




A dynamic programming approach for distributing quantum circuits by bipartite graphs

Zohreh Davarzani¹ · Mariam Zomorodi-Moghadam¹  · Mahboobeh Houshmand² · Mostafa Nouri-baygi¹

Received: 30 March 2020 / Accepted: 18 September 2020
© Springer Science+Business Media, LLC, part of Springer Nature 2020

Abstract

There are many challenges involved in building near-term large-scale quantum computers. Some of these challenges can be overcome by partitioning a quantum circuit into smaller parts and allowing each part to be executed on a smaller quantum 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. This algorithm consists of two steps: first, the quantum circuit is converted into a bipartite graph model, and then a dynamic programming approach is proposed to partition the model into low-capacity quantum circuits. The proposed approach is evaluated on some benchmark quantum circuits, and a remarkable reduction in the number of required teleportations is obtained.

Keywords Quantum computation · Quantum circuit · Distributed quantum circuit · Dynamic programming

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, integer factorization. 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 quantum computation is the interconnection between qubits and the environment, which makes quantum information more delicate and leads to error [3]. Distributed quantum systems overcome these problems in the sense that qubits

✉ Mariam Zomorodi-Moghadam
m_zomorodi@um.ac.ir

¹ Department of Computer Engineering, Ferdowsi University of Mashhad, Mashhad, Iran

² Department of Computer Engineering, Mashhad Branch, Islamic Azad University, Mashhad, Iran

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 recommended to have a set of the limited-capacity quantum system which interact within a quantum or a classical channel and build the behavior of whole quantum system [4]. This concept is known as distributed quantum system. Adaptation of the distributed quantum computing allows large-scale quantum computation and provides the infrastructure for the quantum internet [5].

DQC architecture can be described as follows [6]:

- *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 a 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 it may be created by a central device.

A distributed quantum circuit (DQC) consists of K smaller quantum circuits (called partitions) with fewer qubits and limited capacity where partitions are far from each other [7,8]. It is necessary for a DQC to have a reliable protocol for interconnections between subsystems. Teleportation [9] is a primitive protocol for this interconnection by using entanglement of qubits, which is led to distribution of information through quantum system [10]. Figure 1 shows the quantum circuit for basic teleportation, as described in [11]. In this figure, two top lines are the sender's qubits and the bottom line is the receiver's one. In this protocol, qubits transfer their states from one point to another one 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. In that study, 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 [12], when the state of a qubit is teleported 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 by 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.

The paper is organized as follows. In Sect. 2, some definitions and notations of distributed quantum computing are described. Related work is presented in Sect. 3. In

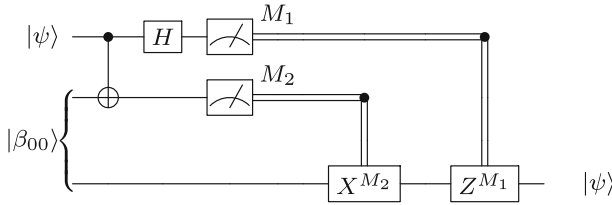


Fig. 1 Quantum circuit for teleporting a qubit [11]

Sect. 4, partitioning of the bigraph is described. The proposed algorithm is presented in Sect. 5, and finally, experimental results for some benchmarks are explained in Sect. 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 [13] and quantum programming languages [14]. But one of the mostly used approaches is quantum circuit [15]: 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 [11]. 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 below [16,17]:

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

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

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

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

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

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

Fig. 2 Circuit and matrix representation of CNOT gate

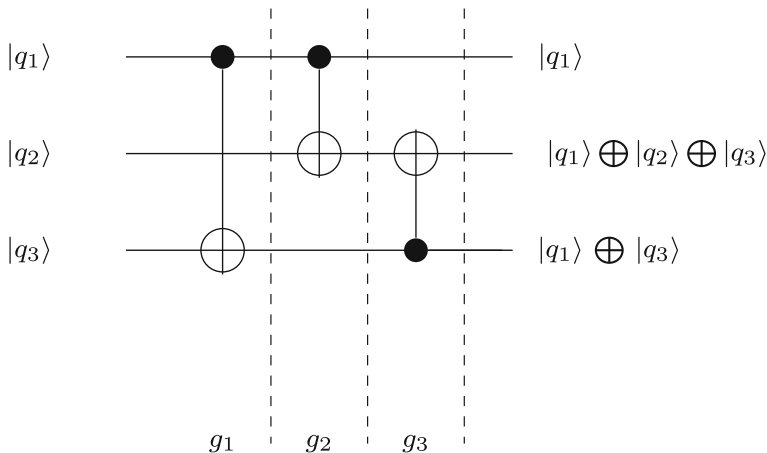
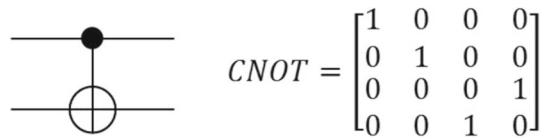


Fig. 3 A sample quantum circuit

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 does not change. Figure 2 shows the circuit and the matrix representation of the CNOT gate.

A quantum circuit consists of several qubits and a number of gates acting on these qubits. Without losing generality, it is assumed that the given quantum circuit composed of single-qubit and two-qubit gates. For example, Fig. 3 shows a sample quantum circuit with three qubits and three gates.

In a quantum circuit, three important resources are as follows [18]:

- Width (W): the total number of qubits in the quantum circuit.
- Size (S): the total number of gates in the circuit.
- Depth (D): the total time steps for executing the circuit. In each time step, a set of gates is executed in parallel.

In the quantum circuit, qubits are shown by set Q and they are numbered from one to n , where i th line from top is called q_i . The set of all gates in the quantum circuit is shown by \mathcal{G} . Moreover, the gates are numbered in the order of their executions in the quantum circuit. The order of execution is based on a scheduling algorithm. In this work, we assume that this order is already known. The i th gate is shown by g_i .

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

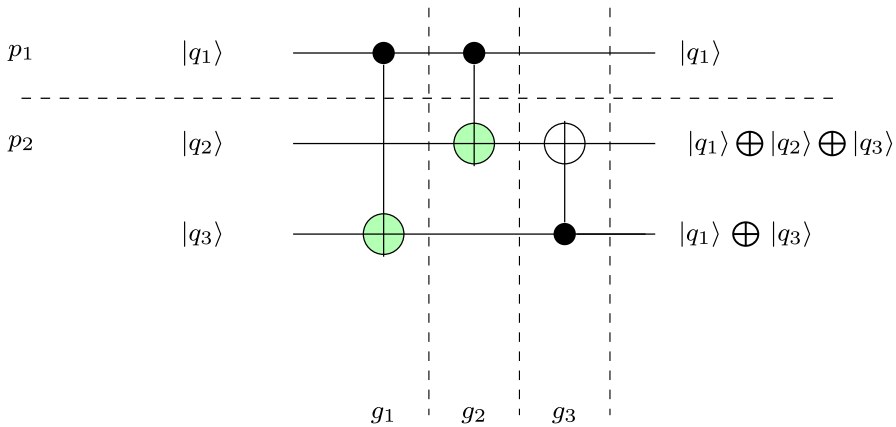


Fig. 4 Circuit of Fig. 3 partitioned into two parts: p_1 and p_2

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 gate is shown by a tuple $g_i(q_j, p_k)$, where g_i is i th single-qubit gate acting on j th qubit which is shown by q_j , and p_k is the partition k that qubit j is located on. In the local CNOT gates, target and control qubits are in the same partition and it is shown by $g_i(q_t, q_c, p_k)$, where g_i is i th gate, q_t is the target qubit, and the q_c is the control qubit, and p_k is the partition k that qubit j is located on.
- *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 that q_t and q_c belong to them, respectively.

Figure 4 partitions the circuit presented in Fig. 3 into two partitions, p_1 and p_2 . By this partitioning, global gates are g_1 and g_2 and the local gate is g_3 . The gates of this circuit are represented as follows:

$$G = CNOT(q_1, q_3, p_1, p_2), CNOT(q_1, q_2, p_1, p_2), CNOT(q_2, q_3, p_2)$$

3 Related work

First ideas on distributed quantum computing were suggested by Grover [19], Cleve and Buhrman [20] and later by Cirac et al. [21]. In [19] a distributed quantum system is proposed. In this system, some particles have located far from each other and send the required data information to a base station when necessary. The author divides the given quantum computation into several parts. Grover showed that using this distributed approach, the overall computation time is faster proportional to the number of such distributed particles.

There are many limitations for realizing a quantum computer. Also, having many numbers of qubits for building a monolithic quantum system has technological limitations. These limitations are one of the reasons for moving toward a distributed quantum computing [22]. Two types of communication for DQC have been presented by Yezep [23]. In Type I, quantum computers use quantum communication between subsystems. In this type, each qubit may be entangled with a number of qubits. In Type II, the quantum computer exploits classical communication between subsystems of the distributed computer. In this type, a quantum computer consists of many quantum systems and they are connected 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 that study, two spatially separated and long-distance quantum subsystems are considered. For different configurations of gate locations, the algorithm is run to calculate the minimum number of teleportations.

The authors of [6] 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 half in comparison with the naive approach.

Another model for distributed quantum circuits can be found in [24]. In that model, non-local gates of Shor algorithm have been implemented by the distributed quantum circuit. Although the number of teleportations is calculated, no attempt has been made to minimize the number of teleportations.

Some definitions of the distributed quantum circuit have been provided by Ying and Feng [25]. They presented an algebraic language for modeling quantum circuits. Van Meter et al. [8] 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 done 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. [26] presented a hypercube graph for a distributed quantum computer which 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.

Steltsov et al. [27] 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.

The authors of [28] studied the challenges of designing quantum internet. Also, they discussed that faster processing speed is achieved by connecting quantum computers via quantum internet. In another work [29], 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. Recently, imperfect entanglement for non-local quantum operations and the effect on the fidelity for a distributed implementation of a quantum phase estimation circuit has been considered in [30]. The authors have only considered imperfect entanglement

(i.e., fidelity < 1). All local operations were assumed perfect, and qubits were assumed not to decohere.

4 Bipartite graph partitioning

As stated, our new DQC model is based on graph partitioning. Therefore, in this section the approach of graph partitioning which has been used in the paper is discussed. The graph partitioning problem is an interesting field which is used in the VLSI circuit design [31], task scheduling, clustering and social networks and many other fields [32].

Since this problem is NP-Hard [31], some heuristics are used for the solution. There are many methods to solve graph partitioning such as Kernighan–Lin [33], Fiduccia–Mattheyses algorithm [34], multi-level methods[35–37], spectral partitioning [38,39].

Definition I: Consider an undirected and 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 . We intend to partition the graph into K disjoint parts (sub-graphs) (V_1, V_2, \dots, V_K) such that each vertex of G is contained in exactly one sub-graph and all vertices are covered. Moreover, 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 between vertices v_1 and v_2 for all $v_1 \in p_i, v_2 \in p_j$. In our problem, no weight is assigned to edges of graph G . In unweighted graphs, the communication cost is the number of edges among all of the different sub-graphs $p_i, i = 1, \dots K$.

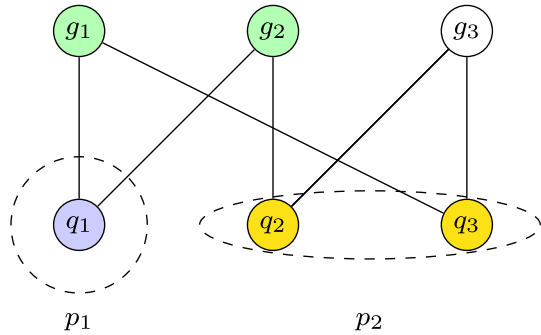
Because our model is based on bipartite graphs, here some definitions related to bipartite graphs are introduced.

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 of the sets X and Y is called a part of the graph. This notation is presented in [40].

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 (G), respectively. The edge set of bigraph (E) is determined as follows: For each $q \in Q$ and $g \in G$, there is an edge $(q, g) \in E$, if qubit $q \in Q$ is control or target input of gate $g \in G$ of quantum circuit. As mentioned above, there are two types of gate called local and global in DQC. These gates 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 the bigraph. Also for 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.

Fig. 5 Bipartite graph of quantum circuit of Fig. 4. The qubits are located in two parts: p_1, p_2



– 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 Fig. 4 is shown in Fig. 5. In this figure, the sets X and Y are $Q = \{q_1, q_2, q_3\}$ and $G = \{g_1, g_2, g_3\}$, 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. Other edges of the bigraph are added as follows:

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

As shown in Fig. 4, it is assumed that qubits are partitioned into two parts: q_1 is assigned to p_1 and q_2, q_3 are assigned to p_2 . As shown in Fig. 5, the control and the target qubits of gates g_1 and g_2 are located in different parts. Therefore, they are called global gates, whereas gate g_3 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 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 the number of partitions (K) as inputs and returns the minimum number of communications 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 Sect. 5. This procedure is done by *QCtoBigraph* function, and it is called in Line 4 of the *main* algorithm.

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

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 teleportations= $DP(G,K)$ ;
    
```

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:      $c \leftarrow$  control qubit of  $g_i$ ;
      $t \leftarrow$  target qubit of  $g_i$ ;
8:     Add to  $E$  edges  $(c, g_i)$  and  $(t, g_i)$ ;
    
```

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 Sect. 5.

Algorithm 3 Dynamic programming to find the minimum number of communications

```

function NUMBER OF TELEPORTATIONS= $DP(G,k)$ 
   ▷ Step II: DP approach to find minimum number of teleportations
3:   Initialize set  $S$  with member of  $X$  of graph  $G$ 
   if  $k == 1$  then
     Return 0;
6:    $index =$  Compute 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  $count$ ;
    
```

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

In the first step of 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 into j parts where set S consists of subset X of bigraph G with size i . In other words, subset X (subset of qubit) is partitioned into j parts. For the full problem, $T(S_n, K)$ will contain the lowest cost.

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

$$T(S_i, j) = \min_{S'_k \subset S_i} (\text{connect}(S'_k, S_i - S'_k) + T(S_i - S'_k, j - 1)) \tag{8}$$

$$S.t. 1 \leq k < i$$

Let $T(S, k)$ use the function $\text{connect}(S_1, S_2)$. 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. Equation (9) shows this function.

$$\text{connect}(S_1, S_2) = |\text{Global_gate}(q_t, q_c)|$$

$$S.t. (q_t \in S_1 \text{ and } q_c \in S_2) \text{ or } (q_c \in S_1 \text{ and } q_t \in S_2) \tag{9}$$

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

As a result of overlapping, we considered a table called C and value of $T(S, k)$ is placed in position $C[\text{index}, k]$ 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' , i th 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 an 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 is equal to it. The minimum number of communications to partition set S into K parts is found among all subsets $S' \subseteq S$ in Lines 8–11. Also, Function $\text{connect}(S_1, S_2)$ is given in Lines 13–20. This function counts the number of global gates between two sets S_1 and S_2 .

For example, Fig. 7 shows a sample quantum circuit partitioned into three parts by our proposed approach. We can consider bipartite graph of this figure, where $G.X =$

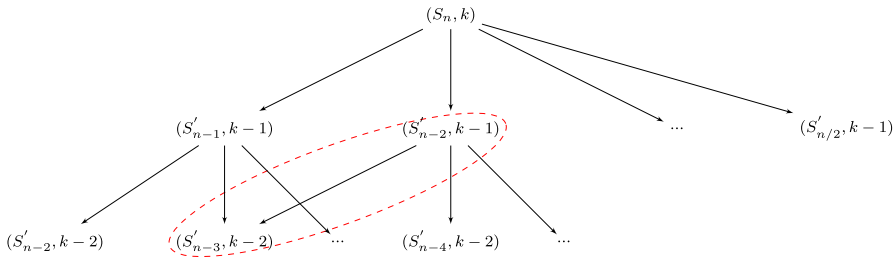


Fig. 6 The recursion tree for computation of $T(S, k)$. Each node of the tree contains subset S' and K . Dashed ellipse shows the subproblem overlaps

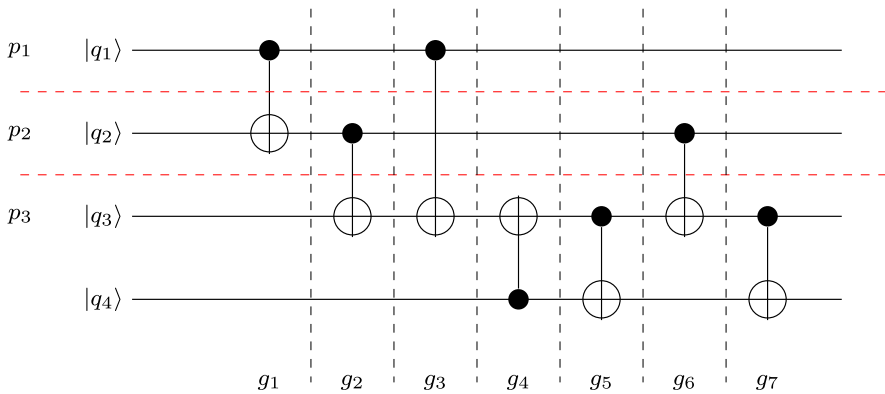


Fig. 7 A sample quantum circuit partitioned by our proposed approach

$\{q_1, q_2, q_3, q_4\}$ 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:

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

As shown above, for obtaining the final solution of $T(\{q_1, q_2, q_3, q_4\}, 3)$, it is required to solve $T(\{q_2, q_3, q_4\}, 2), T(\{q_1, q_3, q_4\}, 2), \dots$ 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 participated 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 p_1 and $\{q_2, q_3, q_4\}$ are partitioned into 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 p_2 and p_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 p_3 and executed there.
- $g_3(q_1, q_3, p_1, p_3)$ is a global gate because its target input is in p_3 and its control input is in p_1 . Therefore, q_1 in p_1 is teleported to p_3 for executing g_3 .
- g_4 and g_5 are local gates and are executed in p_3 . g_6 and g_7 are global and local gates, respectively, and are executed the same as other gates.

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) [11]: 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 Sect. 3.
- Binary welded tree (BWT) [41]: 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) [42]: twice the default number of basic functions and occupied orbitals.
- Another set of test samples for quantum circuits was taken from Revlib [43] 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].

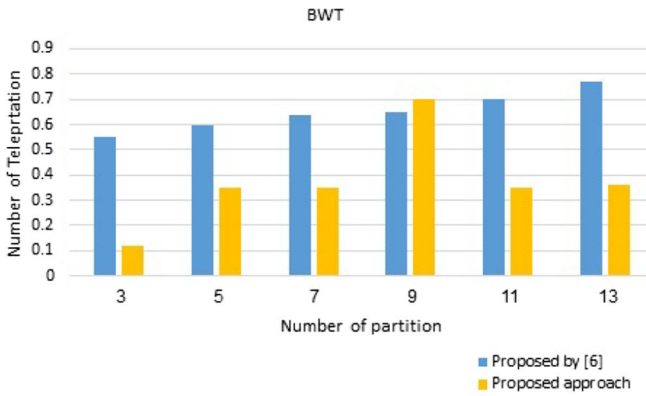
For comparison with method of [6], we used the $ratio(R)$ as follows:

$$R = \frac{\text{Number_teleportations}}{2 * \text{Number_qubits}} \quad (10)$$

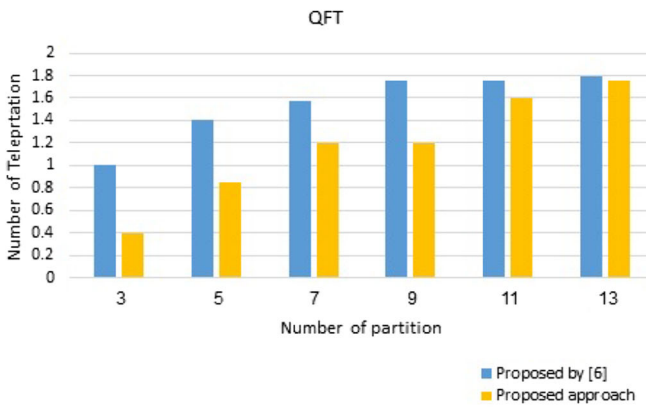
As stated before, each teleportation comprises two qubits and each qubit is located in different parts. Therefore, half of the teleportation is related to the number of qubits.

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

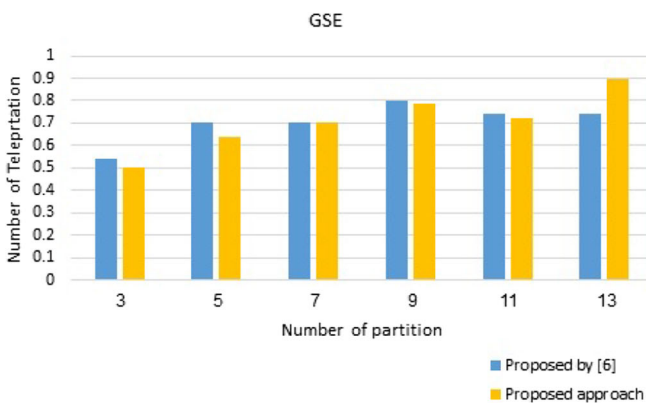
k	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
	{1}	{2}	{2,1}	{3}	{3,1}	{3,2}	{3,2,1}	{4}	{4,1}	{4,2}	{4,2,1}	{4,3}	{4,3,1}	{4,3,2}	{4,3,2,1}
3	N.A	N.A	N.A	N.A	N.A	N.A	N.A	N.A	N.A	N.A	N.A	N.A	N.A	N.A	2
2	N.A	N.A	1	N.A	1	2	2	N.A	0	0	0	3	1	2	2
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0



(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 (R) between number of teleportations (halves) and number of qubits for $K = 3, 5, \dots, 13$ for QFT, BWT and GSE circuits in comparison with [6]

Circuit	$K=3$		$K=5$		$K=7$		$K=9$		$K=11$		$K=13$	
	R	R [6]	R	R [6]	R	R [6]	R	R [6]	R	R [6]	R	R [6]
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

Table 3 Comparison of the proposed approach (P) with [1] and [44]

Circuit	# of qubits	# of gates	K	TC [44]	TC [1]	TC (P)
parity_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

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 distribution has not act well compared to $R < 1$.

The approach of [6] consists of additional pre-processing and post-processing phases: transforming the input circuit to substitute one using only *Clifford + T* gates and rearrangement of CNOTs and single-qubit gates and pulling all CNOTs gates consecutive as possible. These pre- and post-processing phases cause a work overload for the distribution of circuit. Our approach distributes the input circuit without any pre- or post-processing, and our dynamic programming solver can guarantee optimal solutions versus their heuristic approach. Moreover, our algorithm uses one teleportation for each global gate, but the authors of [6] have considered consecutive non-local CNOT gates with common qubits. Therefore, the ratio between the number of teleportation and the number of global gate reduces to less than one in method of [6].

Figure 8 shows the value R for various number of partitions in comparison with the study presented in [6] for three quantum circuits: BWT (Fig. 8a), QFT (Fig. 8b) and GSE (Fig. 8c) circuits. In comparison with [6], the parameter R is better except for GSE circuit which did not distribute well for $K = 13$. Also, the 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 results by the method presented in [6] and required several qubits for communication. Moreover, for $K \geq 5$, 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.

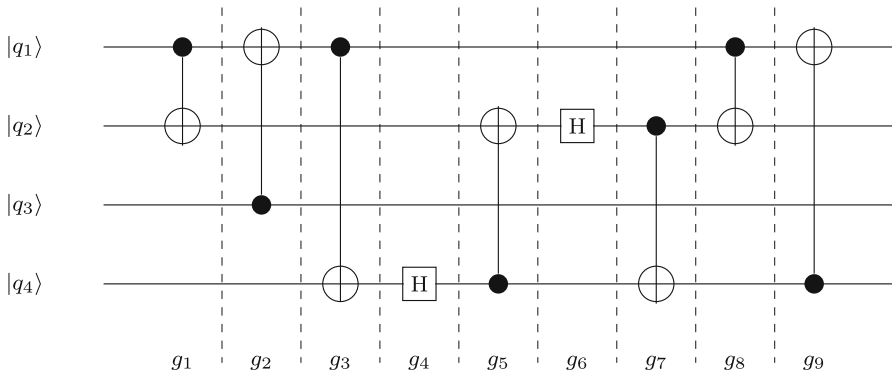


Fig. 9 Sample quantum circuit reproduced from [1]

Table 4 The steps of the proposed algorithm for the random circuit of [1]

Gate number	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

L and G stand for local and global gates, respectively

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 is given for each sample. We compared the results of the proposed approach with two methods presented in [1] and [44] in terms of the teleportation cost (TC).

Let us consider Figure 4 of [1] represented in Fig. 9. In [1], the 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 the other global gates are executed in p_0 . In our model, $\{q_1, q_2, q_4\}$ 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.

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], [6] and [44], it was shown that the proposed approach yielded the better or the same results for benchmark circuits.

References

1. Zomorodi-Moghadam, M., Houshmand, M., Houshmandi, M.: Optimizing teleportation cost in distributed quantum circuits. *Theor. Phys.* **57**(3), 848–861 (2018)
2. Van Meter, R., Ladd, T.D., Fowler, A.G., Yamamoto, Y.: Distributed quantum computation architecture using semiconductor nanophotonics. *Int. J. Quantum Inf.* **8**, 295–323 (2010)
3. Krojanski, H.G., Suter, D.: Scaling of decoherence in wide NMR quantum registers. *Phys. Rev. Lett.* **93**(9), 090501 (2004)
4. Nickerson, N.H., Li, Y., Benjamin, S.C.: Topological quantum computing with a very noisy network and local error rates approaching one percent. *Nat. Commun.* **4**, 1756 (2013)
5. Cuomo, D., Caleffi, M., Cacciapuoti, A.S.: Towards a distributed quantum computing ecosystem. arXiv preprint [arXiv:2002.11808](https://arxiv.org/abs/2002.11808) (2020)
6. Andres-Martinez, P.: Automated distribution of quantum circuits. *Theoret. Comput. Sci.* **410**(26), 2489–2510 (2018)
7. Van Meter, R., Oskin, M.: Architectural implications of quantum computing technologies. *ACM J. Emerg. Technol. Comput. Syst. JETC* **2**, 2006 (2006)
8. Meter, V., Munro, W., Nemoto, K., Itoh, K.M.: Arithmetic on a distributed-memory quantum multi-computer. *ACM J. Emerg. Technol. Comput. Syst. JETC* **3**, 2 (2008)
9. Bennett, C.H., Brassard, G., Crepeau, C., Jozsa, R., Peres, A., Wootters, W.K.: Teleporting an unknown quantum state via dual classical and Einstein–Podolsky–Rosen channels. *Phys. Rev. Lett.* **70**, 1895 (1993)
10. Whitney, M., Isailovic, N., Patel, Y., Kubiatiowicz, J.: Automated generation of layout and control for quantum circuits. *Phys. Rev. Lett.* **85**(26), 1330 (2000)
11. Nielsen, M.A., Chuang, I.L.: *Quantum Computation and Quantum Information*, 10 anniversary edn. Cambridge University Press, Cambridge (2010)
12. Wootters, W.K., Zurek, W.H.: A single quantum cannot be cloned. *Nature* **299**, 802–803 (1982)
13. Farhi, E., Goldstone, J., Gutmann, S., Sipser, M.: Quantum computation by adiabatic evolution. arXiv preprint [arXiv:quant-ph/0001106](https://arxiv.org/abs/quant-ph/0001106) (2000)
14. Zomorodi-Moghadam, M., Taherkhani, M.-A., Navi, K.: Synthesis and optimization by quantum circuit description language. In: *Transactions on Computational Science XXIV*, pp. 74–91. Springer (2014)
15. Deutsch, D.E.: Quantum computational networks. *Proc. R. Soc. Lond. A Math. Phys. Sci.* **425**(1868), 73–90 (1989)
16. Weinstein, Y.S., Buchbinder, S.D.: Steane code single qubit Clifford gates. *J. Mod. Opt.* **61**(1), 49–52 (2014)
17. Zomorodi-Moghadam, M., Navi, K.: Rotation-based design and synthesis of quantum circuits. *J. Circuits Syst. Comput.* **25**(12), 1650152 (2016)
18. Pham, P., Svore, K.M.: A 2D nearest-neighbor quantum architecture for factoring in polylogarithmic depth. *Quantum Inf. Comput.* **13**(11–12), 937–962 (2013)
19. Grover, L.K.: Quantum telecomputation. arXiv preprint [arXiv:quant-ph/9704012](https://arxiv.org/abs/quant-ph/9704012) (1997)
20. Cleve, R., Buhrman, H.: Substituting quantum entanglement for communication. *Phys. Rev. A* **56**, 1201 (1997)

21. Cirac, J., Ekert, A., Huelga, S., Macchiavello, C.: Distributed quantum computation over noisy channels. *Phys. Rev. A* **59**, 4249 (1999)
22. Van Meter, R., Devitt, S.J.: The path to scalable distributed quantum computing. *Computer* **49**(9), 31–42 (2016)
23. Yepez, Jeffrey: Type-II quantum computers. *Int. J. Mod. Phys. C* **12**(09), 1273–1284 (2001)
24. Yimsiriwattana, A., Lomonaco, S.J., Jr.: Distributed quantum computing: a distributed Shor algorithm. arXiv preprint [arXiv:quant-ph/0403146](https://arxiv.org/abs/quant-ph/0403146) (2004)
25. Ying, M., Feng, Y.: An algebraic language for distributed quantum computing. *IEEE Trans. Comput.* **58**(6), 728–743 (2009)
26. Beals, R., Brierley, S., Gray, O., Harrow, A.W., Kutin, S., Linden, N., Shepherd, D., Stather, M.: Efficient distributed quantum computing. *Proc. R. Soc. A Math. Phys. Eng. Sci.* **469**(2153), 20120686 (2013)
27. Streltsov, A., Kampermann, H., Bruß, D.: Quantum cost for sending entanglement. *Phys. Rev. Lett.* **108**, 250501 (2012)
28. Caleffi, M., Cacciapuoti, A.S., Bianchi, G.: Quantum internet: from communication to distributed computing. In: *Proceedings of the 5th ACM International Conference on Nanoscale Computing and Communication*, pp. 1–4 (2018)
29. Cacciapuoti, A.S., Caleffi, M., Tafuri, F., Cataliotti, F.S., Gherardini, S., Bianchi, G.: Quantum internet: networking challenges in distributed quantum computing. *IEEE Netw.* **34**, 137–143 (2019)
30. Neumann, N.M.P., van Houte, R., Attema, T.: Imperfect distributed quantum phase estimation. In: *International Conference on Computational Science*, pp. 605–615. Springer (2020)
31. Andreev, K., Racke, H.: Balanced graph partitioning. *Theory Comput. Syst.* **39**(6), 929–939 (2006)
32. Buluç, A., Meyerhenke, H., Safro, I., Sanders, P., Schulz, C.: Recent advances in graph partitioning. In: *Algorithm Engineering*, pp. 117–158. Springer (2016)
33. Kernighan, B.W., Lin, S.: An efficient heuristic procedure for partitioning graphs. *Bell Syst. Tech. J.* **49**(2), 291–307 (1970)
34. Fiduccia, C.M., Mattheyses, R.M.: A linear-time heuristic for improving network partitions. In: *19th Design Automation Conference*, pp. 175–181. IEEE (1982)
35. Hendrickson, B., Leland, R.W.: A multi-level algorithm for partitioning graphs. *SC* **95**(28), 1–14 (1995)
36. Karypis, G., Kumar, V.: A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM J. Sci. Comput.* **20**(1), 359–392 (1998)
37. Karypis, G., Aggarwal, R., Kumar, V., Shekhar, S.: Multilevel hypergraph partitioning: applications in VLSI domain. *IEEE Trans. Very Large Scale Integr. VLSI Syst.* **7**(1), 69–79 (1999)
38. Zare, H., Shooshtari, P., Gupta, A., Brinkman, R.R.: Data reduction for spectral clustering to analyze high throughput flow cytometry data. *BMC Bioinform.* **11**(1), 403 (2010)
39. Arias-Castro, E., Chen, G., Lerman, G., et al.: Spectral clustering based on local linear approximations. *Electron. J. Stat.* **5**, 1537–1587 (2011)
40. Thulasiraman, K., Swamy, M.N.S.: *Graphs: theory and algorithms*. John Wiley & Sons (2011)
41. Childs, A.M., Cleve, R., Deotto, E., Farhi, E., Gutmann, S., Spielman, D.A.: Exponential algorithmic speedup by a quantum walk. *STOC'03 Proc. Thirty-Fifth Annu. ACM Symp. Theory Comput.* **410**(26), 59–68 (2003)
42. Whitfield, J.D., Biamonte, J., Aspuru-Guzik, A.: Simulation of electronic structure Hamiltonians using quantum computers. *Mol. Phys.* **109**(5), 735–750 (2011)
43. Wille, R., Grobe, D., Teuber, L., Dueck, G.W., Drechsler, R.: Revlib: an online resource for reversible functions and reversible circuits. In: *IEEE International Symposium on Multiple-Valued Logic*, pp. 220–225 (2008)
44. Houshmand, M., et al.: An evolutionary approach to optimizing teleportation cost in distributed quantum computation. *Int. J. Theor. Phys.* **59**(4), 1315–1329 (2020)