A dynamic programming approach for distributing quantum circuits by bipartite graphs

Zohreh Davarzani · **Mariam Zomorodi-Moghadam** · **Mahboobeh Houshmand** · **Mostafa Nouri-baygi**

Abstract Near-term large quantum computers are not able to operate as a single processing unit. It is therefore required to partition a quantum circuit into smaller parts, and then each part is executed on a small unit. This approach is known as distributed quantum computation. In this study, a dynamic programming algorithm is proposed to minimize the number of communications in a distributed quantum circuit (DQC). This algorithm consists of two steps: first, the quantum circuit is converted into a bipartite graph model, and then a dynamic programming approach (DP) is proposed to partition the model into low-capacity quantum circuits. The proposed approach is evaluated on some benchmark quantum circuits with remarkable reduction in the number of required teleportations.

Z. Davarzani
Department of Computer Engineering, Ferdowsi University of Mashhad, Mashhad, Iran

M. Zomorodi-Moghadam
Department of Computer Engineering, Ferdowsi University of Mashhad, Mashhad, Iran
E-mail: m_zomorodi@um.ac.ir

M. Houshmand
Department of Computer Engineering, Mashhad Branch, Islamic Azad University, Mashhad, Iran

M. Nouri-baygi
Department of Computer Engineering, Ferdowsi University of Mashhad, Mashhad, Iran

1 Introduction

Nowadays, with the empirical demonstrations of quantum computing, this field has witnessed a rapid growth with high performance in many areas such as database searching and integer factorization and etc. Quantum computation has many advantages over classical ones, but having a large-scale quantum system with many qubits, has implementation constraints [1] which makes distributed quantum implementation a necessity [2]. One challenge in distributed quantum computation is the interconnection between qubits and the environment, which makes quantum information more delicate and leads to error [3]. Distributed quantum system overcomes these problems in the sense that qubits are distributed to subsystems and each one is responsible of computation between fewer qubits. Therefore instead of having a large-scale quantum computer, it is reasonable and better to have a set of the limited-capacity quantum system which interact within quantum or classical channel and built the behavior of whole quantum system[4]. This concept is known as distributed quantum system.

DQC architecture can be described as follows [5]:

- Multiple quantum processing units (QPUs), each unit keeps a number of qubits and can execute some universal quantum gates on them.
- A classical communication network, the QPUs send or receive messages through this network when measuring their qubits.
- Ebit generation hardware, ebit is shared between two QPUs and consists of two qubits. Each of them is placed in different QPU. Also, an ebit includes the required information for sending a single qubit from one QPU to another one. Each QPU may have the hardware to generate and share ebits or they may be created by a central device.

A Distributed Quantum Circuit (DQC) consists of K smaller quantum circuits (called partition) with fewer qubits and limited capacity where partitions are far from each other [6,7]. It is necessary for DQC to have a reliable protocol for interconnection between subsystems. Teleportation [8] is a primitive protocol for interconnection between qubits by using entanglement of qubits, which is led to distribution of information through quantum system [9]. Figure 1 shows the quantum circuit for basic teleportation, as described in [10]. In this figure, two top lines are the sender's qubits and the bottom line is the receivers one. In this protocol, qubits transfer their states from one point to another without moving them physically. Finally, they perform computations locally on qubits. This approach is called teledata. There is another approach which is called telegate. In [2], telegate and teledata are discussed. In telegate, gates are executed remotely using the teleported gate without needing qubits to be nearby. Authors have shown that teledata is more appropriate for DQC systems and have used teledata for building a DQC system out of a monolithic quantum circuit. Teleportation is an expensive operation in DQC. Also according to no-cloning theorem [11], when a qubit teleports its state to a destination, after a while it may be required in its subsystem again. Therefore, it is essential to

minimize the number of teleportations in DQC. Dynamic programming (DP) is one important method for mathematical optimization and computer sciences and is widely used in many fields. In DP approach, the main problem is decomposed into smaller sub-problems and once all the sub-problems have been solved, one optimal solution to the large problem is left. In this paper, an algorithm is proposed to solve the problem of quantum circuit distribution. The algorithm consists of two steps: in the first step quantum circuit is modeled with a bipartite graph and in the next step, a dynamic programming approach is presented to partition the bipartite graph into K parts in the sense that the number of connections between the parts is minimized.

Fig. 1: Quantum circuit for teleporting a qubit [10].

The paper is organized as follows. In Section 2, some definitions and notations of distributed Quantum computing are described. Related works are presented in Section 3. In Section 4, partitioning of the bigraph is described. The proposed algorithm is presented in Section 5 and finally experimental results for some benchmarks are presented in Section 6.

2 Definitions and notations

In quantum computing, a qubit is the basic unit of quantum information. A qubit is a two-level quantum system and its state can be represented by a unit vector in a two-dimensional Hilbert space for which an orthogonal basis set denoted by $\{|0\rangle, |1\rangle\}$ has been fixed. Qubits can be in a superposition of $|0\rangle$ and $|1\rangle$ in form of $\alpha |0\rangle + \beta |1\rangle$ where α and β are complex numbers such that $| \alpha |^2 + | \beta |^2 = 1$. When the qubit state is measured, with probabilities $| \alpha |^2$ and $| \beta |^2$, classical outcomes of 0 and 1 are observed respectively.

There are many ways to present a quantum algorithm. For example adiabatic model of computation [12] and quantum programming languages [13]. But one of the mostly used approaches is quantum circuit [14]: a model for quantum computation by a sequence of quantum gates to transfer information on the input quantum registers. The quantum circuit is based on unitary evolution by networks of these gates [10]. Every n-qubit quantum gate is a linear transformation represented by a unitary matrix on an n-qubit Hilbert space. A set of useful single-qubit gates called Pauli set are defined bellow [15], [16]:

Fig. 2: Circuit and Matrix representation of CNOT gate

$$\sigma_0 = I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{1}$$

$$\sigma_1 = X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \tag{2}$$

$$\sigma_2 = Y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix} \tag{3}$$

$$\sigma_3 = Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \tag{4}$$

Another important single qubit gate is Hadamard which is defined as:

$$H = 1/\sqrt{2} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \tag{5}$$

A controlled-U is a two-qubit gate which acts on two qubits, namely, control and target qubits. When the control qubit is $|1\rangle$, U is applied to the target qubit, otherwise the target qubit remains unchanged. One of the most useful controlled-U gates is controlled-Not (CNOT) gate. This gate applies operator X to the target qubit, if the control qubit is $|1\rangle$. Otherwise the target qubit do not change. Figure 2 shows the circuit and the matrix representation of the CNOT gate.

A quantum circuit consists of several quantum gates interacted by quantum wires. Without loosing generality, it is assumed that the given quantum circuit consists of single-qubit and two-qubit gates. If three quantum gates that act on more than two qubits, they can be decomposed into some single-qubit or two-qubit gates. Figure 3 shows a quantum circuit with three qubits and four gates.

It is assumed a quantum circuit QC with width W, size S and depth D, where:

- W is the total number of qubits in the quantum circuit.
- S is the total number of gates in the circuit.
- D is the total time steps for executing the circuit. In each time steps, a set of gates are executed in parallel.

In the quantum circuit, qubits are shown by set Q and are they numbered from one to n, where i^{th} line from top is qubit i called q_i. The set of all gates in quantum circuit is shown by $\mathcal{G}$. Moreover, the gates are numbered in order of

Fig. 3: A sample quantum circuit

their executions in the quantum circuit. The gate i^{th} has been shown by g_i and means i^{th} gate in scheduling algorithm for executing of gates. In scheduling algorithm, some gates are executed in parallel. For these gate, the priority of gates is arbitrary. A distributed Quantum circuit (DQC) consists of N limited capacity Quantum Circuits or partitions which are located far from each other and altogether emulate the functionality of a large quantum circuit. Partitions of DQC communicate by sending their qubits to each other using a specific quantum communication channel through teleportation [1].

In DQC, There are two types of quantum gates:

– **Local gate:** A local gate consists of single-qubit and local CNOT gates: A single qubit has been shown by tuple $g_i(q_j, p_k)$, where g_i is i^{th} single-qubit gate acting on j^{th} qubit and p_k is partition k that gate j^{th} is lactated on it. In local CNOT gates, target and control qubits are in the same partition and is shown by $g_i(q_t, q_c, p_k)$, where g_i is i^{th} gate, q_t is target qubit and q_c is control qubit and p_k is partition k that gate j^{th} is located on it.
– **Global gate :** A global gate is the one whose target and control qubits are in the different partitions. This gate is shown by $g_i(q_t, q_c, p_t, p_c)$ where p_t and p_c are partitions which q_t and q_c are belonged on them respectively.

Figure 4 has partitioned the circuit presented in Figure 3 to two partitions p_1 and p_2. By this partitioning, global gates are g_1 and g_2 and local gate is g_3. The set of all gates ($\mathcal{G}$)is as follows:

$\mathcal{G} = CNOT(q_1, q_3, p_1, p_2), CNOT(q_1, q_3, p_1, p_2), CNOT(q_2, q_3, p_2)$

3 Related work

First ideas on distributed quantum computing were suggested by [17], Cleve and Buhrman [18] and later by Cirac et.al. [19]. In [17] a distributed quantum system is proposed. In this system, some particles have located far from each other and sent the required data information to a base station. He divides her quantum computing into several quantum computing parts and shows how her proposed algorithm acted optimally. Grover showed by suggested distributed quantum system, computation time is faster proportional to the particle.

Fig. 4: Circuit presented in Figure3 is partitioned two parts:p_1 and p_2.

There are many limitations for realizing a quantum computer. Also, many numbers of qubits for building a monolithic quantum system have technological limitations. This limitation is one of reason to appear distributed quantum computing [20]. Two types of communication for DQC are presented by Yepez [21]. In Type I, quantum computers use quantum communication between subsystems. In this type, each qubit may be entangled with the number of qubits. In Type II quantum computer exploits classical communication between subsystems of the distributed computer. In type II, a quantum computer consists of many quantum systems and they connect via classical channels.

An algorithm was presented by Zomorodi et al. [1] to optimize the number of qubit teleportations in a distributed quantum circuit. In this study, two spatially separated and long-distance quantum subsystem is considered. For different configurations of gate locations, the algorithm is run to calculate the minimum number of teleportations.

In [5], Pablo and Chris reduced the problem to hypergraph partitioning. They represented two routines called pre- and post-processing to improve the circuit distribution. Then, they evaluated their approach on five quantum circuits and showed that the distribution cost was more than halved in comparison to the naive approach.

Another modeling of distributed quantum circuits can be found in [22]. In this model, non-local gates of Shor algorithm have been implemented by the distributed quantum circuit. Also, the number of teleportations is calculated but no attempt has been done to minimize the number of teleportations.

Some definition of the distributed quantum circuit has been provided by Ying and Feng [23]. They presented an algebraic language for modeling quantum circuits. Van Meter et al. [7] presented a distributed quantum circuit for VBE carry-ripple adder. In this work, the VBE adder was divided into two separate quantum circuits and the circuits were communicated with each other through teleportation. So no attempt has been provided in this work to reduce the number of teleportations and there was a teleportation circuit for each global gate in the DQC. They considered two models called teledata and telegate topologies and proved that teledata is better than telegate. Beals et al. [24] presented a hypercube graph for a distributed quantum computer witch nodes connected via this graph and emulate a quantum circuit with low

overhead. They showed any quantum circuit can be replaced by a DQC whose nodes are connected via a hypercube model.

Streltsov et al. [25] proposed a way for distributed entanglement and provided the minimum quantum cost for sending an entangled composite state in long distance. They showed the amount of entanglement sent in the total process of distribution communication may not be more than the total entanglement for sending the ancilla particle and sending back that particle.

In [26] authors studied the challenges of designing quantum internet. Also they discussed that faster processing speed achieved by using connecting quantum computers via quantum internet. In another work [27], the authors studied the creation of quantum internet and considered teleportation as the main protocol to transfer the information. Then they explored the challenges and open issues in the design of quantum internet.

4 Bipartite graph partitioning

As stated, our new DQC model is based on graph partitioning. So, in this section, The approach of graph partitioning is discussed which has been used in this paper. The graph partitioning problem is an interesting field which is used in the VLSI circuit design [28], task scheduling, clustering and social networks and many other fields [29].

Since this problem is NP-Hard [30], some heuristics are used for the solution. There are many methods to solve graph partitioning such Kernighan-Lin [31], Fiduccia-Mattheyses algorithm [32] as multi-level methods[33, 34, 35], spectral partitioning [36, 37] and etc.

Definition I: Consider undirected and un weighted graph $G = (V, E)$, where V denotes the set of n vertices and E the set of edges. The graph partitioning problem takes a graph $G(V, E)$ as an input and a parameter k giving the number of parts we wish to partition the graph into K disjoint parts(subgraph) $(V_1, V_2, ..., V_K)$ such that each vertex of G is contained in exactly one sub-graph and all vertices are covered. Also the Communication cost among all of different parts(sub-graph) is minimized, This value is calculated as follows:

$$\sum_{i=1}^{K-1} \sum_{j=i+1}^{K} \sum_{v_1 \in p_i, v_2 \in p_j} w(v_1, v_2) \tag{6}$$

Where $w(v_1, v_2)$ is the weight of between vertices v_1 and v_2 for all $v_1 \in p_i, v_2 \in p_j$. In our problem, any weigth assign to edges of graph G. Therefore, in unweigthed graphs the communication cost is the number of edges among all of different sub-graphs $p_i, i = 1, ...K$.

Because our model is based on bipartite graphs, here some definitions related to bipartite graphs are introduced.

Definition II: A graph $G(V, E)$ is a bigraph whose vertices can be divided into two disjoint and independent sets X and Y ($V = X \cup Y$) so that each

edge connects a vertex in X to one in Y. Each set X and Y is called a part of the graph. This notation is presented in [38].

For the representation of quantum circuit by a bigraph, it is required to determine sets X and Y and edges between them. In our proposed model, we have considered sets X and Y as qubit set (Q) and gate set($\mathcal{G}$) respectively. The edge set of bigraph (E)is determined as follows:

For each $q \in Q$ and $g \in \mathcal{G}$, there is an edge $(q,g) \in E$, if qubit $q \in Q$ is control or target input of gate $q \in \mathcal{G}$ of quantum circuit. Two quantum gates describe in before, construct the edges of bigraph as follows:

- For a single-qubit gate $g_i(q_j, p_k)$ where $g_i \in Y, q_j \in X$, an edge (g_i, q_j) is added to bigraph. Also in a two-qubits gate $g_i(q_t, q_c, p_k)$, edges(g_i, q_t) and (g_i, q_c) are added to bigraph.
- For a global gate $g_i(q_t, q_c, p_t, p_c)$ edges(g_i, q_t) and (g_i, q_c) are added to bigraph.

The total number of vertices in graph is $W + S$. For example, the bigraph model of quantum circuit in Figure 4 has been shown in Figure 5. In this Figure, The sets X and Y of bigraph are $Q = \{q_1, q_2, q_3\}$ and $\mathcal{G} = \{g_1, g_2, g_3\}$ respectively. For example g_1 has q_1 and q_3 as control and target qubits respectively. Therefore edges (g_1, q_1) and (g_1, q_3) are added to the bigraph. Another edges of the bigraph are added as follows:

$$
\begin{aligned}
g_1(q_1, q_3, p_1, p_2) &\Rightarrow (g_1, q_1), (g_1, q_3) \\
g_2(q_1, q_3, p_1, p_2) &\Rightarrow (g_2, q_1), (g_2, q_3) \\
g_3(q_2, q_3, p_2) &\Rightarrow (g_3, q_2), (g_3, q_3) \\
E = \{(g_1, q_1), (g_1, q_3), (g_2, q_1), &(g_2, q_3), (g_3, q_2), (g_3, q_3)\}
\end{aligned} \tag{7}
$$

As shown Figure 4, it is assumed that qubits are partitioned into two parts: q_1 is assigned to Part 1 and q_2, q_3 are assigned to part two. As shown in Figure 5, the control and the target qubits of gates g_1 and g_2 are located in the different parts. Therefore, they are called global gates but gate g_1 is a local gate (having control and target qubits in the same part).

5 Proposed Algorithm

In this section, our proposed approach for finding the minimum number of communications in DQC is presented. It is assumed that the quantum circuit consists of single-qubit and two-qubit(CNOT) gates. The main algorithm is given in Algorithm 1 which receives the quantum circuit and number of partitions (K) as inputs and returns the minimum number of communication as an output.

The *main* algorithm consists of two steps (I and II) which are performed by *QCtoBigraph* and *DP* functions respectively. In Step I, the quantum circuit is converted to a bigraph as described in Section 5. This procedure is done by *QCtoBigraph* function and is called in Line 4 of the *main* algorithm.

Fig. 5: Bipartite graph of quantum circuit of Figure 4. The qubits are located in two parts: P_1, P_2.

Algorithm 1 Main algorithm

1: **function** MAIN(QC,K)
2:      ▷ Input: Quantum circuit (QC), The number of partitions (K)
3:      ▷ Output: The minimum number of teleportations
4:      ▷ Step I: G=QCtoBigraph(QC);
5:      ▷ Step II: Number of teleportation=DP(G,K);

Algorithm 2 This algorithm converts quantum circuit to bigraph

---

     function G=QCToBIGRAPH(QC)
2:      ▷ Step I: Convert QC to Bigraph
     Initialize G(V,E) , $G.X = Q$, $G.Y = \mathcal{G}$ and $E = \{\}$; ▷ $V = X \cup Y$
4:      ▷ Qubits are in one part of bigraph G(part X) and gates are in other parts(part Y)
     for each $g_i \in \mathcal{G}$ **do**
6:         $c \leftarrow$ contorol qubit of g_i;
         $t \leftarrow$ target qubit of g_i;
8:         Add to E edges (c, g_i) and (t, g_i);

    QCtoBipartite function is presented in Algorithm 2. This function takes the quantum circuit as an input and illustrates the bigraph as an output(G). As stated, a bigraph has two vertex sets called X and Y. Let $G(V, E)$ be a bigraph. In Line 3, the vertices set X and Y are set to Q and $\mathcal{G}$ respectively and the edge (E) is equal to empty. In Lines 5-8, edges are added to E according to the gates of QC from left to right as mentioned in Section 5.

    In Step II, dynamic programming (DP) algorithm is presented (Algorithm 3) to find the minimum the number of communications. This function is called in Line 5 of the *Main* algorithm.

    In the first step of the DP, the optimal sub-structure must be determined and then the main optimal solution is constructed which is obtained from optimal solutions of sub-problems.

    Let $T(S_i, j)$ be the minimum number of communications for partitioning the set S to j parts where set S consists of subset X of bigraph G with size

Algorithm 3 Dynamic programming to find the minimum number of communication

---

    function NUMBER OF TELEPORTATION=DP(G,k)

        ▷ Step II: DP approach to find minimum number of teleportation

3:     Initialize set S with member of X of graph G

     if $k == 1$ **then**

        $Return$ 0;

6:     $index=$ Comupte decimal number of S;

     $C[index, k] = \infty$;

     for each $S' \subset S$ **do**

9:        $q = connect(S', S - S') + DP(S - S', k - 1)$;

        if $q \leq C[index, k]$ **then**

            $C[index, k] = q$;

12:    $Return$ $C[index, k]$;

    function COUNT=CONNECT(S_1, S_2)

     Output:The number of global gates between S_1 and S_2

15:    $count = 0$;

     for each $q_i \in S_1$ **do**

        for each $q_j \in S_2$ **do**

18:        **for** each $g_k \in \mathcal{G}$ **do**

            if $(g_k, q_i) \in E$ and $(g_k, q_j) \in E$ **then**

                $count + +$;

21:    $Return$ $count$;

---

i. In other words, subset X (subset of qubit) is partitioned to j parts. For the full problem, the lowest-cost way would thus be $T(S_n, K)$.

$T(S_i, j)$ can be defined recursively as follows:

$$T(S_i, j) = min_{S'_k \subset S_i}(connect(S'_k, S_i - S'_k) + T(S_i - S'_k, j - 1)) \tag{8}$$
$$S.t 1 \leq k < i$$

Let $T(S, k)$ use the function $connect(S_1, S_2)$. This function counts the number of global quantum gates between two-qubit sets S_1 and S_2. In other words, for each two-qubit global gate $g_k = (q_t, q_c)$, if there is $q_t \in S_1$ and $q_c \in S_2$ or conversely, this function is increased by one. Eq (9) shows this function.

$$connect(S_1, S_2) = |Global\_gate(q_t, q_c)| \tag{9}$$
$$S.t\ (q_t \in S_1 \text{ and } q_c \in S_2)\ or\ (q_c \in S_1 \text{ and } q_t \in S_2)$$

Figure 6 shows the recursion tree of DP. In the root of tree, the main problem ($T(S_n, K)$) is placed. This value determines the minimum number of communications for partitioning the set S to K parts where set S consists of set X of bigraph G with the size n. In each level of tree, we compute subproblem for each subset $S' \subseteq S$ and $K - 1$.

Moreover, dynamic programming algorithms typically take the advantage of overlapping subproblems by solving each subproblem once and storing the solution in a table where it can be looked up when needed. This problem has

Fig. 6: The recursion tree for computation of $T(S, k)$. Each node of the tree contains subset S' and K. Dashed elipse shows the subproblem overlaps.

been shown in the recursion tree of Figure 6. It references entry $T(S'_{n-3}, k-2)$ many times; during computations of entries $T(S'_{n-2}, k-1)$ and $T(S'_{n-1}, k-1)$ and etc.

As a result of overlapping, we considered a table called C and value of $T(S, k)$ is placed in position $C[index, k]$ so that the value of $index$ is defined as follows.

Let b be a sequence of bits with size n. When $q_i \in Q$ is present in S', ith bit in b becomes one; otherwise it becomes zero. Then the value of $index$ is set to the decimal value of b.

DP function has bigraph G and the number of partitions (K) as inputs and returns the entries of table C as output by the concept of Eq (8) recursively. In the beginning of this function (Line 3), the set S is initialized by the set X of bigraph G. In Lines 4-5, if K is equal to one, then the communication cost will be zero for one part. The decimal number of S is computed and the value of $index$ equal to it. The minimum number of communications to partition set S to K parts is found among all subsets $S' \subseteq S$ in Lines 8-11. Also, Function $connect(S_1, S_2)$ is given in Lines 13-20. This function counts the number of global gates between two sets S_1 and S_2.

For example, we can consider bipartite graph of Figure 7 where $G.X = \{q_1, q_2, q_3, q_4\}$ and $G.Y = \{g_1, g_2, g_3, g_3, g_4, g_5, g_6, g_7\}$. It is assumed that $K = 3$. The steps of the algorithm for calculating $T(\{q_1, q_2, q_3, q_4\}, 3)$ are as follows:

Fig. 7: The example of quantum circuit.

$T(\{q_1, q_2, q_3, q_4\}, 3) =$

$$\min_{S' \subseteq S} = \begin{cases} connect(\{q_1\}, \{q_2, q_3, q_4\}) + T(\{q_2, q_3, q_4\}, 2) = 2 + 2 = 4 \\ connect(\{q_2\}, \{q_1, q_3, q_4\}) + T(\{q_1, q_3, q_4\}, 2) = 3 + 1 = 4 \\ connect(\{q_3\}, \{q_1, q_2, q_4\}) + T(\{q_1, q_2, q_4\}, 2) = 6 + 0 = 6 \\ connect(\{q_4\}, \{q_1, q_2, q_3\}) + T(\{q_1, q_2, q_3\}, 2) = 3 + 2 = 5 \\ connect(\{q_1, q_2\}, \{q_3, q_4\}) + T(\{q_3, q_4\}, 2) = 3 + 3 = 6 \\ connect(\{q_1, q_3\}, \{q_2, q_4\}) + T(\{q_2, q_4\}, 2) = 6 + 0 = 6 \\ connect(\{q_1, q_4\}, \{q_2, q_3\}) + T(\{q_2, q_3\}, 2) = 5 + 2 = 7 \end{cases}$$

As shown above, for obtaining the final solution of $T(\{q_1, q_2, q_3, q_4\}, 3)$, it is required to solve $T(\{q_2, q_3, q_4\}, 2)$, $T(\{q_1, q_3, q_4\}, 2)$, ... recursively. Table 1 indicates these results computed by DP function for this circuit. In this table, rows indicate the number of partitions and the columns represent the set of qubits participate in partitioning. Also, the decimal value of each qubit set is given in the first column.

In this example, the minimum number of communications, which is four, occurs for $\{\{q_1\}, \{q_2\}, \{q_3, q_4\}\}$. The solution shows that $\{q_1\}$ is placed in partition 1 and $\{q_2, q_3, q_4\}$ are partitioned to two parts recursively. By solving $T(\{q_2, q_3, q_4\}, 2)$, qubit sets $\{q_2\}$ and $\{q_3, q_4\}$ are assigned to parts 2 and 3 recursively.

Let us consider the steps of the gate executions according to this partitioning. The algorithm starts with the first gate in $\mathcal{G}$ i.e. $g_1(q_1, q_2, p_1, p_2)$ which is a global gate. For executing this gate, qubit q_1 in p_1 is teleported to p_2, and the number of communication is increased by one, and then step by step all other gates are executed and removed from the list. Other steps of running gates are as follows:

- $g_2(q_2, q_3, p_2, p_3)$ is a global gate and q_2 is teleported to part three and executed there.
- $g_3(q_1, q_3, p_1, p_3)$ is a global gate because its target input is in part three and its control input is in part one. Therefore, q_1 in part one is teleported to part three for executing g_3.
- g_4 and g_5 are local gates and are executed in part three. g_6 and g_7 are global and local gates respectively and are executed the same as other gates.

Table 1: Table C obtained from the DP function for the circuit of Figure 7.

k	1 {1}	2 {2}	3 {2,1}	4 {3}	5 {3,1}	6 {3,2}	7 {3,2,1}	8 {4}	9 {4,1}	10 {4,2}	11 {4,2,1}	12 {4,3}	13 {4,3,1}	14 {4,3,2}	15 {4,3,2,1}
3	N.A	N.A	N.A	N.A	N.A	N.A	N.A	N.A	N.A	N.A	N.A	N.A	N.A	N.A	2
2	N.A	N.A	1	N.A	1	2	2	N.A	0	0	0	3	1	2	2
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

6 Experimental Results

We implemented our algorithm in MATLAB on a workstation with 4GB RAM and 0.5 GHz CPU to find the best partitioning with an optimized number of teleportation. Many different quantum circuits were used for comparing the performance of our algorithm with other approaches. These quantum circuits are as follows:

- Quantum fourier transform (QFT) [10]: in quantum computing, the quantum fourier transform (QFT) is a linear transformation on quantum bits. QFT is used in some quantum algorithms such as Shor's algorithm. The quantum gates used in the implementation of this algorithm are the Hadamard gate and the controlled phase gate R_m as described in Section 3.

- Binary welded tree (BWT) [39]: it consists of two balanced binary trees of the height n with the 2^n leaves of the left tree identified with 2^n leaves of the right tree. In this circuit, Toffoli gates are replaced with CNOT gates.

- Ground state estimation (GSE) [40]: twice the default number of basic functions and occupied orbitals.

- Another set of test samples for quantum circuits was taken from Revlib [41] library which is an online resource of benchmarks. We used some of them such as: Alu_primitive, Parity, Flip_flop, Sym9_147.

- To compare the results with the work in [1], we used the same quantum circuit example of [1].

For comparision with Pablo method, we used the $ratio(R)$ used in [5] as follows:

$$R = \frac{Number\_teleportations}{2 * Number\_qubits} \tag{10}$$

As stated before, each teleportation comprises two qubits and each qubit is located in different part. Therefore half of teleportation is related to the number of qubits. For this purpose, number two is used in the fraction of this equation. Having $R > 1$ means that the number of teleportations is greater than the number of qubits and some qubits are teleported more than once. Therefore this distribuition has not act well comared to $R < 1$.

Figure 8 shows this ratio (R) for various number of partitions (K) in comparison with Pablo [5] for three quantum circuits: BWT (Figure 8.a), QFT (Figure 8.b), and GSE (Figure 8.c) circuits. In comparison with [5], the parameter R is better except for GSE circuit which did not distribute well for $K = 13$. Also proposed method produced the same R for $K = 7$ in GSE circuit. By comparing the values in Table 2, QFT for $K = 3$ did not produce good result by the method presented in [5] and required several qubits for communication. Also, for $K >= 5$, QFT required more qubits than the number of communications $(R > 1)$. But in our proposed approach, for $K = 3, 5$ the distribution is performed better: The ratio was less than one (R < 1). The exact values of R are given in Table 2.

Table 3 shows the minimum number of communications for parity_247, Sym9_147, Flip_flop, Alu_primitive, Alu_primitive_opt, and random circuit example of [1]. In this table, the number of qubits, gates, and partitions are given for each sample. We compared the results of the proposed approach with two proposed methods presented in [1] and [42] in terms of the teleportation cost (TC).

Let us consider the sample circuit of [1] in figure 9. This sample circuit has been reproduced from [1], (their figure 4) for reference. In [1], minimum number of communications, which is four, occurs for Config-Arr={11000} where the first and second global gates are executed in P_1 and other global gates are executed in partition P_0. In our model, $\{q_1, q_2, q_3\}$ and $\{q_3\}$ are located in P_0 and P_1 respectively for $K = 2$. The minimum number of communications was two and the steps of running gates are shown in Table 4. The model of [1] had some limitations: in the beginning of their algorithm, partitions were fixed and they did not afford to find optimized partitions. Therefore, their space model was limited to two pre-defined partitions and they considered different configurations for this pre-defined partitioning.

(a) BWT sample

(b) QFT sample

(c) GSE sample

Fig. 8: Each bar shows the ratio (R) between the number of teleportations and number of qubits. X and Y axes show the number of partitions and R respectively.

Table 2: The ratio between number of teleportaions (halves) and number of qubits for $K = 3, 5, .., 13$ for QFT, BWT and GSE circuits in comparison with [5](RP)

Circuit	K=3		K=5		K=7		K=9		K=11		K=13	
	R	RP	R	RP	R	RP	R	RP	R	RP	R	RP
BWT	0.12	0.55	0.35	0.6	0.35	0.64	0.7	0.65	0.35	0.7	0.36	0.77
QFT	0.4	1	0.85	1.4	1.2	1.58	1.2	1.75	1.6	1.75	1.75	1.8
GSE	0.5	0.54	0.64	0.7	0.7	0.7	0.79	0.8	0.72	0.74	0.9	0.74

Fig. 9: Sampel quantum circuit. The figure was reproduced from [1].

Table 3: Comparison of the proposed approach (P) with [1] and [42].

Circuit	#of qubits	# of gates	K	TC [42]	TC [1]	TC (P)
pariaty_247	17	16	2	2	2	2
Sym9_147	12	108	2	48	N.A.	8
Flip_flop	8	30	3	N.A.	N.A.	8
Alu_primitive	6	21	2	20	18	6
Alu_primitive_opt	6	21	2	10	10	6
Figure 4 of [1]	4	7	2	4	4	2

Table 4: The steps of the proposed algorithm for the random circuit of [1]. L and G stand for local and global gates respectively.

# of Gate	Gate_name	Type of gate
g_1	$CNOT(q_1, q_2, p_0)$	L
g_2	$CNOT(q_3, p_1, q_1, p_0)$	G
g_3	$CNOT(q_1, q_4, p_0)$	L
g_4	$H(q_4, p_0)$	L
g_5	$CNOT(q_2, q_4, p_0)$	L
g_6	$H(q_2, p_0)$	L
g_7	$CNOT(q_2, q_4, p_0)$	L
g_8	$CNOT(q_1, q_2)$	L
g_9	$CNOT(q_1, q_4, p_0)$	L

7 Conclusion

Teleportation is a costly operation in quantum computation and it is very important to minimize the number of this operation in computations. In this

study, an algorithm was proposed for distributing quantum circuits to optimize the number of teleportations between qubits. The proposed algorithm consisted of two steps: in the first step, the quantum circuit was converted to a bipartite graph (bigraph) and in the next step by a dynamic programming approach, bigraph was partitioned into K parts. Finally, compared with previous works in [1],[5], and [42], it was shown that the proposed approach yielded the better or the same results for benchmark circuits.

References

1. M. Zomorodi-Moghadam, M. Houshmand, M. Houshmandi, Theoretical Physics **57**(3), 848861 (2018)
2. R. Van Meter, T.D. Ladd, A.G. Fowler, Y. Yamamoto, International Journal of Quantum Information **8**, 295 (2010)
3. H.G. Krojanski, D. Suter, Physical review letters **93**(9), 090501 (2004)
4. N.H. Nickerson, Y. Li, S.C. Benjamin, Nature communications **4**, 1756 (2013)
5. P. Andres-Martinez, Theoretical computer science **410**(26), 2489 (2018)
6. R.V. Meter, M. Oskin, ACM Journal on Emerging Technologies in Computing Systems (JETC **2**, 2006 (2006)
7. V. Meter, W. Munro, K. Nemoto, K.M. Itoh, ACM Journal on Emerging Technologies in Computing Systems (JETC) **3**, 2 (2008)
8. C.H. Bennett, G. Brassard, C. Crepeau, R. Jozsa, A. Peres, W.K. Wootters, Physical Review Letters **70**, 1895 (1993)
9. M. Whitney, N. Isailovic, Y. Patel, J. Kubiatowicz, Physical Review Letters **85**(26), 1330 (2000)
10. M.A. Nielsen, I.L. Chuang, *Quantum computation and quantum information*, 10th edn. (Cambridge University Press, 2010)
11. W.K. Wootters, W.H. Zurek, nature **299**, 802 (1982)
12. E. Farhi, J. Goldstone, S. Gutmann, M. Sipser, arXiv preprint quant-ph/0001106 (2000)
13. M. Zomorodi-Moghadam, M.A. Taherkhani, K. Navi, in *Transactions on Computational Science XXIV* (Springer, 2014), pp. 74–91
14. D.E. Deutsch, Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences **425**(1868), 73 (1989)
15. Y. Weinstein, S. Buchbinder, Journal of Modern Optics **61**(1), 49 (2014)
16. M. Zomorodi-Moghadam, K. Navi, Journal of Circuits, Systems and Computers **25**(12) (2016)
17. L.K. Grover, arXiv preprint quant-ph/9704012 (1997)
18. R. Cleve, H. Buhrman, Physical Review A **56**, 1201 (1997)
19. J. Cirac, A. Ekert, S. Huelga, C. Macchiavello, Physical Review A **59**, 4249 (1999)
20. R. Van Meter, S.J. Devitt, Computer **49**(9), 31 (2016)
21. Yepez, Jeffrey, International Journal of Modern Physics C **12**(09), 1273 (2001)
22. Yimsiriwattana, Anocha, L. Jr, S. J, arXiv preprint quant-ph/0403146 (2004)
23. M. Ying, Y. Feng, IEEE Transactions on Computers **58**(6), 728 (2009)
24. R. Beals, S. Brierley, O. Gray, A.W. Harrow, S. Kutin, N. Linden, D. Shepherd, M. Stather, Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences **469**(2153), 20120686 (2013)
25. A. Streltsov, H. Kampermann, D. Bruß, Physical review letters **108**, 250501 (2012)
26. M. Caleffi, A.S. Cacciapuoti, G. Bianchi, in *Proceedings of the 5th ACM International Conference on Nanoscale Computing and Communication* (2018), pp. 1–4
27. A.S. Cacciapuoti, M. Caleffi, F. Tafuri, F.S. Cataliotti, S. Gherardini, G. Bianchi, IEEE Network (2019)
28. K. Andreev, H. Racke, Theory of Computing Systems **39**(6), 929 (2006)
29. A. Buluç, H. Meyerhenke, I. Safro, P. Sanders, C. Schulz, in *Algorithm Engineering* (Springer, 2016), pp. 117–158
30. K. Andreev, H. Racke, Theory of Computing Systems **39**(6), 929 (2006)

31. B.W. Kernighan, S. Lin, Bell system technical journal **49**(2), 291 (1970)
32. C.M. Fiduccia, R.M. Mattheyses, in *19th Design Automation Conference* (IEEE, 1982), pp. 175–181
33. B. Hendrickson, R.W. Leland,
34. G. Karypis, V. Kumar, SIAM Journal on scientific Computing **20**(1), 359 (1998)
35. G. Karypis, R. Aggarwal, V. Kumar, S. Shekhar, IEEE Transactions on Very Large Scale Integration (VLSI) Systems **7**(1), 69 (1999)
36. H. Zare, P. Shooshtari, A. Gupta, R.R. Brinkman, BMC bioinformatics **11**(1), 403 (2010)
37. E. Arias-Castro, G. Chen, G. Lerman, et al., Electronic Journal of Statistics **5**, 1537 (2011)
38. M.N.S.S. CK. Thulasiraman, *Theory and Algorithms* (1994)
39. A.M. Childs, R. Cleve, E. Deotto, E. Farhi, S. Gutmann, D.A. Spielman, STOC '03 Proceedings of the thirty-fifth annual ACM symposium on Theory of computing **410**(26), 59 (2003)
40. J.D. Whitfield, J. Biamonte, A. Aspuru-Guzik, Molecular Physics **109**(5), 735 (2011)
41. R. Wille, D. Grobe, L. Teuber, G. Dueck, R. Drechsler, Multiple-Valued Logic, IEEE International Symposium on **0**, 220 (2008)
42. M.Z.M. Zahra Mohammadi, M. Houshmand, M. Houshmandi, Arxiv (2019)