Automated Window-Based Partitioning of Quantum Circuits

Eesa Nikahd[1], **Naser Mohammadzadeh**[2], **Mehdi Sedighi**[1] **and Morteza Saheb Zamani**[1]

[1]Quantum Design Automation Lab, Amirkabir University of Technology, Tehran, Iran
[2]Department of Computer Engineering, Shahed University, Tehran, Iran

E-mail: nikahd@aut.ac.ir, mohammadzadeh@shahed.ac.ir ,msedighi@aut.ac.ir, szamani@aut.ac.ir

June 2020

Abstract. Developing a scalable quantum computer as a single processing unit is challenging due to technology limitations. A solution to deal with this challenge is distributed quantum computing where several distant quantum processing units are used to perform the computation. The main design issue of this approach is costly communication between the processing units. Focused on this issue, in this paper, an efficient partitioning approach is proposed which combines both gate and qubit teleportation concepts in an efficient manner to minimize the communication. Experimental results show the proposed approach on average reduces the communication cost by about 29.5% in comparison with the best approaches in the literature.

Keywords: Quantum Circuit, Distributed Quantum Computing, Window-based Partitioning

1. Introduction

While feature size in VLSI technology enters 7 nm and beyond, quantum effects should be handled [1]. However, managing quantum effects in a controlled manner may also be utilized as a feature. Feynman originally suggested [2] using these effects to efficiently solve problems that are intractable on classical computers and called the device a quantum computer. The circuit model of quantum computation is similar to the circuit model consisting of a discrete set of gates found in conventional computing. Sequences of one- and two-qubit operations constitute the fundamental logic for evolving a quantum state. However, there are some unique characteristics for quantum computing such as superposition, entanglement, and the inability to copy arbitrary quantum states [3].

Although many challenges hinder the realization of a practical quantum system, we believe that the design space for a future quantum computer should be explored now that it helps to organize the plethora of proposed quantum technologies, fault tolerance

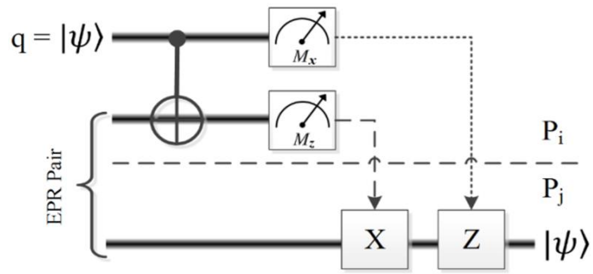


Figure 1. Teleportation circuit to transmit qubit q from partition P_i to P_j using a single EPR pair shared between the partitions P_i and P_j

methods, and other realization choices. However, when both hardware and architecture parameters are considered, the design space grows significantly. Therefore, to manage the design space complexity, a CAD flow is needed to streamline the design process and to enable us to design large quantum circuits.

A homogeneous organization may be acceptable for a small-scale quantum computer but to build a practical full-scale quantum system, distributed computation with the necessary communications is required [4]. The distributed quantum architectures [4, 5, 6, 7, 8] are organized as quantum processing units (QPU) connected by some interfaces such as photonic networks. In such architectures, the communications have considerably more latency and are more error-prone than local operations [4, 5]. For example, in the MUSIQC hardware proposed in [5], the communication latency between QPUs is in the order of milliseconds while other operations are in the order of microseconds. Thus, minimizing the communications between QPUs has a significant effect on the final latency. The communications between QPUs can dramatically decrease if qubits of a circuit are effectively partitioned into QPUs. Focused on this issue, in this paper, we propose a partitioning approach based on a windowing strategy to distribute qubits among QPUs in a manner that the communication cost is minimized.

The remainder of this paper is organized as follows: Section 2 contains some basic concepts in the field. An overview of the prior work is presented in Section 3. Section 4 discusses the proposed approach in detail. Experimental results are discussed in the Section 5, and finally Section 6 concludes the paper.

2. Background

In this section, some terminologies and concepts are explained that would help to give a better understanding of the proposed approach.

Using teleportation to transmit data qubits from a source to a destination is known as data teleportation or teledata [9]. This procedure can be summarized in Figure 1. Performing a multi-qubit gate using teleportation, without placing the target qubits next to each other, is known as gate teleportation or telegate [10]. Figure 2 shows the circuits to perform CNOT and CZ gates remotely, based on the telegate mechanism.

The teledata and telegate concepts lead to two partitioning approaches namely

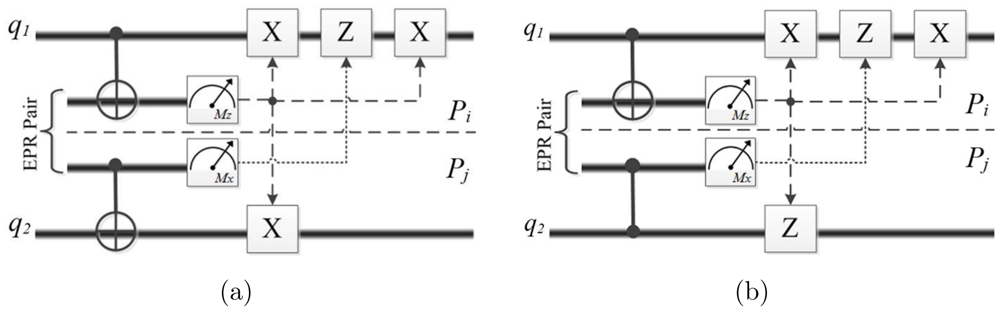


Figure 2. Telegate-based application of a) CNOT and b) CZ gates on qubits q_1 and q_2 using an EPR pair shared between partitions P_i and P_j

gate partitioning and *qubit partitioning*, respectively. In the gate partitioning, a quantum circuit is partitioned according to its gates. In this approach, it is decided to which partition (QPU) a gate must be assigned. If the qubits of that gate are not in the chosen partition, they are transmitted into that using teledata. On the other hand, the qubit partitioning approach partitions the qubits of a circuit and decides where each qubit should be placed. For applying a multi-qubit gate, if the involved qubits are in different partitions, the gate will be applied remotely via telegate; otherwise, the gate can be performed locally.

3. Related Work

Quantum circuit design flow like its classical counterpart can be partitioned into two main processes: synthesis and physical design. The physical design process maps the gate-level netlist generated from the synthesis process onto a physical layout. Several studies have been done on automation of different steps of the physical design process. Some researchers [6, 7, 11, 12, 13, 14, 15, 16] worked on the entire physical design flow and proposed techniques for each of its step while others [17, 18, 19, 20] proposed some techniques for scheduling of a quantum circuit on a layout. Mohammadzadeh et al. [21, 22, 23] introduced the physical synthesis concept for quantum circuits and proposed some practical physical synthesis techniques [21, 23, 24, 25]. Since the main focus of this paper is on the partitioning step, in the rest of this section, the partitioning techniques are reviewed in more detail.

Squash 2 [26] partitions the quantum circuits based on the gate partitioning approach by utilizing METIS [27] as the partitioning tool. Moghadam et al. [28] apply a min-cut placement-aware partitioning approach [29] to divide a quantum dataflow graph of a circuit into smaller manageable parts. Wang et al. [30] modified a graph partitioning algorithm presented in [31] to find the minimum cut of the qubit interaction graph. Ahsan et al. [7, 32] used an efficient graph-theoretic algorithm [33] to assign qubits to QPUs. They first generate the adjacency matrix P of an N-qubit circuit where $P[i][j]$ is the total number of interactions between qubits q_i and q_j. Then P is converted into its corresponding Laplacian matrix as below:

$$L[i][j] = \begin{cases} \sum_{k=1}^{N} P[i][k] & i = j \\ -P[i][j] & o.w. \end{cases}$$

Eigenvalues of L are computed and the eigenvector V_2 corresponding to the second smallest eigenvalue is selected. Sorting V_2 can determine the best order of qubits in a line in such a way that the weighted sum of the distances between the qubits is minimized. As the last step, this line of qubits is broken into some partitions and assigned to the QPUs.

Mohammadzadeh and Sargaran [6] proposed SAQIP architecture and used the multilevel k-way hypergraph partitioning algorithm introduced in [34] to partition the qubits into QPUs. In [35], the authors call METIS [27] iteratively to separate the qubits and place them on a 2D nearest-neighbor architecture. Zomorodi-Moghadam et al. [36] proposed a gate partitioning procedure to minimize the number of teleportation operations. In that work, an additional exhaustive search is applied to decide how each two-qubit quantum gate should be implemented. This increases the runtime exponentially in the number of partitions and gates, making it futile in practice. However, in a recent work, they reduced the complexity of their method by proposing a genetic algorithm to solve the partitioning problem more efficiently [37].

There are some approaches proposed for quantum physical design automation on single-processor but topologically-constrained architectures. Childs et al. [38] used the token swapping framework [39] and a 4-approximation algorithm [40] to insert a minimal sequence of SWAP gates into the circuit and transform an input quantum circuit to a hardware-compliant one. Chakrabarti et al. [41] proposed a balanced graph partitioning technique to find global ordering of qubit lines to achieve the Linear-Nearest-Neighbor architecture with minimum number of SWAP gates by using pmetis [42], an existing multilevel graph partitioning tool. Minimum linear arrangement problem [43] employed in [44] tries to insert minimum number of SWAP gates in different parts of an interaction graph. A novel reverse traversal technique was proposed in [45] to choose the initial mapping with the consideration of the whole circuit. It takes the following gates and previous mappings into account to reduce the overhead of 2-qubit gates and movements. The authors of [46] proposed an efficient heuristic method for logical to physical qubit mapping for linear devices. This has been realized by transforming the mapping problem into an undirected graphical representation and then has implemented spectral graph theory-based approach for placing logical qubits. All these approaches focus on mapping a given circuit on a topologically-constrained architectures while our architecture is a distributed one with less constraints.

4. Our Proposed Approach: Window-based Quantum Circuit Partitioning (WQCP)

The main drawback of *qubit partitioning* approaches is that they convert a circuit into an untimed qubit interaction graph and try to partition it. Although this assignment

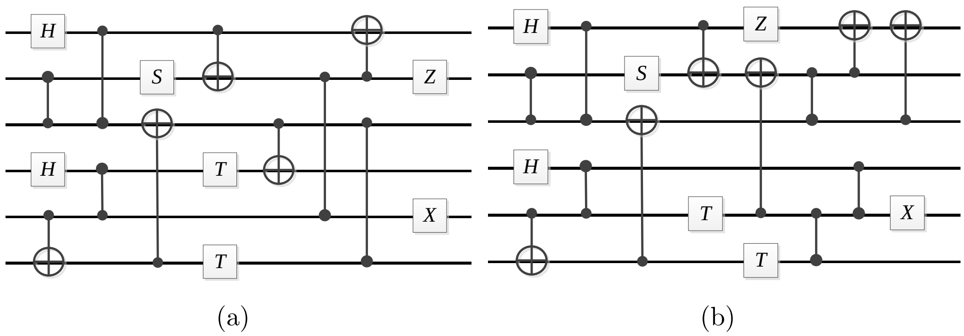


Figure 3. Gate partitioning (qubit partitioning) leads to less communication cost than qubit partitioning (gate partitioning) for the circuit a (b).

is done in a manner that more interacting qubits are attempted to be assigned to the same partition, not using timing information in these approaches makes them inefficient. In other words, since qubits may interact in different parts of a circuit, the partitioning methods that ignore timing information do not generate good results. Therefore, a better solution is to partition the circuit based on the information in local connectivity patterns and to change partitions of qubits if the connectivity pattern of the qubits is changed while the circuit proceeds. On the other hand, the main drawback of *gate partitioning* methods is that they force the qubits of multi-qubit gates to be transported to the same partition to apply the gate. However, allowing remote application of multi-qubit gates can mitigate unnecessary forward and backward transfers.

For example, in Figure 3, the goal is partitioning of the circuits into two parts, each consisting of three qubits. The optimal communication costs of the circuit in Figure 3-a using gate and qubit partitioning approaches are 2 and 4, respectively. On the other hand, those achieved for the circuit in Figure 3-b using gate and qubit partitioning approaches are 4 and 2, respectively. Therefore, gate partitioning generates a better result than qubit partitioning for the circuit of Figure 3-a while for the circuit of Figure 3-b, qubit partitioning outperforms gate partitioning. Therefore, the superiority of a method over the other depends on the interaction pattern of the qubits of the circuit. Considering this observation, it seems that combining two partitioning approaches can improve the communication cost by mitigating the drawbacks of existing approaches.

Focusing on this issue, we propose a hybrid partitioning approach, called WQCP , which combines both *telegate* and *teledata* ideas in an efficient manner to minimize the communication cost. The pseudo-code of the proposed algorithm is given in Algorithm 1. In the first step, single-qubit gates are removed from the circuit because single-qubit gates can be applied without any communication regardless of the partition the target qubit is assigned to. Then, the resulting circuit is levelized and a weighted window with the length of L_W is moved along the circuit from the first level to the last one, level by level. Let L_C, G_L, and C_L be the number of levels of the circuit, the set of gates in level L and the sub-circuit contained in the window of length L_W beginning at

Algorithm 1 WQCP

Input: A quantum circuit (C_{in}), Window length (L_W)
Output: The partitioned circuit, Total communication cost (Number of teledatas and telegates)

$C = rmvSingleQubitGates(Cin)$; //Remove single-qubit gates from the circuit C_{in}
$L_C = levelize(C)$; //Levelize the circuit C and return the total number of levels of C
$nTD = 0$; //Initialize the number of teledatas with zero
$nTG = 0$; //Initialize the number of telegates with zero
for each $L \in Levels = \{1, 2, ..., L_C\}$ **do**
    $C_L = getWindow(C, L, L_w)$; //Get the sub-circuit surrounded by the window with length L_W started from level L
    $P_L = subPartitioning(C_L)$; //Partition C_L based on qubit partitioning approach
    $nTD+ = countTG(P_L, P_{L-1})$; //Return the number of teledatas by comparing P_L with P_{L-1}
    $G_L = getGatesAtLevel(L)$; //Return the gates of level L
    $nTG+ = countTD(G_L, P_L)$; //Return the number of gates of G_L which must be applied remotely by telegate
end for

L, respectively. For each level L, $1 \leq L \leq L_C$, C_L is partitioned based on the qubit partitioning approach. This operation is denoted by $subPartitioning(C_L)$. By this, the partition of each qubit is determined for applying the gates of G_L and is denoted by P_L. For each gate $g \in G_L$, if its qubits are placed in different partitions, the gate will be applied remotely by telegate. Otherwise, g is applied locally. For two successive levels, $L-1$ and L, if $subPartitioning(C_L)$ changes the partition of one qubit with respect to its previous partition which is determined by $subPartitioning(C_{L-1})$, that qubit is transported to the new partition using teledata. In the rest of this section, the subPartitioning algorithm is explained followed by an example.

4.1. subPartitioning(C_L) algorithm

The subPartitioning function implements a min-cut partitioning algorithm [27, 47, 48] which takes a sub-circuit C_L as input and partitions its qubits. To do so, the sub-circuit C_L is modeled using a weighted graph whose vertices are qubits and its edges between two vertices are weighted according to the number of gates applied to the corresponding qubits. The weights of the edges are calculated as follows. Interactions between qubits in different levels of C_L have different importance in our approach. This is because $subPartitioning(C_L)$ determines the partition of each qubit only for applying the gates of G_L, and $subPartitioning(C_{L+1})$ may change the location of qubits for the gates of the next level. Therefore, a weighted window is used to reflect this difference in the importance of interactions between the qubits in different levels. To this end, the weight of a window is defined as $W = \{w_k\}$, where $1 \leq k \leq L_W$ and w_k represents the importance of the interactions between the qubits in the k^{th} level of the sub-circuit C_L. The weight of each edge between two vertices q_i and q_j, denoted by $E(q_i, q_j)$, is defined as the weighted sum of interactions between qubits q_i and q_j in C_L. This weight can be

Table 1. Input parameters and variables of our ILP model

Type	Symbol	Description
NUCC	N	Number of qubits in the sub-circuit C_L
	M	Total number of partitions
	$capacity$	Maximum number of qubits which can be assigned to each partition simultaneously
	w_p	Determines how much qubits tend to stay in their previous partitions
	$adjMat[N][N]$	An $N \times N$ matrix where $adjMat[i][j]$ is equal to $E(q_i, q_j)$ according to Eq. 1
	$prevParts[M][N]$	An $M \times N$ binary matrix where $prevParts[p][i]$ is equal to 1 if and only if the previous partition of qubit q_i is partition p (Previous partition of a qubit is determined by $subPartitioning(C_{L-1})$).

formulated as:

$$E(q_i, q_j) = \sum_{k=1}^{L_W} g_{ij}^k \times w_k \tag{1}$$

where g_{ij}^k is zero if there is not any gate applied to qubits q_i and q_j in the k^{th} level of the sub-circuit C_L and otherwise it is equal to the number of needed EPR pairs to apply the gate by telegate.

In addition to the interactions of the qubits in C_L, $subPartitioning(C_L)$ should consider the previous partition of each qubit, which is determined by $subPartitioning(C_{L-1})$. For doing so, a dummy vertex p_i is added to the graph corresponding to each partition i and each qubit is connected to its previous partition vertex with weight w_p, where w_p represents how much a qubit tends to stay in its previous partition. We formulate $subPartitioning(C_L)$ as an ILP problem. The input parameters and variables of the ILP model are given in Table 1.

Our model minimizes the following cost function:

$$\sum_{i=1}^{N}\sum_{j=i}^{N} cut[i][j] \times adjMat[i][j] + \sum_{n=1}^{N} w_p \times migrate[n]$$

It consists of two terms. The first term is the weighted sum of cuts and the second term is the cost incurred by migrating qubits from their previous partitions.

The constraints of the ILP model are listed below:

- Total number of qubits assigned to each partition must not exceed its capacity:

$$\sum_{i=1}^{N} outParts[p][i] \leq capacity \qquad \forall p : 1 \leq p \leq M$$

- Each qubit must be assigned to only one partition:

$$\sum_{p=1}^{N} outParts[p][i] = 1 \qquad \forall i : 1 \leq i \leq N$$

- If two qubits q_i and q_j are assigned to different partitions, cut[i][j] must be set to 1:

$$outParts[p][i] - outParts[p][j] \leq cut[i][j]$$
$$\forall i, j : 1 \leq i, j \leq N \; and \; \forall p : 1 \leq p \leq M$$

  Suppose that qubits q_i and q_j are assigned to different partitions p_i and p_j, respectively. For $p = p_i$ the left hand side of the above inequality is equal to 1 which forces $cut[i][j]$ to be set to one. On the other hand, when q_i and q_j are in the same partitions, the left hand side of the inequality is zero for all partitions. In this case, the cost function forces $cut[i][j]$ to be zero to minimize the cost.

- If the partitioning algorithm changes the partition of a qubit q_i, $migrate[i]$ must be set to 1:

$$outParts[p][i] - prevParts[p][i] \leq migrate[i]$$
$$\forall i : 1 \leq i \leq N \; and \; \forall p : 1 \leq p \leq M$$

  Suppose that the partition of qubit q_i has been changed from p_{prev} and p_i. Therefore, for $p = p_i$ the left hand side of the above inequality is equal to 1 which forces $migrate[i]$ to be set to 1. On the other hand, when q_i stays in its previous partition, the left hand side of the inequality is zero for all partitions. In this case, the cost function forces $migrate[i]$ to be zero to minimize the cost.

4.2. An example

In this section, the proposed approach is explained by an example. Figure 4 shows a quantum circuit consisting of 6 qubits and 25 gates. Our algorithm partitions the circuit into two parts each containing three qubits. Let the window width be $L_W = 3$, window weight be $W = \{w_1, w_2, w_3\} = \{3, 2, 1\}$, and $w_p = 2$. The algorithm removes single-qubit gates from the circuit and then levelizes it in the first step, as shown in Figure 5. During the next step, the window is laid on the circuit starting from the first level. Figure 6 and Figure 7 show all steps of the algorithm. In the second column, the sub-circuit C_L is depicted. The corresponding interaction graph of C_L and P_L, i.e., the output of $subPartitioning(C_L)$, are shown in the third column. The last column contains the gates of level L (G_L), the qubits that should be teleported using teledata and the gates that should be applied remotely by telegate, respectively.

For the graph of C_1, the gate g_1 in the first level generates an edge between vertices q_2 and q_3 with the weight of 3 (w_1). Similarly, there are edges $E(q_4, q_6) = 2$ and $E(q_3, q_6) = 1$ corresponding to gates g_4 and g_6 at levels 2 and 3, respectively. It should be noted that there is no edge between qubit and partition vertices in the graph of C_1 because no qubit has been assigned to any partition yet. $subPartitioning(C_1)$ partitions the qubits of C_1 into two parts $p_1 = \{q_1, q_2, q_3\}$ and $p_2 = \{q_4, q_5, q_6\}$, denoted by $P_1 = \{p_1, p_1, p_1, p_2, p_2, p_2\}$ in Figure 6 and Figure 7. P_1 is the initial partitioning of qubits and thus no teledata is required. In the next step, it is determined how the gates of G_1, i.e., g_1 and g_2 should be applied. Both g_1 and g_2 can be applied locally because their qubits are assigned to the same partition. For the next levels, each qubit

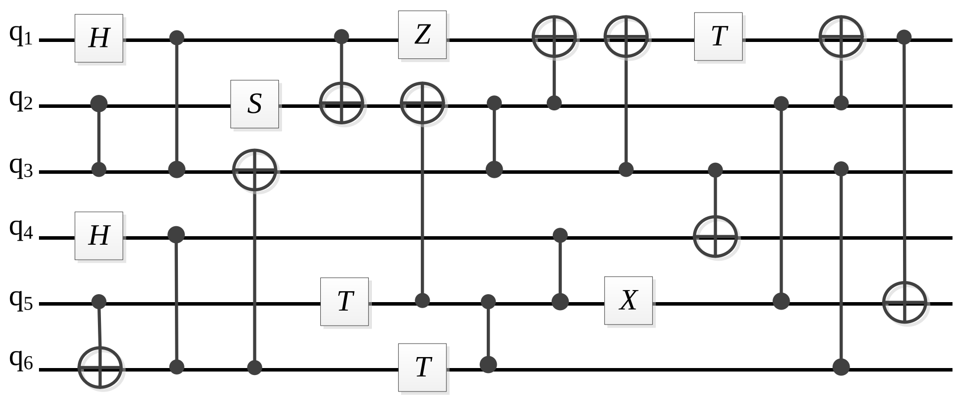


Figure 4. An example circuit to partition into two parts

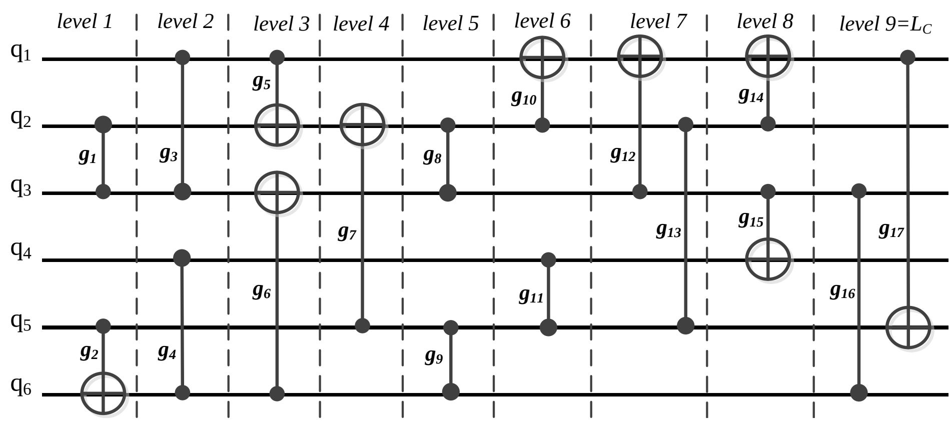


Figure 5. The levelized circuit after removing single-qubit gates

vertex is connected to the vertex corresponding to its previous partition with weight 2 (w_p). In step 3, since q_3 and q_6 are assigned to different partitions, g_6 must be applied remotely by telegate. In step 8, $subPartitioning(C_8)$ partitions the qubits of the circuit into $p_1 = \{q_1, q_2, q_5\}$ and $p_2 = \{q_4, q_3, q_6\}$ which in comparison with the previous partitioning P_7, q_3 and q_5 are assigned to different partitions. Therefore, these qubits will be transported using two teledata operations.

Our hybrid approach needs 5 communication operations including 2 teledata operations and 3 telegate ones while the best solution achieved by qubit partitioning approach or gate partitioning approach requires 6 communication operations. This example shows the superiority of our hybrid approach over both qubit partitioning and gate partitioning approaches.

5. Experimental Results

Our approach (WQCP) was implemented in C++ and CPLEX [49] was used as the ILP solver. It was run on a Core i7 CPU operating at 2.4 GHz with 8 GB of memory.

Single qubit gates and a two-qubit Clifford gate such as CNOT or CZ make a universal gate set for quantum computation. Therefore, the set CNOT, CZ and single

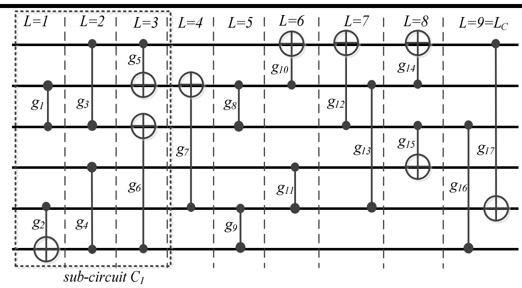
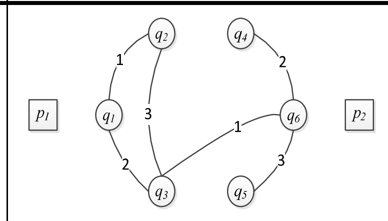
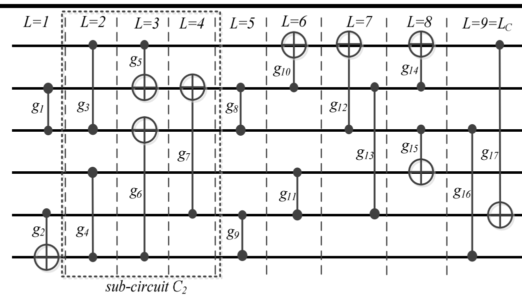
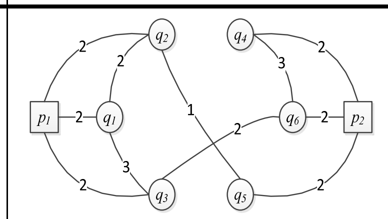
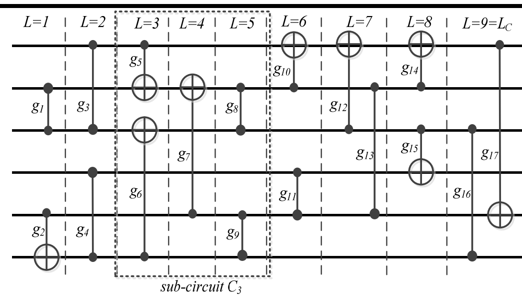
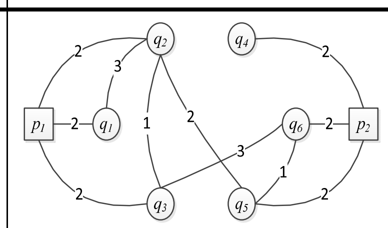
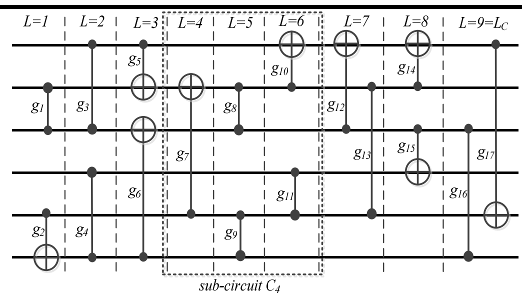
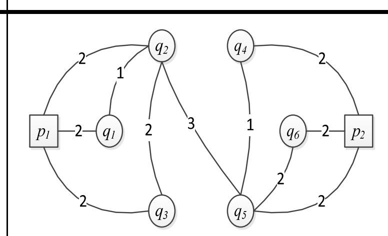
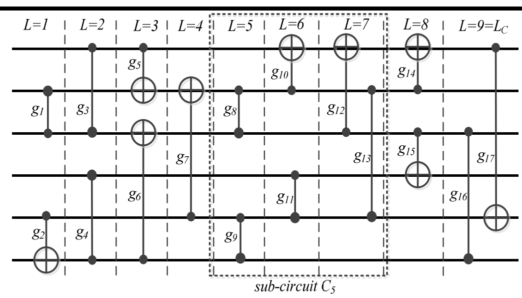
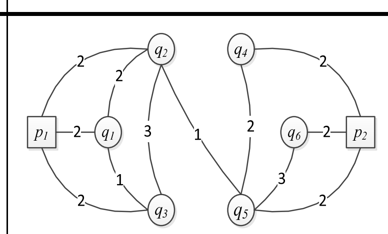
L	sub-circuit C_L	graph of C_L / output of *subPartitioning*(C_L)	G_L teledata / telegate
1			g_1 g_2
			-
		P_1={p_1, p_1, p_1, p_2, p_2, p_2}	-
2			g_3 g_4
			-
		P_2={p_1, p_1, p_1, p_2, p_2, p_2}	-
3			g_5 g_6
			-
		P_3={p_1, p_1, p_1, p_2, p_2, p_2}	g_6
4			g_7
			-
		P_4={p_1, p_1, p_1, p_2, p_2, p_2}	g_7
5			g_8 g_9
			-
		P_5={p_1, p_1, p_1, p_2, p_2, p_2}	-

Figure 6. Steps 1 to 5 of our approach to partition the example circuit of 4

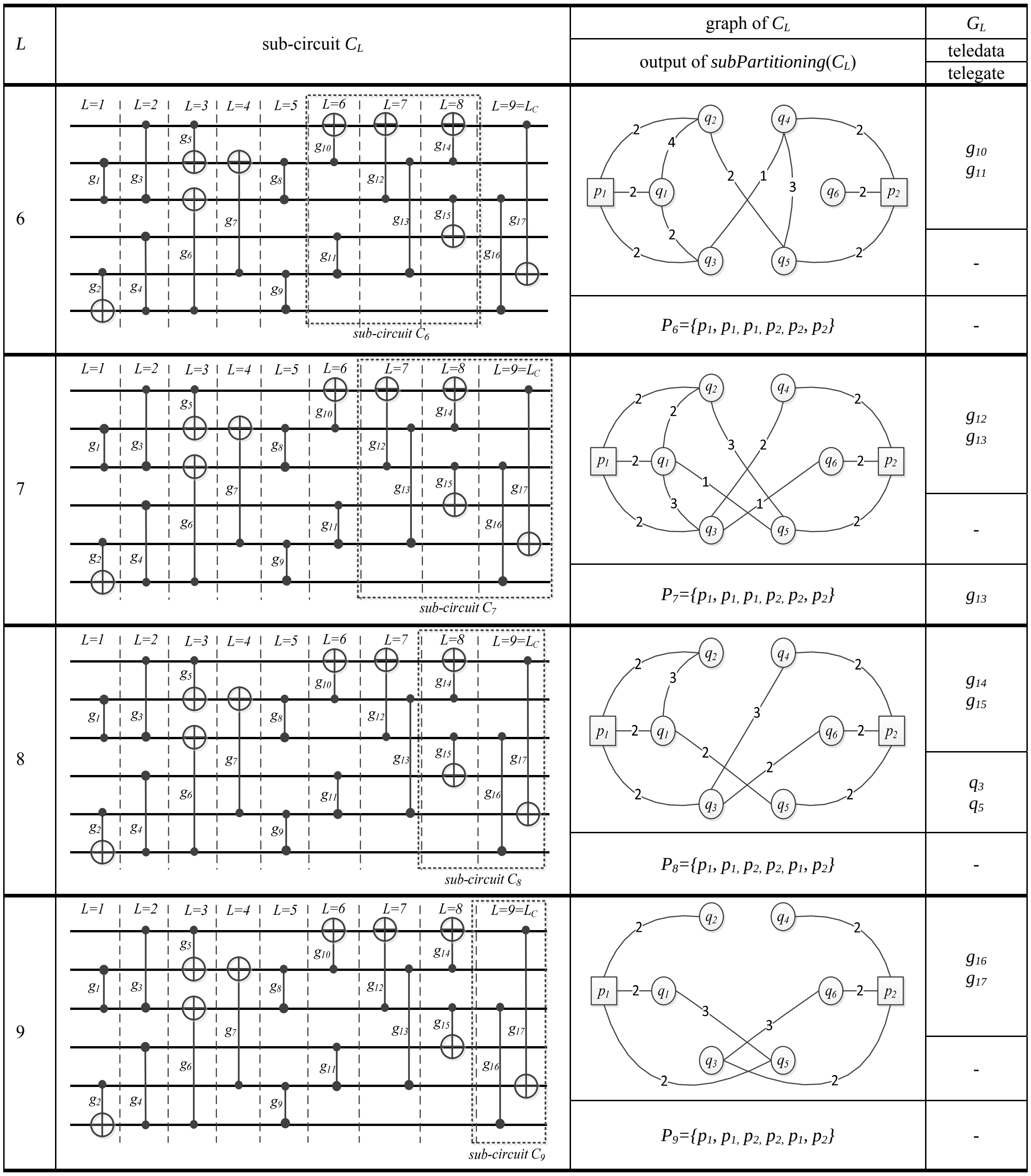


Figure 7. Steps 6 to 9 of our approach to partition the example circuit of Figure 4

qubit gates was chosen as the gate library. To evaluate the performance of WQCP, it was applied to some benchmark circuits from [50] (the first nine circuits in the tables), Revlib [51] (the circuits from 10 to 15), some quantum error-correction encoding circuits [52] (the circuits from 16 to 25), and n-qubit quantum Fourier transform circuits (QFT) [53] where $n \in \{16, 32, 64, 128, 256\}$.

These circuits may include some gates out of the gate library that are synthesized into the gates of the library based on the method proposed in [54].

The window length L_W may potentially have a high effect on the result. Table 2 compares the communication counts obtained by WQCP for different window lengths where window weight $W = \{L_W, L_W-1, \ldots, 1\}$ and $w_p = L_W-1$. The number of qubits, the multi-qubit depth of each benchmark and the number of partitions are shown in the second column. The third column contains the type of teleportation, which can be teledata or telegate. The best obtained results are marked in bold. It is worth noting that the window weight W and w_p were chosen experimentally and different results may be achieved by changing them.

Table 2: Experimental results (number of teleportations) obtained by WQCP for the benchmark circuits and different window lengths L_W

#	**Benchmark**	**# qubits** / **Depth** / **# parts**	**TP type**	L_W						
				5	**6**	**7**	**8**	**9**	**12**	**15**
		6	Telegate	11	16	31	31	30	30	30
1	**2of5-D1**	98	Teledata	38	30	10	10	10	10	10
		2	Total	49	46	41	41	**40**	**40**	**40**
		6	Telegate	9	9	9	9	9	8	9
2	**2-4dec**	19	Teledata	6	2	2	2	4	4	4
		3	Total	15	**11**	**11**	**11**	13	12	13
		10	Telegate	5	4	5	4	6	9	8
3	**6sym**	42	Teledata	18	16	14	12	12	10	8
		2	Total	23	20	19	**16**	18	19	**16**
		12	Telegate	5	17	20	24	29	20	24
4	**9sym**	71	Teledata	31	20	20	16	16	16	17
		3	Total	**36**	37	40	40	45	**36**	41
		15	Telegate	43	40	56	51	59	56	57
5	**Ham15-D3**	177	Teledata	70	63	49	54	43	48	44
		4	Total	113	103	105	105	102	104	**101**
		20	Telegate	246	255	291	589	83	702	740
6	**Cycle17-3**	8561	Teledata	2261	2217	2118	1722	1657	1380	1288
		3	Total	2507	2472	2409	2311	2340	2082	**2028**
		24	Telegate	11	21	19	26	22	46	41
7	**8bitadder**	106	Teledata	87	73	65	58	69	42	48
		6	Total	98	94	**84**	**84**	91	88	89
		56	Telegate	325	386	317	287	325	293	400
8	**Hwb50**	4994	Teledata	1319	1380	1162	1232	1057	1008	936
		5	Total	1644	1766	1479	1519	1382	**1301**	1336
		107	Telegate	1075	676	982	1122	597	753	673
9	**Hwb100**	15923	Teledata	3446	3151	3117	2805	2295	2029	1840
		7	Total	4521	3827	4099	3927	2892	2782	**2513**
		5	Telegate	2	1	1	1	1	1	1
10	**rd32_272**	5	Teledata	4	6	6	6	6	6	6
		2	Total	**6**	7	7	7	7	7	7
		7	Telegate	26	26	26	26	26	26	27
11	**ham7_106**	38	Teledata	4	4	4	4	4	7	6
		4	Total	**30**	**30**	**30**	**30**	**30**	33	33
		8	Telegate	6	3	4	2	3	6	9
12	**rd53_139**	8	Teledata	8	10	10	12	10	6	4
		2	Total	14	13	14	14	13	**12**	13
		13	Telegate	5	5	2	3	3	4	2
13	**rd53_311**	19	Teledata	19	22	23	21	20	18	20
		3	Total	24	27	25	24	23	**22**	**22**
		17	Telegate	1	1	3	2	4	3	4
14	**parity_247**	16	Teledata	3	4	3	4	3	4	3
		3	Total	**4**	5	6	6	7	7	7
		49	Telegate	2	2	2	2	2	3	3
15	**adder16_174**	19	Teledata	9	7	10	11	11	8	7
		3	Total	11	**9**	12	13	13	11	10
		10	Telegate	4	5	5	5	6	7	6
16	**[[10,3,3]]**	25	Teledata	10	8	8	8	8	8	8
		2	Total	14	**13**	**13**	**13**	14	15	14
		16	Telegate	17	16	18	27	28	30	34
17	**[[16,3,5]]**	43	Teledata	34	38	33	21	21	19	17
		4	Total	51	54	51	**48**	49	49	51
		21	Telegate	5	9	14	20	26	32	23
18	**[[21,1,7]]**	58	Teledata	51	51	46	42	32	28	37

Table 2: Experimental results (number of teleportations) obtained by WQCP for the benchmark circuits and different window lengths L_W (continue)

#	**Benchmark**	**# qubits / Depth / # parts**	**TP type**	L_W						
				5	**6**	**7**	**8**	**9**	**12**	**15**
		3	Total	**56**	60	60	62	58	60	60
19	[[**24,3,7**]]	24	Telegate	13	17	34	35	38	37	50
		84	Teledata	88	88	69	71	66	66	57
		4	Total	**101**	105	103	106	104	103	107
20	[[**25,1,9**]]	25	Telegate	19	22	21	21	22	26	43
		83	Teledata	79	70	68	68	66	67	53
		5	Total	98	92	89	89	**88**	93	96
21	[[**27,1,9**]]	27	Telegate	17	12	31	44	35	46	45
		110	Teledata	89	86	77	67	76	68	69
		4	Total	106	**98**	108	111	111	114	114
22	[[**31,11,6**]]	31	Telegate	18	28	42	39	45	60	61
		149	Teledata	127	129	110	99	105	101	94
		4	Total	145	157	152	**138**	150	161	155
23	[[**33,1,9**]]	33	Telegate	9	19	32	36	52	56	55
		153	Teledata	132	119	122	124	102	103	104
		5	Total	141	**138**	154	160	154	159	159
24	[[**35,1,10**]]	35	Telegate	10	22	21	31	43	48	62
		126	Teledata	114	111	99	100	100	102	105
		4	Total	124	133	**120**	131	143	150	167
25	[[**40,3,10**]]	40	Telegate	23	34	40	50	53	59	67
		172	Teledata	166	150	153	136	149	138	122
		4	Total	189	**184**	193	186	202	197	189
26	**QFT16**	16	Telegate	0	0	1	14	15	18	18
		56	Teledata	23	23	23	25	26	22	28
		3	Total	**23**	**23**	24	39	41	40	46
27	**QFT32**	32	Telegate	0	0	0	0	0	0	0
		120	Teledata	48	48	48	48	48	48	48
		4	Total	**48**	**48**	**48**	**48**	**48**	**48**	**48**
28	**QFT64**	64	Telegate	0	0	0	0	0	0	0
		248	Teledata	106	106	106	106	106	106	106
		6	Total	**106**	**106**	**106**	**106**	**106**	**106**	**106**
29	**QFT128**	128	Telegate	0	0	0	0	0	0	0
		504	Teledata	224	224	224	224	224	224	224
		8	Total	**224**	**224**	**224**	**224**	**224**	**224**	**224**
30	**QFT256**	256	Telegate	0	0	NA	NA	NA	NA	NA
		1016	Teledata	468	468	NA	NA	NA	NA	NA
		12	Total	**468**	**468**	NA	NA	NA	NA	NA

Table 3 compares the best results obtained by WQCP with the qubit and gate partitioning approaches. Two different algorithms based on qubit partitioning approach are considered. In the first one, that is denoted by QPILP, the qubit partitioning is modeled using ILP and solved by CPLEX solver. Although this method produces the optimal results of qubit partitioning, it is not scalable. The method proposed by Ahsan et al. [7, 32], denoted by QPGTA, is considered as the second algorithm for comparison. To implement gate partitioning approach (GP), WQCP was used where the window weight w_1 was set to a very large number compared to the other weights, i.e. $W \setminus \{w_1\}$ and w_p. By this, WQCP is forced to apply each multi-qubit locally without any telegate operation. Table 3 shows that WQCP decreases the communication cost, on average, by 37.6% and 21.4% in comparison to the qubit and gate partitioning approach, respectively. For the QFT circuits, the best partitioning approach is gate partitioning because of their particular structures and WQCP produces the same results as GP for these circuits.

Table 3: The partitioning results achieved by WQCP for the benchmark circuits compared with the gate and qubit partitioning approaches

#	Benchmark	TP type	GP	QPILP	QPGTA	WQCP	Improvement (%) GP	Improvement (%) QPILP	Improvement (%) QPGTA
1	**2of5-D1**	Telegate	0	50	50	30	35	20	20
		Teledata	62	0	0	10			
		Total	62	50	50	40			
2	**2-4dec**	Telegate	0	13	18	9	59	15	38
		Teledata	27	0	0	2			
		Total	27	13	18	11			
3	**6sym**	Telegate	0	22	22	4	42	27	27
		Teledata	28	0	0	12			
		Total	28	22	22	16			
4	**9sym**	Telegate	0	57	72	5	16	37	50
		Teledata	43	0	0	31			
		Total	43	57	72	36			
5	**Ham15-D3**	Telegate	0	126	156	57	35	20	35
		Teledata	155	0	0	44			
		Total	155	126	156	101			
6	**Cycle17-3**	Telegate	0	3979	4323	740	15	49	53
		Teledata	2372	0	0	1288			
		Total	2372	3979	4323	2028			
7	**8bitadder**	Telegate	0	NA	147	19	18	NA	43
		Teledata	103	NA	0	65			
		Total	103	NA	147	84			
8	**Hwb50**	Telegate	0	NA	3247	293	44	NA	60
		Teledata	2323	NA	0	1008			
		Total	2323	NA	3247	1301			
9	**Hwb100**	Telegate	0	NA	4995	673	34	NA	50
		Teledata	3825	NA	0	1840			
		Total	3825	NA	4995	2513			
10	**rd32_272**	Telegate	0	10	10	2	25	40	40
		Teledata	8	0	0	4			
		Total	8	10	10	6			
11	**ham7_106**	Telegate	0	32	32	26	57	6	6
		Teledata	71	0	0	4			
		Total	71	32	32	30			
12	**rd53_139**	Telegate	0	17	17	6	25	29	29
		Teledata	16	0	0	6			
		Total	16	17	17	12			
13	**rd53_311**	Telegate	0	41	41	4	15	46	46
		Teledata	26	0	0	18			
		Total	26	41	41	22			
14	**parity_247**	Telegate	0	11	11	1	20	63	63
		Teledata	5	0	0	3			
		Total	5	11	11	4			
15	**adder16_174**	Telegate	0	17	17	2	30	47	47
		Teledata	13	0	0	7			
		Total	13	17	17	9			
16	**[[10,3,3]]**	Telegate	0	16	18	5	27	18	27
		Teledata	18	0	0	8			
		Total	18	16	18	13			
17	**[[16,3,5]]**	Telegate	0	NA	55	27	30	NA	12
		Teledata	69	NA	0	21			
		Total	69	NA	55	48			
18	**[[21,1,7]]**	Telegate	0	NA	80	5	14	NA	30
		Teledata	65	NA	0	51			
		Total	65	NA	80	56			
19	**[[24,3,7]]**	Telegate	0	NA	140	13	13	NA	28
		Teledata	117	NA	0	88			
		Total	117	NA	140	101			
20	**[[25,1,9]]**	Telegate	0	NA	111	22	24	NA	20
		Teledata	117	NA	0	66			
		Total	117	NA	111	88			
21	**[[27,1,9]]**	Telegate	0	NA	163	12	15	NA	40
		Teledata	116	NA	0	86			
		Total	116	NA	163	98			
22	**[[31,11,6]]**	Telegate	0	NA	231	39	20	NA	40

Table 3: The partitioning results achieved by WQCP for the benchmark circuits compared with the gate and qubit partitioning approaches (continue)

#	Benchmark	TP type	GP	QPILP	QPGTA	WQCP	Improvement (%)		
							GP	QPILP	QPGTA
		Teledata	172	NA	0	99			
		Total	172	NA	231	138			
		Telegate	0	NA	220	19			
23	[[**33,1,9**]]	Teledata	157	NA	0	119	12	NA	37
		Total	157	NA	220	138			
		Telegate	0	NA	267	21			
24	[[**35,1,10**]]	Teledata	132	NA	0	99	9	NA	55
		Total	132	NA	267	120			
		Telegate	0	NA	328	34			
25	[[**40,3,10**]]	Teledata	196	NA	0	150	6	NA	44
		Total	196	NA	328	184			
		Telegate	0	87	94	0			
26	**QFT16**	Teledata	23	0	0	23	0	73	75
		Total	23	87	94	23			
		Telegate	0	NA	165	0			
27	**QFT32**	Teledata	48	NA	0	48	0	NA	70
		Total	48	NA	165	48			
		Telegate	0	NA	275	0			
28	**QFT64**	Teledata	106	NA	0	106	0	NA	61
		Total	106	NA	275	106			
		Telegate	0	NA	385	0			
29	**QFT128**	Teledata	224	NA	0	224	0	NA	41
		Total	224	NA	385	224			
		Telegate	0	NA	605	0			
30	**QFT256**	Teledata	468	NA	0	468	0	NA	22
		Total	468	NA	605	468			

The runtime of our WQCP approach in comparison to the previous approaches (GP, QPILP, and QPGTA) is reported in Table 4. Since our approach and GP both use the same approach, their runtimes are approximately the same. QPILP is faster than ours for small circuits because the ILP qubit partitioning is run only once in QPILP while WQCP calls it for each level of a circuit. However, by increasing the size of the circuits, QPILP's resource overhead grows exponentially and it fails to generate an output as the memory runs out. Finally, QPGTA is the fastest approach. Its runtime is less than one second for all benchmark circuits.

The runtime of WQCP grows by increasing the number of qubits, the number of partitions, the window size, and the multi-qubit depth of a circuit. Since our approach calls the ILP solver only for each window, the runtime of our tool is manageable and, as it can be seen, it is applicable to large circuits such as Hwb100 and QFT256. Although the number of qubits in a window is the same as the total number of qubits in the circuit, the adjacency matrix is sparse for a sub-circuit contained in a window. This feature enables our approach to partition large circuits while other approaches using the ILP solver without windowing strategy fails to produce an output. Moreover, using some techniques such as hierarchical partitioning and utilizing a faster algorithm rather than ILP to implement subPartitioning(C_l) can accelerate WQCP probably at the cost of decreasing the quality of results.

Table 4. The runtimes of WQCP (in milisecond) for the benchmark circuits compared with the gate and qubit partitioning approaches

Benchmark	**GP**	**QPILP**	**QPGTA**	**WQCP**
2of5-D1	9122	305	3	7500
2-4dec	2259	122	4	1762
6sym	2956	158	4	3774
9sym	8379	605	6	7231
Ham15-D3	18631	140087	6	20710
Cycle17-3	717058	148226	12	871793
8bitadder	11867	1490	6	21051
Hwb50	758887	NA	20	865777
Hwb100	4235861	NA	39	4266630
rd32_272	1445	140	4	1392
ham7_106	5788	215	5	6580
rd53_139	2573	218	6	2266
rd53_311	6600	520	5	7343
parity_247	1510	166	5	1938
adder16_174	9978	2095	15	9724
[[**10,3,3**]]	1959	334	7	1835
[[**16,3,5**]]	9188	NA	5	10802
[[**21,1,7**]]	7491	NA	6	7294
[[**24,3,7**]]	18235	NA	7	14239
[[**25,1,9**]]	17441	NA	14	22624
[[**27,1,9**]]	20741	NA	14	26925
[[**31,11,6**]]	26014	NA	11	38533
[[**33,1,9**]]	27008	NA	10	27712
[[**35,1,10**]]	28471	NA	12	24308
[[**40,3,10**]]	41794	NA	14	34374
QFT16	11352	179462	12	5268
QFT32	16216	NA	18	15491
QFT64	34099	NA	37	37288
QFT128	131541	NA	57	150314
QFT256	1.386e6	NA	156	1.391e6

6. Conclusion

The main challenge of distributed quantum computing is costly communications between processing units which may be an order of magnitude more time consuming and error prone than logical operations. In this paper, we proposed an automated window-based partitioning method called WQCP to minimize such communications. The proposed method reduces the communication cost by about 29.5% in comparison with the best approaches reported in the literature.

Although the execution time of WQCP is more than existing approaches, it is not a challenge as it runs offline before actual computation. Moreover, one may speed up WQCP by utilizing a faster algorithm rather than the ILP one. Furthermore, in this paper the window weights and w_p are fixed and set manually based on our experiments. While these weights may highly effect the results. Automating window weight setting based on the input circuit can be followed as future work.

Acknowledgement

The authors acknowledge the financial support by the INSF.

References

[1] Timothy P Spiller, William J Munro, Sean D Barrett, and Pieter Kok. An introduction to quantum information processing: applications and realizations. *Contemporary Physics*, 46(6):407–436, 2005.

[2] Richard P Feynman. Simulating physics with computers. *Int. J. Theor. Phys*, 21(6/7), 1999.

[3] Dan C Marinescu and Gabriela M Marinescu. *Approaching quantum computing.* Pearson/Prentice Hall, 2005.

[4] Rodney Van Meter, Thaddeus D Ladd, Austin G Fowler, and Yoshihisa Yamamoto. Distributed quantum computation architecture using semiconductor nanophotonics. *International Journal of Quantum Information*, 8(01n02):295–323, 2010.

[5] C Monroe, R Raussendorf, A Ruthven, K R Brown, P Maunz, L-M Duan, and J Kim. Large-scale modular quantum-computer architecture with atomic memory and photonic interconnects. *Physical Review A*, 89(2):022317, 2014.

[6] Sahar Sargaran and Naser Mohammadzadeh. Saqip: A scalable architecture for quantum information processors. *ACM Transactions on Architecture and Code Optimization (TACO)*, 16(2):1–21, 2019.

[7] Muhammad Ahsan, Rodney Van Meter, and Jungsang Kim. Designing a million-qubit quantum computer using a resource performance simulator. *ACM Journal on Emerging Technologies in Computing Systems (JETC)*, 12(4):1–25, 2015.

[8] Kenneth R Brown, Jungsang Kim, and Christopher Monroe. Co-designing a scalable quantum computer with trapped atomic ions. *npj Quantum Information*, 2(1):1–10, 2016.

[9] Daniel Gottesman and Isaac L Chuang. Quantum teleportation is a universal computational primitive. *arXiv preprint quant-ph/9908010*, 1999.

[10] Rodney Van Meter, Kae Nemoto, WJ Munro, and Kohei M Itoh. Distributed arithmetic on a quantum multicomputer. In *33rd International Symposium on Computer Architecture (ISCA'06)*, pages 354–365. IEEE, 2006.

[11] Mark Whitney, Nemanja Isailovic, Yatish Patel, and John Kubiatowicz. Automated generation of layout and control for quantum circuits. In *Proceedings of the 4th international conference on Computing frontiers*, pages 83–94, 2007.

[12] Krysta M Svore, Alfred V Aho, Andrew W Cross, Isaac Chuang, and Igor L Markov. A layered software architecture for quantum computing design tools. *Computer*, 39(1):74–83, 2006.

[13] Steven Balensiefer, Lucas Kregor-Stickles, and Mark Oskin. An evaluation framework and instruction set architecture for ion-trap based quantum micro-architectures. In *32nd International Symposium on Computer Architecture (ISCA'05)*, pages 186–196. IEEE, 2005.

[14] Naser Mohammadzadeh and Elaheh Taqavi. Quantum circuit physical design flow for the multiplexed trap architecture. *Microprocessors and Microsystems*, 45:23–31, 2016.

[15] Mohammad Javad Dousti and Massoud Pedram. Minimizing the latency of quantum circuits during mapping to the ion-trap circuit fabric. In *2012 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 840–843. IEEE, 2012.

[16] Azim Farghadan and Naser Mohammadzadeh. Mapping quantum circuits on 3d nearest-neighbor architectures. *Quantum Science and Technology*, 4(3):035001, 2019.

[17] Tzvetan S Metodi, Darshan D Thaker, Andrew W Cross, Frederic T Chong, and Isaac L Chuang. Scheduling physical operations in a quantum information processor. In *Quantum Information and Computation IV*, volume 6244, page 62440T. International Society for Optics and Photonics, 2006.

[18] Masahiro Tanaka and Osamu Tatebe. Workflow scheduling to minimize data movement using multi-constraint graph partitioning. In *2012 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (ccgrid 2012)*, pages 65–72. IEEE, 2012.

[19] Naser Mohammadzadeh, Tayebeh Bahreini, and Hossein Badri. Optimal ilp-based approach for gate location assignment and scheduling in quantum circuits. *Modelling and Simulation in*

Engineering, 2014, 2014.

[20] Tayebeh Bahreini and Naser Mohammadzadeh. An minlp model for scheduling and placement of quantum circuits with a heuristic solution approach. *ACM Journal on Emerging Technologies in Computing Systems (JETC)*, 12(3):1–20, 2015.

[21] Naser Mohammadzadeh, Mehdi Sedighi, and Morteza Saheb Zamani. Quantum physical synthesis: improving physical design by netlist modifications. *Microelectronics Journal*, 41(4):219–230, 2010.

[22] Naser Mohammadzadeh, Morteza Saheb Zamani, and Mehdi Sedighi. Improving latency of quantum circuits by gate exchanging. In *2009 12th Euromicro Conference on Digital System Design, Architectures, Methods and Tools*, pages 67–73. IEEE, 2009.

[23] Naser Mohammadzadeh, Morteza Saheb Zamani, and Mehdi Sedighi. Quantum circuit physical design methodology with emphasis on physical synthesis. *Quantum information processing*, 13(2):445–465, 2014.

[24] Zahra Mirkhani and Naser Mohammadzadeh. Physical synthesis of quantum circuits using templates. *Quantum Information Processing*, 15(10):4117–4135, 2016.

[25] Naser Mohammadzadeh, Morteza Saheb Zamani, and Mehdi Sedighi. Auxiliary qubit selection: a physical synthesis technique for quantum circuits. *Quantum Information Processing*, 10(2):139–154, 2011.

[26] Mohammad Javad Dousti, Alireza Shafaei, and Massoud Pedram. Squash 2: a hierarchical scalable quantum mapper considering ancilla sharing. *arXiv preprint arXiv:1512.07402*, 2015.

[27] George Karypis and Vipin Kumar. Multilevel algorithms for multi-constraint graph partitioning. In *SC'98: Proceedings of the 1998 ACM/IEEE Conference on Supercomputing*, pages 28–28. IEEE, 1998.

[28] Mina Chookhachizadeh Moghadam, Naser Mohammadzadeh, Mehdi Sedighi, and Morteza Saheb Zamani. A hierarchical layout generation method for quantum circuits. In *The 17th CSI International Symposium on Computer Architecture & Digital Systems (CADS 2013)*, pages 51–57. IEEE, 2013.

[29] Melvin A Breuer. A class of min-cut placement algorithms. In *Proceedings of the 14th Design Automation Conference*, pages 284–290. IEEE Press, 1977.

[30] Guoming Wang and Oleg Khainovski. A fault-tolerant, ion-trap-based architecture for the quantum simulation algorithm. *Measurement*, 10(6):4–10.

[31] Mechthild Stoer and Frank Wagner. A simple min-cut algorithm. *Journal of the ACM (JACM)*, 44(4):585–591, 1997.

[32] Muhamamd Ahsan, Byung-Soo Choi, and Jungsang Kim. Performance simulator based on hardware resources constraints for ion trap quantum computer. In *2013 IEEE 31st International Conference on Computer Design (ICCD)*, pages 411–418. IEEE, 2013.

[33] Martin Juvan and Bojan Mohar. Optimal linear labelings and eigenvalues of graphs. *Discrete Applied Mathematics*, 36(2):153–168, 1992.

[34] George Karypis and Vipin Kumar. Multilevel k-way hypergraph partitioning. *VLSI design*, 11(3):285–300, 2000.

[35] Ali Javadi-Abhari, Pranav Gokhale, Adam Holmes, Diana Franklin, Kenneth R Brown, Margaret Martonosi, and Frederic T Chong. Optimized surface code communication in superconducting quantum computers. In *Proceedings of the 50th Annual IEEE/ACM International Symposium on Microarchitecture*, pages 692–705, 2017.

[36] Mariam Zomorodi-Moghadam, Mahboobeh Houshmand, and Monireh Houshmand. Optimizing teleportation cost in distributed quantum circuits. *International Journal of Theoretical Physics*, 57(3):848–861, 2018.

[37] Mahboobeh Houshmand, Zahra Mohammadi, Mariam Zomorodi-Moghadam, and Monireh Houshmand. An evolutionary approach to optimizing teleportation cost in distributed quantum computation. *International Journal of Theoretical Physics*, 59(4):1315–1329, 2020.

[38] Andrew M Childs, Eddie Schoute, and Cem M Unsal. Circuit transformations for quantum

architectures. *arXiv preprint arXiv:1902.09102*, 2019.

[39] Katsuhisa Yamanaka, Erik D Demaine, Takehiro Ito, Jun Kawahara, Masashi Kiyomi, Yoshio Okamoto, Toshiki Saitoh, Akira Suzuki, Kei Uchizawa, and Takeaki Uno. Swapping labeled tokens on graphs. *Theoretical Computer Science*, 586:81–94, 2015.

[40] Tillmann Miltzow, Lothar Narins, Yoshio Okamoto, Günter Rote, Antonis Thomas, and Takeaki Uno. Approximation and hardness for token swapping. *arXiv preprint arXiv:1602.05150*, 2016.

[41] Amlan Chakrabarti, Susmita Sur-Kolay, and Ayan Chaudhury. Linear nearest neighbor synthesis of reversible circuits by graph partitioning. *arXiv preprint arXiv:1112.0564*, 2011.

[42] Xin Sui, Donald Nguyen, Martin Burtscher, and Keshav Pingali. Parallel graph partitioning on multicore architectures. In *International Workshop on Languages and Compilers for Parallel Computing*, pages 246–260. Springer, 2010.

[43] Jordi Petit. Experiments on the minimum linear arrangement problem. *Journal of Experimental Algorithmics (JEA)*, 8(2.3).

[44] Alireza Shafaei, Mehdi Saeedi, and Massoud Pedram. Optimization of quantum circuits for interaction distance in linear nearest neighbor architectures. In *2013 50th ACM/EDAC/IEEE Design Automation Conference (DAC)*, pages 1–6. IEEE, 2013.

[45] Gushu Li, Yufei Ding, and Yuan Xie. Tackling the qubit mapping problem for nisq-era quantum devices. In *Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems*, pages 1001–1014, 2019.

[46] Joseph X Lin, Eric R Anschuetz, and Aram W Harrow. Using spectral graph theory to map qubits onto connectivity-limited devices. *arXiv preprint arXiv:1910.11489*, 2019.

[47] Brian W Kernighan and Shen Lin. An efficient heuristic procedure for partitioning graphs. *The Bell system technical journal*, 49(2):291–307, 1970.

[48] Charles M Fiduccia and Robert M Mattheyses. A linear-time heuristic for improving network partitions. In *19th Design Automation Conference*, pages 175–181. IEEE, 1982.

[49] IBM ILOG CPLEX Optimization Studio CPLEX. User's manual, version 12.8.0. *IBM Corporation*, 2018.

[50] Dmitri Maslov. Reversible logic synthesis benchmarks page. *http://webhome.cs.uvic.ca/ dmaslov/*, 2019.

[51] Robert Wille, Daniel Große, Lisa Teuber, Gerhard W Dueck, and Rolf Drechsler. Revlib: An online resource for reversible functions and reversible circuits. In *38th International Symposium on Multiple Valued Logic (ismvl 2008)*, pages 220–225. IEEE, 2008.

[52] Andrew W Cross, David P DiVincenzo, and Barbara M Terhal. A comparative code study for quantum fault-tolerance. *arXiv preprint arXiv:0711.1556*, 2007.

[53] Austin G Fowler and Lloyd CL Hollenberg. Scalability of shor's algorithm with a limited set of rotation gates. *Physical Review A*, 70(3):032329, 2004.

[54] Adriano Barenco, Charles H Bennett, Richard Cleve, David P DiVincenzo, Norman Margolus, Peter Shor, Tycho Sleator, John A Smolin, and Harald Weinfurter. Elementary gates for quantum computation. *Physical review A*, 52(5):3457–3467, 1995.