International Journal of Theoretical Physics (2021) 60:3494–3513
https://doi.org/10.1007/s10773-021-04920-y

Improving the Teleportation Cost in Distributed Quantum Circuits Based on Commuting of Gates

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

Received: 14 April 2021 / Accepted: 24 July 2021 / Published online: 9 August 2021

Abstract
Distributed quantum system is a well-known method to overcome the problems of designing and manufacturing large quantum systems to achieve higher processing power in the world of quantum processing. In distributed quantum computing, a number of limited-capacity quantum circuits communicating with each other through communication channels operate as a large quantum circuit. The most essential step in designing a distributed quantum system is to optimize the communication between the components to reduce the overhead cost of communications. In this study, based on the commuting of quantum gates, an efficient method is proposed to reduce the number of teleportations required to perform distributed quantum circuits (DQCs). The results obtained from the evaluation of our method on benchmark circuits indicate an impressive decrease in the number of teleportations and a significant reduction in the execution time compared to the present methods.

Keywords Distributed quantum circuits · Teleportation · Communication cost · Optimization

1 Introduction

Limitations of Silicon-Based computing like processing power, physical limits of design complexity, memory, energy consumption, density, and heat dissipation [1] have led to new

✉ Keivan Navi
navi@sbu.ac.ir

Omid Daei
o.daei@qiau.ac.ir

Mariam Zomorodi
m.zomorodi@pk.edu.pl

[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 Information and Communications Technology, Cracow University of Technology, Kraków, Poland

technologies which can overcome these limitations. Quantum computing is one of these innovative technologies [2–4].

There are significant algorithmic improvements in quantum computing to solve difficult problems in computations [5] and communications [6] which can be solved in just a few seconds. These problems takes a few days to solve by today's computers [7]. Also, it can be used to solve complex problems in artificial intelligence and in particular machine learning [8, 9]. Furthermore, Quantum computers can also be used for fundamental changes in current authentication mechanisms and network security [10]. Therefore, quantum computing has a potential to transform the entire current market and industry. It can fundamentally change the market for computer systems and can lead to fundamental changes in this industry and related industries.

Manufacturing technology has already entered the engineering phase after about two decades of studies on quantum computing [11]. In the late 2019, Google achieved a quantum computer with a 54-qubit quantum processor called the Sycamore [12]. This company estimates that "a state-of-the-art" supercomputer will take 10,000 years to calculate the same work that Sycamore does in a few seconds. Although this claim has been challenged by multiple research groups. In addition, other big companies such as Intel, IBM and etc. are actively working on quantum computers.

Because of some limitations on the implementation of quantum systems such as interaction of the qubits with the environment which leads to decoherence, it is difficult to build a high-capacity quantum computer. Under these circumstances, even building a small quantum computer is also complicated [13]. As the number of qubits increases, the construction of these systems becomes more complex and more probable to error, leading a system more sensitive to error [14]. Therefore, error correction is also important in quantum systems.

On the other hand, to fabricate the quantum systems, lots of qubits must be involved to correct a qubit, so it is possible that all the qubits do not fit inside one chip [13]. Due to the mentioned limitations, connecting small quantum computers is a reasonable method to develop large-scale quantum computers [15]. A quantum computer built in this way will have fewer qubits per sub-systems and fewer errors. This computer is called a distributed quantum computer [13].

To implement a distributed quantum system, we need a reliable communication channel between sub-systems. Teleportation [16] is one of the primitive communication protocols. Using this protocol, in order to transfer quantum information between sub-systems, entangled qubits are used [17]. This is one of the most promising ways to communicate part of systems in a quantum system, and its validity has been tested experimentally in some articles [18, 19].

In teleportation, the state of a qubit is transferred from one node to another without the qubit itself being transferred [16]. Therefore, after transferring the qubits information, all the calculations will be performed locally at the destination.

Based on the no-cloning theorem [20] in quantum computing, a qubit does not exist at the source after teleportation and cannot be used there. As a result, to reuse this qubit in the source system, it must be returned to the source via a communication channel. Teleportation in distributed quantum systems is a high-cost operation and solutions to reduce the number of teleportations are critically important.

The efficiency of a distributed quantum system is crucial. In this paper, optimizing the number of teleportations between sub-systems is considered as a measure of its efficiency. Some methods have already been studied in [21–23]. In [21], an algorithm with an exponential complexity is proposed to optimize the number of teleportations in two-partitioned

distributed quantum circuits (DQCs). Also in [22], an algorithm with better execution time than the previous one is presented which is based on genetic algorithm. As well as in [23] an algorithm is proposed to decrease the communications in DQCs. In their approach, first, the QC is transformed into a bipartite graph model, and then a dynamic programming approach is applied to partition the graph into low-capacity QCs.

In this study, this specific issue is attentively examined by presenting an innovative algorithm for optimizing distributed quantum circuits, and more improvements have been obtained by our proposed approach compared to the existing methods.

The rest of the paper is organized as follows: some of the primary concepts of quantum computing and distributed quantum computing are described in Section 2. Section 3 contains related work. Our proposed approach is presented in Section 4. Finally, the results and conclusions are specified in Sections 5 and 6, respectively.

2 Background

In this section, concepts such as qubit, quantum gate, quantum circuit (QC), distributed quantum circuit (DQC), teleportation, and then commute of gates are explained.

Qubits are the quantum equivalent of classical bits. The basic states of a quantum system are $\{|0\rangle, |1\rangle\}$. A quantum system can be not only in its basic state, but also can be in a linear combination of basic states called superposition. The general state of a qubit is represented as $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$. $|\psi\rangle$ is a quantum state. α and β are complex numbers such that $|\alpha|^2 + |\beta|^2 = 1$. This state represents a unit vector in a two-dimensional complex space named Hilbert space. By measuring a qubit into the standard basis, the probability of outcome $|0\rangle$ and $|1\rangle$ with value "0" and "1" are $|\alpha|^2$ and $|\beta|^2$, respectively.

A quantum gate is virtually like a classic gate and changes the state of one or more qubits when applied. But quantum logic gates are reversible. It means the number of qubits in the input are equal to output. An n-qubit quantum gate U is defined by a $2^n * 2^n$ matrix. When U performed on a quantum state $|\psi\rangle$, the outcome changes to another state represented by the vector $U|\psi\rangle$.

There are different quantum gates that apply to a various numbers of qubits. Pauli, Hadamard and Rotation are some commonly used and single-qubit gates [24, 25]. If U is a gate that operates on a single qubit, Controlled-U is a gate encompassing control and target and operates on two qubits. If control qubit is $|1\rangle$, U is applied to the target. Otherwise, it remains unchanged. For example, controlled-NOT (CNOT) gate is a two-qubit one. If control qubit is $|1\rangle$, NOT is applied on target qubit, and if control qubit is $|0\rangle$ it remains unchanged.

The circuit model of quantum computation is based on the unitary evolution of qubits by networks of gates [24]. Each n-qubit quantum gate represents a linear transformation which is illustrated by a unitary matrix defined on an n-qubit Hilbert space. Matrix U is unitary if $UU^\dagger = 1$ and $U^\dagger$ is the conjugate transpose of the matrix U.

A QC consists of quantum gates interconnected by quantum wires. Quantum wires carry qubits with time from left to right. We evaluate the unitary matrix of the QC by tensor product [24] of the unitary matrices of quantum gates related to. To realize arbitrary quantum gates, only gates implemented in quantum technology which is called "basic gates" library [26] should be used. So, the other gates have to be decomposed into these gates. All of this process is addressed quantum logic synthesis [27].

DQCs include n smaller QCs working together, and all together implement the functionality of a QC. All partitions of the DQC must be able to send their qubits to other parts. Quantum channels are commonly used for these connections.

In DQC, there are two types of quantum gates called local and global ones. In a local gate, all qubits are located in one sub-system or partition, but all qubits are not positioned in the same partition in a global gate. That is, one or more qubits are located in distinct parts.

To perform a global gate, qubits state that this gate operates on must be collected in a partition. To transmit qubits information from one partition to another partitions, teleportation [16] can be typically used as a communication protocol in these circuits. In Teleportation two points need to share a maximally entangled state [28]. The state of one qubit is sent to another point with the help of two classical bits. In order to perform global gates, the qubits in different partitions must be teleported to one partition. Qubits teleported from a source partition to another partition are called migrated qubits, as long as they are not teleported back to source. The Fig. 1 [24] represents the QC for quantum teleportation.

From the study in [16], it can be concluded that even if a quantum state is an entangled state, the quantum teleportation operates. Accordingly, it can be operated as a communication protocol in distributed quantum systems to transmit qubits state to another partitions [29].

The non-commutativity property of quantum gates is described in this section. If two quantum gates are exchanged and the circuit output changes, both do not commute. We used this property in our proposed algorithm to reduce the number of teleportations. To give an illustration, we examine the noncommutativity of two CNOT gates. When the index of control qubit of CNOT gate is the same as the index of target qubit of another gate, these CNOT gates do not commute [30]. As shown in the Fig. 2, the third output is different. In (a) it is $|a \oplus b \oplus c\rangle$, but in (b) it is $|b \oplus c\rangle$. So, these two gates do not commute. In the same way, if target qubits of the two CNOT gates are the same, they do not commute. But if the control qubits of CNOT gates are the same and the target qubits of them are different, they commute. Also, if the control qubits and target qubits of CNOT gates are not the same they commute. As shown in the Fig. 3a and b, all outputs are the same.

3 Related Work

One of the first studies in the field of DQC was conducted by Grover [2]. He proposed a quantum system, parts of which were spaced apart. He demonstrated the speed and efficiency of his proposed algorithm, divide the quantum system into several parts and used it as a distributed system. In his method, each part performed the calculations separately, and

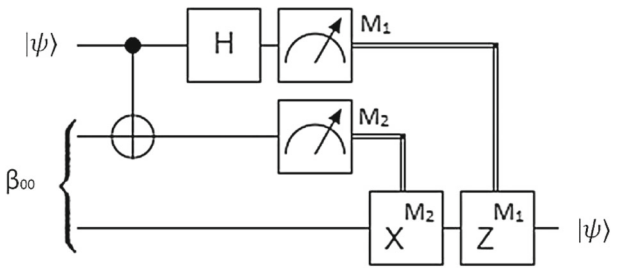

Fig. 1 Quantum circuit to teleporting a qubit [24]

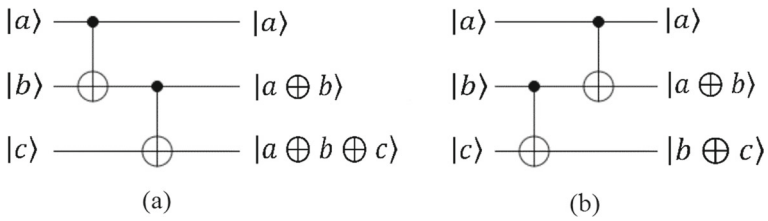

Fig. 2 a and b show that when the index of the target of a CNOT gate is the same as index of control of other, these gates do not commute [30]

when this information was needed in the base system, it was sent to it. He showed that using the distributed method, the total time of the calculations is proportional to the number of these particles distributed.

Beals et al. [31] showed that a DQC could be created using a hypergraph for any desired QC. Also, they proposed some algorithms for addressing quantum memory in parallel. Pablo et al. [32], provided two routines to improve circuit distribution using hypergraphs. They evaluated a number of different QCs using their method and showed that using their method reduced the cost of distribution by more than half compared to the native methods.

In [33], two communication types is proposed as the communication protocol between the components of the distributed quantum system, and the authors called them as Type I and II. In Type I, quantum communication is used to communicate between the components of the distributed system, so that each qubit could be entangled with any number of other qubits. But in Type II, a number of small quantum computers interconnected by classical channels, and each qubit may be entangled with only nearby qubits at a particular part of the system.

In [34], the basic theory of quantum teleportation is discussed and its commercial and scientific applications such as quantum network [35] are presented. In the second part, the results of current experiments on quantum teleportation are given. Also, they have presented five challenges needed to be resolved in the future in order to have an ideal quantum teleportation. Ultimately, the future implementation and development of quantum teleportation are discussed.

In [11], authors studied how to connect quantum computers to each other to achieve higher processing speeds through quantum internet. Their study also examined the challenges of quantum Internet design. The authors of this article examined the same challenges in another article [36] and considered teleportation as the major plan for transmitting information in this network. They finally expressed challenges and open issues regarding the design of quantum internet.

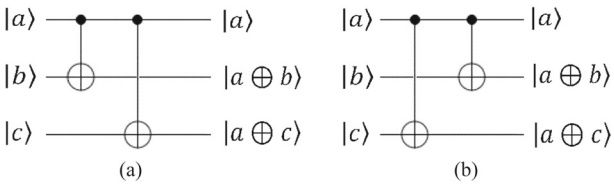

Fig. 3 a and b show that when the control qubits of CNOT gates are the same and the target qubits of them are different, they commute [30]

Yirasmanda et al. in [37] are presented a distributed version of quantum factorization algorithm built by global gates. They designed some of the required circuits, such as full-adders and QFTs, to be distributed using non-local CNOT gates. Then, they divided some qubits into groups and placed each group in a quantum computer. They connected these quantum computers via a network and used non-local CNOTs where needed. They measured total number of required teleportations to perform the DQC, but any offer to optimization is not provided.

Related to teleportation, Van meter et al. [38] divided a 2-qubit VBE adder into two separate QCs and implemented teleportation to connect the two separate parts. They set up a teleportation circuit for each of the global gates and employed them for transmitting qubits between nodes. The first teleportation circuit transfers a qubit from node A to node B, and this qubit is used in this node. Then, the second teleportation circuit transfers a qubit from node B to node A, and this migrated-qubit is used there. In their method, a teleportation circuit is placed for each global gate and no attempt was made to reduce the cost of teleportation. So, their method was fixed and the total number of teleportations was equal to the number of global gates. In the Fig. 4 their final design is shown.

In [39], authors used the concept of teleportation to simplify important classical computing task. They showed that by applying their method, fewer qubits are involved in classical calculations. But like the other ones, they did not any attempt to improve the number of teleportations.

Cacciapuoti et al. [40] identified teleportation as the key to creating and designing the quantum Internet, and outlined some of the challenges in designing it. The main concept of the classical communication system model should be redesigned to take into account the characteristics of quantum teleportation, and this redesign was introduced as a basic precondition for the construction of any effective quantum communication protocol. They also evaluated the fundamental differences between classical information transmissions versus quantum information transmission.

Humble et al. in [41] compared their design with an integrated one and represented that as the number of nodes increases, also its performance increases in proportion to their number. Their design was fixed, and they did not discuss ways to reduce teleportation costs.

In [21], authors have obtained the cost of communicating two-part DQCs based on an algorithm. Because in a two-qubit global gate, each of the qubits is in a different partition,

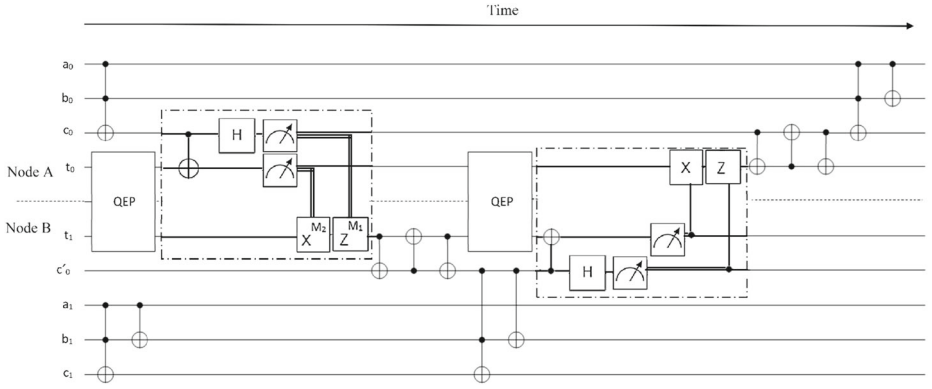

Fig. 4 The distributed 2-qubit VBE adder [38]

there are two possibilities for teleportation, so all global gates in a DQC can establish various possibilities for teleportation. Therefore, when teleportation is performed, it is more satisfactory to run other global gates if possible. Their algorithm achieves the best order of execution of two-qubit CNOT gates. In their method, all possible situations for teleporting and performing of global gates are examined and the best answer is suggested as the least expensive solution. In this method, due to the fact that all possible scenarios are examined, as the number of global gates increases, the number of cases that need to be examined increases exponentially. Therefore, with the increase in the number of global gates, the execution time of the algorithm will also increase exponentially. Now, if we consider that the number of partitions is more than two, this time will be much longer, which is not mentioned in the article. They later improved the execution time in [22] using the genetic algorithm, which is an evolutionary algorithm. In their algorithm, the teleportation cost is considered as the cost function of the genetic algorithm and it tries to find the most exact possible configuration for executing global gates.

In [42], we presented a general method to obtain DQCs from QCs, based on reducing the number of quantum teleportations. In this work, we used graph theory to provide a way to distribute properly qubits across different partitions that reduce the communications between them. As a result, the number of global gates decreases, which ultimately reduces the number of quantum teleportations required to perform DQC.

4 Proposed Approach

In this section, first, the proposed algorithm to optimize the teleportation cost is explained. Then at the end of this section, it is applied to a sample circuit, and details are provided.

DQC consists of a number of low-capacity QC that communicate with each other using communication channels. Typically, teleportation is used to transmit the desired information to each QC. Such system is also called quantum multicomputer [38]. Teleportation is a costly operation for communicating partitions in DQCs. Therefore, teleportation should not be used too much and to increase efficiency, it is beneficial to reduce their use as much as possible.

In DQCs, each partition contains a number of qubits. Depending on the gates of the QC and the qubits on which these gates operate, the gates are classified into two types called local and global which were explained in Section 2.

The qubits information which a local gate operates on is locally available in the same partition where that it is located. So, it is easy to perform them, but global gates are differently performed. It is needed to take all the qubits information that these gates operate on, to the desired partition in which we intend to perform a global gate. For this purpose, the teleportation of qubits is used. As mentioned in Section 2, according to no-cloning, when a qubit is teleported which is called a migrated qubit, it exists no longer at the origin, and it must be teleported back to be used at source. With this in mind, each global gate requires at least two teleportations to be performed. First, a qubit migrates to the destination. Second, when the migrated qubit is returned to the home after the execution of the global gate. However, if a qubit is not needed at the source after being teleported to the destination, its return can be delayed. So, more global gates may be performed at the destination with the same previous teleportation, and there is no need to re-teleport the qubit information again because it is already present there. In this method, several global gates may be performed in a row by just a teleportation.

After teleporting and performing the global gate and before running the next global gate, if the migrated qubit is required to perform a local gate at the source or another global gate using this qubit at the source and it cannot be performed with this teleportation, it must be returned back. After performing those gates, if this qubit is required at the destination, it teleports to perform the next global gate. Assuming one teleportation is performed at a time, there are two possibilities for two-qubit gates and three possibilities for three-qubit ones to teleporting qubits. There are various possibilities to run global gates successively. So, to be able to determine the maximum number of consecutive gates with the least number of teleportations, it is necessary to examine all the cases and achieve the best possible configuration to perform global gates with the minimum number of teleportations.

An algorithm is introduced in [21] that gives the best possible configuration to perform global gates of two-partite DQCs consisting of one and two-qubit gates. Under these circumstances, if the number of global gates is assumed to be m, the 2^m possible configurations must be checked to obtain the best one. This means that as the number of global gates increases, the number of possible configurations and search time increases exponentially. In [22], the same authors improved the running time by proposing a heuristic algorithm based on genetic algorithm.

Our proposed approach to decrease the number of teleportations in DQC is based on the fact that it does not examine all the possible configurations to improve the running time complexity of the algorithm. Also, we use the commuting of gates so that more global gates with fewer teleportations can be performed. This method has not been used in the previous studies.

We initially start with a QC and assume that this circuit has only one, two, and three-qubit quantum gates. The arbitrary n-qubit quantum gates can also be decomposed into the basic gate libraries [43, 44]. Consequently, our method can be applied to all types of QCs.

By using graph theory to convert the monolithic quantum circuit to the corresponding graph model which was explained in our previous work [42] in detail, and by implementing the method introduced to convert QCs to the optimized and arbitrary partitions, DQC with the desired number of partitions (K) and optimal communications is achieved. The partitions of DQC are shown by P_i, and we define them as follow:

$$\begin{cases} (P_i)|\ i : 1...K, \\ K = the\ number\ of\ partitions \end{cases}$$

The qubits of DQC are shown as:

$$\begin{cases} (q_i)|\ i : 1...m, \\ m = the\ number\ of\ qubits \end{cases}$$

In this paper, a heuristic algorithm is proposed which attempts to find a configuration to perform global gates in DQCs that optimizes the number of teleportations. First, a list of the global gates in the DQC is defined. It is called $GGlist$. The global gates named G_i are arranged in $GGlist$, in order.

$$GGlist = \begin{cases} (G_i)|\ i : 1...n, \\ n = the\ number\ of\ global\ gates\ in\ DQC \end{cases}$$

All the gates on the list must be processed. Local gates of DQC perform locally at their partition. However, one or two teleportations are needed to execute global gates that it depends on the type of a gate we are going to execute. If there is a three-qubit gate that all the three qubits are in different partitions, two of these qubits must be teleported to the third qubit partition to execute this gate; therefore, two teleportations are needed. In each round

of the algorithm, the first element of the list, which has not been performed yet is selected, and the rest of the algorithm is based on the qubits of this gate. It is called base. Suppose G_1, the first element of $GGlist$ and base in this round, has two qubits. q_i is in partition P_x $\{q_i \in P_x\}$ and q_j is in partition P_y $\{q_j \in P_y\}$. There are two possible options for teleportation. The first, q_i is teleported to P_y. The second, q_j is teleported to P_x. It does not matter which one is used to execute G_1 but when the next global gates, $G_i|i : 2, ..., n$ are considered, we can determine exactly which one can execute more global gates without additional teleportation. After determining the best option and increasing the number of teleportations by one unit, as well as determining which global gates can be performed with it, these gates are removed from the $GGlist$. Then, the first round of the algorithms is terminated, and the next rounds are continued to process all the $GGlist$.

If q_i is one of the qubits that G_i and G_{i+1} operate on it, we express these conditions mathematically as follows:

$$\{q_i \cap G_i \neq \emptyset \ , \ q_i \cap G_{i+1} \neq \emptyset\} \tag{1}$$

Suppose that qubit q_i is teleported to P_y for the execution of G_i, but the other qubit of G_{i+1} is in P_z;

$$\begin{cases} (q_i, q_j) : \ qubits\ of\ G_i \ , \ q_i \in P_x \ , \ q_j \in P_y \\ (q_i, q_k) : \ qubits\ of\ G_{i+1} \ , \ q_i \in P_x \ , \ q_k \in P_z \end{cases} \tag{2}$$

Therefore, this teleport does not lead to performing G_{i+1}. In this case, the algorithm checks whether it is possible to commute G_{i+1} with the next gate in the $GGlist$, here G_{i+2}. If it is possible, it checks for all the gates between them to commute. If possible, it commutes them and check again to find the most possible match. This means that it seeks to find the maximum global gates able to meet these conditions. But if not, it is no longer possible to perform more global gates by this teleportation and the algorithm stops at this round. The gates that have been performed will be removed from the $GGlist$ and the algorithm continues with the rest of the $GGlist$.

Now suppose qubit q_i is teleported from partition P_x to P_y in order to perform G_i. If there are one or more local gates, in short LG, before G_{i+1} that apply to the following condition:

$$\{LG \in P_x \ \ and \ \ q_i \cap qubits\ of\ LG \neq \emptyset\} \tag{3}$$

To perform this local gate, migrated qubit q_i must be teleported back to P_x before performing G_{i+1}. This will increase the number of teleportations. To improve, it is necessary to check whether it is possible to execute the LG before teleporting q_i, and then, q_i is teleported to P_y or it may be possible to perform this local gate after executing G_{i+1}. This delays the return of q_i to partition P_x and leads to performing more global gates with the same teleportation.

Under the terms described earlier, if the base is a three-qubits one, two qubits may need to be teleported to execute it. In this case, two qubits are found, which lead to the highest number of global gate executions in a row, considering the conditions described above. The algorithm of the proposed approach is presented in Algorithm 1.

Algorithm 1 The Algorithm to find optimized configuration of performing global gates.

```
1: Input: distributed quantum circuit (DQC),
2: Output: a configuration to perform global gates at the optimal number of teleportations.
3: GGlist ← global gates of DQC
4: all_gates ← list of all gates in DQC
5: while GGlist ≠ Null do
6:     base ← GGlist(1)
7:     m ← 0
8:     for i = 1 to last qubits of base do
9:         (sel_gates, num_of_gates) ← find_global_gates (all_gates, GGlist, qubit(i))
10:        if num_of_gates > m then
11:            sel_qubit ← qubit(i)
12:            sel_global_gates ← sel_gates
13:            m = num_of_gates
14:        end if
15:    end for
16:    teleport sel_qubit and perform sel_global_gates
17:    remove sel_global_gates from GGlist
18: end while
```

In order to find a suitable configuration at a low cost, we proposed an algorithm to obtain the right teleportation to perform the maximum number of global gates in a row by commuting the gates. The algorithm starts searching in a loop and continues as long as there are global gates which are not processed. In the *for* loop, it is determined that by teleporting each of the qubits of the base gate, how many subsequent global gates can be performed and the maximum is selected (line 8-15). The function *find_global_gates* returns the number of global gates as well as the list of these gates that can be performed by teleporting *qubit(i)* to destination partition (line 9). After running the *for* loop, the best qubit of the base gate for teleportation is determined, which leads to performing more global gates. the selected qubit is teleported and the selected global gates are performed. Then, these gates are removed from the *GGlist*. Another rounds are performed in the same way, and whenever *GGlist* is empty, the algorithm ends. Some of the quantum gates cannot commute. Accordingly in our approach just the gates which do not change the functionality of the circuit commute. Depending on the type of search performed, the algorithm may not always get the best possible configuration; However, our algorithm can be much more efficient due to the lower execution time of the algorithm compared to previous methods.

To determine the time complexity of the proposed algorithm, let n and m be equal to the number of global gates and local gates, respectively. Each of the global gates in the circuit have a maximum l qubit.

$$\begin{cases} n = the\ number\ of\ qlobal\ gates \\ l = the\ maximum\ number\ of\ qubits\ in\ global\ gates \end{cases}$$

In the worst case,$(n-1)$ comparisons are required for each of the qubits of the first global gate in the $GGlist$ to find the next global gate to perform along with the first one. In addition, $(n-2)$ comparisons are needed for swapping the last global gate in the $GGlist$ with other global gates between this gate and first global gate in the $GGlist$. Also, in the worst case, there are m local gates before the last global gate. So, m comparisons are needed to swap local gates with this global gate. In addition, each global gate has l qubits. So, $l*((n-1)+(n-2)+m)$ comparisons are needed in this step. In the next step, for each of its l qubits of second global gate, $(n-2)+(n-3)$ comparisons are needed for swapping global gates and m comparisons are needed for local gates. So, the number of comparisons for the second global gate is equal to $l*((n-2)+(n-3)+m)$. In the same way, it becomes

$l*((n-3)+(n-4)+m)$ comparisons for the third global gate. If we calculate these values for all the global gates, it will be:

$$\begin{aligned} & l*((n-1)\ +\ (n-2)+m) \\ +& l*((n-2)\ +\ (n-3)+m) \\ +& l*((n-3)\ +\ (n-4)+m) \\ & \vdots \\ +& l*(2\ +\ 1+m) \\ +& l*(1\ +\ 0+m) \end{aligned} \tag{4}$$

Now, let factor l :

$$l*((n-1)+2((n-2)+(n-3)+\ldots+2+1))+(n-1)m \tag{5}$$

The second parentheses is a sum of $(n-2)$ items. By reordering the items and calculating them, we have:

$$\begin{aligned} & l*((n-1)+2\left(\frac{n-2)(n-3)}{2}\right)+(n-1)m\Big) = \\ & l*((n-1)+(n^2-5n+6)+(n-1)m) = \\ & l*((n^2-4n+5)+nm-m) \end{aligned} \tag{6}$$

In our work the maximum value of l is equal to three. Therefore, this parameter can be ignored and the time complexity of the algorithm is $\mathcal{O}(n^2+nm)$.

In the following, the proposed algorithm is applied to a DQC instance shown in the Fig. 5, and the step-by-step explanations are provided.

P, G and q stand for partition, gate and qubit, respectively. The first global gate is G_3, and G_6 is the second one. q_2 in P_1 is shared between these two gates. The algorithm checks whether it is possible to perform them by Teleporting q_2 to P_2. G_4 and G_5 are local gates located between them. G_4 does not have a common qubit to the teleported qubit; therefore, there is not any problem with performing this gate. G_5 has a common qubit to q_2. So, to perform this gate, the migrated-qubit must be returned to P_1 after performing G_3. Since G_6 has not been performed yet, it is necessary to teleport a qubit again to perform it. The algorithm checks to commute G_5 and G_6 which it is not possible. So, it checks to move G_5 to the left of G_3. All the gates between this gate and G_3 must be moved to the left of G_3. First, G_4 is moved to the left of G_3. Then, G_5 is moved to the left of G_3. This process is shown in the Fig. 5. Even if it is not possible to commute G_4 and G_3, the algorithm considers commuting G_4 itself to G_5. Then it commute G_4 with G_6 to move this gate to the right of G_6. So, the order of performing this part of the circuit will be as follows: first, the algorithm performs G_4 and G_5 locally. Then, q_2 is teleported to P_2. As a result, G_3 and G_6 are performed by this teleport.

In this step, the algorithm checks to add another global gates to perform with the previous teleportation. Another global gate is G_9 containing a common qubit to q_2, and the other qubit is in P_2. So, the algorithm checks whether it is possible to perform this gate with along previous teleportation. So, it has to move the gates between G_9 and G_6 to the right of G_9. Also, if the algorithm has moved gates from the previous steps to the right of G_6, it must move them to the right of G_9. If it is not possible to move some gates, it checks to move them to the left of the first global gate of this stage, G_3. According to the Fig. 6, G_7 and G_8 can be moved to the left of G_9. Therefore, G_9 will be added to the set of G_3 and G_6. So, the order of performing gates is changed to: first, G_4 and G_5 are performed locally. Then

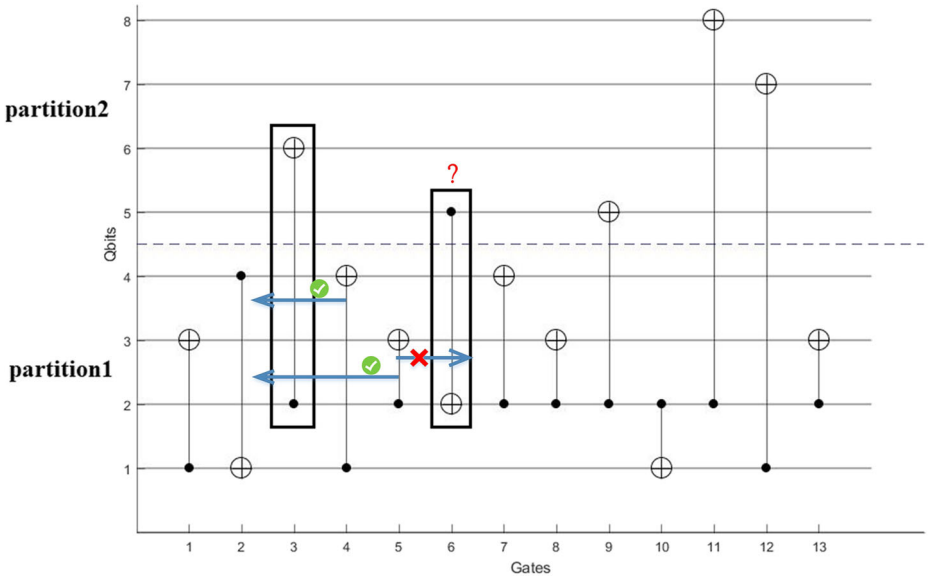

Fig. 5 G_4 and G_5 are moved to the left of G_3

q_2 is teleported to P_2 and G_3, G_6 and G_9 are performed. Finally q_2 is returned back to P_1. So, G_7 and G_8 are performed locally.

In the next step, the algorithm examines the next global gate, G_{11}. It Checks the gates between G_{11} and G_9 to move them to the right of G_{11}. G_{10} is commuted with G_{11}. In this step, the algorithm checks to move the gates to the right of G_{11}, which they were previously

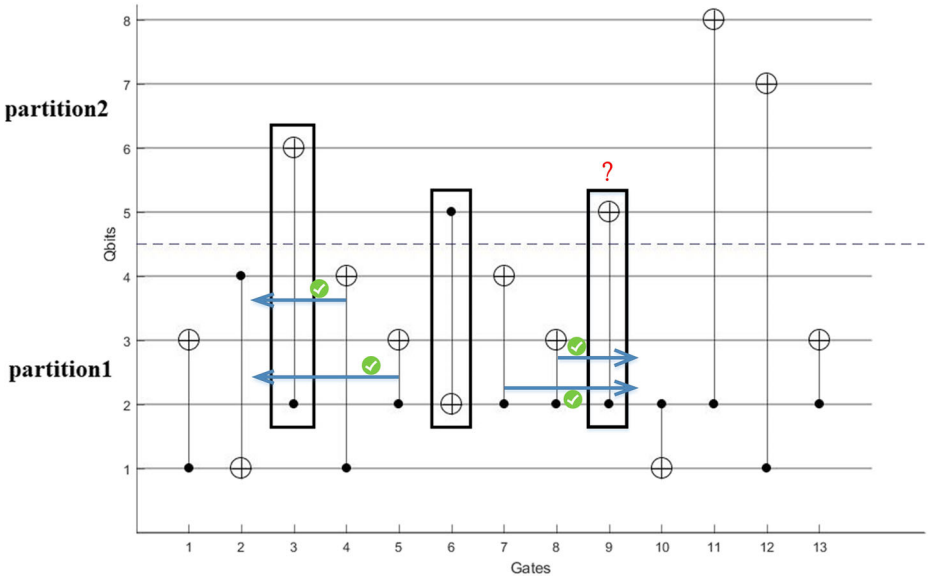

Fig. 6 G_4 and G_5 are moved to the left of G_3, then G_7 and G_8 are moved to the right of G_9

moved to the right of G_9. Because it is possible, it moves all the gates to the right of G_{11}. As shown in the Fig. 7, G_7 and G_8 are moved to the right of G_{11}. So, G_{11} can be added to the set of G_3, G_6 and G_9. Here again, if it is not possible to move some of the gates to the right of G_{11}, it checks to move them to the left of G_3. So, G_{11} can be added to the previous set. The order of execution changes as follows: G_3 and G_4 are performed locally. Then q_2 is teleported to P_2, and G_3, G_6, G_9 and G_{11} are performed. Finally, q_2 is returned back to P_1. So, G_7, G_8 and G_{10} are performed locally.

According to the conditions defined in the algorithm, it is no longer possible to add another gate to this set. This round of the algorithm ends, and the algorithm will continue with the first non-performed global gate in the next round. So, G_{12} is performed by a teleportation. Then G_{13} is performed locally.

5 Results

We implemented our proposed approach in MATLAB on a workstation with a 2.0 GHz CPU and 4 GB RAM, to find the right configuration decreasing the number of teleportations in DQCs and measuring the execution time. In this paper, we consider teleportation cost (TC) to be equal to the number of teleportations required to perform the circuit. Many different QCs were used to test the performance of the proposed algorithm. A series of tests were the Quantum Fourier Transform (QFT (n)) circuits for $n \in \{4, 8, 16, 32, 64\}$ which n is the number of qubits in the circuit. The QFT circuits have been used in some quantum algorithms, such as the Shor. Another set of QCs were taken from Revlib [45] library circuits as test circuits. In our approach, such QCs as parity_247, 4gt5-76, sym9_147, alu_primitive and alu_primitive_opt are used. In order to be able to compare our results with

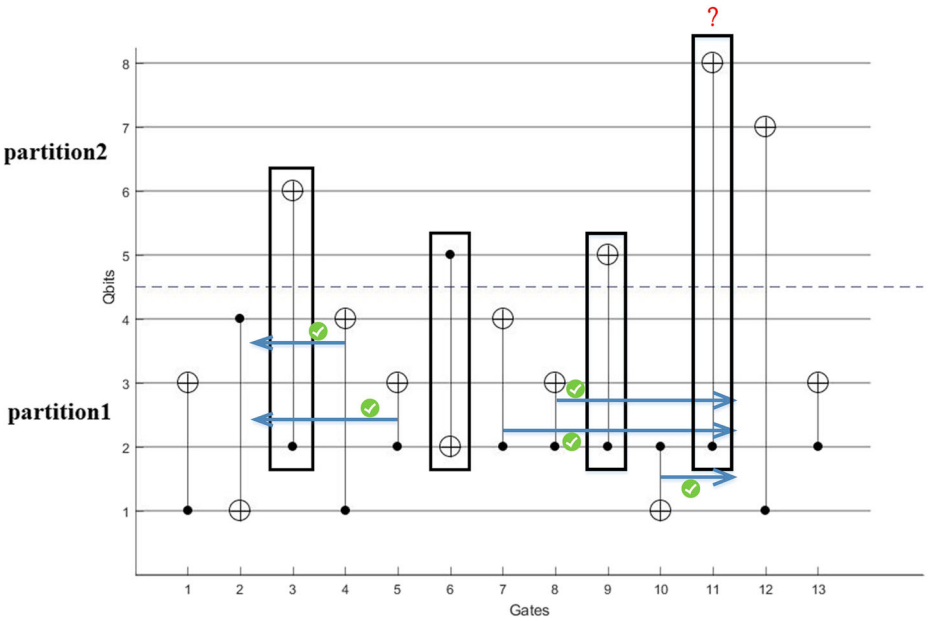

Fig. 7 G_4 and G_5 are moved to the left of G_3, then G_7, G_8 and G_{10} are moved to the right of G_{11}

Table 1 Comparison of Proposed Approach (P) With the [21] and GA [22]

Circuit	# of qubits	K	# of global gates (GA)	# of global gates (P)	TC [21]	TC (GA)	TC (P)	TC. imp [21] (%)	TC. imp (GA) (%)	Time (S) [21]	Time (S) (GA)	Time (S) (P)	Speed-up [21]	Speed-up (GA)
parity_247	17	2	9	8	2	2	2	0.00	0.00	1028.63	212.78	0.41	2522.39	521.78
	17	3	N.A.	8	N.A.	N.A.	6	N.A.	N.A.	N.A.	N.A.	0.61	N.A.	N.A.
	17	4	N.A.	12	N.A.	N.A.	10	N.A.	N.A.	N.A.	N.A.	0.74	N.A.	N.A.
4gt5-76	5	2	13	13	14	14	10	28.57	28.57	1194.92	275.78	0.68	1754.65	404.96
	5	3	N.A.	20	N.A.	N.A.	10	N.A.	N.A.	N.A.	N.A.	0.77	N.A.	N.A.
	5	4	N.A.	20	N.A.	N.A.	22	N.A.	N.A.	N.A.	N.A.	0,788	N.A.	N.A.
alu_primitive_opt	6	2	13	8	10	10	4	60.00	60.00	4874.24	1182.51	0.44	11077.82	2687.52
	6	3	N.A.	10	N.A.	N.A.	12	N.A.	N.A.	N.A.	N.A.	0.45	N.A.	N.A.
	6	4	N.A.	13	N.A.	N.A.	16	N.A.	N.A.	N.A.	N.A.	0.55	N.A.	N.A.
alu_primitive	6	2	18	8	18	20	4	77.78	80.00	154020.53	2183.31	0.46	334827.24	4746.33
	6	3	N.A.	10	N.A.	N.A.	14	N.A.	N.A.	N.A.	N.A.	0.57	N.A.	N.A.
	6	4	N.A.	13	N.A.	N.A.	16	N.A.	N.A.	N.A.	N.A.	0.65	N.A.	N.A.
sym9_147	12	2	54	37	N.A.	48	18	N.A.	62.50	N.A.	10930.57	1.27	N.A.	8627.81
	12	3	N.A.	40	N.A.	N.A.	22	N.A.	N.A.	N.A.	N.A.	1.09	N.A.	N.A.
	12	4	N.A.	66	N.A.	N.A.	36	N.A.	N.A.	N.A.	N.A.	1.37	N.A.	N.A.
4 modulo7	5	2	N.A.	18	10	10	8	20.00	20.00	1013.36	292.26	0.7303	1387.59	400.19
	5	3	N.A.	18	N.A.	N.A.	14	N.A.	N.A.	N.A.	N.A.	0.88	N.A.	N.A.
	5	4	N.A.	27	N.A.	N.A.	30	N.A.	N.A.	N.A.	N.A.	0.98	N.A.	N.A.

Table 1 (continued)

Circuit	# of qubits	K	# of global gates (GA)	# of global gates (P)	TC [21]	TC (GA)	TC (P)	TC. imp [21] (%)	TC. imp (GA) (%)	Time (S) [21]	Time (S) (GA)	Time (S) (P)	Speed-up [21]	Speed-up (GA)
4-qubit QFT	4	2	8	8	8	8	4	50.00	50.00	875.62	170.13	0.33	2653.39	515.55
	4	3	N.A.	13	N.A.	N.A.	10	N.A.	N.A.	N.A.	N.A.	0.46	N.A.	N.A.
	4	4	N.A.	18	N.A.	N.A.	14	N.A.	N.A.	N.A.	N.A.	0.44	N.A.	N.A.
8-qubit QFT	8	2	32	32	N.A.	38	8	N.A.	78.95	N.A.	8059.4	0.57	N.A.	14062.82
	8	3	N.A.	40	N.A.	N.A.	20	N.A.	N.A.	N.A.	N.A.	0.77	N.A.	N.A.
	8	4	N.A.	48	N.A.	N.A.	24	N.A.	N.A.	N.A.	N.A.	0.67	N.A.	N.A.
16-qubit QFT	16	2	128	128	N.A.	133	16	N.A.	87.97	N.A.	19788.34	2.41	N.A.	8194.95
	16	3	N.A.	160	N.A.	N.A.	40	N.A.	N.A.	N.A.	N.A.	3.34	N.A.	N.A.
	16	4	N.A.	192	N.A.	N.A.	48	N.A.	N.A.	N.A.	N.A.	2.48	N.A.	N.A.
32-qubit QFT	32	2	512	512	N.A.	532	32	N.A.	93.98	N.A.	78734.89	25.82	N.A.	3049.14
	32	3	N.A.	640	N.A.	N.A.	80	N.A.	N.A.	N.A.	N.A.	39.19	N.A.	N.A.
	32	4	N.A.	768	N.A.	N.A.	96	N.A.	N.A.	N.A.	N.A.	29.31	N.A.	N.A.
64-qubit QFT	64	2	2048	2048	N.A.	2250	64	N.A.	97.16	N.A.	361645.67	431.69	N.A.	837.74
	64	3	N.A.	2560	N.A.	N.A.	160	N.A.	N.A.	N.A.	N.A.	676.92	N.A.	N.A.
	64	4	N.A.	3072	N.A.	N.A.	192	N.A.	N.A.	N.A.	N.A.	539.51	N.A.	N.A.

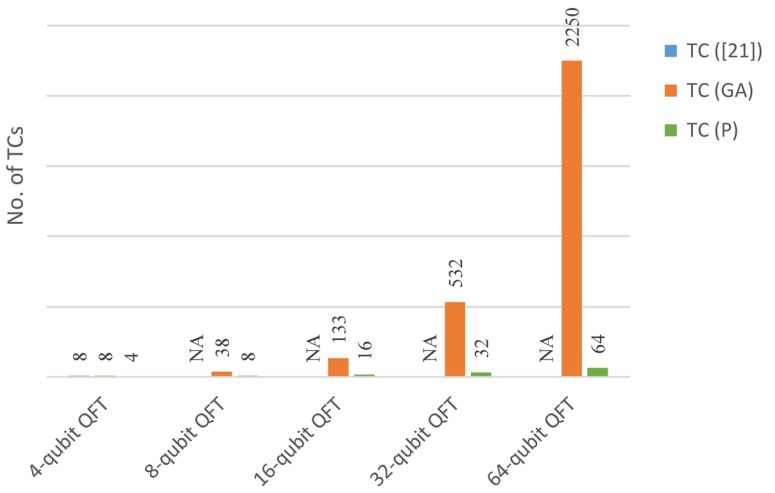

Fig. 8 The number of teleportations of the proposed approach as compared to that of [21] and GA [22] in the QFT circuits

other approaches, we employed the circuits used in [21, 22]. The method presented in [22] is called GA by the authors. Accordingly, we employ the same name to compare with our results. We applied our algorithm to benchmark circuits for $K = 2, 3, 4$ which K stands for the number of partitions. Table 1 indicates the results of the comparison of our work with other two methods in terms of teleportation cost and execution time for eleven different benchmark circuits.

In the Figs. 8 and 9 the number of teleportations of proposed approach compared to [21] and GA are presented for different circuits. The Fig. 8 shows the QFTs and the Fig. 9 represents other benchmark circuits. In both of the figures, The horizontal axis represents the different circuits per $K = 2$, and vertical axis represents the number of teleportations. Since the results of the other works were unavailable for partitions with more than two

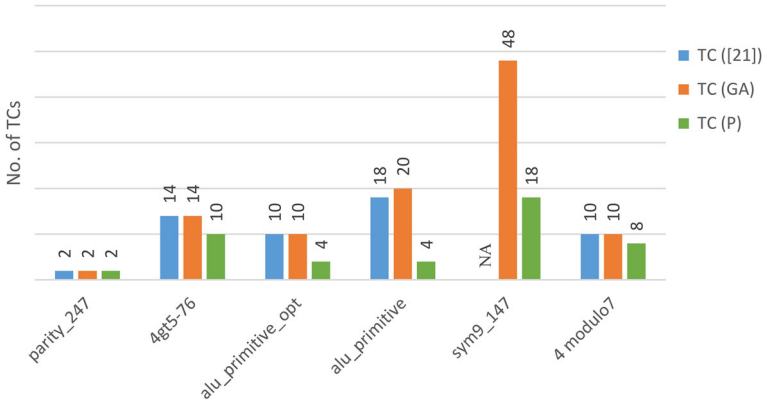

Fig. 9 The number of teleportations of the proposed approach as compared to that of [21] and GA [22] in other circuits except the QFTs

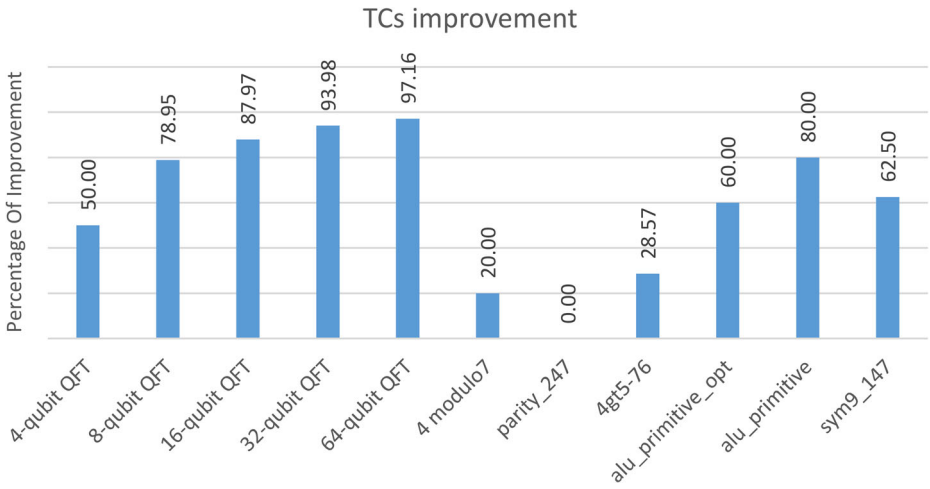

Fig. 10 Percentage of the teleportation cost improvement in the proposed approach (P) compared to the GA [22]

which are shown by NA, the results of $K = 3$ and $K = 4$ are measured and presented in Table 1 but they are not used for comparison in the charts.

Because the commuting of gates, as a new discussion, is considered in the proposed method, teleportation cost may be significantly reduced in some circuits by moving a number of gates. This method has been effective in our results, as it is evolved. In parity_247, the results of TC are equal to those of others, but there is a significant improvement in other circuits. This improvement becomes much more significant especially in the QFT circuits where the number of qubits increases. For instance, the number of teleportations is 64 in the QFT (64), and it is 2250 in GA. As the number of qubits in the QFT circuits increases, the number of teleportations in our method increases linearly, whereas this number in GA

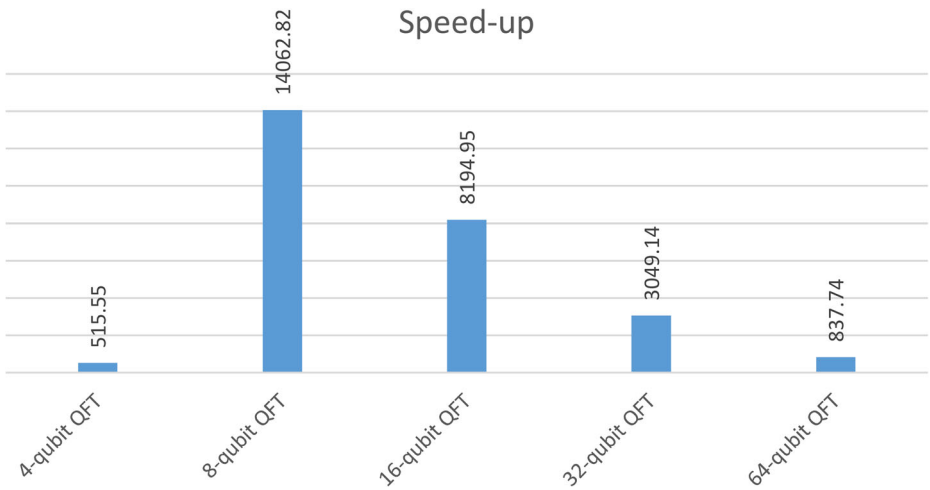

Fig. 11 Speed-up of the proposed approach (P) compared to GA [22] in the QFT circuits

is not linear and it increases too much. The teleportation cost improvement of the proposed method for all the circuits per $K = 2$, is shown in the Fig. 10 compared to GA. The horizontal axis presents the circuits and the vertical axis shows the percentage of teleportation cost improvement.

Algorithms that examine all possible scenarios to find the optimal execution order usually have a high execution time. In DQCs as the number of connections between the parts of the DQC and the number of qubits increase, the execution time in these algorithms also increase exponentially. For this reason, the algorithm in [21] has a high execution time, and as the number of global gates and qubits increases, this time also increases significantly. For instance, in the alu_primitive and alu_primitive_opt circuits, the number of qubits is six on the both, and the number of global gates are 18 and 13, respectively. The execution time of these two circuits by the method presented in [21] are 154020 and 4874 seconds, respectively. But if algorithms that do not examine all cases are used, this time is greatly reduced. Obviously, they may not obtain the most satisfactory results. For instance, if we measure the execution time of the two mentioned circuits by GA method, these values are 2183 and 1182 seconds, respectively. These values show a great improvement over the previous method. These values for the two mentioned circuits employing the proposed method are just 0.46 and 0.44 seconds, respectively. Execution time in this algorithm does not increase exponentially with increasing the number of global gates and the qubits. Compared to the proposed method and method in [21], the value of speed-up for the alu_primitive circuit, which has 18 global gates, is obtained a very significant value. Also, in terms of speed-up of the proposed algorithm, compared to the other two methods, there is a significant improvement for all the circuits. In the eight of eleven benchmark circuits, the execution time of our algorithm is less than two seconds. While the closest time in the other two compared methods are 170 seconds belonging to the QFT (4) circuit and 212 seconds belonging to the parity 247. The Figs. 11 and 12 represents our method's speed-up compared to [21] and GA. Because the results were not available for the QFTs in [21], only the results of the QFTs were compared

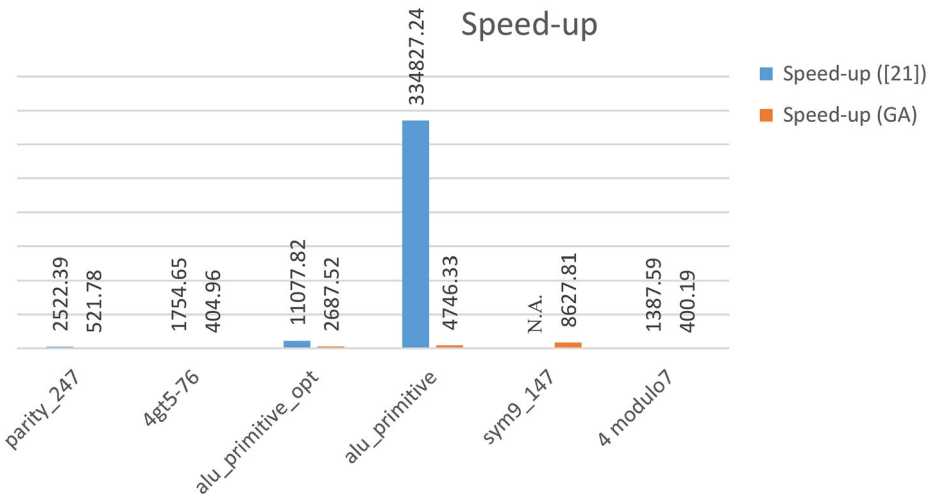

Fig. 12 Speed-up of the proposed approach (P) compared to [21] and GA [22] in other circuits except the QFTs

to the GA method. The minimum speed-up values belong to the 4modulo7 and 4gt5-76 circuits, which are measured 400 and 404, respectively. The speed-up of the proposed method of the QFT circuits per $K = 2$ compared to GA is shown in the Fig. 11 and the other benchmark circuits compared to [21] and GA are presented in the Fig. 12. In the both, the horizontal axis presents the circuits and the vertical axis represents the speed-up.

6 Conclusion

This paper proposed an approach based on a heuristic algorithm to determine the order of global gates execution in the DQCs leading to a minimal number of teleportations. Compared to the method introduced in [21], our proposed algorithm has far better results and much shorter execution time. Also, in comparison to the [22] using genetic algorithm, our proposed algorithm achieved both much better results and much faster execution time. It was also found that the greater consecutive global gates with a common qubit in a circuit, The more improvements can be made by using our approach. the QFT circuits have this feature so the results of our proposed method for the QFTs are better than two compared methods. The results show that the number of teleportations on average is reduced by 81%, compared to the GA [22] in the QFTs, and 39% in other circuits compared to [21] and GA. Moreover, since the execution time of our algorithm is considerably has better performance compared to [21] and GA.

References

1. Hassan, S., Asghar, M., et al.: In: 2010 Second International Conference on Communication Software and Networks, pp. 559–561. IEEE (2010)
2. Grover, L.K.: Phys. Rev. Lett. **79**(2), 325 (1997)
3. MacQuarrie, E.R., Simon, C., Simmons, S., Maine, E.: Nat. Rev. Phys. **2**(11), 596 (2020)
4. Shor, P.: Conference Publications, Spain (1997)
5. Montanaro, A.: npj Quant. Inform. **2**(1), 1 (2016)
6. Krenn, M., Malik, M., Scheidl, T., Ursin, R., Zeilinger, A.: Optics Our Time **18**, 455 (2016)
7. Cuomo, D., Caleffi, M., Cacciapuoti, A.S.: arXiv:2002.11808 (2020)
8. Mishra, N., Kapil, M., Rakesh, H., Anand, A., Mishra, N., Warke, A., Sarkar, S., Dutta, S., Gupta, S., Dash, A.P., et al.: Data management. Analytics and Innovation 101–145 (2021)
9. Dunjko, V., Briegel, H.J.: Rep. Prog. Phys. **81**(7), 074001 (2018)
10. Easttom, W.: Modern Cryptography, pp. 385–390. Springer, Berlin (2021)
11. Caleffi, M., Cacciapuoti, A.S., Bianchi, G.: In: Proceedings of the 5th ACM International Conference on Nanoscale Computing and Communication, pp. 1–4 (2018)
12. Arute, F., Arya, K., Babbush, R., Bacon, D., Bardin, J.C., Barends, R., Biswas, R., Boixo, S., Brandao, F.G., Buell, D.A.: Nature **574**(7779), 505 (2019)
13. Van Meter, R., Ladd, T.D., Fowler, A.G., Yamamoto, Y.: Int. J. Quant. Inform. **8**(01n02), 295 (2010)
14. Krojanski, H.G., Suter, D.: Phys. Rev. Lett. **93**(9), 090501 (2004)
15. DiAdamo, S., Ghibaudi, M., Cruise, J.: arXiv:2101.02504 (2021)
16. Bennett, C.H., Brassard, G., Crépeau, C., Jozsa, R., Peres, A., Wootters, W.K.: Phys. Rev. Lett. **70**(13), 1895 (1993)
17. Whitney, M., Isailovic, N., Patel, Y., Kubiatowicz, J.: In: Proceedings of the 4th international conference on Computing frontiers, pp. 83–94 (2007)
18. Bouwmeester, D., Pan, J.W., Mattle, K., Eibl, M., Weinfurter, H., Zeilinger, A.: Nature **390**(6660), 575 (1997)
19. Metcalf, B.J., Spring, J.B., Humphreys, P.C., Thomas-Peter, N., Barbieri, M., Kolthammer, W.S., Jin, X.M., Langford, N.K., Kundys, D., Gates, J.C.: Nature Photonics **8**(10), 770 (2014)
20. Wootters, W.K., Zurek, W.H.: Nature **299**(5886), 802 (1982)
21. Zomorodi-Moghadam, M., Houshmand, M., Houshmand, M.: Int. J. Theor. Phys. **57**(3), 848 (2018)

22. Houshmand, M., Mohammadi, Z., Zomorodi-Moghadam, M., Houshmand, M.: Int. J. Theor. Phys. **59**(4), 1315 (2020)
23. Davarzani, Z., Zomorodi-Moghadam, M., Houshmand, M., Nouri-baygi, M.: Quantum Inf. Process **19**(10), 1 (2020)
24. Nielsen, M.A., Chuang, I.: Quant. Comput. Quant. Inform. (2002)
25. Zomorodi-Moghadam, M., Navi, K.: J. Circ. Syst. Comput. **25**(12), 1650152 (2016)
26. Barenco, A., Bennett, C.H., Cleve, R., DiVincenzo, D.P., Margolus, N., Shor, P., Sleator, T., Smolin, J.A., Weinfurter, H.: Phys. Rev. A **52**(5), 3457 (1995)
27. Möttönen, M., Vartiainen, J.J., Bergholm, V., Salomaa, M.M.: Phys. Rev. Lett. **93**(13), 130502 (2004)
28. Ballico, E., Bernardi, A., Carusotto, I., Mazzucchi, S., Moretti, V.: Quantum Physics and Geometry. Springer, Berlin (2019)
29. Ying, M., Feng, Y.: IEEE Trans. Comput. **58**(6), 728 (2009)
30. Houshmand, M., Hosseini-Khayat, S., Wilde, M.M.: IEEE Trans. Comput. **61**(3), 299 (2010)
31. Beals, R., Brierley, S., Gray, O., Harrow, A.W., Kutin, S., Linden, N., Shepherd, D., Stather, M.: Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences **469**(2153), 20120686 (2013)
32. Andrés-Martínez, P., Heunen, C.: Phys. Rev. A **100**(3), 032308 (2019)
33. Yepez, J.: Int. J. Modern Phys. C **12**(09), 1273 (2001)
34. Liu, T.: Journal of Physics: Conference Series vol. 1634, vol. 1634, p. 012089. IOP Publishing, Bristol (2020)
35. Valivarthi, R., Zhou, Q., Aguilar, G.H., Verma, V.B., Marsili, F., Shaw, M.D., Nam, S.W., Oblak, D., Tittel, W., et al.: Nat. Photonics **10**(10), 676 (2016)
36. Caleffi, M., Cacciapuoti, A., Cataliotti, F., Gherardini, S., Tafuri, F., Bianchi, G.: arXiv:1810.08421 (2019)
37. Yimsiriwattana, A., Lomonaco, S.J.Jr.: In: Quantum Information and Computation II, vol. 5436. International Society for Optics and Photonics. International Society for Optics and Photonics, vol. 5436, pp. 360–372 (2004)
38. Meter, R.V., Munro, W., Nemoto, K., Itoh, K.M.: ACM Journal on Emerging Technologies in Computing Systems (JETC) **3**(4), 1 (2008)
39. Chen, M.C., Li, R., Gan, L., Zhu, X., Yang, G., Lu, C.Y., Pan, J.W.: Phys. Rev. Lett. **124**(8), 080502 (2020)
40. Cacciapuoti, A.S., Caleffi, M., Van Meter, R., Hanzo, L.: IEEE Transactions on Communications (2020)
41. Humble, T.S., Thapliyal, H., Munoz-Coreas, E., Mohiyaddin, F.A., Bennink, R.S.: IEEE Design and test **36**(3), 69 (2019)
42. Daei, O., Navi, K., Zomorodi-moghadam, M.: Int. J. Theor. Phys. **59**(12), 3804 (2020)
43. Houshmand, M., Zamani, M.S., Sedighi, M., Arabzadeh, M.: ACM Journal on Emerging Technologies in Computing Systems (JETC) **11**(3), 1 (2014)
44. Houshmand, M., Sedighi, M., Zamani, M.S., Marjoei, K.: ACM Journal on Emerging Technologies in Computing Systems (JETC) **13**(4), 1 (2017)
45. Wille, R., Große, D., Teuber, L., Dueck, G.W., Drechsler, R.: In: 38th International Symposium on Multiple Valued Logic (ismvl 2008), pp. 220–225 (2008)