MBQC. A careful study of MBQC compilation [7] reveals that the number of fusions in mapped photonic hardware comes from two sources: (1) The larger the quantum circuit is, the more fusions the photonic hardware needs. (2) The more complicated geometry one quantum circuit owns, the more extra

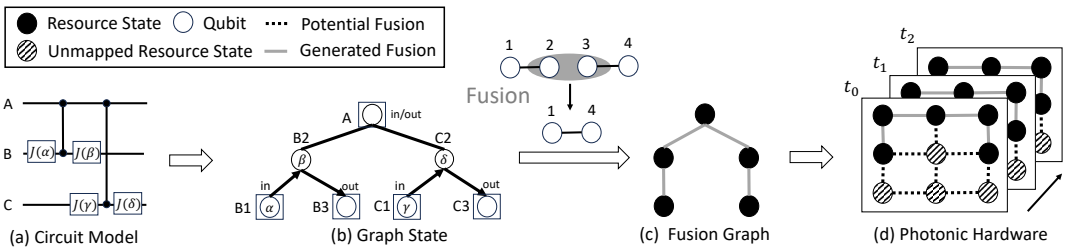

Figure 1: The compilation procedure of MBQC.

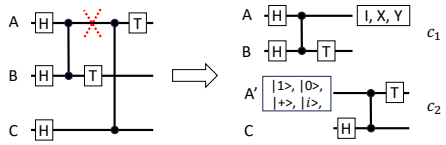

Figure 2: Wire cutting splits one three-qubit circuit into two two-qubit subcircuits.

fusions the mapping algorithm introduces to map its fusion graph to the photonic hardware.

In this paper, we are motivated to exploit wire cutting to decompose a large quantum circuit into several smaller subcircuits such that each subcircuit (1) has fewer circuit gates and thus needs fewer fusions; and (2) has simpler geometry and thus introduces fewer extra fusions during hardware mapping. However, the more cuts are applied, the higher the post-processing overhead is.

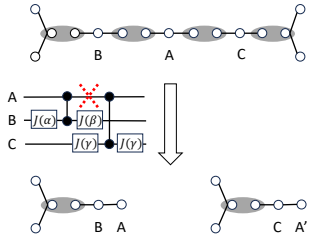

Figure 3: An example on how wire cutting reduces fusions. The original circuit requires four fusions in the fusion graph while two subcircuits generated from one wire cutting only introduce one fusion in each.

Figure 3 illustrates that cutting a large circuit into two smaller circuits reduces fusions in each subcircuit and improves the overall fidelity. For the sample circuit in Figure 3, we need four fusions in the fusion graph. After wire cutting, each of the two subcircuits needs one fusion only. Assuming the success rate of one fusion is 90%, wire cutting improves the fidelity from 65.6% to 90.0%.

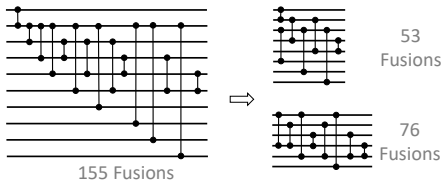

Figure 4: Applying four wire cuts on a complex circuits with 155 fusions after mapping generates two subcircuits, which only have 53 fusions and 76 fusions, respectively.

Figure 4 illustrates that cutting can reduce the extra fusion operations when mapping the fusion graph to photonic hardware. While a node on both fusion graph and photonic hardware represents a resource state, the photonic hardware exhibits layout constraints — one node has at most four neighbors on a 2D physical layer while each corner node has two neighbors.

Consequently, mapping a high-degree node in a graph state introduces extra fusions for the *routing* purpose, i.e., to connect different computation components. In the figure, eight CZ gates share the same control qubit q1 simultaneously, leading to a tree-shape graph (e.g. Figure 1(b)) and extra fusions for the routing. After performing four wire cuts, we obtain two subcircuits where the maximum number of CZ gates sharing the same control qubit is three. As a result, the numbers of fusions in the two subcircuits are 53 and 76, respectively. The sum of them is 17% smaller than the original number, indicating a significant reduction of extra fusions.

Existing wire cutting schemes, e.g., CutQC [18] and Clifford-based cutting [8], focus on decomposing a quantum circuit of width n to run on quantum computers with k physical qubits (n>k). They are fusion-oblivious when searching for cutting solutions. However, for wire cutting on photonic MBQC, our cutting goal is i) to balance fusions among different subcircuits and ii) to reduce the extra fusions during mapping. Thus, a novel wire cutting approach is needed to meet our design goal.

4 DESIGN

In this paper, we develop FCM, a Fusion-aware Cutting approach for MBQC, that cuts a large quantum circuit into smaller subcircuits with simpler geometry. We formulate the problem using MIP model so as to find the best cutting decisions under different problem settings. Once FCM splits the original circuit into subcircuits, it deploys each subcircuit onto photonic hardware using OneQ, and then combines the results through post-processing.

Intuitively, FCM strives to minimize the number of fusions in each subcircuit and the number of cuts at the same time. The former is to improve the overall computation fidelity while the latter is to reduce the post-processing overhead. Our observations reveal that the number of subcircuits largely determines the number of fusions in each subcircuit. The elimination of extra fusions after cutting, while being important, plays a secondary role. In Figure 4, extra fusions are reduced by nearly 20%. However, cutting an n-fusion circuit to four subcircuits instead of two subcircuits reduce the number of fusions from the range of $0.5n$ to the range of $0.25n$. Unfortunately, the more number of subcircuits FCM cuts the original circuit to, the more number of cuts it requires. Therefore, we may need to adjust our optimization goals under different settings.

We next present an illustrating example, assuming we are to cut a 2000-fusion circuit into two subcircuits under three different settings, we have following three kinds of solutions:

(i) Balanced-FCM (B-FCM): If our goal is to improve computation fidelity with a reasonable number of cuts, we seek for the best trade-off between evenly distributing fusions into subcircuits and reducing the number of cuts.
(ii) Threshold-FCM (T-FCM): If the success rate of a subcircuit that has around 1500 fusions is acceptable, we strive to minimize the post-processing cost while we relax the balance constraint and focus on minimizing the number of cuts.
(iii) Fusion-FCM (F-FCM): If we strive to improve fidelity as much as possible regardless of post-processing overhead, we may cut it to two subcircuits only focusing on balancing the fusions.

4.1 FCM

We model the quantum circuit as a directed acyclic graph G where the vertices set $V = \{v_1, ..., v_{n_V}\}$ represents multi-qubit gates modeled and the edges set $E = \{e_1, ..., e_{n_E}\}$ represents circuit wires. We focus on two-qubit gates when finding the cutting solution because single-qubit gates do not affect the circuit connectivity and do not introduce extra fusions during hardware mapping. Note that FCM also works for multi-qubit gates that involve more than two qubits as all the multi-qubit gates can be decomposed to the native gate set before the execution. We next list the parameters, the constraints, and the objective function in the MIP model.

4.1.1 Constant Parameters. In FCM, the parameter n_C specifies the number of subcircuits generated after cutting, i.e., the set of subcircuits are C = $\{c_1, ..., c_{n_C}\}$. Also, to distinguish gates that cause extra fusions in the mapping phase from those that do not, we assign a fusion weight f_v to each two-qubit gate — its value depends on whether the gate is sharing the same control qubit with others. For those do not, we have $f_v = 1$. For gates sharing the same control qubit, firstly we count the number of them and those not. Then, we map the given circuit to physical layers and generate fusions. Next, we search for f_v for gates sharing the same control qubits to enable the sum of fusion weights for all two-qubit gates equal to the number of generated fusions.

4.1.2 Variables. First, we model gates and wires in each subcircuit with the following variables.

$$y_{v,c} = \begin{cases} 1 & \text{if vertex } v \text{ is in subcircuit } c \\ 0 & \text{otherwise} \end{cases}, \forall v \in V, \forall c \in C$$

$$x_{e,c} = \begin{cases} 1 & \text{if edge } e \text{ is cut by subcircuit } c \\ 0 & \text{otherwise} \end{cases}, \forall e \in E, \forall c \in C$$

We then define the total fusion weight for each subcircuit and all subcircuits by the fusion weight of each gate as below:

$$W_c = \sum_{v \in V} f_v * y_{v,c}, \quad \forall c \in C \tag{1}$$

$$W = \sum_{c \in C} W_c \tag{2}$$

4.1.3 Constraints. We next present the constraints that the variables must satisfy when searching for the solution. First, each vertex must belong to one and the only one subcircuit, which can be defined as below:

$$\sum_{c \in C} y_{v,c} = 1, \quad \forall v \in V \tag{3}$$

Then, for each edge, it can only exist in one of the subcircuits. For an edge that is not cut, its two end vertices a and b belong to the same subcircuit. However, if an edge is cut by a subcircuit, a and b would belong to two different subcircuits. As a result, we have the following linear constraints:

$$\begin{aligned} x_{e,c} &\le y_{e_a,c} + y_{e_b,c} \\ x_{e,c} &\ge y_{e_a,c} - y_{e_b,c} \\ x_{e,c} &\ge y_{e_b,c} - y_{e_a,c} \\ x_{e,c} &\le 2 - y_{e_a,c} - y_{e_b,c} \end{aligned} \tag{4}$$

For the fusion weight of each sub-circuit, we define a upper-bound threshold T, which can be customized for different problem settings. Thus, we have the following constraint definition:

$$W_c \le T, \quad \forall c \in C \tag{5}$$

4.1.4 Objective Function. Since our main goal is to reduce the number of fusions in each subcircuit and to balance the fusions among subcircuits, we choose to evenly distribute gates sharing the same control qubits with other gates into different subcircuits. These gates tend to introduce more extra fusions as we discussed in Section 3. Because W is proportional to the one in the original circuit, according to AM-GM Inequality [9], we achieve minimized deviation among W_c values if their product is maximized. Hence, we choose the following objective function to maximize in order to find a better cutting solution:

$$L \equiv \prod_{c \in C} W_c \equiv \prod_{c \in C} \sum_{v \in V} f_v * y_{v,c} \tag{6}$$

It is known that the multiplication computation of variables is non-linear, which is hard to solve for the MIP model. To address that, we apply the piece-wise linear approximation of log function in Gurobi [20] to simplify this objective function as follows:

$$L \equiv \sum_{c \in C} \log \sum_{v \in V} f_v * y_{v,c} \tag{7}$$

It's notable that this piece-wise linear approximation of log function can be solved efficiently, similar as the linear function. By using the log function, we can force the first term in equation 9 to be in the same order of magnitude as the number of cuts in most cases. Besides, because post-processing overhead grows as more cuts are applied, we should also consider the number of cuts in the objective function so that we can reduce the fusion of all subcircuits with fewer cuts. As a result, the number of cuts and the final objective function is defined below:

$$K = \frac{1}{2} \sum_{c \in C} \sum_{e \in E} x_{e,c} \tag{8}$$

$$L \equiv \alpha \sum_{c \in C} \log \sum_{v \in V} f_v * y_{v,c} + \beta K \tag{9}$$

where α and β are meta parameters. Overall, the MIP model can be formulated as:

$$\begin{aligned} &maximize\ \ objective\ \ L(Eq.\ 9) \\ &s.t.\ \ constraint\ \ Eqs.\ (3,4,5) \end{aligned} \tag{10}$$

4.2 Case Studies

By choosing different meta parameters, we can adapt our FCM model to meet different optimization goals.

(i) B-FCM: We choose α=1,β=-1, indicating that we take both computation fidelity and post-processing cost into consideration. We set T=W, indicating we do not put a hard threshold on the number of fusions per subcircuit.
(ii) T-FCM: We set α=0 and β=-1, indicating we no longer emphasize balancing the fusions across subcircuits. Instead, we set T to a value determined by the single fusion success rate. In the experiments, we choose T=500 and 1000, to study its effectiveness on choosing different cutting decisions.
(iii) F-FCM: We set α=1 and β=0, indicating the post-processing overhead is ignored. While we set T=W, it has little impact as our goal is to balance fusions across subcircuits, which achieves the best computation fidelty with a given number of subcircuits.

5 EVALUATION

5.1 Benchmarks and Metrics

We conduct the evaluation of our methods using five benchmarks, including `supremacy` [15], Approximate Quantum Fourier Transform (`aqft`) [1], Bernstein–Vazirani (`bv`) [3], Quantum Approximate Optimization Algorithm (`qaoa`) [4], and `random`. The depth of `supremacy` and `random` is configured as 20 and 40, respectively. For `bv`, we randomly generate secret strings with the number of qubits as the length of the string. The quantum circuits in `qaoa` and `random` are generated randomly without any restriction, while gates in other benchmarks are generated by repeated specific patterns. In the evaluation, we set the area size of each physical layer as 16×16 and evaluate each benchmark with three different numbers of qubits as 16, 25, and 36. We apply the mathematical optimization solver Gurobi [20] as MIP model solver backend and set the maximum timeout of solving the MIP model as 300 seconds.

We compare our methods against the baseline, which is to take the original circuit as input and deliver the mapping result using OneQ. We use three metrics to evaluate our method: the number of fusions, the physical depth, and the number of cuts. For the number of fusions and the physical depth, we only report the maximum value of them among all subcircuits as the maximum determines the fidelity as we discussed in Section 3.

5.2 Result

5.2.1 B-FCM. Figure 5 reports the reduction of fusions achieved by B-FCM when cutting the original circuit into k subcircuits (k=2, 3, and 4). From the figure, the average reductions of the maximum number of fusions are 50.6%, 66.2%, and 75.6% when k=2, 3, and 4, respectively. Theoretically, cutting a circuit into two subcircuits, i.e, k=2, can achieve reduction higher than 50% if the fusions are perfectly balanced. Similar theoretical lower bound of reductions 66.7% and 75% exist for k=3 and 4. While our averages are close to their theoretical average lower bounds, B-FCM on quantum circuits

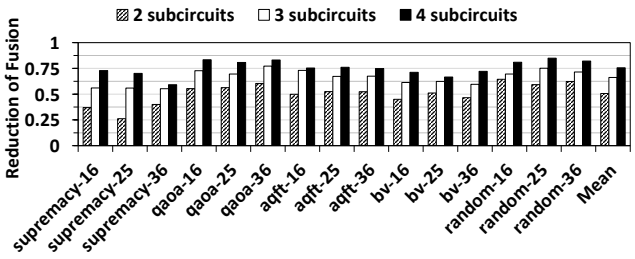

Figure 5: Reduction of the maximum number of fusions achieved by B-FCM on five benchmarks (*ABC-#* indicates benchmark *ABC* with # numbers of qubits).

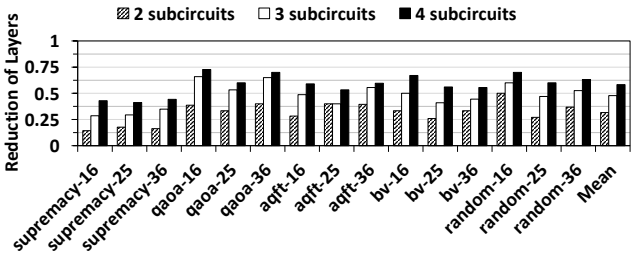

Figure 6: Reduction of the maximum number of physical depths achieved by B-FCM on five benchmarks (*ABC-#* indicates benchmark *ABC* with # numbers of qubits).

having random patterns, e.g., `qaoa` and `random`, achieves higher reductions. This is because these circuits have more cases in which multiple gates share the same control qubit and thus require more extra fusions during hardware mapping. B-FCM distributes these gates to different subcircuits, leading to more reductions.

Other benchmarks, e.g., `supremacy-25`, achieve 25% reduction for k=2, which is lower than the theoretical 50% for a balanced fusion split. B-FCM chooses the reported cutting solutions because it considers both the balanced fusion split and the number of cuts. We evaluate the best effectiveness of FCM on fusion reduction by F-FCM, which is demonstrated in section 5.2.2.

Figure 6 reports the reductions of the maximum physical depth among all subcircuits with different numbers of subcircuits k. From the figure, the average reductions of the physical depth are 31.6%, 47.8%, and 58.3% when k=2, 3, and 4, respectively. For `qaoa` and `random`, B-FCM can achieve 55.4% and 51.8% reductions, respectively. They are bigger than the overall average 45.9%. This is because when mapping the fusion graph to physical layers, to accommodate the cases that some gates share the same control qubits, qubits tend to be delayed to upcoming layers to avoid extra fusions. Therefore, cutting simplifies the subcircuit and thus reduces more physical layers.

Table 1 presents the number of cuts applied by B-FCM on five benchmarks to obtain subcircuits of different numbers. In most cases except `bv`, the number of cuts increases as the number of qubits or the number of subcircuits grows. This is because in `bv`, all the two-qubits gates are using different qubits as the same control qubit and the same qubit as the target qubit. Consequently, cuts required to decompose the original circuit into multiple subcircuits are all performed on the target qubits, and the number of cuts always equals to the number of subcircuit minus one.

Regarding the time cost, the average searching time on all benchmarks is 9.06 seconds, 104.63 seconds, and 195.01 seconds where the number of subcircuits is 2, 3, and 4, respectively. For all cases, B-FCM manages to provide solutions with a significant reduction of fusions within the given period.

Benchmark	#Qubit	#Cuts used for different #Subcircuits		
		2	3	4
supremacy	16	10	15	20
	25	12	19	25
	36	15	23	29
qaoa	16	14	23	32
	25	18	26	36
	36	19	33	44
aqft	16	5	10	15
	25	5	10	15
	36	6	12	18
bv	16	1	2	3
	25	1	2	3
	36	1	2	3
random	16	14	21	31
	25	17	30	41
	36	22	27	56

Table 1: The number of cuts applied by B-FCM to decompose circuits of various widths into subcircuits.

5.2.2 F-FCM. When the optimization goal is the fusion reduction, F-FCM reduces fusions by 59.6%, 76.0% and 83.6% on average (up to 69.1%, 85.5%, and 90.7%) for k=2, 3, and 4, respectively. The smallest improvement is 55.8% reduction for bv-25 with k=2, which is better than the theoretical 50% due to the elimination of extra fusions. Regarding the reduction of physical depths, F-FCM can achieve 42.5%, 56.5%, and 67.8% on average for k=2, 3, and 4, respectively.

Note, choosing β=0 results in significantly more numbers of cuts, the average numbers of cuts increase by 1.68x, 1.49x, and 1.38x when k = 2, 3, and 4, respectively.

5.2.3 T-FCM. In this section, we evaluate T-FCM only on benchmarks random and qaoa since they are the most complicated ones generating large numbers of fusions. Table 2 reports the reduction of cuts and the number of subcircuits under different thresholds T. A threshold T indicates that the fidelity is acceptable if the number of fusions for a mapped photonic hardware is below T.

When setting the threshold to be 1000 and 500, T-FCM reduces cuts by 12.9% and 15% on average over B-FCM, respectively. For benchmark random-16, since the number of fusions in the original circuit is below 1000, there is no need to perform any cut if T=1000.

5.2.4 Comparison to CutQC. CutQC [18] was proposed to cut a circuit into smaller subcircuits so that they can run on superconducting quantum computers with fewer qubits. The number of subcircuits and the number of quantum devices with limited size are constants. Naively adapting CutQC to solve the problem for MBQC leads to sub-optimal results.

A comparison between B-FCM and CutQC shows that B-FCM outperforms CutQC by 40.1%, 24.9%, and 15.6% on average when k=2, 3, and 4, respectively. The two schemes have similar numbers of cuts while B-FCM has one or two more cuts for some large circuits, due to choosing different cutting decisions.

5.2.5 Result Correctness. Based on the wire-cutting theory and the re-construction process introduced and proved in [18], our cutting solution outputs the same result as the original circuit.

Benchmark	#Qubit	Reduction of #Cuts (#Subcircuits) under different thresholds T	
		1000	500
random	16	- (-)	7.1% (2)
	25	15.0% (2)	12.2% (3)
	36	14.3% (2)	11.6% (4)
qaoa	16	13.3% (2)	21.7% (3)
	25	11.1% (2)	19.2% (3)
	36	10.5% (2)	18.2% (4)

Table 2: With different threshold T, the reduction of #Cuts gained by T-FCM compared to B-FCM on qaoa and random.

6 CONCLUSION

This paper is the first to present the insight that wire cutting can be leveraged to reduce fusions in MBQC. We propose FCM to decompose a large circuit into subcircuits for improved computation fidelity. It leverages the fusion weight to distinguish the two-qubit gates that tend to generate more extra fusions and constructs an MIP model to find the optimal cut solution under three typical settings. With appropriate adjustments, FCM can be adapted to meet the cutting demands with special design goals.

ACKNOWLEDGMENTS

The authors would like to thank the anonymous reviewers for their constructive feedback and suggestions. This work is supported in part by PQI Community Collaboration Award #007913, NSF grants #2011146, #2154973, #1725657, #1910413, and #2312157.

REFERENCES

[1] Adriano Barenco *et al.* 1996. Approximate quantum Fourier transform and decoherence. *Phys. Rev. A* 54 (1996), 139–146. Issue 1.
[2] Aleks Kissinger *et al.* 2020. PyZX: Large Scale Automated Diagrammatic Reasoning. *Electronic Proceedings in Theoretical Computer Science* 318 (2020), 229–241.
[3] Ethan Bernstein *et al.* 1993. Quantum complexity theory. In *Proceedings of the twenty-fifth annual ACM symposium on Theory of computing*. 11–20.
[4] Edward Farhi *et al.* 2014. A Quantum Approximate Optimization Algorithm.
[5] Fabian Ewert *et al.* 2014. 3/4-Efficient Bell Measurement with Passive Linear Optics and Unentangled Ancillae. *Phys. Rev. Lett.* 113 (2014), 140403. Issue 14.
[6] Hector Bombin *et al.* 2021. Interleaving: Modular architectures for fault-tolerant photonic quantum computing. *arXiv* (2021).
[7] Hezi Zhang *et al.* 2023. OneQ: A Compilation Framework for Photonic One-Way Quantum Computation. In *ISCA, 2023*. 12:1–12:14.
[8] Kaitlin N. Smith *et al.* 2023. Clifford-based Circuit Cutting for Quantum Simulation. In *ISCA, 2023*. 1–13.
[9] Limin Zou *et al.* 2015. Improved arithmetic-geometric mean inequality and its application. *Journal of Mathematical Inequalities* (2015), 107–111.
[10] Michael A. Perlin *et al.* 2021. Quantum circuit cutting with maximum-likelihood tomography. *npj Quantum Information* 7, 1 (2021), 1–8.
[11] Mercedes Gimeno-Segovia *et al.* 2015. From Three-Photon Greenberger-Horne-Zeilinger States to Ballistic Universal Quantum Computation. *Phys. Rev. Lett.* 115 (2015), 020502. Issue 2.
[12] M. H. Devoret *et al.* 2013. Superconducting Circuits for Quantum Information: An Outlook. *Science* 339, 6124 (2013), 1169–1174.
[13] Pieter Kok *et al.* 2007. Linear optical quantum computing with photonic qubits. *Reviews of Modern Physics* 79, 1 (2007), 135–174.
[14] Robert Raussendorf *et al.* 2003. Measurement-based quantum computation on cluster states. *Phys. Rev. A* 68 (2003), 022312. Issue 2.
[15] Sergio Boixo *et al.* 2018. Characterizing quantum supremacy in near-term devices. *Nature Physics* 14, 6 (2018), 595–600.
[16] Sara Bartolucci *et al.* 2023. Fusion-based quantum computation. *Nature Communications* 14, 1 (2023), 912.
[17] Thomas Kilmer *et al.* 2019. Boosting linear-optical Bell measurement success probability with predetection squeezing and imperfect photon-number-resolving detectors. *Phys. Rev. A* 99 (2019), 032302. Issue 3.
[18] Wei Tang *et al.* 2021. CutQC: using small Quantum computers for large Quantum circuit evaluations. In *ASPLOS, 2021*. 473–486.
[19] W. P. Grice. 2011. Arbitrarily complete Bell-state measurement using only linear optical elements. *Phys. Rev. A* 84 (2011), 042331. Issue 4.
[20] Gurobi Optimization LLC. 2023. Gurobi Optimizer Reference Manual. http://www.gurobi.com.