International Journal of Theoretical Physics (2020) 59:3804–3820
https://doi.org/10.1007/s10773-020-04633-8

Optimized Quantum Circuit Partitioning

Omid Daei[1] · Keivan Navi[2] · Mariam Zomorodi-Moghadam[3]

Received: 13 March 2020 / Accepted: 19 October 2020 / Published online: 6 November 2020

Abstract
The main objective of this paper is to improve the communication cost in distributed quantum circuits. To this end, we present a method for generating distributed quantum circuits from monolithic quantum circuits in such a way that communication between partitions of a distributed quantum circuit is minimized. Thus, the communication between distributed components is performed at a lower cost. Compared to the existing works, our approach can effectively map a quantum circuit into an appropriate number of distributed components. Since teleportation is usually the protocol used to connect components in a distributed quantum circuit, our approach ultimately reduces the number of teleportations. The results of applying our approach to the benchmark quantum circuits determine its effectiveness and show that partitioning is a necessary step in constructing distributed quantum circuit.

Keywords Distributed · Quantum circuits · Teleportation · Communication cost

1 Introduction

In 1994, Peter Shor developed an algorithm for quantum processing that solved integer factorization problem by using quantum computing in polynomial time [1]. Shor's algorithm showed that quantum computers can solve this specific problem exponentially faster than the best-known classical algorithms.

✉ Keivan Navi
navi@sbu.ac.ir

Omid Daei
o.daei@qiau.ac.ir

Mariam Zomorodi-Moghadam
m_zomorodi@um.ac.ir

[1] Faculty of Computer and Information Technology Engineering, Qazvin Branch, Islamic Azad University, Qazvin, Iran

[2] Nanotechnology & Quantum Computing Lab and the Department of Computer Engineering and Science, Shahid Beheshti University, G. C, Tehran, Iran

[3] Department of Computer Engineering, Ferdowsi University of Mashhad, Mashhad, Iran

Quantum computers outperform their classical counterparts in solving certain problems such as search in database [2], discrete logarithm finding [1], and integer factorization [1] .

Quantum computing has many advantages over classical one; however, the realization of a quantum system on a very large scale is a serious challenge [3]. Experimental teams are developing powerful quantum processors to run small quantum algorithms [4], but it is too hard to make an integrated quantum system with a large number of qubits [5, 6]. Today's technology implementing quantum computing in the real world considers limitations on the number of qubits able to be processed [7] and it is hard to produce an integrated large-scale quantum computer [8] .This makes the distributed implementation of quantum systems necessary [9].

Interaction of qubits with the environment is one of the limitations of implementing quantum systems [10]. While the number of qubits is increased, the quantum information becomes more sensitive to errors and the interaction of qubits results in decoherence [10].

Error correction codes also create much overhead in the system, and a large number of qubits should be involved in computing. In addition, there is a possibility that it could not fit into a quantum chip [9].

Due to the above mentioned restrictions, one of the rational ways of making great quantum systems is to use distributed nodes where fewer qubits will be placed in each node or subsystem. Therefore, to build a large quantum computer, it is necessary to make a network of limited capacity quantum computers connected together through a classical or quantum channel and they all interact with and simulate the behavior of a large quantum computer [11]. This scheme is known as distributed quantum computer.

To model distributed quantum circuits, we can implement the model of monolithic quantum circuits. Each subsystem sends out data on demand to other ones through the communication channel created between the subsystems. To ensure that subsystems can communicate with each other in a distributed quantum system, there must be a reliable communication mechanism between the nodes of the distributed quantum system.

The realization of quantum communications can be regarded as a serious challenge [12]. One of the primitive communication protocols is teleportation [13] in which some quantum technologies such as NMR [14] and trapped ions [15] have implemented teleportation [16].

Van Meter *et.al*, [17] describe two different interconnect topologies known as teledata and telegate. In telegate approach, a teleported gate is applied on the qubits without transferring them. In teledata, information is teleported to another subsystem without physically moving qubit states and it is used to perform computations. In a VBE carry-ripple adder circuit, he has shown that the performance of teledata approach is better than of telegate by a factor of 3.5. Basically, since teleportation is a costly operation in distributed quantum systems, it is critically important to reduce the number of teleportations in a distributed quantum system. On the other hand, based on the no-cloning theorem, when a qubit is teleported to another node, it cannot be used in the source node [18]. Regarding this issue, a method is proposed in this study to algorithmically distribute the qubits in nodes to reduce the total number of teleportations needed.

The paper is arranged as follows: in Section 2, some of the essential concepts of quantum computing and distributed quantum computing are described. Related work is presented in Section 3. Our proposed approach to design a DQC with an improved number of teleportations is described in Section 4. The results are specified in Section 5 and finally, we conclude the paper with some suggestions for the future work in Section 6.

2 Background

In quantum computers, instead of bits, there are quantum bits or qubits. A bit can store zero or one, but a qubit can take zero, one or any combination of zero and one at the same time that is called superposition. The basic states of a quantum system are displayed as $\{|0\rangle,|1\rangle\}$. Based on the quantum principles, a quantum system can be not only in its basic state, but also in a linear combination of basic states. The general state of a qubit is represented as $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$, where $|\psi\rangle$ stands for a quantum state and α and β stand for complex numbers so that $|\alpha|^2 + |\beta|^2 = 1$. In the classical circuits, gates act on bits and produce the desired output. Also, in the quantum circuits, quantum gates have the same effect on qubits.

Quantum Gate A quantum gate is similar to a classic gate. When applied, a quantum gate changes the state of one or more qubits. An n-qubit quantum gate U is defined by a $2^n \times 2^n$ matrix and by performing U on a quantum state $|\psi\rangle$, the outcome is another state represented by the vector $U|\psi\rangle$.

Pauli, Hadamard and Rotation are some commonly used single-qubit gates [16, 19]. If U is a gate that operates on a single qubit, then controlled-U is a gate that operates on two qubits, working as follows: U is applied to the target if the control qubit is $|1\rangle$ and leaves it unchanged otherwise. For example, Controlled-NOT (CNOT) is a two qubit gate. Target qubit changes if control qubit is $|1\rangle$ and unchanged otherwise. A quantum circuit (QC) consists of quantum gates interconnected by quantum wires. A quantum wire is a mechanism for moving quantum data from one location to another [20].

For the convenience of quantum computing, it is better to model them. There are several models to evaluate quantum computations; For instance, adiabatic model of computation [21] and quantum programming languages [22]. One of the preferred methods for displaying quantum computations is the circuit model that is based on unitary evolution of qubits by networks of gates [16].

Figure 1 shows a simple quantum system in the circuit model that represents a quantum full-adder.

Distributed Quantum Circuit A distributed quantum circuit (DQC) is an extension to the quantum circuit model. A DQC consists of n smaller QCs and located far away from each other, each possesses a limited capacity. All parts together implement the functionality of a QC. The different parts of a DQC should communicate with each other sending their qubits to each other by a quantum channel via teleportation.

Local and Global Gates In a DQC system, a local gate is a gate whose all qubits reside in the same partition. On the other hand, in a global gate, all qubits are located in different partitions.

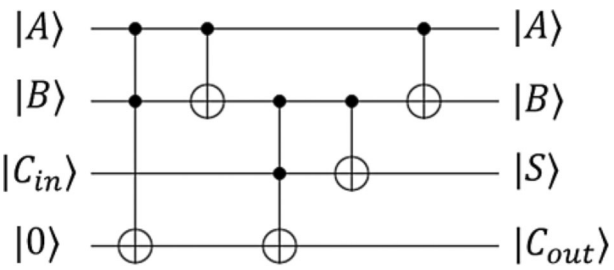

Fig. 1 Circuit model of a quantum full-adder

Therefore, to run this gate, the qubits' information must be brought from the partitions in which they are located to the current partition where that gate runs.

Teleportation sends the state of a qubit from a point to another just by communicating two classical bits. Two points need sharing a maximally entangled state [23]. QC for quantum teleportation is shown in Fig. 2 [16]. The first and second lines denote the sender's system and the third line is the receiver's system. Meters determine the measurement and output of the measurement carries classical bits. Teleporting a quantum state works in any situation [13, 8], so it can be used as a communication protocol for interconnecting partitions of a DQC.

One of the advantages of using graph theory is to provide the same form for many problems. Formally, a graph is a pair of sets(V, E), where V is the set of vertices and E is the set of edges, formed by pairs of vertices. In mathematics, a graph partitioning comprises the division of its nodes into mutually exclusive sets. The edges that connect the sets make the connection between those sets. Usually, graph partitioning problems are classified as NP-hard problems. So the solutions that have come up for such issues use heuristics and approximation algorithms, in general [24].

Suppose a graph $G = (V, E)$, where V and E represent a set of n vertices and a set of edges, respectively. A balanced partition problem divides G into K parts at a similar size so that the weight of the edges between separated components must be minimum [25]. Local and global algorithms are two methods for this purpose.

Two well-known local methods for graph partitioning are the Kernighan–Lin and Fiduccia-Mattheyses algorithms. They have local search for 2-way cuts. An effective drawback of these two algorithms that can affect the quality of the final result is the arbitrary initialization of the vertex set, but global approaches do not require an arbitrary initial partition and depend on the properties of the entire graph. Spectral partitioning is the most widely used method.

Multi-level graph partitioning algorithms perform partitioning using one or more stages. In each step, the size of the graph is reduced by collapsing the edges and vertices, the smaller graph is partitioned and then this part of the main graph is mapped back and refined [26]. In multi-level scheme, a wide range of partitioning methods can be used. In some problems, this method can lead to fast execution time and extremely desirable results. In this paper, Kernighan–Lin is extended to a multi-level scheme and this altered algorithm is used for QC partitioning.

2.1 Kernighan-Lin Graph Partitioning Algorithm

The Kernighan–Lin (K-L) [27] algorithm is a heuristic algorithm for graph partitioning. The input to the algorithm is an undirected weighted graph $G = (V, E, W)$ where V is the set of

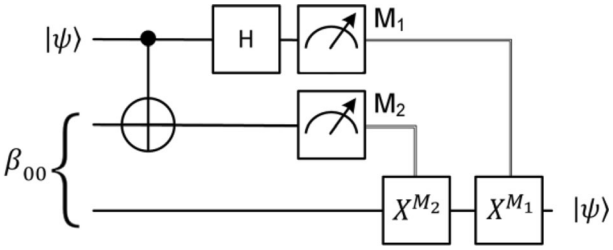

Fig. 2 Quantum circuit teleporting a qubit [16]

vertices, E is the set of edges, and W is the set of weights assigned to edges. The K-L algorithm partitions V into two subsets A and B. Partitioning is done in a way that minimizes the sum of the weights of the edges that are crossing from A to B. If the edges are without weights, we assume the weights of the edges as one.

At first, the vertices are randomly placed in partitions. Then, we have to calculate the two parameters: first, the cost of communication (weight of edges) of each node with the other nodes in its partition, known as internal cost. Second, the cost of the communication of each node with other partitions, known as external cost. These two parameters are represented by I and E respectively. Suppose that the difference between these two parameters is shown with D ($D = E - I$). The values of D for all nodes in the circuit are calculated and the gain between the two nodes x and y that are divided into two sections A and B, is calculated as $g_{xy} = D_x + D_y - 2C_{xy}$. Since we are calculating the cost of moving two nodes x and y, so the cost between them should be uncalculated and will be reduced by $2C_{xy}$. The gain value is computed for each of the two nodes and the largest number that is called $\widehat{g}1$ and indicates the relationship between the two vertices is selected. Next, these two vertices are locked, and they are deleted from the set A and B. Now the values of D are updated and the calculations are performed for the remaining nodes. This process continues to the remainder of the vertices until there are no more vertices in the two sets. At each step, $\widehat{g}2$, $\widehat{g}3$ and etc. are identified.

The parameter k represents the number of each stage of execution and $G_k = max\sum_{i=1}^{k}\widehat{g}_i$ is calculated in every stage. The largest value specifies which two nodes must be exchanged. In the first run of the algorithm, these two nodes will be exchanged, and all locked nodes will be unlocked for the next round. This algorithm will continue while G_k>0. Pseudo code for the K-L is shown in Algorithm 1.

Algorithm1 Pseudo code for Kernighan-Lin [27]

```
function Kernighan-Lin (G (V, E)):
 Distribute the nodes balances in sets a and b
  while (Gk > 0) {
    for all a in A and b in B compute D values
    the lists gvect, avect and bvect are erased
    for (n: = 1 to |V| / 2)
      find a from A and b from B, such that g = D [a] + D [b] - 2*c (a, b) will be maximum
       remove a and b from further consideration in this pass
       add g to gvect, a to avect, and b to bvect
       update D values for the elements of A = A / a and B = B / b
   end for
   find k in such a way that maximizes Gk, the sum of gvect [1]..., gvect [k]
    if (Gk > 0) then
      Exchange avect [1], avect [2]..., avect[k] with bvect [1], bvect [2] ..., bvect [k]
  }//while
return G (V, E)
```

The K-L algorithm can be expanded to divide each partition into smaller partitions so that the communication between those partitions is optimized.

2.2 Graph Representation of a Quantum Circuit

A weighted graph is usually defined as $G = (V, E, W)$, where V is the set of vertices $V = \{v_1, v_2 \ldots, v_n\}$, E is the set of edges e_1, e_2, …, e_k and W is the set of edge weights represented as $w(e_1)$, $w(e_2)$, …, $w(e_k)$. To transform a monolithic quantum circuit into a graph model, we must first represent the qubits and the connections between them in terms of vertices and edges in a graph model.

Suppose $q_1, q_2, \ldots, q_n$ are qubits of a QC, where n is the number of qubits in the monolithic representation of the circuit. Qubits are denoted by vertices in a graph, so the associated graph has n vertices. The set V of graph is $V = \{q_1, q_2, \ldots, q_n\}$. Each gate in a QC is composed of one or more qubits and qubits of QC are connected to each other through quantum gates. Edges of the graph can represent such a relationship. If two qubits are related to each other through a gate, then there is an edge between their corresponding vertices in the graph, and otherwise, the vertices are disjoint. Also, the one-qubit quantum gates require only one qubit for execution, so they do not add any edges in the graph model. However, for quantum gates with more than one qubit, their data must be brought to the point where the quantum gate is going to be executed. For instance, suppose control and target qubits of a CNOT gate are $q_1$and q_2. To perform this gate, $q_1$should be brought to the partition where q_2 is located or vice versa. Therefore, a connection is needed for this transfer which we show it with an edge between q_1 and q_2 with initial weight of one in the graph model. If both q_1 and q_2 are needed to run another gate, the weight of the edge is increased one unit. This process is repeated for all quantum gates to construct all the interconnections needed to run the gates in the graph model. Regarding all the interpretations, to convert a QC to a graph model, an undirected weighted graph is used. Figure 3 represents a sample circuit and its corresponding undirected weighted graph.

3 Related Work

There are serious challenges to design and manufacture devices that are not based on CMOS technology and these devices are going to implement quantum computing [28]. Therefore, we will face very serious problems to realize a quantum computer. One of these problems is due to technology constraints. We cannot use a large number of qubits to fabricate a single quantum device [29] and the greatest reason for the emergence of DQC is the existence of these constraints. Research on distributed quantum computing has been ongoing for nearly two decades [8]. Cleve and Buhrman [30] and Grover [31] were people who began to research on distributed quantum computing. After that, Cirac et al. [32] also published research in this

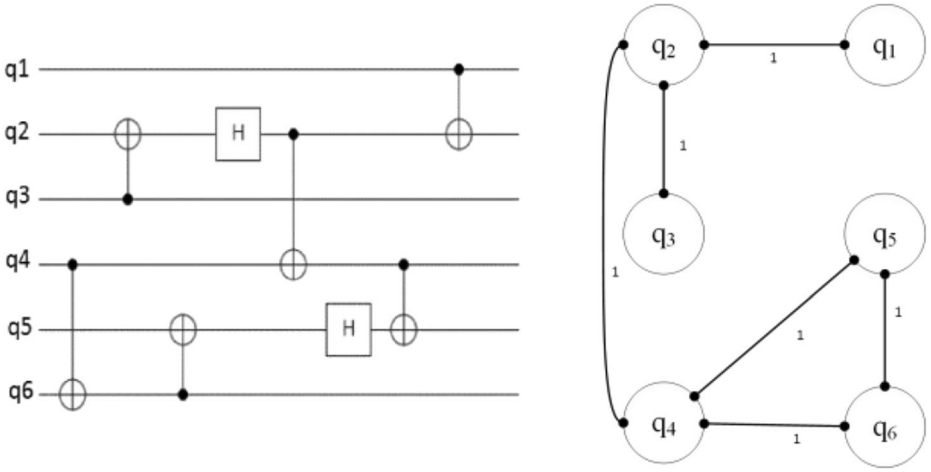

Fig. 3 Sample quantum circuit and corresponding weighted graph

field. In [31], Grover stated that to solve an issue in a distributed quantum system, the important question is how to split the problem into different quantum units in an optimal way. He divided a quantum system into separate parts, each of which was separated and far from each other, and each section performed its computations separately. Every section requiring information from another part received it. Grover showed that the overall calculation time was reduced by the use of this distribution approach. He also introduced a quantum algorithm and matched it with distributed computation although it possessed very costly communications.

In [30], Cleve and Buhrman have studied quantum communications. They have shown that quantum entanglement can be employed as a substitute for communication when we want to compute a function whose input data are distributed among remote locations. Also, Cirac et al., [32] showed that the use of maximally entangled states was advantageous over uncorrelated ones for the ideal quantum channels and a sufficiently large number of nodes. Beals et al., [33] introduced a distributed quantum system in which each part was located on the nodes of a hypercube graph. They used a large number of qubits to apply a logical overhead and showed that each QC could be converted into a DQC so that each part was at the vertex of a hypercube.

The author in [34] has presented an architecture for distributed quantum computing in two communication methods. Quantum communications between partitions are used in one of the methods and classical communications are used in the other. In the former, each qubit could be entangled to any number of qubits. In the latter, small quantum systems are connected to each other by a network of classical communication channels.

The authors in [8] specified a language to illustrate QCs for distributed quantum computations. Also, some definitions were presented for distributed quantum systems.

Connecting quantum computers via quantum internet to achieve faster processing speed is discussed in [35]. The challenges of designing quantum internet have also been studied in this work. The authors also discussed the creation of quantum internet in another article [36] and considered teleportation as the main strategy for the transmission of information. Then, they explored the challenges and open issues in the design of quantum internet.

Hoi-Kwong Lo in [37] examined the cost of classical communication. The authors stated that, given the two-step teleportation, it takes a bit of classic communication at each step and $2log_2N$ bits of classical communications is needed for remote preparation of a desired N-dimensional state which it is not the case in a normal teleportation.

The authors in [17] presented a fixed distributed quantum circuit of a monolithic quantum circuit. The circuit is a 2-qubit VBE carry-ripple adder and has been distributed in two nodes. It is divided into two equal parts, and each part is located in a node. Teleportation is used for the communication between these two nodes, illustrating in the Figs. 4 and 5. They calculated the cost of their teleportation design and compared two types of teleportations named teledata and telegate. Finally, they concluded that teledata is better than telegate. The authors in [28] compared their circuit with an integrated design and indicated that when the node size rises, performance increases in large problems. Their architecture was fixed and they did not consider such issues as teleportation cost reduction methods. They have not especially made any effort to reduce the number of teleportations in their design resulting in higher costs in large quantum circuits.

A distributed model of Shor's algorithm has been presented in [38]. Teleportation has been used as the communication protocol in this design. It uses global gates to implement the distributed quantum circuit for shor's algorithm. But no attempt has been made to determine the best location of qubits among the distributed nodes. So their method is not systematic as

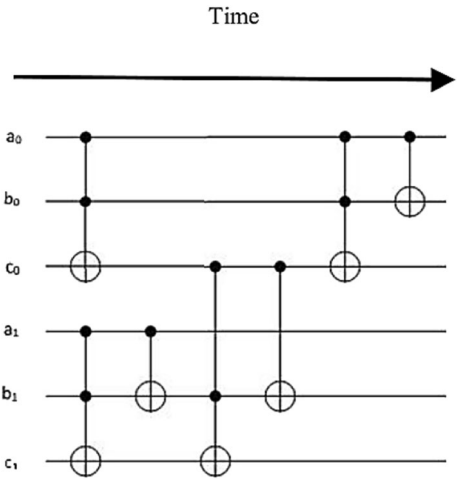

Fig. 4 Details of a 2-qubit VBE adder in the monolithic form [17]

the one in [39]. Their method is fixed such that each register is divided into four parts and qubits of each part are hold in one computing node and output of each node is teleported to the next one. The authors did not try to optimize the number of teleportations. Also, they did not calculate the cost of returning the qubits back to their original part teleported.

In [40], the authors have examined the cost of quantum communication for DQCs by having just two partitions. They have computed the quantum cost of distributed components by providing an algorithm searching for the best execution order of two-qubit CNOT global gates to reduce the total number of teleportations between the two partitions by examining all the possible ordering scenarios to perform global gates. Finally, the best execution order leading to the least amount of teleportation have been explored, but the authors did not provide a solution how to distribute qubits in different partitions to the minimum total cost needed. Also their algorithm is limited to just two partitions and so that it is not clear what problems are arisen when multiple partitions are considered. The authors improved their approach by using an evolutionary algorithm in [41]. Genetic algorithm has been used to find the best configuration to execute gates in DQCs with two partitions. The algorithm takes

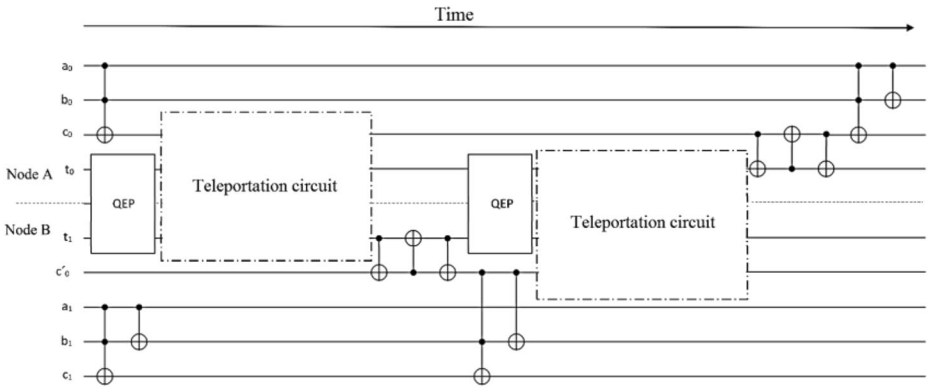

Fig. 5 Details of a distributed 2-qubit VBE adder using the teledata method [17]

the teleportation cost as the cost function of the genetic algorithm and tries to assign the best partition for the execution of each global gate.

The authors in [42] presented an automated way to distribute monolithic quantum circuits into multiple agents. They transform the input circuit into an equivalent one consisting of only Clifford+T gates and then swap the order between CNOTs and the 1-qubit gates from Clifford+T, pulling all CNOTs as early in the circuit as possible. This brings CNOTs closer together. But the authors have not considered the problem of monolithic quantum circuit partitioning with the purpose of reducing the number of communications. Also, they have not discussed the general arrangement of the global gates in different partitions. They only discussed global gates bearing special conditions.

4 Proposed Approach

In this section, we first describe the proposed method for the modeling of QCs to get a DQC, and ultimately, at the end of this section, we describe the method in practice by applying it to a sample QC. Since teleportation in DQCs is a costly operation, it is worth reducing it as much as possible. Our approach is based on decreasing the number of global gates in DQC.

First, we model the monolithic quantum circuit with an undirected weighted graph. Without loss of generality, it is assumed that the circuit is composed of one, two or three qubit gates.

Assuming a QC as input, it contains n qubits $\{q_1, q_2, \ldots, q_n\}$, and m gates. Our gate library is a set $\mathcal{G}$ consists of one, two, and three qubit gates. The QC has a set $G = \{g_i \in \mathcal{G}; i : 1, \ldots, m\}$

$$\left(G = \{g_i(j,k,l) \; or \; g_i(j,k) \; or \; g_i(j) \in \mathcal{G}; i : 1, \ldots, m; j,k,l : 1, \ldots, n\} \right)$$

of gates.

The number of qubits in the QC determines the number of nodes in the graph model. Thus, the set of vertices for n qubits is $V = \{q_1, q_2 \ldots q_n\}$ in the graph model.

To create an undirected weighted graph model, the connections between qubits in the gates are considered as the edges set E. Considering three different types of gates, E is defined as follows:

- For single-qubit gates: no edge is defined for them.
- For two-qubit gates:

$$E_i = \left\{ \left(q_j, q_k \right) \middle| g_i(j,k) \epsilon \; QC; j,k : 1, \ldots, n \right\}$$

- For three-qubit gates: here, we have three edges for each of the connections between qubits. We define them as follows:

$$E_{i1} = \left\{ \left(q_j, q_k \right) | g_i(j,k,l) \epsilon QC; j,k : 1, \ldots, n \right\}$$

$$E_{i2} = \left\{ \left(q_j, q_l \right) | g_i(j,k,l) \epsilon QC; j,k : 1, \ldots, n \right\}$$

$$E_{i3} = \{ (q_k, q_l) | g_i(j,k,l) \epsilon QC; j,k : 1, \ldots, n \}$$

As an example, if there is a two-qubit gate $g_i(1, 2)$ requiring two qubits q_1 and q_2 to be executed, we represent the edge between q_1 and q_2 as $e_i = (q_1, q_2)$.

Starting from the first gate in the QC, the edge set is constructed. The weight of each edge is one when it is created. If there is more than one gate connecting two qubits in the QC, the weight of the corresponding edge is increased accordingly $w(e_i) = w(e_i) + 1$. Finally, we will obtain an undirected weighted graph in which the weight of each edge represents the number of gates requiring these two qubits for execution.

Since communications between partitions in a DQC are performed by teleportation, the number of these communications should be optimized. It is clear that we do not need to use teleportation for qubit interactions within a partition because the qubits are locally available. Therefore, vertices must be arranged in different partitions so that it leads to the least number of communications between the partitions to run the DQC. Now, with a graph model of the QC, we can partition it into arbitrary parts using graph partitioning algorithms which it considers the above issues. Given the implementation constraints and considering that the number of partitions affects the overall performance of the given method and this value varies for different circuits, we can calculate the values for the different partitions and finally select the desired partitions.

In order to have a balanced partitioned quantum circuit, each partition of the circuit must have almost the same number of qubits. Since the nodes in the graph represent qubits, the partitioning algorithm should distribute the nodes almost evenly in the partitions. Various algorithms can be used to achieve it. We employed the Kernighan-Lin (K-L) algorithm used in the VLSI design algorithms [27].

After applying the proper algorithm to the graph model, we will have K partitions $\{P_1, P_2 \ldots P_K\}$ in which there are a number of nodes and edges per partition and each partition may be connected to several other partitions through some edges.

Since graph partitioning algorithms attempt to partition a graph in order to optimize communications between partitions, it includes the least communication outside the partitions after applying the algorithm to the graph. This will typically result in the optimal number of global gates required for the efficient implementation of the DQC.

To construct the DQC from the monolithic circuit, each part of the graph must be converted into an equivalent circuit. Each partition P_i of the graph represents a part of the DQC and the vertices in each one represent qubits in that partition. The communication edges between two vertices inside a partition indicate a local gate. In addition, the most important part is the communications between the qubits not existing just within a partition. The edges connecting the partitions represent global gates whose qubits are in more than one partition. It means that $w(e_{k1})$ global gates uses q_i and q_j qubits if there is an edge e_{k1} from q_i of partition P_x to q_j of partition P_y, and the weight of this edge shown by $w(e_{k1})$. Sum of the weights of these edges among all partitions $W(E_K) = w(e_{k1}) + w(e_{k2}) + \ldots$ will eventually determine the number of global gates that is necessary to create DQC.

Once the algorithm is executed, DQC is achieved by improved communication that reduces the number of global gates. If the nodes are logically distributed in the partitions, the number of global gates will be reduced that will ultimately require fewer teleportations to perform the circuit. It should be noted that the number of teleportations required to perform the circuit may be less than twice the number of global gates. This is taking into account that by doing a teleport, we will execute the next global gates with the same teleport as much as possible. Then, if it is not possible to perform more global gates, the migrated qubit will be returned to the origin.

The algorithm for the proposed approach is shown in Algorithm 2. First, we create graph-based model. Then, partitioning algorithm based on the K-L is applied to.

a) Initially, n (the number of vertices of related graph) is determined by the number of qubits and K is the number of arbitrary partitions.
b) The weight of all the edges between the vertices is set to zero (lines (4–8)).
c) If q_i and q_j for each gate in QC are necessary to perform, the weight of the edge between two vertices (qubits) is increased as one unit. It creates the adjacency matrix of the graph (lines (9–13)).
d) The graph is partitioned into K partitions by calling the modified Kernighan-Lin function (line 14). It takes the adjacency matrix of the graph and the required number of partitions as inputs and generates the partitioned graph as output. Since the K-L is an algorithm for binary partitioning, we employed the original K-L to use it recursively to partition each part into smaller sections. So, the vertices (qubits) in each partition are specified (lines (15–17)).
e) Finally, all QC gates are placed in the DQC (lines (18–20)).

Algorithm2 Algorithm for obtaining DQC from QC

```
1: Input: monolithic quantum circuit (QC), number of desired partitions (K)
2: Output: low cost distributed quantum circuit (DQC).
3: n= number of qubits of QC
4: for i ← 1 to n do
5:   for j ← 1 to n do
6:       Weight (e_ij) =0
7:   end for
8: end for
9: for m ← 1 to last gate in QC do
10:   if gate (m) needs (qubit (i) and qubit (j)) to run then
11:            weight (e_ij) = weight (e_ij) +1
12:   end if
13: end for
14: Partition (1) to Partition (K)← Kernighan-Lin (weight [1..n] [1..n], K)
15: for i ← 1 to K do
16:   place all qubits related to partition (i) as a section
17: end for
18: for i ← 1 to last gate in QC do
19:   Add gate (i) in DQC
20: end for
```

The proposed method is explained by running on a sample QC shown in Fig. 6. Each line represents a qubit. The connections between the qubits needed to run the gate are shown with a vertical line drawn between the qubits. After applying the proposed method, the corresponding graph model is achieved as shown in Fig. 7.

The QC has eight qubits; therefore, we will have a graph with eight vertices that are labeled one to eight, representing qubits from one to eight. Also, weight of the edges connecting the vertices together shows the number of connections between those qubits on all the quantum gates of the sample QC. Now, bearing a graph model and applying the partitioning algorithm, it can be divided to the desired number of partitions.

The graph is divided into four parts by the K-L partitioning algorithm, as follows: $P1\{q1,q6\}$, $P2\{q2,q5\}$, $P3\{q3,q7\}$, $P4\{q4,q8\}$. Therefore, the QC is divided into four sections bearing specified qubits. The qubits that have the most connections are placed in a partition to reduce the number of global gates. The DQC is shown in Fig. 8.

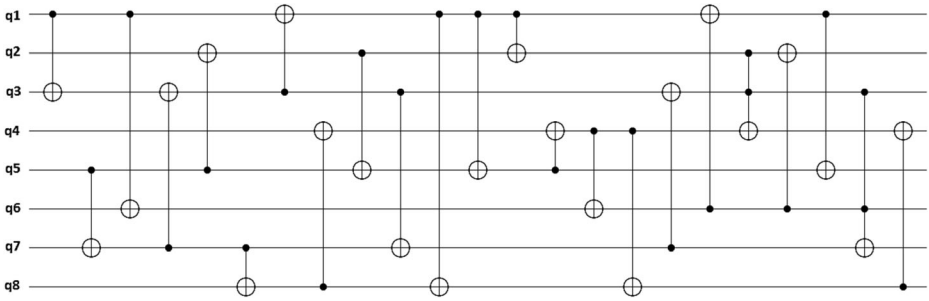

Fig. 6 A sample QC

5 Results

We used MATLAB software to verify the proposed method experimentally. We also used some Revlib [43] library circuits as test circuits. Revlib is an online resource to benchmark quantum circuits and reversible quantum circuits. Another series of tests were conducted on the Quantum Fourier Transform (QFT for short) circuits with n qubits where $n \in \{4, 8, 16\}$ is the number of qubits in the circuit. To implement and check the efficiency of the proposed method, the benchmark circuits must be first decomposed to basic gate library. The synthesis approach described in [44, 45] was used for this purpose. Then, we applied our proposed method to benchmark circuits for $K = 2$, 3 *and* 4 numbers of partitions. As expressed, this partitioning is done in such a way that the least amount of communications between partitions is required and this will lead to fewer global gates in distributed quantum circuits. In order to obtain a more accurate estimate, we implemented a random partitioning method for each of the given circuits. We performed this operation 200 times, and finally we computed the average for each of them. The obtained average was the basis of comparison with our method. We also compared our results with the approach in [41]. The results of the comparison of our approach

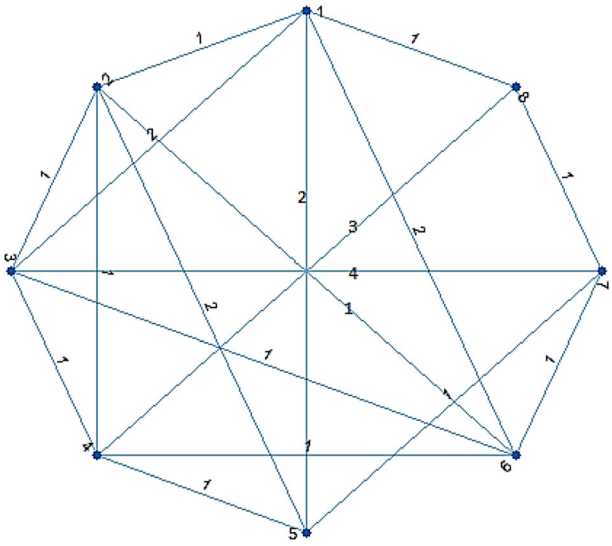

Fig. 7 The graph corresponding to Fig. 6

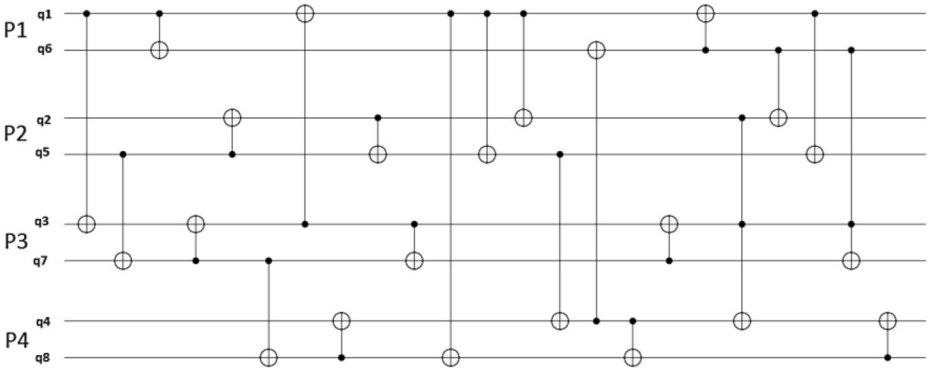

Fig. 8 Sample QC split into 4 partitions

and random partitioning and [41] in terms of the number of global gates and the percentage of teleportation improvement in nine different circuits are presented in Table 1.

Because of a particular pattern in the QFT (n) circuits, almost all the communications between qubits are equal and non-zero. As the number of qubits increases, there is not much difference between the number of global gates in the random partitioning and our approach.

Table 1 Comparison of the proposed approach (P) with the random partitioning (RP) and [41] . The last two column indicate percentage of teleportation cost improvement (TC.imp) of the proposed approach as compared to RP and [41] respectively

Circuit	# of qubits	# of partitions (K)	# of global gates (RP)	# of global gates (P)	TC (RP)	TC [41]	TC (P)	TC. imp (RP) (%)	TC. imp [41] (%)
sym9_147	12	2	58	37	94	48	28	70.21	41.67
	12	3	79	40	N.A.	N.A.	43	N.A.	N.A.
	12	4	88	66	N.A.	N.A.	58	N.A.	N.A.
alu_primitive	6	2	13	8	26	20	12	53.85	40.00
	6	3	16	10	N.A.	N.A.	16	N.A.	N.A.
	6	4	18	13	N.A.	N.A.	24	N.A.	N.A.
alu_primitive_opt	6	2	13	8	18	10	10	44.44	0.00
	6	3	16	10	N.A.	N.A.	14	N.A.	N.A.
	6	4	18	13	N.A.	N.A.	28	N.A.	N.A.
parity_247	17	2	8	8	12	2	2	83.33	0.00
	17	3	11	8	N.A.	N.A.	10	N.A.	N.A.
	17	4	13	12	N.A.	N.A.	16	N.A.	N.A.
Fig. 4 [40]	4	2	5	4	4	4	4	0.00	0.00
	4	3	7	5	N.A.	N.A.	6	N.A.	N.A.
flip_flop	8	2	17	9	8	N.A.	6	25.00	N.A.
	8	3	22	9	14	N.A.	8	42.86	N.A.
	8	4	26	18	50	N.A.	16	68.00	N.A.
4-qubit QFT	4	2	12	8	12	8	6	50.00	25.00
	4	3	15	13	N.A.	N.A.	16	N.A.	N.A.
	4	4	18	18	N.A.	N.A.	20	N.A.	N.A.
8-qubit QFT	8	2	39	32	52	38	16	69.23	57.89
	8	3	51	40	N.A.	N.A.	34	N.A.	N.A.
	8	4	58	48	N.A.	N.A.	48	N.A.	N.A.
16-qubit QFT	16	2	141	128	216	133	42	80.56	68.42
	16	3	187	160	N.A.	N.A.	68	N.A.	N.A.
	16	4	211	192	N.A.	N.A.	102	N.A.	N.A.

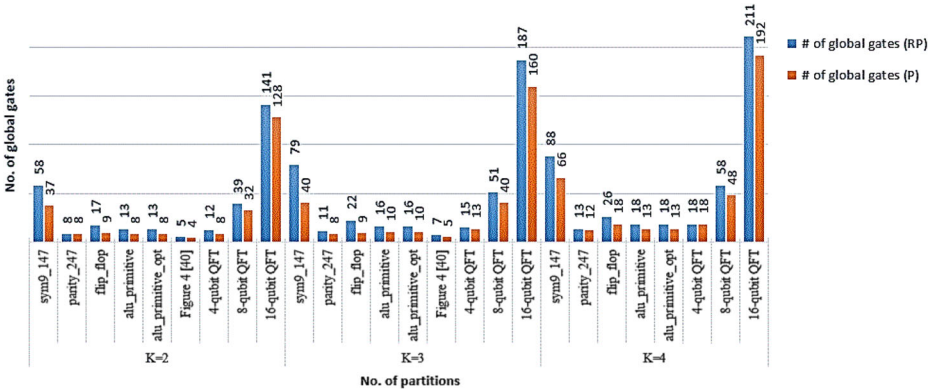

Fig. 9 Number of the global gates in the proposed approach as compared to the random partitioning

However, due to the fact that these circuits have a special pattern, a large number of global gates can be executed one after another by performing a teleport if the arrangement of the gates in the partitions is made correctly. The correct arrangement leads to a significant reduction in the teleportation cost. Also, in circuits with fewer connections between all the qubits, there is a massive improvement in the number of global gates by logically arranging the gates in the partitions. In the circuits with four qubits, such as QFT (4) and $K = 4$ partitions, there is no improvement. That is because the circuit has only four qubits, and by creating four partitions, each partition has a qubit. Thus, there is no difference between our approach and other methods in terms of the number of global gates.

In Fig. 9, the number of the global gates in our approach (P) and random partitioning (RP) is shown for each of the nine circuits of Table 1. The vertical axis represents the number of the global gates and the horizontal axis represents the number of the partitions that are $K = 2$, 3 and 4, for all the circuits.

Since increase in the number of the random partitioning iterations did not indicate any significant changes in the results, we limited the iteration number to 200.

The cost of communication is critically dependent on the order in which the gates are placed in the partitions. The consecutive global gates can be performed with a single teleport if they have a common qubit and are also properly distributed in the partitions. Thus, to

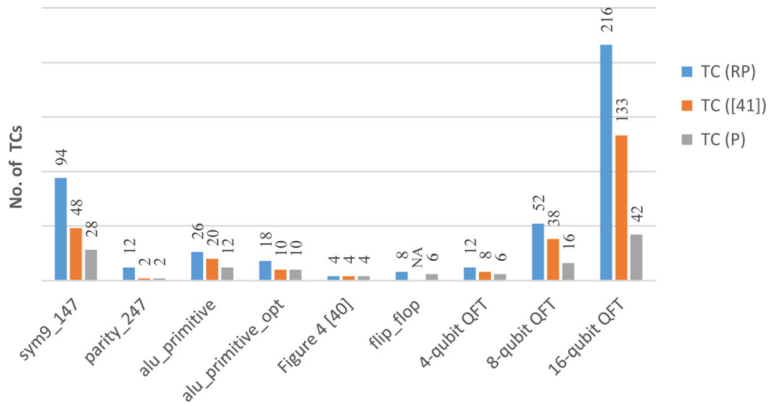

Fig. 10 Teleportation cost of the proposed approach as compared to the random partitioning and [41]

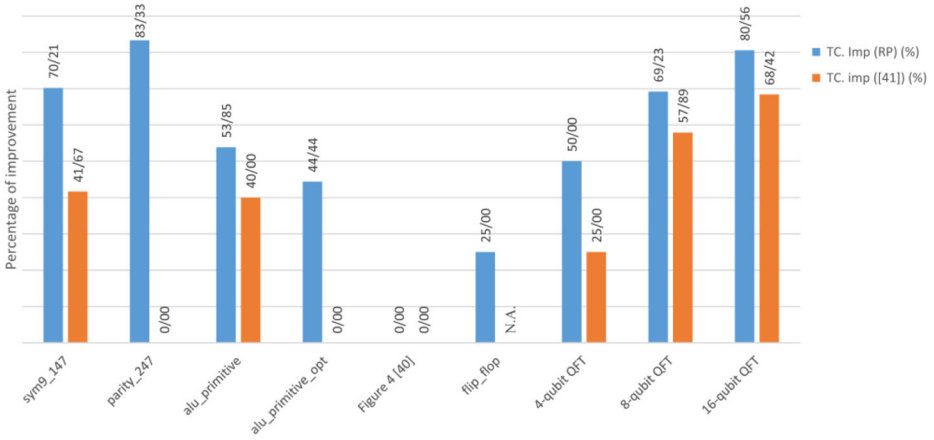

Fig. 11 Percentage of improvement in teleportation cost of the proposed approach (P) compared to the random partitioning (RP) and [41]

determine the total teleportation cost, it is extremely important how these gates are properly arranged in different partitions. Taking this into account, Fig. 10 graphically shows the results of the proposed method to calculate the teleportation cost required to perform each of the nine circuits compared to the random and the proposed method in [41]. Because the results of the compared methods are unavailable in $K = 3$ and 4, Fig. 10 shows the results of $K = 2$ partitions. In Fig. 10, the vertical axis and the horizontal axis represent the teleportation cost and the benchmark circuits, respectively.

The percentage of improvement in the proposed approach (P) relative to the random partitioning (RP) and the proposed method in [41] is illustrated in Fig. 11 for each of the nine benchmark circuits. The vertical axis represents the percentage of improvement in the number of teleportations and the horizontal axis represents the benchmark circuits. Because our approach is to reduce the number of global gates, the number of teleportations is also decreased, leading to reducing the cost of teleportations in DQC.

Using our method and various number of partitions for each of the benchmark circuits compared to the random method, an average of 22% improvement in the number of global gates has been achieved. It is necessary to note that since QFT (4) is meaningless for 4 partitions, its results are not affected in this average. Also, if we calculate the percentage of improvement for benchmark circuits except QFTs, this value is significantly increased to an average of 35%. Furthermore; in the proposed method compared to (RP) and [41], the number of teleportations required to perform benchmark circuits has been reduced by an average of 59% and 29%, respectively.

6 Conclusion and Future Works

In this paper, we presented a method to construct distributed quantum circuits using a graph model of monolithic quantum circuits. Due to the fact that the teleportation in quantum technologies creates an overhead, the number of teleportations should be reduced as much as possible. Our approach to solving this issue is to model the circuit as a graph and to use a graph partitioning method in order to reduce the number of global gates in a DQC that

ultimately reduces the number of required teleportations. Despite the similar existing study, our approach can be used to divide the quantum circuit into any desirable number of distributed systems. We compared our method to the random partitioning method and [41] in order to ensure that our algorithm works well with the sample circuits.

After applying the proposed method to benchmark circuits, it was shown that our approach yielded the better or the same results in some benchmark circuits. It was also determined that if the number of global gates with common qubits in a circuit was higher and they were logically distributed in partitions in terms of their arrangement, a considerable percentage of improvement was achieved. Because QFT circuits have this feature, increasing the number of qubits will result in a higher percentage of improvement. As future work, we aim to extend our algorithm considering the location of the global gate execution as part of the partitioning problem and to investigate its effect on the DQC cost.

References

1. Shor, P.: Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer," 35th Annual Symposium on Foundations of Computer Science, Piscataway, NJ. IEEE Press (1994)
2. Grover, L.K.: A fast quantum mechanical algorithm for database search, In In Proc. 28th annual ACM Symposium on the theory of computation, New York (1996)
3. Steffen, L., Fedorov, A., Oppliger, M., Salathe, Y., Kurpiers, P.E.A.: Realization of deterministic quantum teleportation with solid state qubits. Nature. **500**, 319 (2013)
4. Van Meter, R., Devitt, S.: Local and distributed quantum computation. IEEE Comput. **49**(9), 31–42 (2016)
5. Salonik, R., Ulya, R.K.: Quantum computing: An overview across the system stack., arXiv preprint arXiv: 1905.07240, (2019)
6. Aaronson, S.: The Limits of Quantum. Sci. Am. **298**, 62–69 (2008)
7. Fuji, K., Yamamoto, T., Koashi, M., Imoto, N.: A distributed architecture for scalable quantum computation with realistically noisy devices. Quantum Phys. arXiv:1202.6588v1 (2012)
8. Ying, M., Feng, Y.: An algebraic language for distributed quantum computing. IEEE Trans. Comput. **58**, 728–743 (2009)
9. Van Meter, R., Ladd, T., Fowler, A., Yamamoto, Y.: Distributed quantum computation architecture using semiconductor nanophotonics. Int. J. Quantum Inf. **8**(01n02), 295–323 (2010)
10. Krojanski, H., Suter, D.: Scaling of decoherence in wide nmr quantum registers. Phys. Rev. Lett. **93**(25), 090501 (2004)
11. Nickerson, N., Li, Y., Benjamin, S.: Topological quantum computing with a very noisy network and local error rates approaching one percent. Nat. Commun. **4**, 1756 (2013)
12. Brassard, G., Lutkenhaus, N., Mor, T., Sanders, B.: Limitations on practical quantum cryptography. Phys. Rev. Lett. **1330**, 85 (2000)
13. Bennett, C., Brassard, G., Cr'epeau, C., Jozsa, R., Peres, A., Wootters, W.: Teleporting an unknown quantum state via dual classical and einstein-podolsky-rosen channels. Phys. Rev. Lett. **1895**, 70 (1993)
14. Nielsen, E., Knill, M., Laflamme, R.: Complete quantum teleportation using nuclear magnetic resonance. Nature. **396**, 52–55 (1998)
15. Riebe, M., Haffner, H., Roos, C., Hansel, W., Benhelm, J.E.A.: Deterministic quantum teleportation with atoms. Nat. **429**, 734–737 (2004)
16. Nielsen, M.A., Chuang, I.: Quantum computation and quantum information, ed: American Association of Physics Teachers, (2002)
17. Van Meter, R., Munro, W., Nemoto, K., Itoh, K.: Arithmetic on a distributed-memory quantum multicomputer. ACM J. Emerg. Technol. Comput. Syst. **3**(JETC), 2 (2008)
18. Wootters, W., Zurek, W.: A single quantum cannot be cloned. Nature. **299**, 802–803 (1982)
19. Zomorodi-Moghadam, M., Navi, K.: Rotation-based design and synthesis of quantum circuits. J. Circuits, Syst. Comput. **25**(12), 1650152 (2016)
20. Oskin, M., Chong, F.T., Chuang, I.L., Kubiatowicz, J.: Building Quantum Wires: the Long and the Short of it, In 30th Annual International Symposium on Computer Architecture, San Diego, CA (2003)

21. Farhi, E., Goldstone, J., Gutmann, S., Sipser, M.: Quantum Computation by Adiabatic Evolution., arXiv: quant-ph/0001106 (2000), 2000
22. Zomorodi-Moghadam, M., Taherkhani, M.-A., Navi, K.: Synthesis and optimization by quantum circuit description language., In Transactions on Computational Science XXIV: Springer, pp. 74–91 (2014)
23. Edoardo, B., Alessandra, B., Iacopo, C., Sonia, M., Valter, M.: Quantum Physics and Geometry, pp. 5–41. Springer, Cham (2019)
24. Feldmann, A.E., Foschini, L.: Balanced Partitions of Trees and Applications, Proceedings of the 29th International Symposium on Theoretical Aspects of Computer Science, pp. 100–111, (2012)
25. Andreev, K., Räcke, H.: Balanced graph partitioning, Theory of Computing Systems **39**(6), 929–939 (2006)
26. Hendrickson, B., Leland, R.: A Multilevel Algorithm for Partitioning Graphs, In Proceedings of the 1995 ACM/IEEE conference on Supercomputing. ACM (1995)
27. Kernighan, B., Lin, S.: An efficient heuristic procedure for partitioning graphs. Bell. Syst. Tech. J. **49**(2), 291–307 (1970)
28. Humble, T.S., Thapliyal, H., Munoz-Coreas, E., Mohiyaddin, F.A., Bennink, R.S.: Quantum Computing Circuits and Devices. IEEE Design Test. **36**(3), 03 04 (2019)
29. Van Meter, R., Devitt, S.: The path to scalable distributed quantum computing. IEEE. **49**(2), 31–42 (2016)
30. Cleve, R., Buhrman, H.: Substituting quantum entanglement for communication. Phys. Rev. A. **56**, 1201 (1997)
31. Grover, L.: Quantum Telecomputation. arXiv:Quant-Ph/9704012 (1997)
32. Cirac, J., Ekert, A., Huelga, S., Macchiavello, C.: Distributed quantum computation over noisy channels. Phys. Rev. A. **59**, 4249–4254 (1999)
33. Beals, R., Brierley, S., Gray, O., Harrow, A., Kutin, S., Linden, N., Shepherd, D., Stather, M.: Efficient distributed quantum computing. Proc. R. Soc. **469**, 20120686 (2013)
34. Yepez, J.: Type-II quantum computers. Int. J. Mod. Phys. **C 12**(09), 1273–1284 (2001)
35. Caleffi, M., Cacciapuoti, A., Bianchi, G.: Quantum internet: from communication to distributed computing, In NANOCOM '18 Proceedings of the 5th ACM International Conference on Nanoscale Computing and Communication, Reykjavik, Iceland (2018)
36. Cacciapuoti, A., Caleffi, M., Tafuri, F., Cataliotti, F., Gherardini, S., Bianchi, G.: Quantum Internet: Networking Challenges in Distributed Quantum Computing. arXiv:1810.08421 (2019)
37. Lo, H.K.: Classical-communication cost in distributed quantum-information processing: a generalization of quantum-communication complexity. Phys. Rev. A. **62**(1), 012313 (2000)
38. Yimsiriwattana, A., Lomonaco, S.J.: Distributed quantum computing: A distributed Shor algorithm, in Quantum Information and Computation II. International Society for Optics and Photonics, **5436**, 360–372 (2004)
39. Deutsch, D.: Quantum computational networks. arXiv:quant-ph/0607065 (2006)
40. Zomorodi-Moghadam, M., Houshmand, M., Houshmand, M.: Optimizing teleportation cost in distributed quantum circuits. Int. J. Theor. Phys. **57**(3), 848–861 (2018)
41. Houshmand, M., Mohammadi, Z., Zomorodi-Moghadam, M., Houshmand, M.: An Evolutionary Approach to Optimizing Communication Cost in Distributed Quantum Computation; arXiv:1910.07877 [Quant-Ph] (2019)
42. Andrés-Martínez, P., Heunen, C.: Automated distribution of quantum circuits via hypergraph partitioning. Phys. Rev. A. **100**(3), 032308 (2019)
43. Wille, R., Große, D., Teuber, L., Dueck, G.W., Drechsler, R.: Revlib:An online resource for reversible functions and reversible circuits, In 38th International Symposium on Multiple Valued Logic (ismvl 2008), Dallas, TX, USA (2008)
44. Barenco, A., Bennett, C.H., Cleve, R., DiVincenzo, D., Margolus, N., Shor, P., Sleator, T.: Elementary gates for quantum computation. Phys. Rev. A. **52**, 3457, quant-ph/9503016 (1995)
45. Shende, V.V., Bullock, S.S., Markov, I.L.: Synthesis of quantum-logic circuits. IEEE Trans. Comput. Aided Des. **25**(6), 1000–1010 (2006)