Int J Theor Phys (2018) 57:848–861
https://doi.org/10.1007/s10773-017-3618-x

Optimizing Teleportation Cost in Distributed Quantum Circuits

Mariam Zomorodi-Moghadam[1] ·
Mahboobeh Houshmand[2] · **Monireh Houshmand**[3]

Received: 8 June 2017 / Accepted: 29 November 2017 / Published online: 7 December 2017

Abstract The presented work provides a procedure for optimizing the communication cost of a distributed quantum circuit (DQC) in terms of the number of qubit teleportations. Because of technology limitations which do not allow large quantum computers to work as a single processing element, distributed quantum computation is an appropriate solution to overcome this difficulty. Previous studies have applied ad-hoc solutions to distribute a quantum system for special cases and applications. In this study, a general approach is proposed to optimize the number of teleportations for a DQC consisting of two spatially separated and long-distance quantum subsystems. To this end, different configurations of locations for executing gates whose qubits are in distinct subsystems are considered and for each of these configurations, the proposed algorithm is run to find the minimum number of required teleportations. Finally, the configuration which leads to the minimum number of teleportations is reported. The proposed method can be used as an automated procedure to find the configuration with the optimal communication cost for the DQC. This cost can be used as a basic measure of the communication cost for future works in the distributed quantum circuits.

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

Mahboobeh Houshmand
houshmand@mshdiau.ac.ir

Monireh Houshmand
m.hooshmand@imamreza.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

[3] Department of Electrical Engineering, Imam Reza International University, Mashhad, Iran

Keywords Quantum computation · Distributed quantum circuit · Teleportation cost · Optimization

1 Introduction

Quantum computation [1] is an emerging field of study with great potential to outperform classical computers in certain problems such as integer factorization [2], discrete logarithm finding, and database search [3].

Despite the theoretical advantages of quantum computations, realization of a large-scale quantum system is a real challenge [4]. It is mainly because that quantum computation technologies have limitations in the number of qubits they can process [5] and building a large-capacity quantum computer is hard [6] which makes distributed implementations of quantum technologies necessary [7].

One of the limitations of quantum implementations is due to the interactions of qubits with the environment that leads to the decoherence and when the number of qubits increases, the quantum information becomes more fragile and more susceptible to errors [8]. To overcome the mentioned problems, using distributed nodes or subsystems in a quantum system is a reasonable solution in which fewer qubits are used in each node or subsystem. Therefore, to have a large quantum computer, one appropriate solution is to build a network of limited-capacity quantum computers which are interconnected through a quantum or classical channel and altogether can implement the behavior of the whole quantum system [9]. This structure is known as *distributed quantum computer*.

In order to perform a distributed quantum computation, there should be a reliable mechanism for communication between separate nodes of a distributed quantum system. Long-distance quantum communication is a technological challenge for physical realizations of quantum communication [10]. In this regard, teleportation [11] is a primitive protocol for such this communication that uses entangled qubits to distribute quantum information [12]. This protocol has been experimentally implemented in many quantum technologies such as quantum photonics [13], NMR [14], and trapped ions [15].

In teleportation, qubit states are teleported from one node to another, without physically moving them and then computations are performed locally on qubits, which is also known as *teledata*. There is an alternative approach, called *telegate* that executes gates remotely using the teleported gate approach without the need for qubits to be nearby [16]. For some applications, like some adder algorithms it has been shown that teledata outperforms telegate and also its performance is independent of the network topology [17]. In this study, teledata is considered for communication between distributed nodes.

Teleportation is a costly operation in a distributed quantum system. Moreover, according to no-cloning theorem [18], when a qubit is teleported to a destination subsystem, it can no longer be used in its own subsystem. Therefore, minimizing the number of teleportations in a distributed quantum system is an important concern. In this study, this problem is considered and an algorithm is presented to optimize the number of teleportations for a distributed quantum system consisting of two spatially separated and long-distance quantum subsystems.

The paper is organized as follows: Section 2 explains some basic concepts and definitions of quantum computation and distributed quantum computing. Related work is presented in Section 3. Section 4 then establishes some definitions and notation needed for the proposed algorithm. Our proposed algorithm for optimal distributed quantum circuit is described in Section 5. Finally in Section 6, we conclude the paper.

2 Background

A generic scenario for a distributed quantum computation consists of n nodes in a quantum network. These nodes agree on performing a given computation by using a distributed quantum circuit (DQC). A 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 a DQC communicate by sending their qubits to each other using a specific quantum communication channel through teleportation.

In this study, it is assumed that data transfer between distributed nodes in the system is performed by teleportation. Teleportation allows us to send the state of a qubit from point A to point B by communicating only two classical bits. To complete this task, the sender and the receiver at the beginning need to share a maximal entangled state which serves as the quantum channel. Figure 1 shows the quantum circuit for a basic quantum teleportation as described in [1]. In this figure, the two top lines represent the sender's system and the bottom line is the receiver's one. The meters represent measurement, and the double lines coming out of them carry classical bits.

As stated in [11] and mathematically proved in [6], it can be concluded that teleporting a quantum state works even if that state is a mixed or an entangled state and so it can be used as communication protocol between distributed quantum systems.

The quantum channel connecting A and B is considered to be maximally entangled in order to achieve faithful teleportation. However, due to interactions with the environment, an entangled state may lose its coherence and become a mixed one, which reduces the efficiency of the protocol. Therefore, in order to obtain secure teleportation, the noisy quantum channel should be fixed using some approaches including quantum distillation [19] or by using weak measurement to improve the fidelity of teleportation via noisy channels [20].

3 Related Work

Distributed quantum computing has been studied for more than fifteen years [6] and the first suggestions were made by Grover [21], Cleve and Buhrman [22] and later by Cirac et al., [23]. In [21], Grover presented a distributed quantum system where there are some particles at remote locations and each one performs its computation and sends the required information to a base station when necessary. He mentioned that the issue in distributed quantum computing is how to optimally divide the problem into the different quantum computing units. The answer to this question effects the performance of the proposed problem in this work that can be considered in the future. Grover showed that using this distributed approach, the overall computation time is faster proportional to the number of such distributed particles. In the same paper, he also presented a quantum algorithm and adapted it to distributed computation where communication is costly.

Beals et al. [24] provided some algorithms for efficiently moving and addressing quantum memory in parallel. This imply that the quantum circuit model can be simulated with a

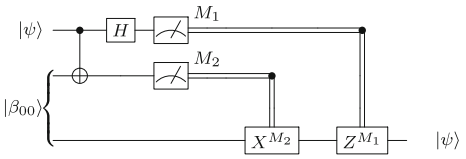

Fig. 1 Quantum circuit for teleporting a qubit [1]

low overhead by a distributed quantum computer with nodes connected using a hypercube graph.

Yepez [25] presented an architecture for distributed quantum computing with two types of communication which were called type I and type II. Type I quantum computer uses quantum communication between subsystems and type II quantum computer exploits classical communication between subsystems of the distributed computer. Using this classification, our distributed quantum circuit is of type I.

Related to teleportation cost, Streltsov et al., [26] posed the question of the cheapest way for distributing entanglement and provided the minimal quantum cost for sending an entangled composite state in a long distance. They showed that regarding the most general distribution protocol, the amount of entanglement sent in the total process of distributed communication cannot be larger than the total entanglement cost for sending the ancilla particle and sending back that particle.

Classical communication cost is the subject of the study in [27]. Authors conjecture that in a two-stage teleportation, each step requires a single bit of classical communication and in general, for an arbitrary N-dimensional pure state, $2log_2N$ bits of classical communication is required for remote preparation which is different from the usual teleportation in which the precise state of the qubit to be prepared in the receiver is known to the sender.

Ying and Feng [6] provided some definitions of a distributed quantum computing system and defined an algebraic language to describe quantum circuits for distributed quantum computing. Van Meter et al., [16] designed a fixed distributed quantum circuit for VBE carry-ripple adder. They evaluated the cost of the teleportation for the designed distributed system. They compared two approaches, namely teledata and telegate topologies of quantum circuits and showed that teleportation of data (the one we use in our work) is better than teleportation of gates. They also compared it to a monolithic machine and showed that as the node sizes are increased the multicomputer architecture has better performance in large problem sizes. But their distributed quantum circuit was a predesigned architecture and they did not make any specific change in the system to reduce the number of teleportation circuits in the design.

Also in [28], authors used the teleportation communication model for implementing a distributed version of Shor's algorithm. But like the one in [29], their method for assigning logical qubits to distributed nodes is an ad-hoc method specific for implementing Shor's algorithm. In their implementation, each computing node holds $n/4$ qubits from each register and output of each computing node is teleported to the next computing node. So there is no optimization in the number of teleportations and also the authors did not consider the cost of returning the teleported qubits back into their own subsystems. In [30, 31], the path through scalable quantum computers using distributed-memory multicomputer architectures is considered.

4 Definitions and Notations

In this section, we establish some assumptions and notations before proceeding with the main development of the study. First of all, we assume that there is a quantum circuit, QC as input with width W, size S, and depth D with the following definitions as stated in [32]:

The *width* of a quantum circuit is defined as the total number of qubits in the circuit. The *size* of a quantum circuit is the total number of its gates from a set of universal quantum

gates. The *depth* of a quantum circuit is the number of timesteps, D required for executing the circuit. In each timestep one or more gates can be executed in parallel.

It is assumed that there is an appropriate ordering of gate executions in the sense the number of timesteps is minimal. In each timestep, a set of gates that can be performed in parallel are executed.

For simplicity, in this paper it is assumed that there are just two partitions in DQC, each with the size of $W/2$. Without losing the generality, W is assumed to be an even number. Qubits are numbered from top to bottom from one to n in each partition, where the i^{th} line of the circuit from top to bottom represents the i^{th} qubit, q_i.

The gates are numbered in order of their executions in the QC and g_i means the i^{th} gate which is executed according to a scheduling algorithm. For gates which can be executed in parallel, the priority of their numbering is arbitrary. The set of all gates in the circuit is represented by $\mathcal{G}$.

There are three kinds of quantum gates in the DQC, namely, single-qubit, local and global CNOT gates and their definitions and representations are as follows:

- A single-qubit gates is shown as $gate\_name_i(p, j)$. p, indicates the index of partition to which the gate $gate\_name_i$ belongs and j indicates the index of the qubit the gate acts on.
- A *local* CNOT gate is the one whose control and target qubits belong to the same partition and is shown as $\text{CNOT}_i(p, j_c, j_t)$. p indicates the index of partition to which the gate CNOT_i belongs. $j_c(j_t)$ indicates the index of control (target) qubit in its partition.
- A *global* CNOT gate is the one whose control and target qubits belong to different partitions is called a *global* CNOT gate shown as $\text{CNOT}_i(p_c, j_c, p_t, j_t)$. $j_c(j_t)$ indicates the index of control (target) qubit in its home partition. The partition to which each qubit q of a global CNOT gate belongs is called the *home* partition of q. $p_c(p_t)$ indicates the index of partition to which the control (target) of CNOT_i belongs.

It is supposed that local gates including single-qubit and local CNOT gates are performed in their local partitions. The total number of gates in a QC and the number of global gates are denoted by m_t and m_g respectively. Also a subset of $\mathcal{G}$ for representing the set of global gates is denoted by $\mathcal{G}_d$.

$Config\text{-}Arr$, is an array with the size of m_g, whose elements show the partitions each global gate is supposed to be executed. The value of '0' ('1') for the elements of this array means the corresponding gate is assumed to be executed in P_0 (P_1). The partition in which the gate g is supposed to be executed based on the $Config\text{-}Arr$ array is denoted by $g.l$.

Definition 1 In order to perform a global CNOT gate, one of its two qubits should be teleported from its home partition to another. This qubit is called a *migrated* qubit, as long as it is not teleported back to its home partition.

To keep load balancing of quantum subsystems, it is assumed that at each time, the number of migrated qubits is at most one.

For example, in the circuit of Fig. 4, the set $\mathcal{G}$ is represented as:
$\{\text{CNOT}_1(0, 0, 1)\ \text{CNOT}_2(1, 0, 0, 0)\ \text{CNOT}_3(0, 0, 1, 1)\ H_4(1, 1)$
$\text{CNOT}_5(1, 1, 0, 1)\ H_6(0, 1)\ \text{CNOT}_7(0, 1, 1, 1)\ \text{CNOT}_8(0, 0, 1)\ \text{CNOT}_9(1, 1, 0, 0)\}$

As non-commutativity property of gates is an important concept in our algorithm, it is considered in the following. First we examine the non-commutativity of two CNOT gates.

Two CNOT gates do not commute when the index of control qubit of CNOT gate is the same as the index of target qubit of other gate [33] as shown in Fig. 2. As shown in this figure, the third output in circuit in the left is $|b \oplus c\rangle$, while it is $|a \oplus b \oplus c\rangle$ in the circuit in the right.

Now the commutativity relations of two adjacent CNOT and a single-qubit gate are considered.

1. If the single-qubit gate, U is applied to the same qubit of control of the CNOT gate, they commute if:
$$(U \otimes I)\text{CNOT} = e^{i\theta}\text{CNOT}(U \otimes I) \tag{1}$$
where $\theta \in [0, 2\pi)$. Examining (1), θ is obtained 0, and U is a diagonal unitary matrix as follows:
$$U = \begin{bmatrix} u_0 & 0 \\ 0 & u_1 \end{bmatrix}$$
.
2. If the single-qubit gate, U is applied to the target qubit of CNOT gate they commute if that the X gate and U gate commute. X gate and a U gate commute if:
$$XU = e^{i\theta}UX, \tag{2}$$
Examining (2), we found that θ is either 0 or π. In the case of $\theta = 0$, the structure of U is a unitary matrix as follows:
$$U = \begin{bmatrix} u_0 & u_1 \\ u_1 & u_0 \end{bmatrix},$$
and where $\theta = \pi$, the structure of U is a unitary matrix as follows:
$$U = \begin{bmatrix} u_0 & u_1 \\ -u_1 & -u_0 \end{bmatrix}.$$

The number of required teleportations for a given DQC, namely its teleportation cost is considered as the figure of merit for the proposed approach as explanted in Section 5.

5 Proposed Approach

In this section, first the proposed algorithm for optimizing the teleportation cost of a given DQC is explained and then, it is performed on a simple running example.

5.1 The Proposed Algorithm

In this section, the proposed algorithm to find the optimal number of teleportations and their sequence for a given DQC is presented. The algorithm is motivated by the fact that

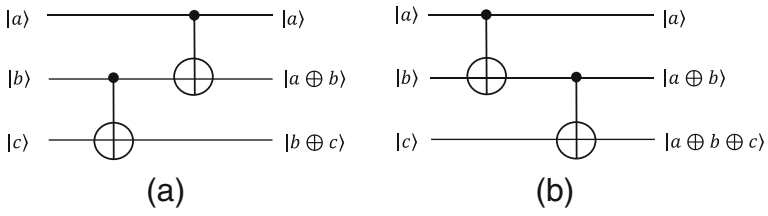

Fig. 2 **a** and **b** show when the index of the control qubit of one CNOT is the same as the index of the target of the other, two CNOTs do not commute [33]

teleportation is a costly operation and reducing the number of teleportations in a given DQC is of great importance.

We intend to start with a quantum circuit QC, composed of basic gate library [34], i.e., CNOT and single-qubit gates operating on $2n$ qubits. As mentioned earlier, for simplicity, in this paper it is assumed that there are just two partitions in DQC, each with the size of n. The proposed approach should first assign qubits to two partitions. To this end, Kernighan-Lin (K-L) algorithm as used in the VLSI design algorithms [35] is applied. The K-L algorithm is an $O(n^2 log(n))$ heuristic algorithm for solving the graph partitioning problem, in which we attempt to partition a graph into two partitions in such a way the interconnection cost between different partitions is minimized.

The K-L algorithm starts with a QC already split into two partitions, P_0 and P_1 each with n qubits. To apply this algorithm, the problem of assigning qubits to partitions should be represented by a graph. In this graph, each qubit represents a node and each CNOT gate relates to an edge between the nodes of graphs (the qubits) of the gate that it is applied to. The weight of each edge is equal to the total number of CNOT gates which operate on these two qubits. For example, for the circuit shown in Fig. 3a, the corresponding graph is shown as Fig. 3b.

After this graph representation, the K-L algorithm is applied to swap nodes (qubits) in partitions in such a way that the interconnection cost between partition is minimized.

After the initial partitioning by the K-L algorithm, the rest of proposed approach is explained. Suppose there are m_g global CNOT gates in the given DQC. In general, each of these gates can be executed in either P_0 or P_1 and hence there are 2^{m_g} different configurations for executing m_g global gates. In the proposed approach, each of these configurations is considered and for each of which, the minimum number of teleportation and their corresponding sequence are analytically determined. Finally, the configuration which leads to the minimum number of teleportation is reported. This procedure is performed by Algorithm 1 as explained in the following.

In Algorithm 1, teleportation cost for different configurations of executing m_g global gates is considered and the configuration with the minimum cost is reported as output. To this end, function $MIN\text{-}TELEPORTATION$ is called which takes a DQC with the ordered list of gates, $\mathcal{G}$, and $Config\text{-}Arr$ as inputs and returns the minimum number of teleportations and their sequence for that configuration. This algorithm is first called with $n_t = 0$. The subset of global gates, $\mathcal{G}_d$ is also used in the algorithm for reporting the sequence of teleportations. In different steps of this algorithm, the general and the global gates which are executed are removed from $\mathcal{G}$ and $\mathcal{G}_d$, respectively.

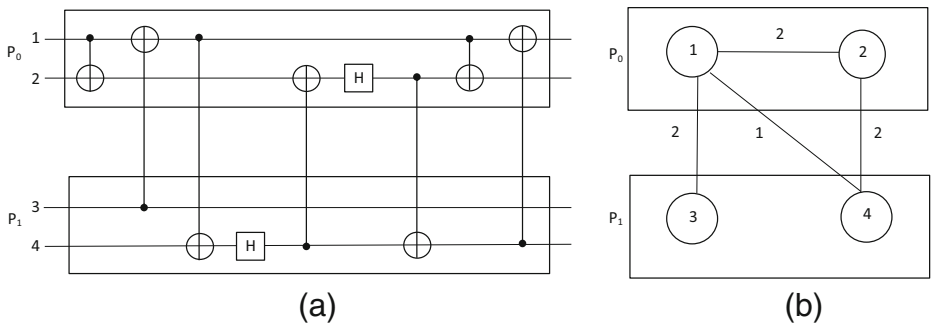

Fig. 3 **a** A sample QC split into two partitions **b** The corresponding graph

This algorithm starts from the first gate in list $\mathcal{G}$ of DQC. If it is a local gate, then no teleportation is needed for it and this gate is removed from $\mathcal{G}$. Otherwise, the qubit is teleported to the destination and the teleportation cost (n_t) is increased by one. Furthermore, to show the sequence of teleportations, $\mathcal{G}_d[0](C/T)$ is printed. Based on $Config\text{-}Arr$, C/T determines whether the control/ target qubit of $\mathcal{G}_d[0]$ should be telepoted to the other partition. When the number of teleportations (n_t) for a configuration exceeds or equals to the minimum number of them produced so far, the $MIN\text{-}TELEPORTATION$ function is forced to end.

When a qubit of a global gate, $temp$ in algorithm, is teleported to the other partition, the whole circuit is tracked and as much as possible number of gates that can be executed without the need of teleporting back the qubit are executed. This means that the migrated qubit has been used optimally by other gates before it is teleported back to its own partition.

Three characteristics of gates that can not be executed are explained in the $NON\text{-}EXECUTE$ function. This function returns TRUE when any of these characteristics occurs. When there is a gate, $\mathcal{G}[i]$, for which $NON\text{-}EXECUTE(temp, \mathcal{G}[i])$ function returns FALSE, there are possibly some gates before $\mathcal{G}[i]$ which have not been executed. In this case, $\mathcal{G}[i]$ can be executed just in case that it can commute with all of these gates. The commutativity of gates is checked in the $NON\text{-}COMMUTE$ function. The mentioned functions are introduced in the following:

– $NON\text{-}EXECUTE(g, g')$, takes two gates, g and g' as inputs. It returns TRUE, if the migrated qubit of g should be returned to its home partition due to the three different cases mentioned below and then g' can be performed. It returns FALSE otherwise.

  1. g' is a local gate whose one of qubits is the same as the migrated qubit of g.
  2. g' is a global gate with different label of g, i.e., $g'.l != g.l$.
  3. g' is a global gate and $g'.l = g.l$, but it requires another teleportation in order to execute g'.

– $NON\text{-}COMMUTE(g, g')$, takes two gates, g and g' as inputs. It returns TRUE, if the gates g and g' do not commute and returns FALSE otherwise. According to discussions in Section 4, the non-commutativity of two gates happens in three cases:

  1. The two gates are CNOTs where the index of control qubit of one of them is the same as the index of target qubit of the other.
  2. One of the gates is CNOT and the other is a non-diagonal single-qubit gate that acts on the control qubit of CNOT.
  3. One of the gates is CNOT and the other is a single-qubit gate which acts on the target qubit of CNOT and does not have any of the structures as explained in (1) or (2).

At the end of each run of the algorithm, n_t is increased by one once more because another teleportation is required to return the teleported qubit back. Then the algorithm recursively is called with the new existing gates until there is no gate in the set $\mathcal{G}$ and the minimum number of teleportations (n_t) is obtained.

Algorithm 1 Algorithm for determining the minimum number of teleportations required in $\mathcal{G}$

$n_{min} =$ MAX INT
for $i \leftarrow 1$ **to** 2^m configurations of $Config\text{-}Arr$ **do**
  $n_t = 0$
  temp=$MIN\text{-}TELEPORTATION(\mathcal{G}, Config\text{-}Arr, n_t, n_{min})$
  if $temp < n_{min}$ **then**
    $n_{min} = temp$
  end if
end for
$MIN\text{-}TELEPORTATION(\mathcal{G}, Config\text{-}Arr, n_t, n_{min})$
sw $= 0$
if $Empty(\mathcal{G})$ **or** $n_t >= n_{min}$ **then**
  return n_t
end if
if $local(\mathcal{G}[0])$ **then**
  remove$(\mathcal{G}[0])$
  $MIN\text{-}TELEPORTATION(\mathcal{G}, Config\text{-}Arr, n_t, n_{min})$
end if
$temp = \mathcal{G}[0]$
$n_t = n_t + 1$
if $n_t >= n_{min}$ **then**
  return n_t
end if
remove $(\mathcal{G}[0])$
print $(\mathcal{G}_d[0](C/T))$
remove $(\mathcal{G}_d[0])$
for $i \leftarrow 0$ **to** $\mathcal{G}.length$ **do**
  if $NON\text{-}EXECUTE(temp, \mathcal{G}[i])$ =**FALSE then**
    for $k \leftarrow i$ **to** 0 **do**
      if $NON\text{-}COMMUTE(\mathcal{G}[i], \mathcal{G}[k])$ =**TRUE then**
        sw $= 1$
        break
      end if
    end for
    if sw $= 0$ **then**
      remove$(\mathcal{G}[i])$
      if $global(\mathcal{G}[i])$ **then**
        remove$(\mathcal{G}[i]$ **from** $\mathcal{G}_d)$
      end if
    end if
  end if
end for
$n_t = n_t + 1$
$MIN\text{-}TELEPORTATION(\mathcal{G}, Config\text{-}Arr, n_t, n_{min})$
return n_{min}

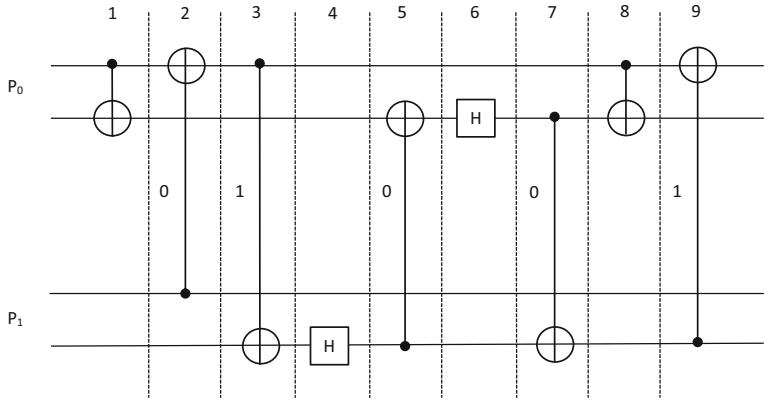

Fig. 4 A sample DQC

5.2 Running Example

In this section, the proposed algorithm in Section 5.1 is explained by a running example. This example is illustrated in Fig. 4 and the proposed algorithm is applied to it.

The algorithm starts by taking an already two-partite DQC as an input. This circuit consists of nine gates including two single-qubit gates, two local CNOT gates and five global CNOT gates. Since there are five global gates, the $Config\text{-}Arr$ has five elements and $2^5 = 32$ different configurations. Algorithm 1 for all these configurations calls function $MIN\text{-}TEELEPORTATION$ which finds the minimum number of teleportation and their sequence for that configuration. If in this function, the computed number of teleportations for a special configuration exceeds a previously computed miniminm number of teleportation, this function is forced to end. The proposed algorithm finally reports the configuration with the smallest number of teleportations. Therefore, for the given DQC this approach finds the minimum number of teleportations.

Let us consider $Config\text{-}Arr$={0, 1, 0, 0, 1} as a sample configuration. Function $MIN\text{-}TELEPORTATION$ starts with the first gate in $\mathcal{G}$, i.e., $\text{CNOT}_1(0, 0, 1)$ which is a local gate. Therefore, it is executed in its own partition and is then removed. The next gate in this

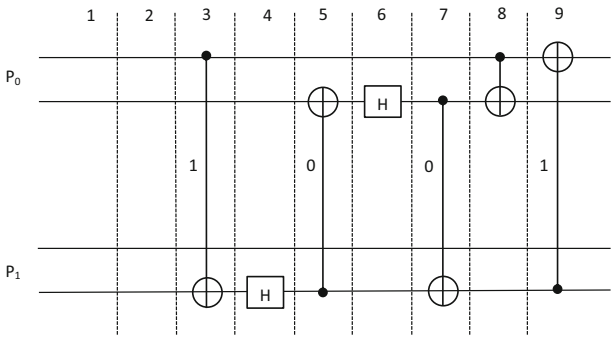

Fig. 5 New DQC after the first run of MIN-TELEPORTATION

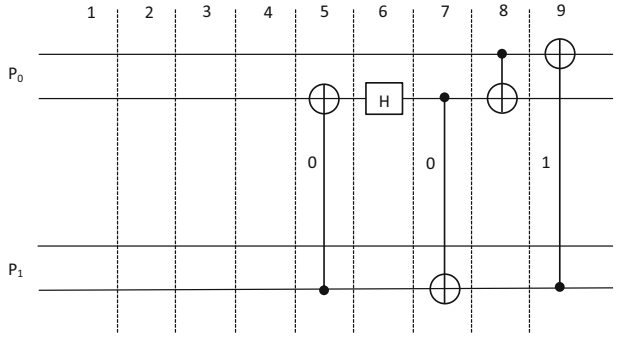

Fig. 6 New DQC after the second run of MIN-TELEPORTATION

array is $\text{CNOT}_2(1, 0, 0, 0)$ which is a global gate and based on the first element of $Config\text{-}Arr$, this gate is executed in P_0. For executing this gate, qubit #0 in P_1 is teleported to P_0, the number of teleportations, n_t is increased by one and then this gate is removed. In the outer loop of Algorithm 2 (Line 15), for all other gates, they are removed from the list, if they can be executed before teleporting back this migrated qubit.

The steps for the first run of function $MIN\text{-}TELEPORTATION$ are as follows.

- $NON\text{-}EXECUTE(\text{CNOT}_2(1, 0, 0, 0), \text{CNOT}_3(0, 0, 1, 1))$= TRUE according to the second condition of $Non\text{-}Execute$ function as they are intended for execution in different partitions.
- $NON\text{-}EXECUTE(\text{CNOT}_2(1, 0, 0, 0), H_4(1, 1))$= FALSE. Therefore, in the inner loop of function $MIN\text{-}TELEPORTATION$ (Line 17), this gate is checked with the set of all previous existing gates whether they can commute. As $NON\text{-}COMMUTE(\text{CNOT}_3(0, 0, 1, 1), H_4(1, 1))$= TRUE, $H_4(1, 1)$ can not be executed and is not removed.
- $NON\text{-}EXECUTE(\text{CNOT}_2(1, 0, 0, 0), \text{CNOT}_5(1, 1, 0, 1))$= TRUE according to the third condition of $NON\text{-}EXECUTE$ function as a different qubit should be teleported for the execution of this gate.
- $NON\text{-}EXECUTE(\text{CNOT}_2(1, 0, 0, 0), H_6(0, 1))$= FALSE. Therefore, the algorithm enters the inner loop where this gate is checked with the set of all previous existing

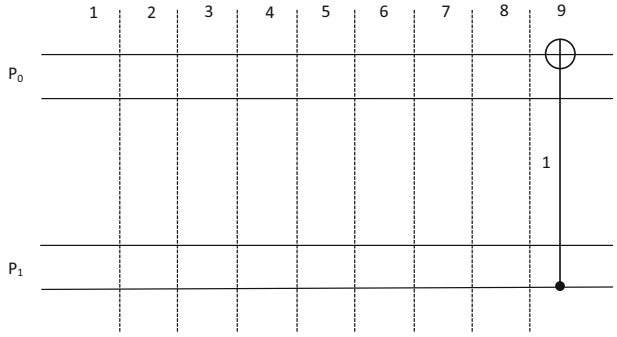

Fig. 7 New DQC after the third run of MIN-TELEPORTATION function

gates for possible commutativity. In this case, $NON\text{-}COMMUTE(\text{CNOT}_5(1, 1, 0, 1), H_6(0, 1))$= TRUE and hence, $H_6(0, 1)$ cannot be executed and is not removed.

– $NON\text{-}EXECUTE(\text{CNOT}_2(1, 0, 0, 0), \text{CNOT}_7(0, 1, 1, 1))$= TRUE since the case three of $NON\text{-}EXECUTE$ function is satisfied.
– $NON\text{-}EXECUTE(\text{CNOT}_2(1, 0, 0, 0), \text{CNOT}_8(0, 0, 1))$= FALSE but as $NON\text{-}COMMUTE(\text{CNOT}_7(0, 1, 1, 1), \text{CNOT}_8(0, 0, 1))$= TRUE, $\text{CNOT}_8(0, 0, 1)$ cannot be executed.
– $NON\text{-}EXECUTE(\text{CNOT}_2(1, 0, 0, 0), \text{CNOT}_9(1, 1, 0, 0))$= TRUE since the second condition of $NON\text{-}EXECUTE$ function is satisfied.

Table 1 The sequence and the number of minimum teleportations for different configurations of Fig. 4

Configuration # (in base 10)	The sequence of qubit teleportations	n_t
0	$g_2(C), g_3(T), g_5(C)$	6
1	$g_2(C), g_3(T), g_5(C), \textit{not continued}$	6
2	$g_2(C), g_3(T), g_5(C), \textit{not continued}$	6
3	$g_2(C), g_3(T), g_5(C), \textit{not continued}$	6
4	$g_2(C), g_3(T), g_5(T), \textit{not continued}$	6
5	$g_2(C), g_3(T), g_5(T), \textit{not continued}$	6
6	$g_2(C), g_3(T), g_5(T), \textit{not continued}$	6
7	$g_2(C), g_3(T), g_5(T), \textit{not continued}$	6
8	$g_2(C), g_3(C), g_5(C)$	6
9	$g_2(C), g_3(C), g_5(C), \textit{not continued}$	6
10	$g_2(C), g_3(C), g_5(C), \textit{not continued}$	6
11	$g_2(C), g_3(C), g_5(C), \textit{not continued}$	6
12	$g_2(C), g_3(C), g_5(T), \textit{not continued}$	6
13	$g_2(C), g_3(C), g_5(T), \textit{not continued}$	6
14	$g_2(C), g_3(C), g_5(T), \textit{not continued}$	6
15	$g_2(C), g_3(C), g_5(T), \textit{not continued}$	6
16	$g_2(T), g_3(T), g_5(C)$	6
17	$g_2(T), g_3(T), g_5(C), \textit{not continued}$	6
18	$g_2(T), g_3(T), g_5(C), \textit{not continued}$	6
19	$g_2(T), g_3(T), g_5(C), \textit{not continued}$	6
20	$g_2(T), g_3(T), g_5(T), \textit{not continued}$	6
21	$g_2(T), g_3(T), g_5(T), \textit{not continued}$	6
22	$g_2(T), g_3(T), g_5(T), \textit{not continued}$	6
23	$g_2(T), g_3(T), g_5(T), \textit{not continued}$	6
24	$g_2(T), g_5(C)$	4
25	$g_2(T), g_5(C), \textit{not continued}$	4
26	$g_2(T), g_5(C), \textit{not continued}$	4
27	$g_2(T), g_5(C), \textit{not continued}$	4
28	$g_2(T), g_5(T), \textit{not continued}$	4
29	$g_2(T), g_5(T), \textit{not continued}$	4
30	$g_2(T), g_5(T), \textit{not continued}$	4
31	$g_2(T), g_5(T), \textit{not continued}$	4

The steps described above correspond to the first run of function $MIN\text{-}TELEPORTATION$ and lead to the DQC of Fig. 5. The next runs of the algorithm are depicted in Figs. 6 and 7. There is a final run in the algorithm that results in an empty circuit and it is not shown.

For each configuration of global gates of Fig. 4, Table 1 shows the minimum number of teleportations and their corresponding sequence. In this table, the general name g for gates of DQC is used. $g_k(C/T)$ implies that the control/target qubit of g_k is teleportated to the other partition. The configuration used in the example is configuration #9 in Table 1.

In this example, the minimum number of teleportations, which is 4, occurs for the configuration #24. This configuration corresponds to the *Config-Arr* $=\{11000\}$ where the first and second global gates are executed in P_1 and other global gates are executed in partition P_0.

In the worst case, this example requires 10 teleportations while the optimizations of the proposed approach decrease this value to 4. Therefore, a remarkable improvement, i.e., 60% is obtained for this small circuit by the proposed approach. Table 1 shows the sequence and the number of minimum teleportations for different configurations of Fig. 4. When the number of teleportations exceeds or equals to the minimum number of them produced so far, the $MIN\text{-}TELEPORTATION$ function is forced to end and in this table, it is shown by "not continued" phrase in the second column.

6 Conclusion and Future Works

In this study, we presented an algorithm to optimize the number of teleporta- tions for a distributed quantum system consisting of two spatially separated and long-distance quantum subsystems. As teleportation is a costly operation in quantum computation technologies, reducing the number of such operations is very important for designing distributed quantum computers.

As future works, proposing a faster algorithm to perform the proposed idea is being considered. Moreover, extending the proposed approach to the case when the number of partitions is more than two can be studied. Finally, the used library of gates can be extended to include three-qubit quantum gates and hence the proposed algorithm should be modified accordingly.

Acknowledgements This work has been partially supported by a grant from Ferdowsi University of Mashhad.

References

1. Nielsen, M.A., Chuang, I.L. Quantum computation and quantum information, 10th anniversary edition. Cambridge University Press, Cambridge (2010)
2. Grover, L.K.: A fast quantum mechanical algorithm for database search. In: ACM symposium on theory of computing, pp. 212–219 (1996)
3. Shor, P.W.: Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. SIAM J. Comput. **26**, 1484–1509 (1997)
4. Steffen, L., Fedorov, A., Oppliger, M., Salathe, Y., Kurpiers, P., et al.: Realization of deterministic quantum teleportation with solid state qubits. arXiv:1302.5621 (2013)
5. Koashi, M., Fujii, K., Yamamoto, T., Imoto, N.: A distributed architecture for scalable quantum computation with realistically noisy devices. arXiv:1202.6588 (2012)
6. Ying, M., Feng, Y.: An algebraic language for distributed quantum computing. IEEE Trans. Comput. **58**, 728–743 (2009)

7. Van Meter, R., Ladd, T.D., Fowler, A.G., Yamamoto, Y.: Distributed quantum computation architecture using semiconductor nanophotonics. International Journal of Quantum Information **8**(01n02), 295–323 (2010)
8. Krojanski, H.G., Suter, D.: Scaling of decoherence in wide nmr quantum registers. Phys. Rev. Lett. **93**(25), 090501 (2004)
9. Nickerson, N.H., Li, Y., Benjamin, S.C.: Topological quantum computing with a very noisy network and local error rates approaching one percent. Nature Nat. Commun. **4**, 1756 (2013)
10. Brassard, G., Lütkenhaus, N., Mor, T., Sanders, B.C.: Limitations on practical quantum cryptography. Phys. Rev. Lett. **85**, 1330 (2000)
11. Bennett, C.H., Brassard, G., Crépeau, 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)
12. Whitney, M., Isailovic, N., Patel, Y., Kubiatowicz, J.: Automated generation of layout and control for quantum circuits. Phys. Rev. Lett. **85**(26), 1330 (2000)
13. Pan, J.-W., Bouwmeester, D., Mattle, K., Eibl, M., Weinfurter, H., Zeilinger, A.: Experimental quantum teleportation. Nature **390**, 575–579 (1997)
14. Nielsen, E., Knill, M.A., Laflamme, R.: Complete quantum teleportation using nuclear magnetic resonance. Nature **396**, 52–55 (1998)
15. Riebe, M., Häffner, H., Roos, C., Hänsel, W., Benhelm, J., et al.: Deterministic quantum teleportation with atoms. Nature **429**, 734–737 (2004)
16. Van Meter, R., Munro, W., Nemoto, K., Itoh, K.M.: Arithmetic on a distributed-memory quantum multicomputer. ACM Journal on Emerging Technologies in Computing Systems (JETC) **3**, 2 (2008)
17. Van Meter, R.D. III.: Architecture of a quantum multicomputer optimized for shor's factoring algorithm. arXiv:quant-ph/0607065
18. Wootters, W.K., Zurek, W.H.: A single quantum cannot be cloned. Nature **299**, 802–803 (1982)
19. Bennett, C.H., Brassard, G., Popescu, S., Schumacher, B., Smolin, J.A., Wootters, W.K.: Purification of noisy entanglement and faithful teleportation via noisy channels. Phys. Rev. Lett. **76**(5), 722 (1996)
20. Pramanik, T., Majumdar, A.S.: Improving the fidelity of teleportation through noisy channels using weak measurement. Phys. Lett. A **377**(44), 3209–3215 (2013)
21. Grover, L.K.: Quantum telecomputation. arXiv:quant-ph/9704012 (1997)
22. Cleve, R., Buhrman, H.: Substituting quantum entanglement for communication. Phys. Rev. A **56**, 1201 (1997)
23. Cirac, J., Ekert, A., Huelga, S., Macchiavello, C.: Distributed quantum computation over noisy channels. Phys. Rev. A **59**, 4249 (1999)
24. Beals, R., Brierley, S., Gray, O., Harrow, A.W., Kutin, S., Linden, N., Shepherd, D., Stather, M.: Efficient distributed quantum computing. In: Proceedings of the Royal Society A, vol. 469, p. 20120686 (2013). The Royal Society
25. Yepez, J.: Type-II quantum computers. Int. J. Mod. Phys. C **12**(09), 1273–1284 (2001)
26. Kampermann, H., Streltsov, A., Brub, D.: Quantum cost for sending entanglement. Phys. Rev. Lett. **108**, 250501 (2012)
27. Lo, H.K.: Classical-communication cost in distributed quantum-information processing: a generalization of quantum-communication complexity. Phys. Rev. A **62**(1), 012313 (2000)
28. Yimsiriwattana, A., Lomonaco, S.J. Jr.: Distributed quantum computing: a distributed shor algorithm. arXiv: quant-ph/0403146 (2004)
29. Deutsch, D.: Quantum computational networks. arXiv:quant-ph/0607065 (2006)
30. Van Meter, R., Devitt, S.J.: The path to scalable distributed quantum computing. Computer **49**(9), 31–42 (2016)
31. Van Meter, R.: Distributed quantum computing systems: Technology to quantum circuits. In: 2017 symposium on VLSI circuits, pp. T184–T185. IEEE, Piscataway (2017)
32. Pham, P., Svore, K.: A 2d nearest-neighbor quantum architecture for factoring in polylogarithmic depth. Quantum Inf. Comput. **13**(11-12), 937–962 (2013)
33. Houshmand, M., Hosseini-Khayat, S., Wilde, M.M.: Minimal-memory requirements for pearl-necklace encoders of quantum convolutional codes. IEEE Trans. Comput. **61**(3), 299–312 (2012)
34. Houshmand, M., Saheb Zamani, M., Sedighi, M., Arabzadeh, M.: Decomposition of diagonal Hermitian quantum gates using multiple-controlled pauli Z gates. ACM J. Emerg. Technol. Comput. Syst. **11**(3), 28 (2014)
35. Kernighan, B., Lin, S.: An efficient heuristic procedure for partitioning graphs. Bell Syst. Tech. J. **49**(2), 291–307 (1970)