Orchestrating Measurement-Based Quantum Computation over Photonic Quantum Processors

Yingheng Li, Aditya Pawar, Mohadeseh Azari, Yanan Guo,Youtao Zhang,
Jun Yang, Kaushik Parasuram Seshadreesan, Xulong Tang
University of Pittsburgh, Pittsburgh, Pennsylvania, USA
Email: {yil392, adp110, moa125, yag45, juy9, kausesh, tax6}@pitt.edu, zhangyt@cs.pitt.edu

2023 60th ACM/IEEE Design Automation Conference (DAC) | 979-8-3503-2348-1/23/$31.00 ©2023 IEEE | DOI: 10.1109/DAC56929.2023.10247944

***Abstract*—Quantum computing has rapidly evolved in recent years and has established its supremacy in many application domains. While matter-based qubit platforms such as superconducting qubits have received the most attention so far, there is a rising interest in photonic qubits lately, which show advantages in parallelism, speed, and scalability. Photonic qubits are best served by the paradigm of measurement-based quantum computation (MBQC). To deliver the promise of measurement-based photonic quantum computing (MBPQC), the photon cluster state depth and photon utilization are two of the most important metrics. However, little attention has been paid to optimizing the depth and utilization when mapping quantum circuits to the photon clusters. In this paper, we propose a compiler framework that achieves automatic and dynamic depth and utilization optimizations. Our approach consists of an MBPQC mapping mechanism that maps optimized measurement patterns on a cluster state and a cluster state pruning strategy that removes all possible redundancies without impacting the circuit functions. Experimental results on five quantum benchmark with three different qubit numbers indicate our approach achieves an average of 63.4% cluster depth reduction and 22.8% photon utilization improvements.**

***Index Terms*—Quantum Computer, Compiler**

I. Introduction

The emerging field of quantum computing has embraced rapid development in the past decade, including the announcement of quantum supremacy [1]. As a promising computing paradigm, quantum computing has shown its advantages over classical computers in many applications such as integer factorization [17] and database search [8]. While the primary focus in quantum computing so far has been on matter-based platforms such as super-conducting [1] and trapped-ion [7] qubits, which can be categorized as gate-based quantum computers (GBQC), there is an increasing interest in photonic quantum computers (PQCs) which leverages measurement-based quantum computing (MBQC)—referred to as measurement-based photonic quantum computing (MBPQC) here. PQCs show advantages in parallelism, speed, and scalability [5] and do not require a cryogenic environment. Moreover, PQCs can be easily and securely networked with the help of fiber optic channels [6].

At a high level, quantum computing uses quantum circuits to implement quantum algorithms. A quantum circuit is comprised of a series of unitary quantum evolutions called quantum gates. In GBQC, which is widely adopted by superconducting quantum computers, quantum gates are sequentially applied to randomly initialized qubits, followed by measurements to obtain the computation results. In contrast, MBQC leverages local measurements on qubits that are in highly entangled resource states called *cluster states* [15] to perform unitary evolution [5]. The measurements process and propagate logical information forward along the resource cluster state, thereby accomplishing the computation.

PQCs use MBQC because photons are *flying* qubits, and the same photons cannot be statically retained to perform multiple quantum gates as in the GBQC model. Also, unlike GBQC qubits, a photonic qubit is destroyed once measured and cannot be re-initialized. MBQC, thankfully, does not require quantum gates or reuse of the same qubits after measurement. MBQC only requires each qubit in a resource cluster to be active for a relatively short time until it is measured. Also, cluster states of many photons can be generated efficiently [18]. These reasons make MBQC the most suitable computational paradigm for photonic qubits.

Two important metrics are related to MBPQC: i) cluster depth and ii) photon utilization. While the cluster depth depends on the mapping of a quantum circuit to MBPQC, a larger than required cluster depth has several issues that warrant careful optimization of the mapping to minimize cluster depth where possible. These include a greater possibility of photon loss with a larger cluster state and wastage of photon resources in the cluster states. The wasteful photons do not contribute to the quantum algorithm implemented in MBPQC but introduce additional detection noise associated with measurement. Quantum computing compiler frameworks have been designed primarily for GBQC, whereas MBPQC requires specialized compilers that carefully optimize these two metrics, for which only a few prior works are known.

In this paper, we comprehensively study cluster depth and photon utilization when mapping quantum circuits to MBPQCs. We observe a significant amount of photons that do not contribute to the quantum algorithm (i.e., they are wasted in the cluster states). We also observe that baseline mapping from the circuit to the cluster state is agnostic to the PQC characteristics and misses opportunities for depth reduction and utilization improvement. The major contributions of this paper can be summarized as follows:

- We provide a detailed analysis of the cluster depth and photon utilization when mapping quantum circuits to cluster states. It quantifies the deficiencies in existing mapping strategies and reveals the opportunities when performing photon-aware mapping.
- We design a compiler framework that automatically and dynamically achieves the optimized circuit for cluster state mapping. The proposed framework includes the following features. First, it maps a quantum algorithm onto a cluster state using a series of optimizations. Second, it removes redundancy without affecting the circuit's correctness.
- We use five quantum benchmarks with different numbers of qubits to evaluate the proposed framework. Experimental results indicate that the proposed approach, on average, reduces 63.4% of the cluster depth and improves 22.8% of the photon utilization.

II. Photonic processors and Opportunities

In this section, we introduce the basics of MBPQCs, including photonic cluster state and measurements. Then, we qualitatively and quantitatively reveal the opportunities and urgent need to develop a compiler framework for MBPQCs.

Authorized licensed use limited to: University of Pittsburgh. Downloaded on May 13,2024 at 15:28:35 UTC from IEEE Xplore. Restrictions apply.

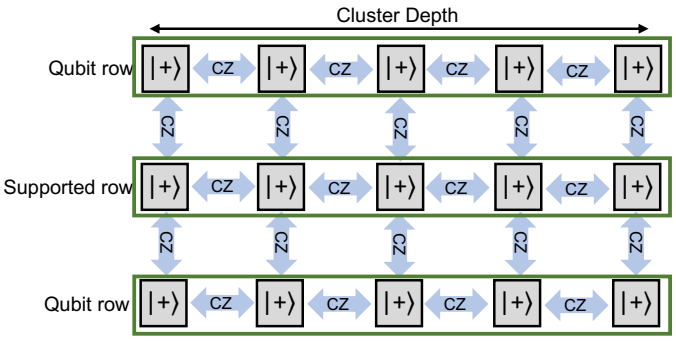


Fig. 1. Example of cluster states

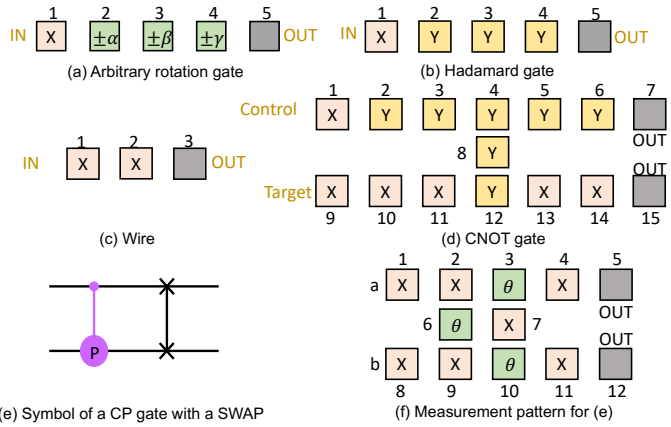


Fig. 2. The measurements pattern for quantum gates in MBQC

A. Photonic Cluster State

The first step of MBPQC is to prepare a cluster state, which is a highly entangled multipartite quantum resource state. A cluster state is represented as a graph state $|G\rangle$ that includes N vertices (qubits) and a set of edges E that form a regular lattice, e.g., a square lattice. To prepare an entangled cluster state $|G\rangle$, all N qubits are initially assigned with state $|+\rangle$, and Controlled Z (CZ) gates are applied to any two qubits connected by an edge [5]. Large-scale entangled cluster states have been demonstrated with photonic qubits [18]. In the rest of this paper, we focus on cluster states with photons. Fig. 1 shows an example of building a cluster state with three rows and a cluster depth of five. The 2-D cluster state is shown to be universal in quantum computation [16]. While an arbitrary single-qubit gate shown in Fig. 2(a) can be implemented on a 1-D cluster state, it is not universal because it cannot implement two-qubit gates.

A cluster state requires at least $2n-1$ rows to realize an n-qubit quantum circuit [16]. In this paper, we refer to *qubit rows* as the rows that store the logical qubits, and *supported rows* as the rows between qubit rows as shown in Fig. 1. Since the number of rows is fixed for a given number of qubits, the size of the cluster state is only proportional to its depth (x-axis length). In general, preparing a smaller (i.e., shallow depth) cluster state for MBPQC is relatively easier and typically requires fewer probabilistic fusion operations compared to preparing a larger (i.e., high depth) cluster state [18].

Also, once prepared, a larger cluster state suffers from higher error rates as compared to a smaller one since there is a greater probability of a photon being lost due to transmission losses in optical fiber. Moreover, even those photons that are not actually a part of the computation would still have to be measured off from the cluster, which, due to imperfect optical detection efficiencies, can introduce more noise in the propagated logical quantum information [18]. Thus, reducing the cluster depth lowers the error rate as well as saves resources. In the rest of the paper, we use **cluster depth** as the metric to quantify the depth of the cluster state.

B. MBQC on photonic cluster state

In MBPQC, the unitary quantum gates can be realized using a series of local measurements in a certain basis and order. In this section, we first show how measurements can realize the one-qubit and two-qubit gates. Then, we show how the z measurement is used in MBPQC. Due to lack of space, we refer the readers to [16] for formal mathematical proofs. Finally, we show how to use these measurements to realize quantum circuits in the photonic cluster state.

In MBPQC, five measurements in a chain can realize an arbitrary single-qubit rotation. In quantum computing, any unitary operation on a single qubit can be decomposed into the combination of three rotation gates [13]:

$$R[\alpha,\beta,\gamma] = R_X[\alpha]R_Z[\beta]R_X[\gamma] \tag{1}$$

where the R_X and R_Z are rotations about the x-axis and z-axis.

In MBPQC, the qubit rotation can be realized by measuring the qubits on the x-y plane with angle θ. The measurement on the x-y plane is an x measurement or a y measurement when the measurement angle θ is 0 or $\frac{\pi}{2}$, respectively. The first measurement in an arbitrary rotation is to prepare the input state $|in\rangle$. The second, third, and fourth measurements correspond to rotations about the x-axis, z-axis, and x-axis, respectively. The measurements angles from photon p_1 to photon p_4 are as follows: 0, $-\theta(-1)^{S_1}$, $-\theta(-1)^{S_2}$, and $-\theta(-1)^{S_1+S_3}$. S_1 to S_3 are the measurement outcomes for photons p_1 to p_3. After the four measurements, the quantum state of photon p_5 will be:

$$R' = \sigma_x^{S_2+S_4}\sigma_z^{S_1+S_3}R \tag{2}$$

where σ_x and σ_z are the Pauli matrices. Note that the $\sigma_x^{S_2+S_4}\sigma_z^{S_1+S_3}$ is the random byproduct produced by the measurements from photons p_1 to p_4. The fifth measurement is used for byproduct correction. Prior works have shown that the byproduct can be propagated and corrected at the end of the cluster state [16]. Hence, the quantum gates before the end of the circuit do not require the byproduct correction measurements. The input photon of a quantum gate is then placed at the output location of the previous gate. This may form consecutive x measurements that can be removed (details are discussed in Section III-A).

The realization of an arbitrary rotation is shown in Fig. 2(a). Note that the arbitrary rotations can be adjusted to implement other common unitary evolution gates, *e.g.*, the H and identity gates. In the H gate, y measurements are applied to photons p_2 to p_4. In MBPQC, any even number of x measurements form an identity gate, also called a *wire*. The patterns of these quantum gates are shown in Fig. 2.

Similarly, one can use measurements to implement two-qubit gates. Fig. 2(d) shows the measurement pattern for the CNOT gate. Specifically, photons p_1 and p_9 are inputs with x measurement while photons p_7 and p_{15} are used for byproduct corrections, which are only required at the end of the circuit. Another example is a unique measurement pattern shown in Fig. 2(f) that combines a CP gate and a SWAP gate shown in Fig. 2(e). This combines a Controlled Phase (CP) gate with an additional SWAP gate. The measurement angles of photons p_3, p_6, and p_{10} are based on the rotation phase and the previous measurement results. The simulation pattern becomes a

Authorized licensed use limited to: University of Pittsburgh. Downloaded on May 13,2024 at 15:28:35 UTC from IEEE Xplore. Restrictions apply.

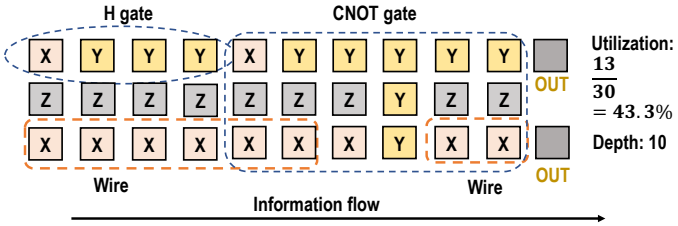


Fig. 3. Example of a quantum circuit.

SWAP gate or a CZ gate with a SWAP gate when the measurement angles are 0 or π, respectively. This means the SWAP gate is *not* required to be decomposed into three CNOT gates in MBPQC.

Another important characteristic of MBPQC is the z measurements and consecutive even number x measurements (i.e., wires) in the cluster state. First, as discussed above, the z measurement in MBPQC is not used to realize quantum gates but to remove photons from the cluster state. It destroys entanglement between photons and makes them ineffective for realizing a circuit [16]. Second, wires are used to "glue" neighboring photons to pass quantum information from left to right in the cluster. As a result, both the wire and the z measurements can be seen as "redundant" in MBPQC because they do not contribute to the real implementation of quantum algorithms. Thus, in the rest of the paper, we use **photon utilization** to quantify the number of photons not involved in z measurement and wires. We also show how our proposed optimizations improve photon utilization in MBPQC.

Fig. 3 shows a simple example of MBPQC. This is a 2-qubit quantum circuit with qubits q_0 and q_1. Specifically, an H gate is applied to qubit q_0, and a CNOT gate is applied between the q_0 and q_1 qubits. The wires are inserted to match the H gate so the bottom part of the CNOT gate does not shift to the left. The rest of the photons on the supported row are then removed by applying z measurements. The redundancy includes all z measurements and the wires. The utilization and depth are given in this example.

C. Related work

Many compiler-guided optimizations for quantum computers have been proposed [12], [14]. However, these works are designed for GBQCs, and they are not suitable for photonic quantum computers.

Firstly, existing works aim to reduce the number of CNOT gates due to their high error rate. Most CNOT gates are generated due to the decomposition of SWAPs and other two-qubit gates since CNOT is the only supported two-qubit gate in the IBM superconducting quantum computers. However, the SWAP and other two-qubit gates in MBPQC (shown in Fig. 2(e)) can be realized using their own measurement patterns and do not need to be decomposed into CNOT gates. Using the current compiler for MBPQC would lead to even more CNOT gates generated from gate decomposition, which results in more measurements and a larger cluster state.

Second, the existing works do not consider the special quantum gate patterns of MBPQC. For example, the CP gate in MBPQC comes with an additional SWAP gate, and the special CNOT pair, which will be discussed in section III-A, has its unique structure as shown in Fig. 4(d). These patterns lead to a different optimization space compared to other quantum computers. Since the existing compiler framework is incapable of the MBPQCs, there is an urgent need for a compiler specific to MBPQC.

III. Photonic Quantum Computer Compiler Framework

In this paper, we set to reduce the photonic cluster depth and improve photon utilization. To this end, we propose an automatic compiler framework that conducts cluster-aware mapping from the quantum algorithm to the cluster state. Specifically, the proposed framework consists of the following techniques: i) A mapping mechanism that utilizes circuit commutation to search for special CNOT pairs, reduce SWAP gates, and improve circuit parallelism. ii) A cluster state pruning approach that removes all the redundancy without affecting the circuit function.

A. Optimization Opportunities for MBPQC

We use Fig. 4 to illustrate the optimization opportunities in MBPQC.

Opportunity-I: Our first optimization motivation stems from the observation that two adjacent dependent gates can sometimes be switched by commutation. However, the two equivalent-function circuits may have different cluster depths and photons utilization.

For example, Fig. 4(a) shows a pair of two CNOT gates marked in two boxes. They are on three neighboring physical qubits. When mapping the circuit to cluster state, the corresponding circuit depth is 16, and the photon utilization is 28.8% (Fig. 4(c)). In contrast, if we switch the two CNOT gates in Fig. 4(b) to form a special CNOT pair [16], the circuit has the same functionality but the resulting cluster depth, and photon utilization can be significantly improved (cluster depth eight and 57.5% photon utilization in Fig. 4(d)).

Another example is that the order of logical CP gates also matters. For example, in Fig. 4(e) and Fig. 4(f), the two circuits have the same quantum function. However, the gate order in Fig. 4(f) can reduce two required SWAP gates because the first CP gate also performs an additional SWAP operation that the second CP gate needs. The corresponding MBPQC implementation is shown in Fig. 4(g) and Fig. 4(h). However, if option 2 is chosen in Fig. 4(e), only one additional SWAP gate is required. Although the photon utilization stays the same, the cluster depth was reduced from 16 to 8.

Opportunity-II: Our second optimization motivation is that consecutive single gates can be combined in MBPQC. Specifically, This approach is hardly used in GBQCs because many of them do not support the arbitrary rotation gate. However, all MBPQC support arbitrary rotation gates as in Fig. 2(a). For example, an arbitrary rotation gate and H gate in Fig. 4(i) are combined into only one arbitrary rotation gate. This optimization can reduce the cluster depth by N, where N is the number of single-qubit gates to be combined.

Opportunity-III: Our third optimization motivation is to reorder the independent gates to enable more parallelism. For example, the reordering of the two independent gates from Fig. 4(j) and Fig. 4(k) results in a reduction of depth by four photons because the latter order enables parallel execution of the CP gate and H gate. The MBPQC implementation is shown in Fig. 4(l) and Fig. 4(m). The cluster depth is reduced from 12 to 8, and the photon utilization is increased from 26.7% to 35%. Reordering of this nature, using CP gates, can not benefit GBQCs because CP gates need to be decomposed into two CNOT and three Phase gates. One of the Phase gates decomposed from the CP gate in the GBQC always executes in parallel with the H gate for both orders in Fig. 4(j) and Fig. 4(k).

Opportunity-IV: The last optimization opportunity is to remove the redundancy introduced in section II-B. As discussed in section II-B, the even number of connected x measurements are wire (identity gate), and they are redundant because they do not perform any useful quantum evolution. However, these patterns can not be removed naively since they may lead to pattern misalignment. For example, removing the wire part in Fig. 3 will destroy the functionality of the CNOT gate. In order to maintain the functionality of quantum circuits, the number of removed measurements in each row should

Authorized licensed use limited to: University of Pittsburgh. Downloaded on May 13,2024 at 15:28:35 UTC from IEEE Xplore. Restrictions apply.

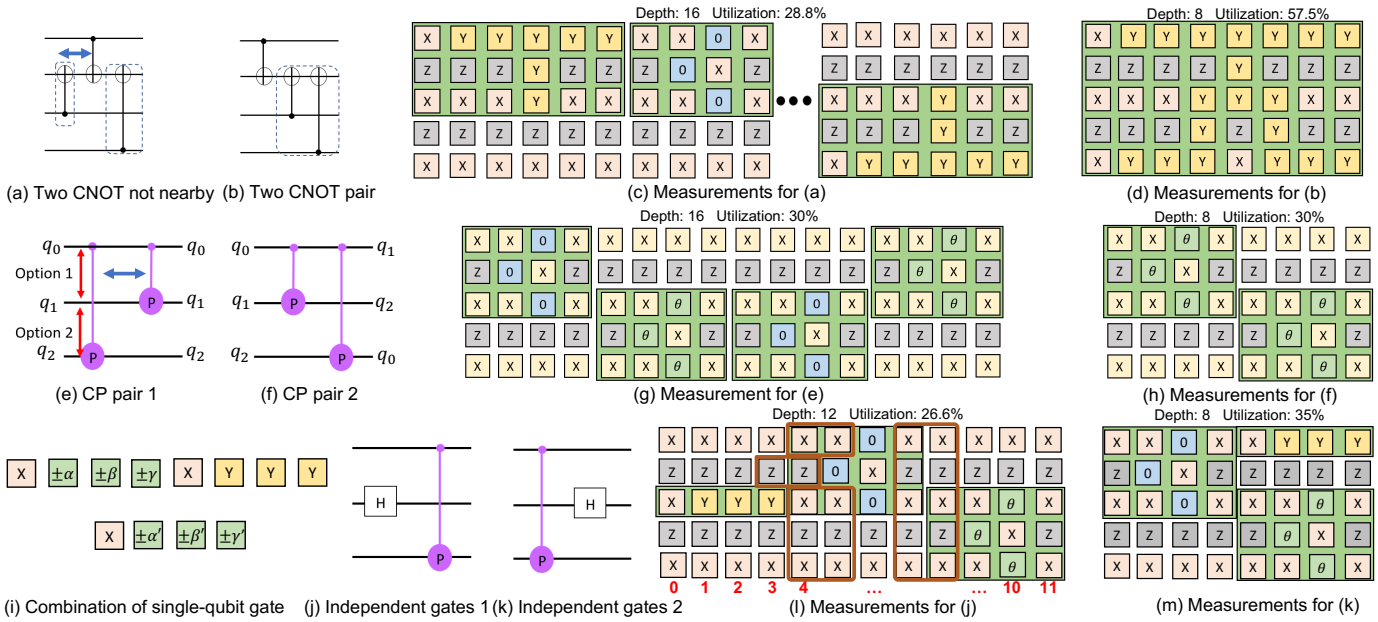


Fig. 4. Example of optimizations to reduce depth

be the same. As shown in Fig. 4(l), the wire part for each row can be removed since it will not cause any misalignment problems.

Overall potential: The original utilization, the new utilization, and cluster depth reduction for each of the five cases in the four optimizations in Fig. 4 are summarized in Table I. For the redundancy removal, we use the example in Fig. 4(l), where the redundancy marked in the orange boxes is removed. The utilization of all single-qubit gates is one because we assume they are implemented on a 1-D cluster state. As shown in Table I, the cluster depths of all five cases are reduced. Although the CP-CP commutation does not reduce the utilization in this example, it shortens the cluster state by 50%.

TABLE I
DEPTH AND UTILIZATION IMPROVEMENT FROM OPTIMIZATIONS

Optimization type	Formal utilization	New Utilization	Depth reduction
CNOT-CNOT pair	20.5%	41.1%	50%
CP-CP commutation	30%	30%	50%
Gate combination	1	1	N
Independent reorder	26%	35%	33.3%
Redundancy removal	26%	40%	33.3%

B. Our Approach

The proposed compiler framework consists of two stages: i) an MBPQC mapping mechanism that utilizes opportunities I, II, and III, and ii) a redundancy removal strategy that leverages opportunity IV to remove all the redundancy without impacting the circuit functions.

The optimization opportunities can be discovered by using circuit commutation [9]. Circuit commutation enables switching between two adjacent dependent quantum gates while maintaining the same quantum circuit function. For example, target-target and control-control commutation. These two commutations of two-qubit gates are shown in Fig. 4(a) and Fig. 4(e). We leverage these commutations to realize opportunities I to III step by step.

In the initialization, an input logical quantum circuit is first transformed into a Directed-Acyclic-Graph (DAG). The nodes in a DAG represent a quantum gate, and the edges represent the dependency between quantum gates. The quantum gates can be executed when their dependent gates have been executed. Here, we define the nodes without predecessors as the *front layer*. Hence, nodes within the same layer can be executed simultaneously at the logic level.

Our first step searches for all possible commutations within a window of MAX layers. If the commutation between two dependent CP gates is possible, we check which order minimizes the number of swap gates in the circuit. For example, the order of Fig. 4(f) is better than the order of Fig. 4(e). If the commutations involve CNOT gates, we then search for the two CNOT pairs as illustrated in Fig. 4(b) and Fig. 4(d). CNOT gates that follow the pattern shown in Fig. 4(d) can be implemented with significantly fewer photons than their naive implementation, as shown in Fig. 4(c).

If SWAP gates are still needed after applying the first step, we further apply a look-ahead strategy to minimize the impact of swaps on subsequent gates as step 2. The SWAP option that is then chosen is one that minimizes the SWAP requirement in the remaining circuit. This can be seen in Fig. 4(e), where option 1 causes two extra swaps to be added to the final circuit, and SWAP option 2 only requires one SWAP. After the above two steps, we then combine the single-qubit gate as our third step. As shown in Fig. 4(i), the consecutive single gates can be combined into one arbitrary rotation gate.

During each of the optimization steps 1-3, we continually update the DAG since these optimizations affect the circuit dependency after every commutation and combination. After obtaining the final updated DAG from step 3, we search for the best independent gate order to optimize the parallelism of the circuit in the window of MAX layers to optimize the depth and photon utilization further. The examples have been discussed in opportunity III.

The pseudo-code of our mapping mechanism is shown in Algorithm 1. The quantum algorithm is first transferred into a DAG D. Our framework will check all possible commutations between each layer l and the next MAX layer $l + MAX$ (lines 2-4). For each possible commutation, we apply each optimization step to a temporary DAG D', where $D' = D[l : l + MAX + 1]$ (line 6). We evaluate one more layer because commutation happens in layer $l + MAX$ and

Authorized licensed use limited to: University of Pittsburgh. Downloaded on May 13,2024 at 15:28:35 UTC from IEEE Xplore. Restrictions apply.

may affect the next layer $l + MAX + 1$. For example, the single-qubit gate in $l + MAX$ and $l + MAX + 1$ could be combined.

After commutation, the framework first searches for special CNOT pairs in D' and updates D' once finding a pair (lines 7-9). If the commutation is between two CP gates, their impact on the number of SWAP gates will be shown as a shorter depth of D', which will be computed at line 16. If SWAP gates are needed in D', the SWAP option, which minimizes the next SWAP distance, is used (lines 10-12). After that, the third step (lines 13-14) is applied to combine the consecutive single gates in D', and the fourth step (line 15) is applied to choose the best independent gates order for each layer in D'. After performing all the optimizations, the layers $D[l : l+MAX+1]$ then is replaced by the commutation D' with the shortest depth (lines 16-22). After optimizing $D[l : l + MAX + 1]$ for every layer l, we map the updated DAG on a cluster state M (line 23).

Algorithm 1 MBQC mapping

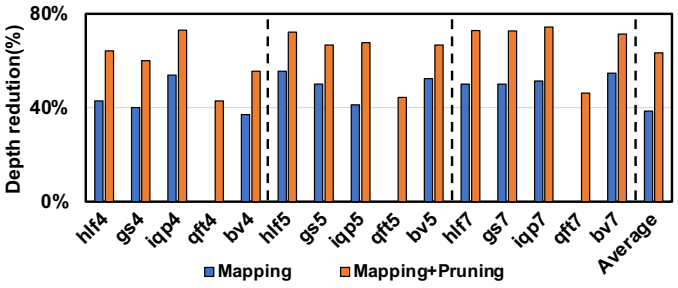
Input: A quantum circuit C, qubits number q, maximum search depth MAX
Output: mapping M on a cluster state
Initialisation : DAG D of C
1: **for** each layer l in D **do**
2: **if** ($Commutable(D[l : l + MAX])$) **then**
3: $Temp\_depth = []$
4: $Temp\_DAG = []$
5: **for** Each commutation $Commu$ in $D[l : l + MAX]$ **do**
6: $D' = copy(D[l : l + MAX + 1])$
7: **if** ($Commu == CNOT$) **then**
8: $update\_special\_CNOT(D')$
9: **end if**
10: **if** (gate g in D' requires SWAPs) **then**
11: $Minimize\_next\_SWAP(D)$
12: **end if**
13: $Combine\_single\_gate(D')$
14: $Update\_DAG(D')$
15: $Reorder\_independent(D')$
16: $Temp\_depth.append(cal\_depth(D'))$
17: $Temp\_DAG.appned(D')$
18: **end for**
19: $best\_commute = Temp\_DAG.index(min(Temp\_depth))$
20: $D[l : l + MAX + 1] = Temp\_DAG[best\_commute]$)
21: **end if**
22: **end for**
23: $M = place(D)$
24: **return** M

C. Cluster State pruning

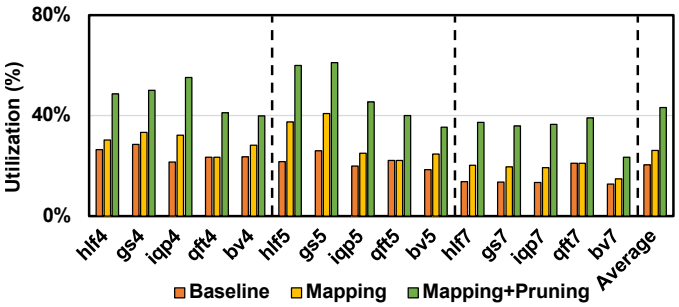
After mapping the gates to the cluster state, we design a Cluster State pruning strategy that leverages opportunity IV to significantly reduce the redundancy without affecting the circuit functionality.

The approach is as follows: For each column, we look for the consecutive pair of x and z measurements in every row. If a valid pair is found in a row, record the column location of the pair. If not every row has found a valid pair, search all rows in the next column and update the column location if a new valid pair is found. Once every row has identified a valid pair, we remove the pair at the latest pair location so the redundant measurements are removed while retaining the relevant functional structure. After removing the pairs for each row, we restart the search at the current column location. The search is done when the algorithm reaches the last column.

For example, the search starts from column 0 in Fig. 4(l), and redundancies are found in every row except for the middle qubit row. The location keeps increasing until the redundancy in the middle row is found, which is in column 4. After removing the latest found pair in each row marked with red boxes, the SWAP gate shifts left while maintaining its structure. The search then restarts at column 4 and will find another redundancy marked in another orange box. In this case, 20 redundant photons are removed.



(a) Depth Reduction



(b) Utilization

Fig. 5. Experiment results show the depth reduction and utilization.

IV. Evaluation

A. Benchmarks and Metrics

The benchmarks we used to evaluate our proposed framework are: Quantum Fourier transform (`qft`) [10], Instantaneous quantum polynomial-time (`iqp`) [4], Hidden linear function (`hlf`) [3], Graph state (`gs`) [11], and Bernstein–Vazirani (`bv`) [2]. The `hlf`, `gs`, and `iqp` benchmarks include many CP gates, which test the ability of our framework to order CP gates. The `bv` benchmarks consist of many CNOT gates, which we can leverage to examine the ability to discover the special CNOT pairs. The `qft` is a special benchmark for which opportunities I to III cannot apply, but opportunity IV does. Each benchmark is evaluated with 4, 5, and 7 qubits.

Two metrics we introduced above are used to evaluate the cluster state: cluster depth and utilization. The hyper-parameter MAX is set to 2 in our evaluation. The reason why we use this parameter will be analyzed in section IV-C.

B. Experiment Results

Fig. 5 shows the cluster depth reduction and the photon utilization of our approach compared against the baseline where the default gate order of benchmarks are sequentially mapped to cluster state as used in [16]. Here, mapping means only implementing the mapping algorithm. Mapping+pruning is the complete implementation of our framework.

The results show that applying the first stage optimization results in an average of 38.6% (up to 55.6% in `hlf5`) cluster depth reduction. When excluding `qft`, which does not have mapping opportunities, the average depth reduction is 48.2%. After applying both optimizations, the depth reduction is an average of 63.4% (up to 74.3% in `iqp7`).

The average photon utilization increases from 20.4% to 26.2% and 43.2% after applying the mapping algorithm and the complete optimization, respectively. The Cluster state pruning can vastly improve photon utilization because it is specifically designed for redundancy removal. As shown in Fig. 5(b), the utilization decreases with the

Authorized licensed use limited to: University of Pittsburgh. Downloaded on May 13,2024 at 15:28:35 UTC from IEEE Xplore. Restrictions apply.

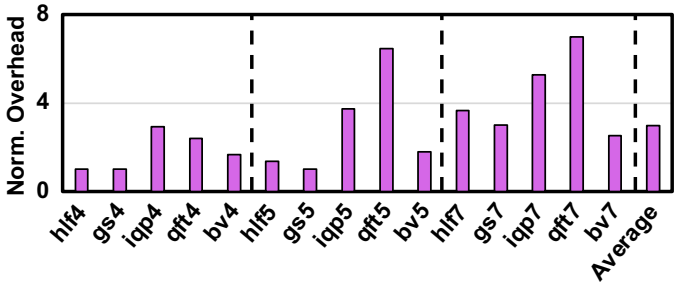


Fig. 6. The compilation time overhead of MAX 3 normalized to MAX 2.

seven-qubit benchmarks. The reason is that the row number of a cluster state is proportional to the qubit number. Some quantum gates cannot be executed in parallel, which results in many rows of redundancy that cannot be removed. The `hlf5` and `gs5` show the best utilization because they have a dense structure to enable more parallelism.

Table II summarizes the optimization opportunities enlargement discussed in section III-A for benchmarks with 5 qubits. It shows the specific optimization results that reduce the cluster depth and increase the photon utilization. CNOT pair ratio is the number of CNOT gates that have been paired to the total number of CNOT gates, while the parallelism ratio is the number of quantum gates executed in parallel to all the quantum gates. In this table, the gain is the increase in the ratio comparing our approach to the baseline. The SWAP reduction rate is the decreased number of SWAP gates compared to the baseline. Similarly, the single gate reduction rate is the decreased number of single-qubit gates compared to the baseline.

TABLE II
OPTIMIZATION OPPORTUNITIES ENLARGED

Benchmarks	CNOT pair ratio gain	SWAPs reduction rate	Single gate reduction rate	Parallelism ratio gain
hlf5	0%	66.7%	16.7%	55.6%
gs5	0%	66.7	0%	66.7%
iqp5	0%	33.3%	9.1%	33.1%
bv5	50%	66.7%	9.1%	36.2%

C. Sensitivity and Complexity

The window size MAX would bring a worst-case complexity of $O(n^2 * MAX!)$, where n is the number of qubits. Suppose gates in MAX neighboring layers in a DAG can be switched with each other. There is a total of $n^2 * MAX!$ possible commutations. Although a larger MAX may reduce the depth and increase the utilization further, it will significantly increase the compilation time. Hence, we only implement MAX 2 and 3 in our experiments.

Compared to the results shown in Fig. 5, our compiler framework with MAX 3 only improves 3 out of the 15 benchmarks compared to MAX 2. The depth of bv4, bv7, and iqp7 is reduced by 8.4% on average, while the utilization is only increased by 4.5% on average. The results for other benchmarks even do not change. Fig. 6 shows the compilation overhead of MAX 3 normalized to MAX 2. Compared to MAX 2, the compiler framework with MAX 3 increases the compilation time by 3X on average, which is not a desirable trade-off. Based on the results shown above, we set MAX as 2.

V. CONCLUSION

In this work, we have proposed our novel compiler framework for efficiently mapping quantum circuits into MBPQC (Measurement-Based Photonic Quantum Computation). We analyze the detailed opportunities available within this new cluster state-based approach and describe how our compiler takes advantage of these opportunities to compile a more efficient MBPQC. Specifically, our compiler achieves an average of 63.4% cluster depth reduction and 22.8% increase in photon utilization over our baseline, which is the current state-of-the-art. We demonstrate these results on five different benchmarks that vary across the number of qubits and functionality, showing our solution's universality.

ACKNOWLEDGEMENT

The authors would like to thank the anonymous reviewers for their constructive feedback and suggestions. This work is supported in part by NSF grants #2011146, #2154973, #2204985, #1910413, and #1725657.

REFERENCES

[1] F. Arute, K. Arya, R. Babbush, D. Bacon, J. C. Bardin, R. Barends, R. Biswas, S. Boixo, F. G. Brandao, D. A. Buell *et al.*, "Quantum supremacy using a programmable superconducting processor," *Nature*, vol. 574, no. 7779, pp. 505–510, 2019.

[2] E. Bernstein and U. Vazirani, "Quantum complexity theory," in *Proceedings of the twenty-fifth annual ACM symposium on Theory of computing*, 1993, pp. 11–20.

[3] S. Bravyi, D. Gosset, and R. König, "Quantum advantage with shallow circuits," *Science*, vol. 362, no. 6412, pp. 308–311, 2018.

[4] M. J. Bremner, R. Jozsa, and D. J. Shepherd, "Classical simulation of commuting quantum computations implies collapse of the polynomial hierarchy," *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 467, no. 2126, pp. 459–472, 2011.

[5] H. J. Briegel, D. E. Browne, W. Dür, R. Raussendorf, and M. Van den Nest, "Measurement-based quantum computation," *Nature Physics*, vol. 5, no. 1, pp. 19–26, 2009.

[6] A. Broadbent, J. Fitzsimons, and E. Kashefi, "Universal blind quantum computation," in *2009 50th Annual IEEE Symposium on Foundations of Computer Science*. IEEE, 2009, pp. 517–526.

[7] J. I. Cirac and P. Zoller, "Quantum computations with cold trapped ions," *Physical review letters*, vol. 74, no. 20, p. 4091, 1995.

[8] L. K. Grover, "A fast quantum mechanical algorithm for database search," in *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, 1996, pp. 212–219.

[9] 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.

[10] A. JavadiAbhari, S. Patil, D. Kudrow, J. Heckey, A. Lvov, F. T. Chong, and M. Martonosi, "Scaffcc: Scalable compilation and analysis of quantum programs," *Parallel Computing*, vol. 45, pp. 2–17, 2015.

[11] D. E. Koh, "Further extensions of clifford circuits and their classical simulation complexities," *arXiv preprint arXiv:1512.07892*, 2015.

[12] J. Liu, P. Li, and H. Zhou, "Not all swaps have the same cost: A case for optimization-aware qubit routing," in *2022 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*. IEEE, 2022, pp. 709–725.

[13] M. A. Nielsen and I. Chuang, "Quantum computation and quantum information," 2002.

[14] S. Park, D. Kim, M. Kweon, J.-Y. Sim, and S. Kang, "A fast and scalable qubit-mapping method for noisy intermediate-scale quantum computers," in *Proceedings of the 59th ACM/IEEE Design Automation Conference*, 2022, pp. 13–18.

[15] R. Raussendorf and H. J. Briegel, "A one-way quantum computer," *Physical review letters*, vol. 86, no. 22, p. 5188, 2001.

[16] R. Raussendorf, D. E. Browne, and H. J. Briegel, "Measurement-based quantum computation on cluster states," *Physical review A*, vol. 68, no. 2, p. 022312, 2003.

[17] P. W. Shor, "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer," *SIAM review*, vol. 41, no. 2, pp. 303–332, 1999.

[18] S. Slussarenko and G. J. Pryde, "Photonic quantum information processing: A concise review," *Applied Physics Reviews*, vol. 6, no. 4, p. 041303, 2019.

Authorized licensed use limited to: University of Pittsburgh. Downloaded on May 13,2024 at 15:28:35 UTC from IEEE Xplore. Restrictions apply.