International Journal of Theoretical Physics (2020) 59:1315–1329
https://doi.org/10.1007/s10773-020-04409-0

An Evolutionary Approach to Optimizing Teleportation Cost in Distributed Quantum Computation

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

Received: 13 October 2019 / Accepted: 4 February 2020 / Published online: 27 February 2020

Abstract
Distributed quantum computing has been well-known for many years as a system composed of a number of small-capacity quantum circuits. Limitations in the capacity of monolithic quantum computing systems can be overcome by using distributed quantum systems which communicate with each other through known communication links. In our previous study, an algorithm with an exponential complexity was proposed to optimize the number of qubit teleportations required for the communications between two partitions of a distributed quantum circuit (DQC). In this work, a genetic algorithm is used to solve the optimization problem in a more efficient way. The results are compared with the previous study and we show that our approach works almost the same with a remarkable speed-up. Moreover, the comparison of the proposed approach based on GA with a random search over the search space verifies the effectiveness of GA.

Keywords Communication cost · Distributed quantum computation · Genetic algorithms (GA) · Optimization · Teleportation

1 Introduction

Quantum computing is one of the emerging technologies in computation with a great potential to outperform classical computers [1–3]. The theory of quantum computing is getting

✉ Mahboobeh Houshmand
houshmand@mshdiau.ac.ir

Zahra Mohammadi
mohammadi.m8609@gmail.com

Mariam Zomorodi-Moghadam
m_zomorodi@um.ac.ir

Monireh Houshmand
m.hooshmand@imamreza.ac.ir

[1] Department of Computer Engineering, Islamic Azad University, Mashhad Branch, Mashhad, Iran

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

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

more and more mature since it was initiated by Feynman and Deutsch in the 1980s. Quantum computation has revolutionized the computer science, showing that the processing of quantum states can lead to a remarkable speed up in the solution of a class of problems, as compared to the traditional algorithms that process classical bits.

One of the implementation limitations of quantum computing is due to the interactions of qubits with the environment. When the number of qubits increases, the quantum information becomes more fragile and more susceptible to errors [4]. On the other hand, error-correction codes require the involvement of many qubits just for one logical qubit and so a number of logical qubits may not fit on a single quantum chip [5]. To overcome this problem, a distributed 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 finite quantum computers which are interconnected through a quantum or classical channel and they can implement the behavior of the whole quantum system. This structure is known as "distributed quantum computing" [5]. It should be noted that distributed quantum computing in the sense implied in this study is different with anything that is "distributed and quantum", e.g., quantum key distribution protocols [6, 7] in which a secret key is classically distributed between two distant parties.

The circuit model for quantum computation can be expanded to the case of distributed quantum computation where each subsystem sends its data on demand to the other sites through a communication channel. A reliable mechanism for such communication is by using quantum teleportation between nodes of a distributed quantum system.

Quantum teleportation is one of the most promising ways for quantum state movement and its validity has been demonstrated experimentally in some studies including [8–10]. The basic idea in the quantum teleportation is to transfer qubit states without moving them physically. The original state is destroyed such that quantum teleportation does not contract with the no-cloning theorem. The no-cloning theorem states that it is impossible to produce an identical copy of a qubit which is initially in an arbitrary state. Quantum teleportation is composed of some phases including creating an EPR pair, doing some local operations, measurement, and classical communication [11, 12].

In the distributed realization, communication can be done by teleporting qubits between communication links. Although quantum circuit for teleportation is a much simpler circuit compared to any nontrivial quantum computational task [13], having many such teleportation circuits, maybe a thousand, in interconnection links results in high communication costs for distributed quantum circuits. A non-functional property [14] of a distributed quantum system, which is a constraint on the manner the system performs its functionality, is its efficiency. Efficiency is a quality that reflects a system's ability to meet its performance requirements while minimizing its usage of resources. In this study, minimizing the number of teleportations between nodes of a distributed quantum computer is considered as a measure of its efficiency.

In our previous study [15], with two given quantum subsystems, an algorithm with an exponential complexity was proposed to optimize the number of qubit teleportations required for the communication between these two subsystems. In this work, the genetic algorithm [16] is used to solve the optimization problem in a more efficient way.

In Section 2, the required background is given. The related work of the distributed quantum computing is presented in Section 3. Our proposed approach and the results of the work are explained in Sections 4 and 5, respectively. We conclude the paper in Section 6.

2 Background

In this study, it is assumed that data transfer between distributed nodes in the system is performed by quantum teleportation. The basic idea in quantum teleportation is to transfer an unknown quantum state of a quantum bit (qubit) using two classical bits in a way that the receiver reproduces exactly the same state as the original qubit state. The original state is destroyed such that quantum teleportation does not contract with the no-cloning theorem. The no-cloning theorem [17] is one of the earliest results of quantum computation and quantum information which states that an unknown quantum system cannot be cloned by unitary transformations.

Basically, in quantum teleportation, two parties, referred to as Alice and Bob, share an entangled pair of qubits, e.g., $\left|\beta_{00}\right\rangle$ (defined in (1)). *Entanglement* is a quantum mechanical phenomenon that plays a key role in many of the most interesting applications of quantum computation and quantum information. A multi-qubit quantum state $|v\rangle$ is said to be entangled if it cannot be written as the tensor product $|v\rangle = |\phi_1\rangle \otimes |\phi_2\rangle$ of two pure states. For example, the EPR pair [17] shown below is an entangled quantum state:

$$\left|\beta_{00}\right\rangle = (|00\rangle + |11\rangle)/\sqrt{2} \tag{1}$$

Alice attempts to send an unknown qubit, $|\psi\rangle$, to Bob. The overall state of the system ($|\phi\rangle$) is as follows:

$$|\phi\rangle = |\psi\rangle \otimes \left|\beta_{00}\right\rangle = (\alpha\,|0\rangle + \beta\,|1\rangle) \otimes (\tfrac{|00\rangle+|11\rangle}{\sqrt{2}}) = \\ \tfrac{1}{\sqrt{2}}(\alpha(|000\rangle + |011\rangle) + \beta(|000\rangle + |111\rangle))$$

The first two qubits belong to Alice and Bob has the third qubit. Alice applies a CNOT gate to the first two qubits and then she applies a Hadamard gate to the first qubit which results in:

$$\tfrac{1}{2}[a(|000\rangle + |011\rangle + |100\rangle + |111\rangle) + b(|010\rangle + |001\rangle - |110\rangle - |101\rangle)] \\ = \tfrac{1}{2}[|00\rangle\,(a\,|0\rangle + b\,|1\rangle) + |01\rangle\,(a\,|1\rangle + b\,|0\rangle) + |10\rangle\,(a\,|0\rangle - b\,|1\rangle) + |11\rangle\,(a\,|1\rangle - b\,|0\rangle)]$$

Finally, Alice measures both qubits that belong to her. She will obtain one of the four outcome states of $|00\rangle$, $|01\rangle$, $|10\rangle$ or $|11\rangle$ with the equal probability of $\frac{1}{4}$. Depending on the result of Alice's measurement, Bob's qubit collapses to $a\,|0\rangle + b\,|1\rangle$, $a\,|1\rangle + b\,|0\rangle$, $a\,|0\rangle - b\,|1\rangle$ or $a\,|1\rangle - b\,|0\rangle$, respectively. Alice then sends the results of her measurement to Bob using two classical bits. Alice's initial qubit, $|\psi\rangle$ is totally destroyed upon her measurement which makes the quantum teleportation consistent with the no-cloning theorem. Finally, after receiving the two classical bits, Bob can know the state of the qubit at his hand by applying I, X, Z and Y, if the classical bits are 00, 01, 10 and 11 respectively to reconstruct the initial state of Alice. Figure 1 shows the circuit for the entire teleportation protocol as stated above.

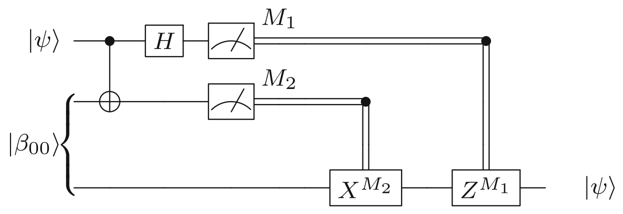

Fig. 1 Quantum circuit for teleporting a qubit [17]. The two top lines belong to Alice, while Bob has the third qubit. The meters represent quantum measurement, and the double lines coming out of them carry classical bits

Another important quantum circuit which is also one of the components of Shor's algorithm [1] is Quantum Fourier Transform (QFT). As Shor's algorithm has potential application in breaking the RSA encryption protocol, implementation of this algorithm has been extensively investigated in the literature. Figure 2 shows QFT for an n-qubit state.

3 Related Work

In this section, a brief review on applying evolutionary algorithms in different domains of quantum computing and also the research studies on distributed quantum computing is given. An evolutionary algorithm is a generic population-based metaheuristic algorithm which uses mechanisms inspired by biological evolution, such as mutation, crossover, natural selection and survival of the fittest for solving optimization problems. Candidate solutions to the optimization problem play the role of individuals in a population, and the fitness function determines the quality of the solutions. Evolution of the population then takes place after the repeated application of the above mentioned operators. Evolutionary algorithms have been successfully used in different aspects of quantum computing. For example, genetic programming can be used to evolve quantum programs and algorithms [18]. Having a simulator for a quantum computer, genetic programming can be used to calculate the fitness of a program in the population on the simulated quantum computer. In [19], a review on how evolutionary algorithms have been used to evolve quantum circuits is given. In that study, issues in representing quantum artefacts in a form suitable for evolutionary search are discussed and it is stated that the basic approach uses bit strings to encode the search space and quantum gates. Moreover, different quantum artefacts that have been discovered through evolutionary search, an example of which QFT, are presented. In [20], genetic programming has been used to find the best quantum circuit for a given quantum algorithm. In that study, the authors have tested different selection strategies for the evolutionary quantum circuit design and showed that the tournament selection and the self-adaptation have been effective on the test problems. The approach in [21] uses a new technique to evolve a quantum circuit, different from [20] and the presented programs and circuits were capable of solving specific problem instances and one general problem. Also in another work [22], a new quantum circuit has been evolved for the two-query AND/OR problem. An efficient approach based on genetic algorithms is proposed to find the stabilizers of a given sub space in quantum information in [23].

Realizing a quantum computer has many obstacles. There are technological limitations on the number of qubits that can be used for building a monolithic quantum computing device [24]. These limitations are the main causes for the emerging of distributed quantum computing. Grover [3], Cleve and Buhrman [25] and later Cirac et.al. [26] were the first ones to propose distributed model of quantum computing. In [3], Grover presented a distributed

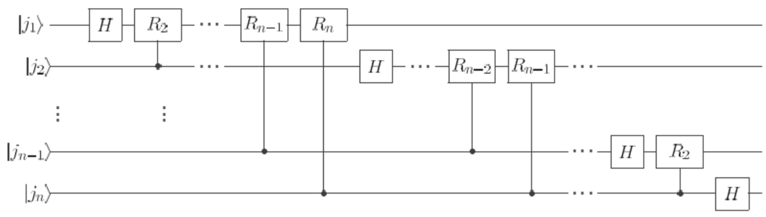

Fig. 2 QFT Circuit

quantum system where there are some particles at separated locations and each one performs its computation and sends the required information to a base station when necessary. Grover showed that using this distributed approach, the overall computation time is proportional to the number of such distributed particles. Beals et.al [27] showed that an arbitrary quantum circuit can be emulated by a distributed quantum circuit with nodes connected using a hypercube graph. Yepez [28] presented an architecture for distributed quantum computing with two types of communication which were called Type I and Type II. A Type-I quantum computer uses quantum communication between subsystems and Type-II quantum computer exploits classical communication between the subsystems of the distributed computer. In Type-I quantum computers, each qubit can be entangled with any number of qubits. On the other hand, a Type-II quantum computer is a network of small quantum systems and communication between them is accomplished by means of classical communication channels.

Related to teleportation cost, Streltsov et al. [29] 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 cost for sending the ancillary particle and sending back that particle. Classical communication cost is the subject of the study in [30]. Authors then conjectured 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, $2 \log 2N$ bits of classical communications 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 [31] provided some definitions of a distributed quantum computing system and defined an algebraic language to describe quantum circuits for distributed quantum computing.

In [32], a distributed circuit model of a monolithic quantum circuit has been presented. In that study, the functionality of a 2-qubit VBE adder is distributed over two nodes. The adder has been split into two almost equal quantum circuits and the communication between these nodes was performed using teleportation. The authors have not considered multiple issues regarding the distributed version of the monolithic circuit. The first issue is about finding the minimal number of qubits for teleporting. The presented idea in that study works for small circuits, however, for large quantum circuits, finding the qubits for teleportation that leads to the minimal quantum cost is an important challenge. Another issue in [32] is about having some consecutive gates in the target node tha use the teleported qubit as one of their inputs. In their sample case, the teleported qubit is used consecutively in two gates of the destination node, but this is not always the case. Sometimes the teleported qubit is required in the source node and so somewhere in the middle the qubit should be returned back by another teleportation and this will lead to larger distributed circuits and cost in the system.

Another distributed implementation of a quantum circuit can be found in [33] where the distributed quantum circuit for Shor's algorithm using non-local gates has been implemented. The additional overhead for this distributed quantum circuit has been calculated in terms of the number of teleportation circuits, but no attempt has been done to reduce the number of teleportations or to justify that the mentioned circuit is minimal in terms of teleportation circuits.

In our previous study [15], with two given quantum subsystems, an algorithm with an exponential complexity was proposed to optimize the number of qubit teleportations required for the communication between these two subsystems. Reducing the problem of quantum circuit partitioning to the graph partitioning model has been proposed in [34]. The authors have mapped the quantum circuit into a hypergraph and then they have used the state-of-the-art hypergraph partitioning algorithms for partitioning the graph into multiple quantum circuits. The authors have considered the case where different quantum gates have a common control or common target and then for each case they have built a hyperedge connecting common qubits and non-common qubits which meet at one end. But the authors have not considered any optimization like moving gates back and forth for making them close to each other as the approach presented in [15]. They have not also taken into account the entire search space of different partitioning and different partition for executing global gates and hence cannot produce optimal solutions.

4 Proposed Approach

In this section, the proposed approach for minimizing the number of teleportations in a distributed quantum circuit is presented.

4.1 Problem Definition

A distributed quantum circuit which is also called a quantum multicomputer [32, 35], consists of N-limited capacity quantum circuits (QCs) or partitions which are physically located far from each other and altogether emulate the functionality of a large quantum system. Partitions of a DQC communicate by sending their qubits to each other. In each partition, qubits are numbered from top to bottom where the i^{th} line represents the i^{th} qubit.

We intend to start with a quantum circuit QC, composed of basic gate library [36], i.e., CNOT and single-qubit gates, already split into two parts. It is already known [36–38] that arbitrary n-qubit quantum gates can be decomposed to the basic gate library. After that, the quantum circuit is represented by a graph [15] and the initial partitioning of the circuits into two partitions is performed by the Kernighan-Lin (K-L) algorithm as used in the very large-scale integration (VLSI) design algorithms [39]. The K-L algorithm is a heuristic algorithm for solving the graph partitioning problem. We use it to partition a graph into two partitions in such a way the interconnection cost between different partitions is minimized. The complexity of the K-L algorithm is $O(n^2log(n))$ where n is the number of nodes in the graph which is equal to the number of qubits in our problem.

A sample input to the proposed approach, i.e., a QC split into two parts is given Fig. 3. We define two types of CNOT gates, namely, local and global gates. A *local* CNOT gate is the one whose control and target qubits belong to the same partition. A *global* CNOT gate is the one whose control and target qubits belong to different partitions.

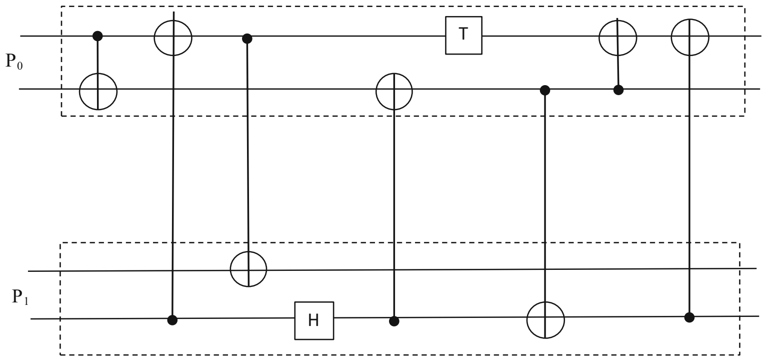

Fig. 3 A sample QC split into two parts

The partition to which each qubit q of a global CNOT gate belongs is called the *home* partition of q. 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.

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. $Config\text{-}Arr$, 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 $g.l$.

Here two questions arise:

- When a global CNOT gate is supposed to be executed, which qubit of that global gate, namely the control or the target qubit, should be teleported to the other system? Does the answer influence the teleportation cost?
- Another question is when the teleported qubit should be returned back to its home partition? Does the answer influence the teleportation cost?

The answer to the first question is related to the second one. Therefore, we first address the second question. It is clear that when a qubit is transferred into another partition, it no longer exists in its home partition, so it should be returned to its home partition for local gate executions. In this regard, the existing works such as [32] assume that as soon as an operation is applied to the teleported qubit, it returns back by another teleportation.

In order to perform global CNOT gates, both their qubits should exist in the same partition. Teleportation is the natural way of transporting qubit states. In a two-partite system, with Partitions A and B, there are two cases for executing each gate. One is the teleportation of the qubit in Partition A to Partition B and then executing the gate in Partition B and vice versa.

In [15], an exact solution has been given and an algorithm has been proposed to minimize the number of teleportations in a two-partite distributed quantum circuit. The approach calculates the minimal number of teleportations for each configuration of global gates, where each configuration has a unique placement for global gates being in either Partition A or B. Finally, the minimal number of teleportations for all of the configurations is returned. For a two-partite system with m_g global gates, there are 2^{m_g} different configurations of executing global gates whose consideration takes an exponential complexity of $O(2^{m_g})$.

In order to overcome this problem, in this paper, a heuristic approach based on genetic algorithms is proposed which attempts to find a configuration of global gates with an optimized teleportation cost. Each chromosome, i.e., the individuals in the GA population of potential solutions, shows a configuration whose fitness is computed by *MIN-TELEPORTATION* function in [15] and then by the evolution in the generations, the configuration with the optimized cost is found. Figure 4 shows the flowchart of the whole process and the details of the proposed approach are given in the following sections.

4.2 The Outline of the Algorithm

In the following, the details of the proposed genetic algorithm are presented.

4.2.1 Encoding of Chromosomes and Initial Population

One of the most important parts in GA is the encoding mechanism for representing chromosomes. In this paper, the algorithm uses a chromosome structure with m_g genes which is basically a string of m_g bits where m_g is the number of global gates. The i^{th} gene corresponds to the i^{th} global gate, where the gates are labeled from left to right. If a global gate is considered to be executed in Partition A (B), the corresponding gene takes the value of 0 (1) respectively. For example, for Fig. 5, m_g is equal to 7. It is inferred that the first global gate is executed in Partition B, the second global gate is intended for execution in Partition A, and so on. The chromosome illustrates the global gate configuration in the distributed circuit. Each configuration can be a potential solution to the problem. The algorithm starts with a random initial population of chromosomes.

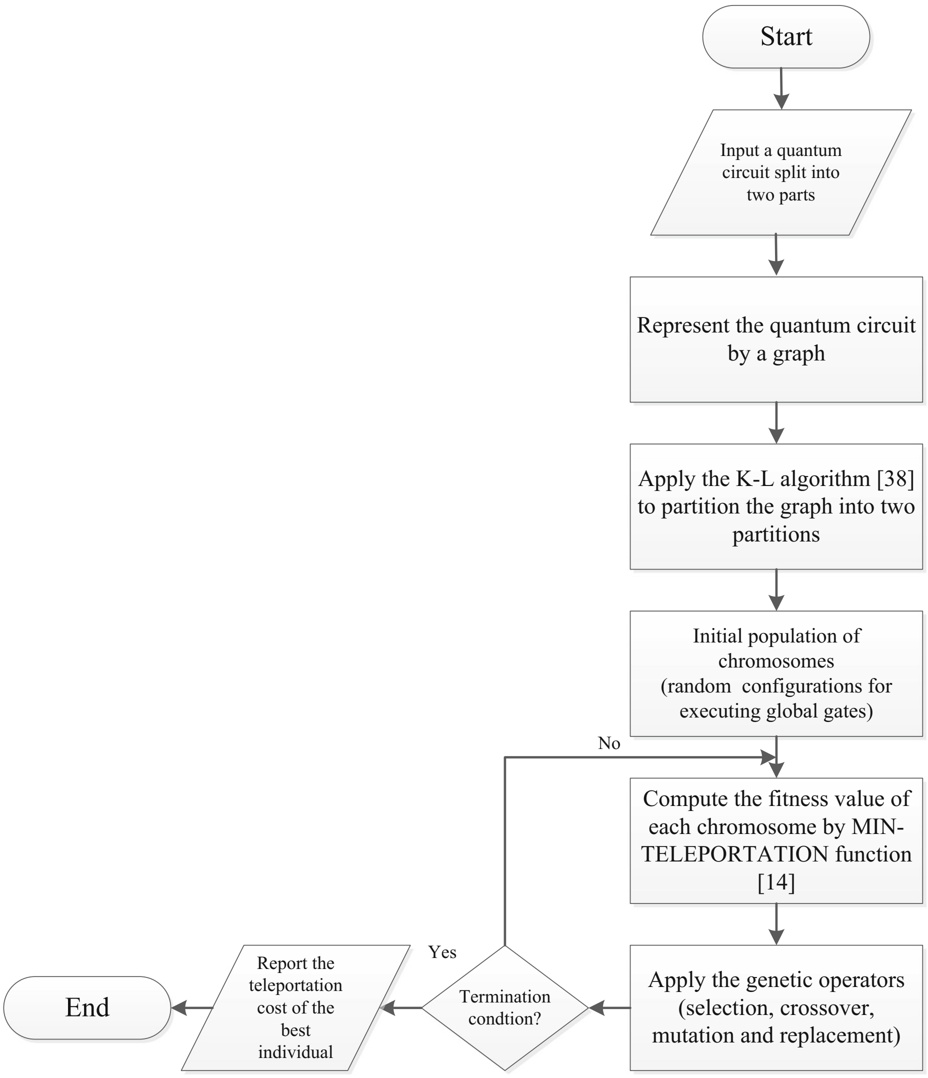
Start
Input a quantum circuit split into two parts
Represent the quantum circuit by a graph
Apply the K-L algorithm [38] to partition the graph into two partitions
Initial population of chromosomes (random configurations for executing global gates)
No
Compute the fitness value of each chromosome by MIN-TELEPORTATION function [14]
Apply the genetic operators (selection, crossover, mutation and replacement)
Termination condtion?
Yes
Report the teleportation cost of the best individual
End

Fig. 4 The flowchart of the proposed approach

4.2.2 Fitness Function and Selection Strategy

The fitness function for this genetic algorithm is computed as the minimal number of required teleportations for a given configuration of global gates, represented by chromosomes, of a distributed quantum system. This value is computed by the $MIN\text{-}TELEPORTATION$ function in [15].

The function $MIN\text{-}TELEPORTATION$ takes a DQC with the ordered list of gates, $\mathcal{G}$, and $Config\text{-}Arr$ as inputs and returns the minimum number of teleportations 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.

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.

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. The non-commutativity of two gates happens in three cases: [15]

  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.

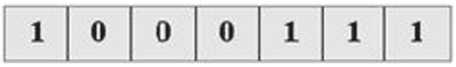

Fig. 5 The representation of chromosomes

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 (2) or (3).

$$U = \begin{bmatrix} u_0 & u_1 \\ u_1 & u_0 \end{bmatrix}, \tag{2}$$

$$U = \begin{bmatrix} u_0 & u_1 \\ -u_1 & -u_0 \end{bmatrix}. \tag{3}$$

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 is recursively 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 The function for finding the minimum number of teleportations for a given configuration.

```
1:  $MIN\text{-}TELEPORTATION(\mathcal{G}, Config\text{-}Arr, n_t)$
2:  sw = 0
3:  **if** $Empty(\mathcal{G})$ **then**
4:      **return** $n_t$
5:  **end if**
6:  **if** $local(\mathcal{G}[0])$ **then**
7:      **remove**($\mathcal{G}[0]$)
8:      $MIN\text{-}TELEPORTATION(\mathcal{G}, Config\text{-}Arr, n_t)$
9:  **end if**
10: $temp = \mathcal{G}[0]$
11: $n_t = n_t + 1$
12: **remove**($\mathcal{G}[0]$)
13: **remove**($\mathcal{G}_d[0]$)
14: **for** $i \leftarrow 0$ **to** $\mathcal{G}.length$ **do**
15:     **if** $NON\text{-}EXECUTE(temp, \mathcal{G}[i])$=**FALSE then**
16:         **for** $k \leftarrow i$ **to** 0 **do**
17:             **if** $NON\text{-}COMMUTE(\mathcal{G}[i], \mathcal{G}[k])$=**TRUE then**
18:                 sw = 1
19:                 **break**
20:             **end if**
21:         **end for**
22:         **if** sw = 0 **then**
23:             **remove**($\mathcal{G}[i]$)
24:             **if** $global(\mathcal{G}[i])$ **then**
25:                 **remove**($\mathcal{G}[i]$ **from** $\mathcal{G}_d$)
26:             **end if**
27:         **end if**
28:     **end if**
29: **end for**
30: $n_t = n_t + 1$
31: $MIN\text{-}TELEPORTATION(\mathcal{G}, Config\text{-}Arr, n_t)$
```

The roulette wheel strategy is used for the selection.

4.2.3 Crossover, Mutation, and Replacement

Crossover recombines two randomly selected ancestors into two fresh off-spring. We used two-point crossover where two crossover points on both parents' chromosomes are randomly selected. Then, the data between those points in either chromosome are swapped between the two parents' chromosomes. Mutation type is flip bit which is implemented by randomly inverting a randomly chosen gene of a selected individual. The random numbers generated for choosing ancestors, genes and changing the genes use a uniform distribution. Replacement forms the next generation of individuals by replacing some offspring using a specific replacement strategy. In this study, we replaced a number of the worst chromosomes with the best ones which are carried over to the next generation unaltered. This strategy is known as elitism which guarantees that the solution quality obtained by the GA will not decrease from one generation to the next. Mutation, crossover, and replacement take place with their corresponding probabilities, i.e., P_m, P_c, and P_r, respectively.

5 Results

We implemented our algorithm in MATLAB on a workstation with 4 GB RAM and 2.0 GHz CPU to find the configuration with an optimized number of teleportations. According to the complexity of the problem in this study, the wrong choice of GA parameters might greatly increase the convergence time of GA. It is explained in the following how the GA parameters are set.

The number of initial population is one of the important parameters in evolutionary algorithms like GA. There have been many studies [40–42] analyzing the impact of different population sizes on the performance of GA. The basic idea is that a larger population size may increase the population diversity and consequently help GA, however, it is shown [43] that there are conditions where larger populations might be harmful.

On the other hand, it is already known that in any search algorithm, including GA, we seek a proper balance between exploration and exploitation [44]. Exploration is the process of visiting entirely new zones of a search space, while exploitation is the process of visiting those zones of a search space within the neighborhood of previously visited points. In GA, the mutation operator is mostly used to provide exploration so as to increase the probability of finding the optimal solution while the crossover operator is widely used to lead population to converge to the optimal solution (exploitation). This balance is determined by the

Table 1 Parameters of GA

#of initial population	Selection strategy	Crossover type	Mutation type	Termination condition	P_m	P_c	P_r
$\lceil \frac{m_g}{2} \rceil$	Roulette wheel	Two-point	Flip bit	The max # of 1000 generations or less improvement in the fitness of the best individual in 10 generations than 0.001	0.1	0.9	0.4

m_g denotes the number of global gates in the given quantum circuit. P_m, P_c, and P_r denote probabilities of mutation, crossover and replacement, respectively

Table 2 Comparison of the proposed approach (P) with the random search (RS) and [15] in terms of the teleportation cost (TC) and the run time in seconds (Time (S))

Circuit	# of qubits	# of global gates	TC [15]	TC (RS)	TC (P)	Time (S) [15]	Time (S) (P)	TC. imp (RS) (%)	Speed-up [15]
Figure 4 [15]	4	5	4	4	4	0.65	3.15	–	–
4gt5-76	5	11	14	18	14	1194.92	275.88	22.22	4.17
4modulo7	5	11	10	16	10	1013.36	292.26	37.5	3.46
alu_primitive_opt	6	13	10	18	10	4874.24	1182.51	44.44	4.12
alu_primitive	6	18	18	26	20	154020.53	2183.31	23.06	70.54
sym9_147.real	12	54	N.A.	94	48	N.A.	10930.57	48.93	N.A.
parity_47.real	17	9	2	12	2	1028.63	212.78	83.33	4.83
4-qubit QFT	4	8	8	12	8	875.62	170.13	33.33	7.49
8-qubit QFT	8	32	N.A.	52	38	N.A.	8059.40	26.92	N.A.
16-qubit QFT	16	128	N.A.	216	133	N.A.	19788.34	38.42	N.A.
32-qubit QFT	32	512	N.A.	852	532	N.A.	78734.89	37.55	N.A.
64-qubit QFT	64	2048	N.A.	3752	2250	N.A.	361645.67	40.31	N.A.

The last two columns indicate the percentage of teleportation cost improvement (TC. imp) and the speed-up of the proposed approach as compared to RS and [15], respectively. N.A. stands for Not Applicable mutation and the crossover rate. Moreover, the replacement strategy as used in this study (elitism) can help to keep best chromosomes in all generations.

As the parameters of GA depend on the problem, we tuned these parameters by performing different experiments and have set parameters such that they produce a better solution within a certain amount of time. We stopped GA if *a*) the maximum number of 1000 generations is reached or *b*) the improvement in the fitness value of the best individual in the population over 10 generations is less than 0.001. The GA parameters are given in Table 1.

For comparing the performance of our algorithm with the exact solution of [15], we used some circuits from Revlib [45] library which is an online resource for benchmarks within the domain of reversible and quantum circuits, and the other set is QFT(n) where $n \in \{4, 8, 16, 32, 64\}$.

The benchmark circuits were first decomposed into the basic gate library by applying the synthesis approach in [37, 46] and then the rest of the proposed approach is run.

We compared the results of the proposed approach with [15]. Moreover, in order to show the effectiveness of GA, we performed a random search over different configurations and found the minimal number of required teleportations in all of the iterations. The number of iterations of the random search was equal to the size of initial population in GA multiplied by the number of generations of our GA when it was terminated.

Table 2 compares the results of our algorithm with [15] and the random search in terms of the execution time and the teleportation cost (TC), respectively, for 12 different benchmark circuits.

It should be noted that for the small circuits, e.g., Figure 4 of [15], where there are a few global gates, GA takes more time due to its overhead and there is no gain in applying GA. However, when the number of global gates increases, GA shows more advantage by remarkable speed-ups. For the cases in this table where the number of global gates is equal to or greater than 32, the approach of [15] cannot produce the answer even after ten days of continuous running, which are reported as N.A. The resultant TCs for the proposed approach and the approach of [15] for the benchmark circuits (excluding the N.A. rows) are the same except for alu-primitive (with 18 global gates), even though GA has a considerably lower execution time. The approach of [15] computes the teleportation cost of 18 for alu-primitive while the proposed reaches to the teleportation cost of 20 for this benchmark which indicate that the teleportation costs are only slightly different. A speed-up of 70.54 is obtained for this benchmark circuit. It is concluded that the genetic algorithm is capable of producing almost the same results with the optimal solution of [15] in much less time.

The proposed approach also improves the teleportation cost of the purely random search by on average 36.33% for the benchmark circuits which verifies that by taking inspirations from natural evolution concepts, GA can converge to near-optimal results. For the parity 47.real circuit which has a regular structure, the proposed approach can effectively produce a TC of 2, while the random search cannot find the optimal answer and there a is 83.33% improvement in the TC for this circuit.

6 Conclusion

This paper proposed an approach based on the genetic algorithm to obtain a configuration of global gates in a distributed quantum circuit which leads to minimal number of teleportation. Compared with the previous work in [15], with an exponential complexity, it was shown that the proposed approach yielded almost the same results in much less time. Moreover, the results demonstrate that GA decreases teleportation cost by on average 36.33% as compared with a random search over configurations and verified the effectiveness of GA as a guided random search. As future works, we are going to consider the case where the number of partitions is more than two. There are some challenges for having more than two partitions including the representation of global gates and finding the best configuration. Also, interwinding the initial partitioning scheme of qubits into subcircuits with the phase of obtaining an optimized configuration of global gates can be considered as a future work.

References

1. Shor, P.W.: Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. SIAM J. Comput. **26**, 1484–1509 (1997)
2. Lov, K.: Grover. A fast quantum mechanical algorithm for database search. In: Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing, pp. 212–219. ACM (1996)
3. Grover, L.K.: Quantum telecomputation. arXiv:quant-ph/9704012 (1997)
4. Stolze, J., Suter, D.: Quantum Computing: A Short Course from Theory to Experiment. Wiley (2008)
5. Van Meter, R., Ladd, T.D., Fowler, A.G., Yamamoto, Y.: Distributed quantum computation architecture using semiconductor nanophotonics. Int. J. Quantum Inform. **8**(01n02), 295–323 (2010)
6. Bennett, C.H.: Quantum crytography. In: Proc. IEEE Int. Conf. Computers, Systems, and Signal Processing, Bangalore, India, 1984, pp. 175–179 (1984)
7. Ekert, A.K.: Quantum cryptography based on bell's theorem. Phys. Rev. Lett. **67**(6), 661 (1991)
8. Furusawa, A., Sørensen, L.J., Braunstein, S.L., Fuchs, C.A., Kimble, H.J., Polzik, E.S.: Unconditional quantum teleportation. Science **282**(5389), 706–709 (1998)
9. Bouwmeester, D., Pan, J.-W., Mattle, K., Eibl, M., Weinfurter, H., Zeilinger, A.: Experimental quantum teleportation. Nature **390**(6660), 575–579 (1997)
10. Metcalf, B.J. et al.: Quantum teleportation on a photonic chip. Nat. Photon. **8**(10), 770–774 (2014)
11. Van Meter, R., Nemoto, K., Munro, W.J.: Communication links for distributed quantum computation. IEEE Trans. Comput. **56**(12), 1643–1653 (2007)
12. Cacciapuoti, A.S., Caleffi, M., Tafuri, F., Cataliotti, F.S., Gherardini, S., Bianchi, G.: Quantum internet: networking challenges in distributed quantum computing. IEEE Network (2019)
13. Bashar, M.A., Chowdhury, M.A., Islam, R., Rahman, M.S., Das, S.K.: A review and prospects of quantum teleportation. In: International Conference on Computer and Automation Engineering, 2009. ICCAE'09, pp. 213–217. IEEE (2009)
14. Troya, J., Rivera, J./E., Vallecillo, A.: On the specification of non-functional properties of systems by observation. In: International Conference on Model Driven Engineering Languages and Systems, pp. 296–309. Springer (2009)
15. Zomorodi-Moghadam, M., Houshmand, M., Houshmand, M.: Optimizing teleportation cost in distributed quantum circuits. Int. J. Theor. Phys. **57**(3), 848–861 (2018)
16. Mitchell, M.: An Introduction to Genetic Algorithms. MIT Press (1998)
17. Nielsen, M.A., Chuang, I.L.: Quantum Computation and Quantum Information. Cambridge University Press 10th anniversary edition (2010)
18. Spector, L., Barnum, H., Bernstein, H.J., Swamy, N.: Quantum computing applications of genetic programming. Adv. Gen. Program. **3**, 135–160 (1999)
19. Stepney, S., Clark, J.A.: Searching for quantum programs and quantum protocols: a review. J. Comput. Theor. Nanosci. **5**(5), 942–969 (2008)
20. Leier, A., Banzhaf, W.: Comparison of selection strategies for evolutionary quantum circuit design. In: Genetic and Evolutionary Computation Conference, pp. 557–568. Springer (2004)
21. Massey, P., Clark, J.A., Stepney, S.: Evolving quantum circuits and programs through genetic programming. In: Genetic and Evolutionary Computation Conference, pp. 569–580. Springer (2004)
22. Spector, L., Klein, J.: Machine invention of quantum computing circuits by means of genetic programming. AI EDAM **22**(3), 275–283 (2008)
23. Houshmand, M., Zamani, M.S., Sedighi, M., Houshmand, M.: Ga-based approach to find the stabilizers of a given sub-space. Genet. Program Evolvable Mach. **16**(1), 57–71 (2015)
24. Van Meter, R., Devitt, S.J.: The path to scalable distributed quantum computing. Computer **49**(9), 31–42 (2016)

25. Cleve, R., Buhrman, H.: Substituting quantum entanglement for communication. Phys. Rev. A **56**, 1201 (1997)
26. Cirac, J., Ekert, A., Huelga, S., Macchiavello, C.: Distributed quantum computation over noisy channels. Phys. Rev. A **59**, 4249 (1999)
27. Beals, R., Brierley, S., Gray, O., Harrow, A.W., Kutin, S., Linden, N., Shepherd, D., Stather, M.: Efficient distributed quantum computing. In: Proc. R. Soc. A, vol. 469, p. 20120686. The Royal Society (2013)
28. Yepez, J.: Type-II quantum computers. Int. J. Modern Phys. C **12**(09), 1273–1284 (2001)
29. Streltsov, H., Kampermann, A., Brub, D.: Quantum cost for sending entanglement. Phys. Rev. Lett. **108**, 250501 (2012)
30. Lo, H.K.: Classical-communication cost in distributed quantum-information processing: A generalization of quantum-communication complexity. Phys. Rev. A **62**(1), 012313 (2000)
31. Ying, M., Feng, Y.: An algebraic language for distributed quantum computing. IEEE Trans. Comput. **58**, 728–743 (2009)
32. Van Meter, R., Munro, W., Nemoto, K., Itoh, K.M.: Arithmetic on a distributed-memory quantum multicomputer. ACM J. Emerg. Technol. Comput. Syst. (JETC) **3**, 2 (2008)
33. Yimsiriwattana, A., Lomonaco, S.J. Jr..: Distributed quantum computing: A distributed shor algorithm. arXivquant-ph/0403146 (2004)
34. Andrés-Martínez, P., Heunen, C.: Automated distribution of quantum circuits via hypergraph partitioning. Phys. Rev. A, **100**(3) (2019)
35. van Meter, R., Oskin, M.: Architectural implications of quantum computing technologies. ACM J. Emerg. Technol. Comput. Syst. (JETC) **2**(1), 31–63 (2006)
36. Houshmand, M., Saheb Zamani, M., Sedighi, M., Arabadeh, M.: Decomposition of diagonal hermitian quantum gates using multiple-controlled pauli Z gates. ACM J. Emerg. Technol. Comput. Syst., **11**(3) (2014)
37. Shende, V.V., Bullock, S.S., Markov, I.L.: Synthesis of quantum-logic circuits. IEEE Trans. CAD **25**(6), 1000–1010 (2006)
38. Houshmand, M., Sedighi, M., Zamani, M.S., Marjoei, K.: Quantum circuit synthesis targeting to improve one-way quantum computation pattern cost metrics. ACM J. Emerg. Technol. Comput. Syst. (JETC) **13**(4), 55 (2017)
39. Kernighan, B.W., Lin, S.: An efficient heuristic procedure for partitioning graphs. Bell Syst. Techn. J. **49**(2), 291–307 (1970)
40. Rylander, S.G.B. et al.: Optimal population size and the genetic algorithm. Population **100**(400), 900 (2002)
41. He, J., Yao, X.: From an individual to a population: An analysis of the first hitting time of population-based evolutionary algorithms. IEEE Trans. Evol. Comput. **6**(5), 495–511 (2002)
42. Oliveto, P.S., He, J., Yao, X.: Analysis of the $(1+1)$-ea for finding approximate solutions to vertex cover problems. IEEE Trans. Evol. Comput. **13**(5), 1006–1029 (2009)
43. Chen, T., Ke, T., Chen, G., Yao, X.: A large population size can be unhelpful in evolutionary algorithms. Theor. Comput. Sci. **436**, 54–70 (2012)
44. Črepinšek, M., Liu, S.-H., Mernik, M.: Exploration and exploitation in evolutionary algorithms: A survey. ACM Comput. Surv. (CSUR) **45**(3), 35 (2013)
45. Wille, R., Große, D., Teuber, L., Dueck, G.W., Drechsler, R.: Revlib: An online resource for reversible functions and reversible circuits. In: 38th International symposium on multiple valued logic, 2008. ISMVL 2008, pp. 220–225. IEEE (2008)
46. Barenco, A. et al.: Elementary gates for quantum computation. Phys. Rev. A **52**, 3457–3467 (1995)