Quantum Information Processing (2020) 19:141
https://doi.org/10.1007/s11128-020-02630-8

Mathematical formulation of quantum circuit design problems in networks of quantum computers

R. van Houte[1,2] · J. Mulderij[1,2] · T. Attema[1,3,4] · I. Chiscop[1] · F. Phillipson[1]

Received: 3 October 2019 / Accepted: 3 March 2020 / Published online: 17 March 2020

Abstract
In quantum circuit design, the question arises how to distribute qubits, used in algorithms, over the various quantum computers, and how to order them within a quantum computer. In order to evaluate these problems, we define the global and local reordering problems for distributed quantum computing. We formalise the mathematical problems and model them as integer linear programming problems, to minimise the number of SWAP gates or the number of interactions between different quantum computers. For global reordering, we analyse the problem for various geometries of networks: completely connected networks, general networks, linear arrays and grid-structured networks. For local reordering, in networks of quantum computers, we also define the mathematical optimisation problem.

Keywords Nearest neighbour compliant · Quantum computation architectures and implementations · Distributed quantum computing

1 Introduction

The early quantum computers have a (very) limited number of qubits [32]. This is the result of the conditions that are required to store quantum information, and means required to manipulate the information. It is possible to connect multiple quantum computers to form a network and do computations together. This is, analogously to current methods in ICT, called *distributed quantum computing* [4,7]. In such a system, we require the network to be able to share both classical and quantum information. If

✉ F. Phillipson
frank.phillipson@tno.nl

1 TNO, The Hague, South Holland, Netherlands

2 Delft University of Technology, Delft, South Holland, Netherlands

3 CWI, Amsterdam, North Holland, Netherlands

4 Leiden University, Leiden, South Holland, The Netherlands

the network is set up correctly, the collection of quantum computers will behave as one big computer [37], and thus greatly increase the possibilities and practical instances that it can be used for.

To act as one big quantum computer, two quantum computers are connected by an entangled pair of qubits. Depending on the topology of the network, we may have a situation where two computers are not connected directly, but indirectly, via other computers in the network. We can apply a method called *entanglement swapping* [14] to create an entangled pair of qubits between these computers. This procedure requires all consecutive computers along the path to have the shared entangled state $\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$. Consider the computers that are not at the endpoints of the path. Were those computers to measure in the Bell basis and then communicate their outcome (this requires two bits of information) to their neighbours along the path, they can perform Pauli gates on their qubits to create a shared entangled pair. This pair would again be in the state $\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$. By repeating this process for all computers along the path, we end up with a shared entangled state between two computers at the endpoints. This procedure indicates that it is preferable to use the shortest path in a network. Computers that perform entanglement swapping will need two extra qubits to store and measure information.

Next, if we want to perform a calculation on a distributed quantum computer, we have to partition the calculations into parts and assign those parts to the individual quantum computers, in such a way that communication between parts of the calculation is possible and can be done efficiently. This means that we have to design a quantum circuit for the total network of quantum computers. Already on a single quantum computer quantum circuit design is not trivial. There are a couple of considerations on how to compile a circuit. The nearest neighbour constraint is one of them. This constraint imposes a restriction on quantum gates, such that gates can only act on two adjacent qubits. Given the locations on which the qubits are present, one might need to change the locations of the qubits before a gate can be applied. Changes to the locations of qubits can be made using so-called SWAP gates [24]. SWAP gates interchange the position of two qubits, but since they are also quantum gates, they can only act on two adjacent qubits. The SWAP gates are considered overhead because they do not directly contribute to the calculation that is being performed. SWAP gates do not only require resources, but also increase the running time significantly. Since coherence times are currently very low, information on qubits can only be held stable for a short amount of time, after which the information is lost due to interaction with the environment [9]. It is therefore important to minimise the running time of the circuit and hence the size of the overhead. In quantum algorithm design, minimising the number of required SWAP gates in order for a circuit to comply with the nearest neighbour constraints has become a research topic of its own. So far though, the focus has been on architectures that involve only a single quantum computer.

There are two main strategies of coping with the minimisation of the number of SWAP gates: global reordering and local reordering [36]. In global reordering, one is only concerned with finding an optimal initial qubit placement without focusing on the micromanagement of swapping the qubits into the right positions after every gate, which is what local reordering entails. Both strategies can be done on a single

Table 1 Four areas of research minimising calculation overhead

	Global	Local
Single	I	II
Distributed	III	IV

quantum computer or on a network of quantum computers, leading to four areas of research and applications as indicated in Table 1.
For the single quantum computer (Areas I and II), a variety of research is available. Area I was studied, mostly because of its relative simplicity, in [16,17,28,29,36]. All kinds of qubit architectures have been considered in the more popular Area II: Qubits are placed on a linear array in [3,5,13,15,18,20,25,27,31,35,36], on a 2D grid in [1,2,6,8,12,19,26,30], on a 3D grid in [11], or, more recently, on the IBM QX architectures in [10,33,38,39].

Areas III and IV have (as far as the authors are aware of) not been studied before. The contribution of this paper lies in the definition of this research area and the first mathematical formulation of the problems of minimising the number of SWAP gates in the distributed computing areas III and IV.

Area III can be viewed in two ways. If we are interested in the order of all qubits on all quantum computers, we have *Complete distributed global reordering*. This can be seen as global single reordering with two different cost values for the SWAP gates between qubits on different computers and SWAP gates between qubits on the same computer. Next we have, as we will call it, *Celestial Reordering*. Here, one allocates qubits to quantum computers while trying to minimise the number of interactions between computers. This addressed because of the high costs that come with setting up the required entanglement between the computers. The order of the qubits within the computers is not considered. The problem is related to the well known graph partitioning problem as will be shown in Sect. 2.

Our contribution comprises of integer linear programming (ILP) models for the proposed problems. The size of the models is reflected by the number of variables and the number of constraints that they contain. In Table 2, an overview is provided. All the models provide optimal solutions given that the circuit and the gate decomposition are both optimal.

In this paper, we define the 'Celestial Reordering' problem (from research area III) and present the mathematical problem formulation for minimising the number of SWAP gates in specific topologies of quantum computer networks. We include ILP models that are suited for exact solution methods. After that, in Sect. 3, the problem of local reordering in the context of distributed quantum computing (Area IV) is formulated and explored. Here, we minimise, using a weighted objective function, the number of required SWAP gates within a computer and the number of required SWAP gates between computers. An integer linear programming model is also provided, such that the problem can be solved with exact methods. We end in Sect. 4 with concluding remarks and suggestions for future research.

Table 2 The order $\mathcal{O}(\cdot)$ of variables and constraints of each model is shown

ILP model sizes for different problems			
Research area	Network/qubit architecture	Variables	Constraints
Area I	Linear array [29]	n^2	n^2
Area II	Linear array [22]	n^2m	n^2m
	2D grid [21]	n^4m	n^4m
	3D grid [21]	n^4m	n^4m
Area III	Complete	$nM + n^2$	Mn^2
	General	n^2M^2	n^2M^2
	Linear array	n^2	$n^2 + M$
	2D grid	$nM + n^2(m_1 + m_2)$	$n^2(m_1 + m_2)M$
	General grid	$nM + n^2pM^{(p-1)/p}$	$M + n^2pM^{(p-1)/p}$
Area IV	Linear array	$n^2m + nMm$	$n^2m + nMm$

Here, n resembles the number of qubits, m is the number of quantum gates, and M is the number of quantum computers. The grid dimensions, where applicable, are indicated by m_1 and m_2. The dimension of the grid is denoted by p

2 Celestial reordering of qubits in a distributed quantum circuit

In this section, we will introduce the problem of Celestial reordering. In celestial reordering, given a quantum circuit consisting of qubits, quantum gates acting on the qubits and a number of quantum computers with given capacities, the task is to assign the qubits to the computers in such a way that the number of gate operations on pairs of qubits on different computers is minimised. We assume that the cost of setting up entanglement between two computers is significantly more costly than applying gates within a computer. Therefore, we neglect costs related to gates that are applied on qubits that are located on the same computer.

It is of great importance how the quantum computers are connected in a network. In this section, we consider the most straightforward geometries: the completely connected network, the general network, the linear array, the two-dimensional grid and the general grid. For each of the networks, we formalise and visualise the problem, and model it as an integer linear program (also ILP).

First, we introduce some notation that we will use throughout the paper.

i. n denotes the total number of qubits in the quantum algorithm. In diagrams, vertices that represent qubits are denoted by circles.
ii. M denotes the number of quantum computers. In diagrams, quantum computers are represented by rounded squares.
iii. K is used to denote the effective capacity of each quantum computer. That is, the maximum number of qubits that an individual quantum computer can use and store in working memory. This does not include the qubits that are necessary for communication or entanglement swapping. This adds an extra number of qubits per computer, depending on the network architecture.

Suppose we have a quantum algorithm acting on n qubits that is represented by a series of unitary gates. We allow for unitary operations on single qubits or controlled gates on two qubits. The unitary operations on single qubits will be ignored. All other operations are assumed to be decomposed into this set of gates [24]. One way to ensure sufficient capacity is to take $K \geq \lceil n/M \rceil$ for every computer. If necessary, this quantity may vary per computer as long as the total capacity exceeds n.

2.1 Completely connected network

We start out in the setting where we have all-to-all coupling between the different quantum computers.
Consider the complete graph K_n, where the vertices are labelled $[n] := \{1, \ldots, n\}$ and each vertex corresponds to a qubit in the algorithms. We can count the number of controlled gates that are applied to each pair of qubits. Similar to the model of single quantum computer global reordering [17], we create a cost function $c : E(K_n) \to \mathbb{Z}_{\geq 0}$ by letting $c_{ij} = c(\{i, j\})$ be the number of controlled gates between qubits i and j. This graph is called the *interaction graph*.
Our goal now is to find an assignment of qubits to computers $f : \{1, \ldots, n\} \to \{1, \ldots, M\}$ such that the total number of controlled gates between all different pairs of computers is minimal.
For a qubit $i \in [n]$ and computer $k \in [M]$ let

$$x_{ik} = \begin{cases} 1 & \text{if qubit } i \text{ is assigned to computer } k \\ 0 & \text{otherwise.} \end{cases} \tag{2.1}$$

For each computer, we thus want to limit the total number of assigned qubits by the computer's total capacity K, so

$$\sum_{i=1}^{n} x_{ik} \leq K, \quad \forall k \in [M]. \tag{2.2}$$

Furthermore, every qubit can be assigned to only one computer, thus

$$\sum_{k=1}^{M} x_{ik} = 1, \quad \forall i \in [n]. \tag{2.3}$$

The objective is

$$\min \sum_{\substack{i,j\in[n] \\ i<j}} c_{ij} \sum_{k\in[M]} \frac{|x_{ik} - x_{jk}|}{2}, \tag{2.4}$$

since for a given $i, j \in [n], i \neq j$. The second summation in the objective is given by

$$\sum_{k\in[M]} \frac{|x_{ik} - x_{jk}|}{2} = \begin{cases} 1 & \text{if qubit } i \text{ and } j \text{ are assigned to different computers} \\ 0 & \text{otherwise.} \end{cases} \tag{2.5}$$

The 2 in the denominator is to compensate for counting twice that a qubit is on a computer where the other qubit is not. This constant can be taken out of the sums.
We can remove the absolute value in the objective by introducing the variable L_{ijk} and add an extra pair of constraints $-L_{ijk} \leq x_{ik} - x_{jk} \leq L_{ijk}$ for every $i, j \in [n], i < j$ and $k \in [M]$. Since any optimal solution will have integer values for L_{ijk}, this variable does not necessarily have to be formulated as integer. This gives us an MILP (mixed integer linear program) of the form

$$\begin{aligned} \min\ & \frac{1}{2} \sum_{\substack{i,j\in[n]\\ i<j}} c_{ij} \sum_{k\in[M]} L_{ijk} \\ \text{s.t. } & \sum_{i=1}^{n} x_{ik} \leq K, \quad \forall k \in [M] \\ & \sum_{k=1}^{M} x_{ik} = 1, \quad \forall i \in [n] \\ & \left.\begin{array}{l} x_{ik} - x_{jk} \leq L_{ijk} \\ x_{ik} - x_{jk} \geq -L_{ijk} \end{array}\right\}, \quad \forall i, j \in [n], i < j, \forall k \in [M] \\ & x_{ik} \in \{0, 1\}, \quad \forall i \in [n], k \in [M] \\ & L_{ijk} \in \mathbb{R}, \quad \forall i, j \in [n], i < j, \forall k \in [M]. \end{aligned} \tag{2.6}$$

The total number of integer variables is nM and the total number of continuous variables is $\binom{n}{2}m = n(n-1)m/2$. There are $M + n + n(n-1)M = \mathcal{O}(Mn^2)$ constraints in this problem.
It is possible to extend the celestial reordering model by allowing different capacities of computers. This can easily be done by replacing the capacity constraints by

$$\sum_{i=1}^{n} x_{ik} \leq K_k \quad \forall k \in [M], \tag{2.7}$$

where the capacity K_k is now computer specific.

2.2 General networks of quantum computers

Suppose the network of quantum computers is represented by a connected graph $G = (V, E)$, where quantum computers are represented by nodes. A pair of quantum computers can communicate directly if and only if their corresponding nodes are connected by an edge in the graph. If two quantum computers are not connected directly,

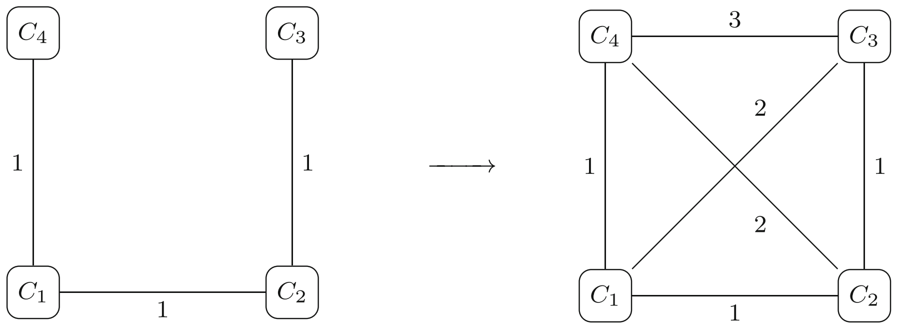

Fig. 1 The conversion of a general graph to a complete graph with edge weights corresponding to the distance of the shortest path between each pair of nodes. In this example, the shortest path between C_3 and C_4 in the left graph is 3; therefore, the weight on the edge $\{C_3, C_4\}$ in the right graph is equal to 3

we can indirectly connect them via intermediate connections with other computers. We can do this by applying *entanglement swapping*. In this case, we search for the shortest path between the pair of computers.
We let the vertex set $V = [M]$ be labelled by the computers and define $w_{k\ell}$ as the length (i.e. the number of edges) of the shortest path between vertices k and ℓ in G. We can therefore consider the problem on the complete graph K_M with edge weights $w_{k\ell}$ for all $k, \ell \in [M], k \neq \ell$. In Fig. 1, an example is given for clarification.
Here, we see that for a network of four computers, we can construct a complete graph where every computer is connected to every other computer. The weights on the edges now indicate the length of the path from one computer to another. Pairs of qubits which are placed on different computers contribute to the costs if they interact with each other. The cost per interaction is equal to the distance between the computers on which the interacting qubits are located, since that counts the number of times an entangled pair of ancillary qubits is required (Fig. 2).

We consider the same decision variables $x_{ik}, i \in [n], k \in [M]$ as in Sect. 2.1. Our objective will be

$$\min \sum_{\substack{i,j\in[n]\\ i<j}} c_{ij} \sum_{\substack{k,\ell\in[M]\\ k\neq\ell}} w_{k\ell} x_{ik} \cdot x_{j\ell}, \tag{2.8}$$

and since x_{ik} and x_{jl} are both binary, their product is

$$x_{ik} \cdot x_{j\ell} = \begin{cases} 1 & \text{if qubit } i \text{ is on computer } k \text{ and qubit } j \text{ is on computer } \ell \\ 0 & \text{otherwise.} \end{cases} \tag{2.9}$$

The contribution of an assigned pair of qubits depends on two factors: the path length between the computers in the network and the number of interactions in-between the qubits in the algorithms. The product of these quantities is the number of EPR pairs that is required for this pair of computers.

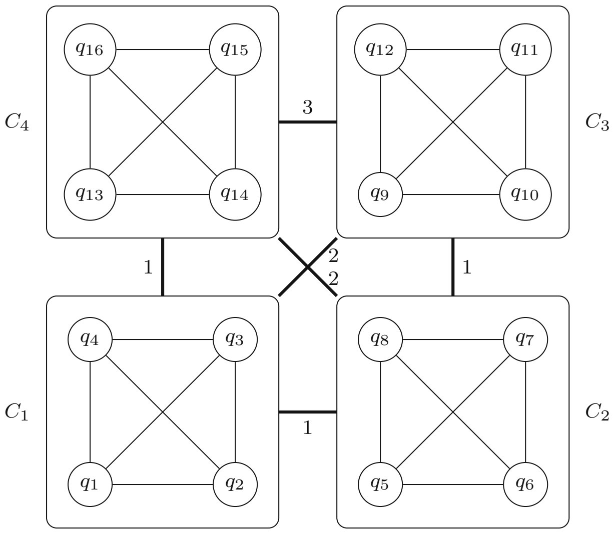

Fig. 2 A complete graph on a network of four quantum computers, indicated by the C's. The computers each have a capacity of four qubits

The constraints are the same as in the original Program 2.6. We thus obtain a quadratic binary optimisation program. This quadratic problem has nM variables and $n + M$ constraints.

To transform the quadratic program into an ILP, we introduce a variable $z_{ijk\ell} \in \{0, 1\}$ for $i, j \in [n], i < j$ and $k, \ell \in [M], k \neq \ell$, that satisfies the inequality

$$z_{ijk\ell} \geq x_{ik} + x_{j\ell} - 1, \quad \forall i, j \in [n], i < j \text{ and } k, \ell \in [M], k \neq \ell. \tag{2.10}$$

If $x_{ik}, x_{j\ell}$ or both are equal to 0, then $z_{ijk\ell} \geq 0$ and since we are minimising over an increasing function this yields $z_{ijk\ell} = 0$. Only if $x_{ik} = x_{j\ell} = 1$, then $z_{ijk\ell} = 1$ is required. We are left with the equivalent program

$$\begin{aligned}
\min \sum_{\substack{i,j\in[n]\\ i<j}} c_{ij} \sum_{\substack{k,\ell\in[M]\\ k\neq\ell}} w_{k\ell} z_{ijk\ell} & \\
\text{s.t. } z_{ijk\ell} \geq x_{ik} + x_{j\ell} - 1, \quad & \forall i, j \in [n], i < j \text{ and } k, \ell \in [M], k \neq \ell \\
\sum_{k\in[M]} x_{ik} = 1, \quad & \forall i \in [n] \\
\sum_{i\in[n]} x_{ik} \leq K, \quad & \forall k \in [M] \\
x_{ik} \in \{0, 1\}, \quad & \forall i \in [n], k \in [M] \\
z_{ijk\ell} \in \{0, 1\}, \quad & \forall i, j \in [n], i < j \text{ and } k, \ell \in [M], k \neq \ell.
\end{aligned} \tag{2.11}$$

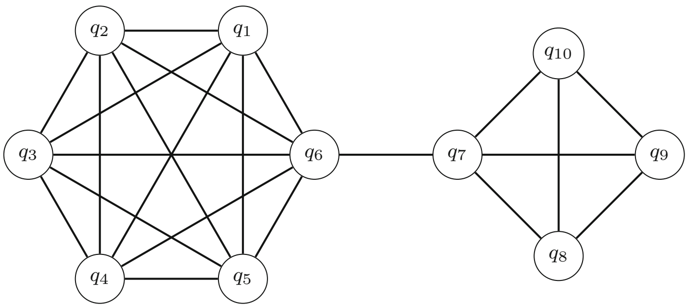

Fig. 3 Here, we see an interaction graph that consists of two complete graphs that are connected to each other by one edge. The graph contains ten qubits. Each edge represents a single interaction between a pair of qubits for some quantum algorithm on ten qubits

This is an ILP with $nM + \binom{n}{2}M(M-1) = \mathcal{O}(n^2M^2)$ variables and $\mathcal{O}(n^2M^2)$ constraints. Notice that the number of variables and constraints has increased by turning the quadratic program into an ILP.

An interesting question is how much the objective can vary as the capacity K of each computer changes. We can illustrate this with the example of the graphs K_6 and K_4 that are connected by one edge, see Fig. 3.
If we have $M = 2$ computers, both with capacity $K = 5$, we are required to make a cut of at least 5 edges. We partition the interaction graph into $\{1, \ldots, 5\}$ and $\{6, \ldots, 10\}$ forming two computers.
However, if we were somehow able to increase the capacity of both computers to $K = 6$ qubits, we can partition the graph into $\{1, \ldots, 5\}$ and $\{6, \ldots, 10\}$. This requires a cut of only one edge. This example and generalisations to more qubits show that the capacity of the computers by a small amount can yield a big difference in the number of EPR pairs required.

2.3 Example of a quantum network and distributed algorithm

We consider an example of a quantum network of four computers between four cities in The Netherlands: Amsterdam (A), Delft (D), Leiden (L) and The Hague (G), as is illustrated in Fig. 4.
The cities of Leiden, Delft and The Hague are all mutually connected while Amsterdam is only connected to Leiden. The shortest distance between every pair of cities is represented in Table 3.

This circuit consists of fifteen CNOT gates and ten Toffoli gates. The Toffoli gate acts on three qubits and can be decomposed in five controlled gates as shown in Fig. 6. The V-gate is the square root of the Pauli X-gate:

$$V = \sqrt{X} = \frac{1}{2}\begin{pmatrix} 1+i & 1-i \\ 1-i & 1+i \end{pmatrix}. \tag{2.12}$$

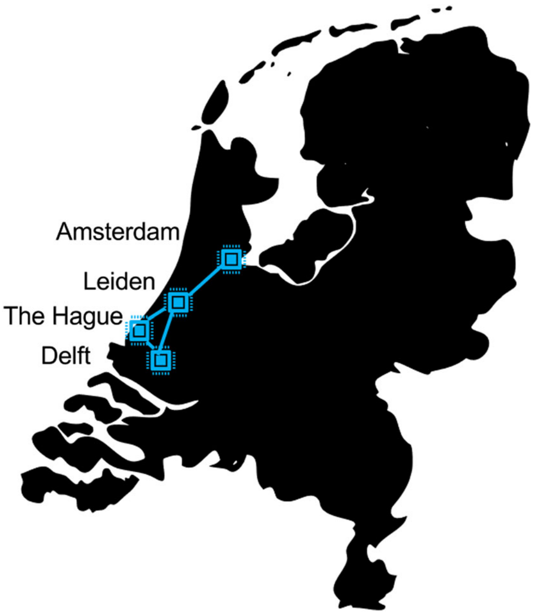

Fig. 4 A map of a potential quantum computer network in the Netherlands

Table 3 The shortest distances between the cities of Amsterdam (A), Delft (D), Leiden (L) and The Hague (G)

w	A	D	L	G
A	–	2	1	2
D	2	–	1	1
L	1	1	–	1
G	2	1	1	–

From the circuit, the cost c_{ij} is obtained for every pair of qubits, by counting the number of gates act on the qubit pair i, j (Fig. 7).
Each computer has an effective capacity of four qubits to execute the circuit. Each computer also has one extra qubit that is used for communication. This qubit is not assigned to qubits in the circuit and is not taken into account in the optimal assignment. On this network of quantum computers, we want to execute a quantum circuit on fifteen qubits that counts the number of qubits in state $|1\rangle$. The circuit is shown in Fig. 5. It is called the "rd84_143" circuit and was obtained from the reversible circuit library *RevLib* [34].

The ILP was constructed and solved to optimality using the Python API of CPLEX. The solver was run on a computer with 2 GB of RAM, and completed its Branch & Bound search in 0.39 s. The optimal qubit assignment is shown in Fig. 8.

The costs of communication between every two computers are shown on the edges in Fig. 8. The sum of these costs, which is the objective function of the optimisation program, is 48. There was no communication between computers with a distance of two between them.

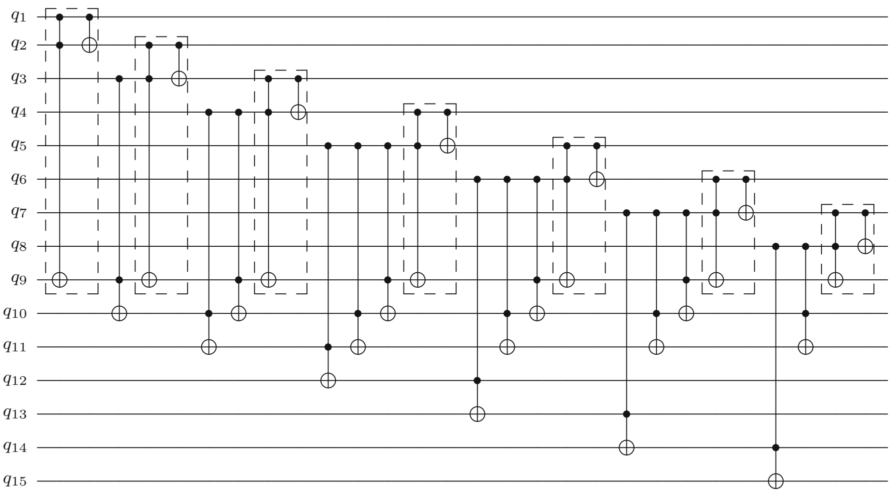

Fig. 5 The "rd84_143" circuit. The circuit consists of fifteen qubits. After the Toffoli and Peres gates are decomposed, 98 two-qubit gates remain

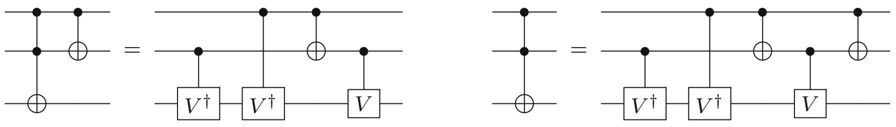

Fig. 6 On the left, the decomposition of the Peres gate. On the right, the Sleator–Weinfurter decomposition of the Toffoli gate [23]

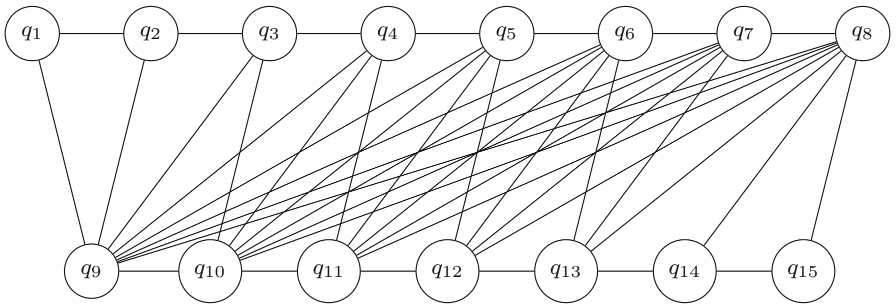

Fig. 7 The interaction graph of counting circuit of Fig. 5 on fifteen qubits. An edge represents one or more controlled gates between each pair of qubits. An optimal assignment of qubits to computers in not immediately clear

2.4 Linear array

In this section, we consider a different network of quantum computers. In this network, all computers are arranged on a line, and each one of them is connected to its one or two neighbouring computers. This network is a special case of the general network

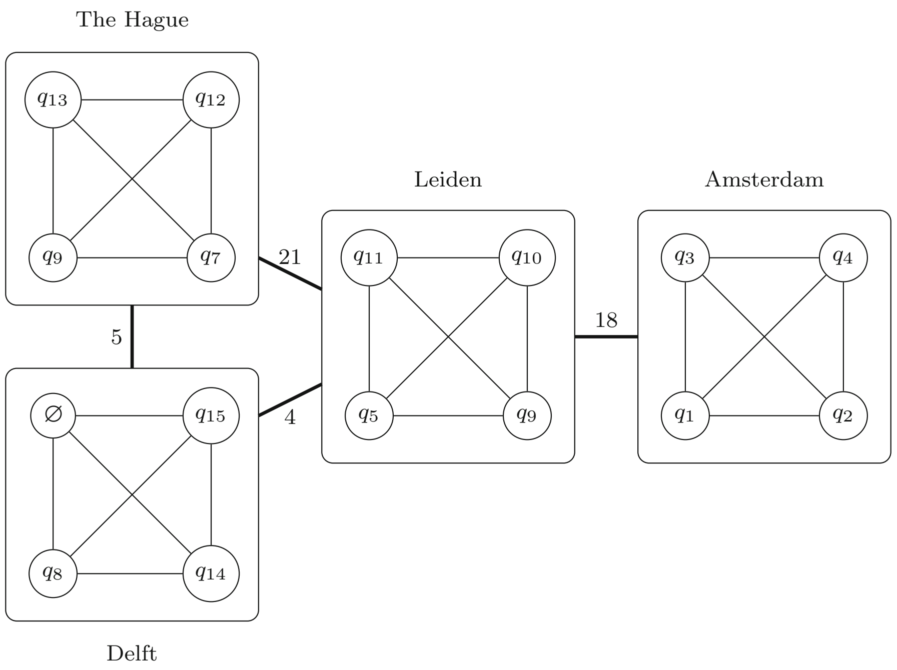

Fig. 8 The graph shows the optimal qubit configuration, where each qubit is assigned to a computer. The capacity of the computers is not exceeded. The number of gates between every pair of computers is shown on the edges between computers

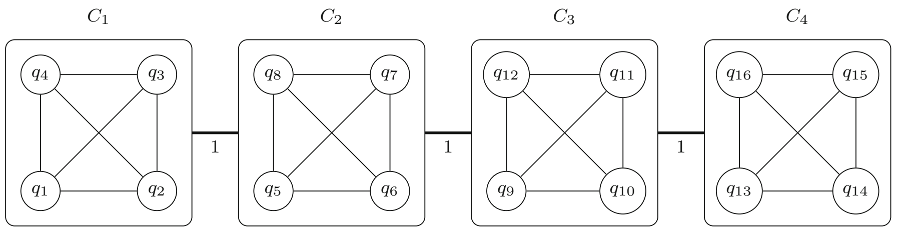

Fig. 9 A line graph on a network of four quantum computers, indicated by the C's. The computers each have a capacity of four qubits, corresponding to the circular nodes

and leads to a reduction in the number of variables and constraints in the resulting model because of the structure in the network.
If we associate the computers with the numbers $\{1, \dots, M\}$, then quantum computer k can only communicate with computers $k-1$ and $k+1$, except at the boundaries. An example of such a network is given in Fig. 9.

This means that if two qubits are located on computer k and ℓ, then applying a two qubit gate requires us to make $|k-\ell|$ non-local interactions by using the computers in-between. This changes the original objective in Eq. 2.4 to an objective that takes the distance between computers into account. For a given pair of qubits (i, j) this is given by equation

$$\left|\sum_{k\in[M]} kx_{ik} - \sum_{k\in[M]} kx_{jk}\right| = \left|\sum_{k\in[M]} k(x_{ik} - x_{jk})\right|. \tag{2.13}$$

Again, we introduce a new variable to encode the absolute value as a linear constraint, analogous to the completely connected network of Sect. 2.1. The full mixed integer linear program now reads

$$\begin{aligned}
\min \; & \sum_{\substack{i,j\in[n]\\ i<j}} c_{ij} L_{ij} \\
\text{s.t.} \; & \sum_{i=1}^{n} x_{ik} \le K, \quad \forall k \in [M] \\
& \sum_{k=1}^{M} x_{ik} = 1, \quad \forall i \in [n] \\
& \left.\begin{array}{l} \sum_{k\in[M]} k(x_{ik} - x_{jk}) \le L_{ij} \\ \sum_{k\in[M]} k(x_{ik} - x_{jk}) \ge -L_{ij} \end{array}\right\}, \quad \forall i, j \in [n], i < j \\
& x_{ik} \in \{0, 1\} \quad \forall i \in [n], k \in [M] \\
& L_{ij} \in \mathbb{R} \quad \forall i, j \in [n], i < j.
\end{aligned} \tag{2.14}$$

This MILP consists of nM integer variables and $\binom{n}{2} = \mathcal{O}(n^2)$ continuous variables and has $n + M + 2\binom{n}{2} = \mathcal{O}(n^2 + M)$ constraints.

2.5 Two-dimensional grid

In this section, we consider a two-dimensional grid as network topology. Such a network allows for more connections between computers and directly extends the linear network of Sect. 2.4. Nevertheless, this network also leads to a reduction in the complexity in the assignment of qubits to computers.
We first have to introduce some tools to describe this network. Consider the metric based on the 1-norm[1] defined by

$$d(x, y) = \|x - y\|_1 = \sum_{i=1}^{p} |x_i - y_i|, \quad x, y \in \mathbb{Z}^p. \tag{2.15}$$

We first consider a (square) grid with side length m defined by $G_2 := [m_1] \times [m_2] \subseteq \mathbb{Z}^2$. Thus, the number of quantum computers equals $M = m_1 m_2$. We say that two quantum computers are connected if and only if their distance in the graph is 1. A distance of 2 means that communication has to go via one other quantum computer. A small example of such a network is shown in Fig. 10.

[1] This metric is also called the *taxicab distance* or *Manhattan distance* for its similarity to travelling along the shortest route between two points in the streets of Manhattan, New York.

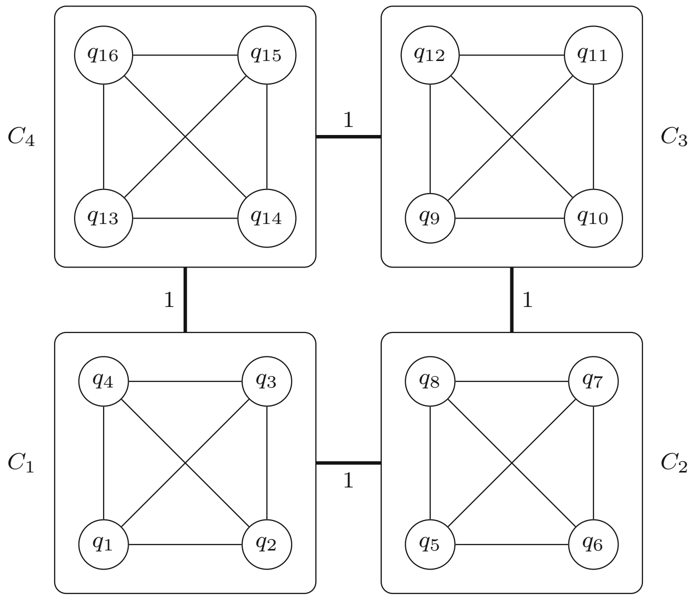

Fig. 10 The graph of a two-dimensional grid on a network of four quantum computers, indicated by the C's. The computers each have a capacity of four qubits

For qubit $i \in [n]$ and computer $(u, v) \in G_2$, we let

$$x_{i,uv} = \begin{cases} 1 & \text{if qubit } i \text{ is assigned to position } (u, v) \\ 0 & \text{otherwise.} \end{cases} \tag{2.16}$$

Then, similar to the constraints in Eqs. (2.2) and (2.3), we have the following constraints:

$$\sum_{i=1}^{n} x_{i,uv} \leq K, \quad \forall (u, v) \in G_2, \tag{2.17}$$

and

$$\sum_{(u,v)\in G_2} x_{i,uv} = 1, \quad \forall i \in [n]. \tag{2.18}$$

Furthermore, the objective is now a weighted sum. The weights are determined by the number of interactions between a pair of qubits. The weighted sum consists of terms given by the distance between computers in the network to which the qubits are assigned. The objective is

$$\sum_{\substack{i,j\in[n]\\ i<j}} c_{ij}\|f(i)-f(j)\|_1 = \sum_{\substack{i,j\in[n]\\ i<j}} c_{ij}\left(\sum_{u\in[m_1]}\left|\sum_{v\in[m_2]} vx_{i,uv} - \sum_{v\in[m_2]} vx_{j,uv}\right| + \sum_{v\in[m_2]}\left|\sum_{u\in[m_1]} ux_{i,uv} - \sum_{u\in[m_1]} ux_{j,uv}\right|\right). \tag{2.19}$$

Again, we introduce new variables to encode the absolute values, this is done by two families $L^{(1)}_{ij,u}$ and $L^{(2)}_{ij,v}$. Analogous to the complete linear network described in Sect. 2.4, these variables can be relaxed to real numbers. The mixed integer linear program then reads

$$\begin{aligned}
\min \ & \sum_{\substack{i,j\in[n]\\ i<j}} c_{ij}\left(\sum_{u\in[m_1]} L^{(1)}_{ij,u} + \sum_{v\in[m_2]} L^{(2)}_{ij,v}\right)\\
\text{s.t.} \ & \sum_{i=1}^{n} x_{i,uv} \le K, \quad \forall u\in[m_1], v\in[m_2]\\
& \sum_{u\in[m_1]}\sum_{v\in[m_2]} x_{i,uv} = 1, \quad \forall i\in[n]\\
& \left.\begin{array}{l}
\left.\begin{array}{l}
\sum_{v\in[m_2]} v(x_{i,uv}-x_{j,uv}) \le L^{(1)}_{ij,u}\\
\sum_{v\in[m_2]} v(x_{i,uv}-x_{j,uv}) \ge -L^{(1)}_{ij,u}
\end{array}\right\} \ \forall u\in[m_1]\\
\left.\begin{array}{l}
\sum_{u\in[m_1]} u(x_{i,uv}-x_{j,uv}) \le L^{(2)}_{ij,v}\\
\sum_{u\in[m_1]} u(x_{i,uv}-x_{j,uv}) \ge -L^{(2)}_{ij,v}
\end{array}\right\} \ \forall u\in[m_1]
\end{array}\right\} \ \forall i,j\in[n], i<j\\
& x_{i,uv}\in\{0,1\}, \quad \forall i\in[n], u\in[m_1], v\in[m_2]\\
& L^{(1)}_{ij,u}, L^{(2)}_{ij,v}\in\mathbb{R}, \quad \forall u\in[m_1], v\in[m_2], \forall i,j\in[n], i<j.
\end{aligned} \tag{2.20}$$

This MILP has $nm_1m_2 = nM$ integer variables and $\binom{n}{2}(m_1+m_2) = \mathcal{O}(n^2(m_1+m_2))$ continuous variables. The program contains $m_1m_2+n+2\binom{n}{2}(m_1+m_2) = \mathcal{O}(n^2(m_1+m_2)+M)$ constraints. For $m_1 = 1$ or $m_2 = 1$, this grid reduces to the linear grid and the corresponding optimisation programs are equivalent. If the grid is square, i.e. $m_1 = m_2 = m$, for some positive integer m, then the number of constraints is $\mathcal{O}(nM\sqrt{M})$. In this case, there is a difference in complexity depending on the sizes of n and M.

2.6 General grid

The two-dimensional network of Sect. 2.5 was a generalisation of the linear network of Sect. 2.4. Since the 1-norm allows for generalisation to any finite dimensional lattice, this section describes the most general case for grids.

We assume the dimensions of the p-dimensional grid are the same to provide a clearer description. However, similar to the two-dimensional grid, it is possible to use grids of different spatial proportions. Let $G_p = \underbrace{[m] \times \cdots \times [m]}_{p \text{ times}}$, and for $u = (u_1, \ldots, u_{p-1}) \in G_{p-1}$, $r \in [d]$, $v \in [m]$ define

$$u \oplus_r v = (u_1, \ldots, u_{r-1}, v, u_r, \ldots, u_{p-1}) \in G_p, \tag{2.21}$$

that is, in the integer string u we insert the number v at place r.
The general objective now becomes

$$\sum_{\substack{i,j\in[n]\\ i<j}} c_{ij} \left(\sum_{r\in[p]} \left(\sum_{u\in G_{p-1}} \left| \sum_{v\in[m]} v(x_{i,u\oplus_r v} - x_{j,u\oplus_r v}) \right| \right) \right). \tag{2.22}$$

Here, we can again introduce a family of variables $L^{(r)}_{ij,u}$ for all $i, j \in [n], i < j$, $u \in G_{d-1}$ and $r \in [m]$ to linearise the absolute value. This gives us the MILP

$$\begin{aligned}
\min \ & \sum_{\substack{i,j\in[n]\\ i<j}} c_{ij} \left(\sum_{r\in[p]} \left(\sum_{u\in G_{p-1}} L^{(r)}_{ij,u} \right) \right) \\
\text{s.t.} \ & \sum_{i\in[n]} x_{i,\omega} \le K, \quad \forall \omega \in G_p \\
& \sum_{\omega\in G_p} x_{i,\omega} = 1, \quad \forall i \in [n] \\
& \left. \begin{array}{l} \sum_{v\in[m]} u(x_{i,u\oplus_r v} - x_{j,u\oplus_r v}) \le L^{(r)}_{ij,u} \\ \sum_{v\in[m]} u(x_{i,u\oplus_r v} - x_{j,u\oplus_r v}) \ge -L^{(r)}_{ij,u} \end{array} \right\} \quad \forall u \in G_{p-1}, r \in [p], i, j \in [n], i < j \\
& x_{i,\omega} \in \{0, 1\}, \quad \forall i \in [n], \omega \in G_p \\
& L^{(r)}_{ij,u} \in \mathbb{R}, \quad \forall u \in G_{p-1}, r \in [p], i, j \in [n], i < j.
\end{aligned} \tag{2.23}$$

The number of quantum computers in this network is $M = m^p$. This program contains nM integer variables and $\binom{n}{2} pm^{p-1} = \mathcal{O}(n^2 pM^{(p-1)/p})$ continuous variables. Furthermore, there are $m^p + n + 2\binom{n}{2} pm^{p-1} = \mathcal{O}(M + n^2 pM^{(p-1)/p})$ constraints. For $p = 2$ we indeed get the result of the previous section.

3 Local reordering of qubits in a distributed quantum circuit

Now we switch our focus to the problem of local reordering in the distributed case. In local reordering, SWAP gates can be inserted before every quantum gate, such that the quantum gate acts on adjacent qubits. The goal is to find the minimal number of required SWAP gates. These SWAP gates, only inserted to meet the nearest neighbour constraint, are considered overhead. They cost precious calculation time and resources.

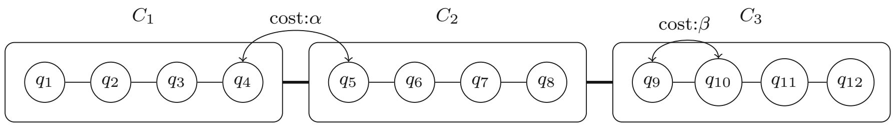

Fig. 11 A line graph on a network of three quantum computers, indicated by the C's. The computers each have a capacity of four qubits which are also connected

Suppose we have a quantum circuit, consisting of m unitary 2-qubit gates $g_{il} \in G$, acting on a total of n qubits $\{q_1, \ldots, q_n\} \equiv Q$. The physical locations of the qubits are distributed between N quantum computers in a linear fashion, i.e., locations $L_1 = (1, \ldots, k_1)$ belong to computer C_1 and locations $L_2 = (k_1+2, \ldots, k_1+k_2+1)$ belong to computer C_2, locations $L_N = (k_1 + \ldots + k_{N-1} + N, \ldots, k_1 + \ldots + k_N + N - 1)$ to computer C_N, where k_i is the qubit capacity of computer C_i. Here, one qubit location is skipped between every two consecutive computers, we will see this helps with the modelling later on. Also suppose we have to comply with nearest neighbour interaction constraints, where gates can only act on two qubits if the corresponding qubits are physically adjacent, so their locations are l_i, l_{i+1}, respectively, for some i. The local reordering problem concerns the micromanagement of the qubit location at the gate level. Before each gate, the qubit order must be adjusted such that the nearest neighbour constraints are satisfied. However, there are costs involved with the reorganisation of the qubit order. SWAP gates are used to interchange the location of two qubits, but they also have to comply with the nearest neighbour constraints and can thus only interchange locations of two adjacent qubits.
Furthermore, interactions between computers are limited to the action of SWAP gates, where two qubits are exchanged between two quantum computers. The SWAP gates between computers will likely be more expensive than the ones within a computer since entanglement between the computers in needed for this purpose, this assumption is however not required for the model to provide valid results.
The goal consists of two parts:

1. Minimise the number of SWAP gates between different computers, associated with a cost of α
2. Minimise the number of SWAP gates within each computer, associated with a cost of β

An illustration is provided for clarification in Fig. 11.
In order to extend the ILP formulation of minimising the number of SWAP gates in the case of one computer [22] such that it also encapsulates the distributed variant of the local reordering problem, no big extension is required. The proposed mathematical model is presented below.
Let us first introduce variables x_i^t, indicating the location $l \in \cup_{i \in [N]} L_i$ of a qubit q_i just before gate g^t is applied. Note that this also specifies the quantum computer on which the qubit is located. To count the required number of SWAP gates when changing the qubit order between gates, variables y_{il}^t are introduced which keep track of the pairwise ordering of two qubits q_i, q_l before gate g^t.

$$y_{il}^t = \begin{cases} 1 & \text{if } x_i^t > x_l^t \\ 0 & \text{otherwise.} \end{cases} \tag{3.1}$$

Changes in the y's, when moving from one gate to the next, mean that two qubits have changed pairwise order. These pairwise changes in order are also known as inversions. Inversions exactly count the number of SWAP gates that are required to change one qubit order to the next.[2]

The nearest neighbour constraints state that a gate can only act on two adjacent qubits, $|x_i^t - x_l^t| \leq 1$. This constraint is linearised using the inequalities

$$x_i^t - x_l^t \leq 1 \quad \forall g_{il}^t \in G, \tag{3.2}$$

$$x_i^t - x_l^t \geq -1 \quad \forall g_{il}^t \in G. \tag{3.3}$$

To keep track of the qubit order and to make sure the distance between qubits is at least 1, the following big-M type constraints are added.

$$x_i^t - x_l^t \leq M y_{il}^t - 1 \quad \forall i, l \in Q, i < l, t \in [m], \tag{3.4}$$

$$x_l^t - x_i^t \leq M(1 - y_{il}^t) - 1 \quad \forall i, l \in Q, i < l, t \in [m], \tag{3.5}$$

where the constant M is chosen to be large enough, in this case $M = N + \sum_i k_i$ will suffice, to make one constraint trivially satisfied. The binary variable y determines for which of the two constraints that holds.

Recall the auxiliary locations left between the quantum computers which were supposed to help us. These borders b are locations indexed by the values

$$b_s = s + \sum_{j \in [s]} k_j \quad s \in [N-1], \tag{3.6}$$

where b_s is the location of the border between quantum computers s and $s+1$. Next, to keep track of qubits changing computer, variables y_{is}^t are introduced.

$$y_{is}^t = \begin{cases} 1 & \text{if } x_i^t > b_s \\ 0 & \text{otherwise.} \end{cases} \tag{3.7}$$

They tell us on which side of the auxiliary location between two computers a qubit is located before gate g^t. If the qubit changes order with the auxiliary location, we can add a cost to the objective later on. The y_i^t are binary and constrained in the following way:

$$x_i^t - b_s \leq M y_{is}^t - 1 \quad \forall i \in Q, i < l, t \in [m], s \in [N-1], \tag{3.8}$$

$$b_s - x_i^t \leq M(1 - y_{is}^t) - 1 \quad \forall i \in Q, i < l, t \in [m], s \in [N-1]. \tag{3.9}$$

[2] The number of required SWAP gates to go from one qubit order to the next is actually a metric on the corresponding elements of the symmetric group $\mathcal{S}_n$, called the Kendall tau metric.

Here, the variable y_i^t is 0 if the location x_i^t of qubit q_i is smaller than the location of the border between computers s and $s+1$.
The absolute change (from gate to gate) in the y variables adds a cost to the objective function. To linearise the absolute values, variables p and r are introduced. The p-variables are used for SWAP gates within a computer:

$$p_{il}^t = \begin{cases} 1 & \text{if the order of qubits } i \text{ and } l \text{ changed from gate } g^t \text{ to } g^{t+1} \\ 0 & \text{otherwise.} \end{cases} \tag{3.10}$$

The r variables keep track of SWAP gates between different quantum computers:

$$r_{is}^t = \begin{cases} 1 & \text{if qubit } i \text{ crossed border } b_s \text{ between gates } g^t \text{ and } g^{t+1} \\ 0 & \text{otherwise.} \end{cases} \tag{3.11}$$

The p's are constrained as

$$y_{il}^t - y_{il}^{t+1} \leq p_{il}^t \quad \forall i, l \in Q, i < l, t \in [m-1], \tag{3.12}$$

$$y_{il}^t - y_{il}^{t+1} \geq -p_{il}^t \quad \forall i, l \in Q, i < l, t \in [m-1], \tag{3.13}$$

and the r's are constrained as

$$y_{is}^t - y_{is}^{t+1} \leq r_{is}^t \quad \forall i \in Q, t \in [m-1], s \in [N-1], \tag{3.14}$$

$$y_{is}^t - y_{is}^{t+1} \geq -r_{is}^t \quad \forall i \in Q, t \in [m-1], s \in [N-1]. \tag{3.15}$$

Next, we formulate the objective function. The objective is of course to minimise the variables p and r, as they count the changes in qubit order and the qubits swapping to another computer, respectively. Note that every time two qubits on different computers are swapped, both the corresponding r- and p-variables become 1. Swapping two qubits on different quantum computers should only cost β and not $\alpha + \beta$, the objective function is therefore

$$\min \sum_{t \in [m-1]} \left(\left(\frac{\alpha - \beta}{2} \sum_{\substack{i \in Q \\ s \in [N-1]}} r_{is}^t \right) + \left(\beta \sum_{\substack{i,l \in Q \\ i<l}} p_{il}^t \right) \right), \tag{3.16}$$

where the $(\alpha - \beta)$ term counteracts the extra counting of the SWAP gate with cost α and the factor of one half prevents us from counting the SWAP over the border between computers twice (once for both qubits).

The complete integer linear program then reads

$$
\begin{aligned}
\min \sum_{t\in[m-1]} & \left(\left(\frac{\alpha-\beta}{2}\sum_{\substack{i\in Q\\ s\in[N-1]}} r_{is}^t\right)+\left(\beta\sum_{\substack{i,l\in Q\\ i<l}} p_{il}^t\right)\right) & \\
\text{s.t. } & x_i^t - x_l^t \le 1 & \forall g_{il}^t \in G\\
& x_i^t - x_l^t \ge -1 & \forall g_{il}^t \in G.\\
& x_i^t - x_l^t \le M y_{il}^t - 1 & \forall i,l \in Q, i<l, t\in[m]\\
& x_l^t - x_i^t \le M(1-y_{il}^t) - 1 & \forall i,l \in Q, i<l, t\in[m]\\
& x_i^t - b_s \le M y_{is}^t - 1 & \forall i \in Q, t\in[m], s\in[N-1]\\
& b_s - x_i^t \le M(1-y_{is}^t) - 1 & \forall i \in Q, t\in[m], s\in[N-1]\\
& y_{il}^t - y_{il}^{t+1} \le p_{il}^t & \forall i,l \in Q, i<l, t\in[m-1]\\
& y_{il}^t - y_{il}^{t+1} \ge -p_{il}^t & \forall i,l \in Q, i<l, t\in[m-1]\\
& y_{is}^t - y_{is}^{t+1} \le r_{is}^t & \forall i \in Q, t\in[m-1], s\in[N-1]\\
& y_{is}^t - y_{is}^{t+1} \ge -r_{is}^t & \forall i \in Q, t\in[m-1], s\in[N-1]\\
& x_i^t \in \cup_{i\in[N]} L_i & \forall i \in Q, t\in[m]\\
& y_{il}^t \in \{0,1\} & \forall i,l \in Q, i<l, t\in[m]\\
& y_{is}^t \in \{0,1\} & \forall i \in Q, t\in[m], s\in[N-1]\\
& r_{is}^t \in \{0,1\} & \forall i \in Q, t\in[m-1], s\in[N-1]\\
& p_{il}^t \in \{0,1\} & \forall i,l \in Q, i<l, t\in[m-1].
\end{aligned}
\tag{3.17}
$$

The size of the ILP model scales as a polynomial in the number of qubits, quantum gates and quantum computers in the instance. The number of variables and the number of constraints are both of the order $\mathcal{O}(n^2m + nMm) = \mathcal{O}(n^2m)$.

4 Concluding remarks and future research

In quantum circuit design, the step to distributed quantum networks gives rise to an extended area of research. How to distribute qubits over the various quantum computers, and how to order qubits within a quantum computer, are naturally arising problems on the interface of distributed quantum computing and nearest neighbour compliant quantum circuit design. These problems have not been discussed in the literature before and are formally introduced in this paper. In order to evaluate these problems, we define the global and local reordering problems for distributed quantum computing. We formalise the mathematical problems and model them as integer linear programming problems, to minimise the number of SWAP gates or the number of interactions between different quantum computers. For global reordering, the problem we identify and analyse is called celestial reordering. In celestial reordering, only the initial distribution of qubits between the quantum computers is optimised. We analyse the problem for various geometries of networks: completely connected networks, general networks, linear arrays and grid-structured networks. We provide an ILP model for each geometry. For local reordering, in networks of quantum computers, we also define the mathematical optimisation problem and we provide an ILP model. However, as these are NP-hard problems, the size of the instances that can be analysed, will be restricted by calculation times. Evaluation of existing or proposed quantum networks will lead to insights in capabilities of networks and algorithms. For development of large scale networks, these optimisation methods will be essential for efficient use. Further research on heuristic approaches for solving these integer linear programs is recommended by the authors.

Compliance with ethical standards

Conflicts of interest The authors declare that they have no conflict of interest.

References

1. AlFailakawi, M.G., Ahmad, I., Hamdan, S.: Harmony-search algorithm for 2D nearest neighbor quantum circuits realization. Expert Syst. Appl. **61**, 16–27 (2016)
2. Bhattacharjee, A., Bandyopadhyay, C., Wille, R., Drechsler, R., Rahaman, H.: A novel approach for nearest neighbor realization of 2D quantum circuits. In: 2018 IEEE Computer Society Annual Symposium on VLSI (ISVLSI), pp. 305–310. IEEE, Hong Kong (2018)
3. Bhattacharjee, A., Bandyopadhyay, C., Wille, R., Drechsler, R., Rahaman, H.: Improved look-ahead approaches for nearest neighbor synthesis of 1D quantum circuits. In: 2019 32nd International Conference on VLSI Design and 2019 18th International Conference on Embedded Systems (VLSID), pp. 203–208. IEEE, Delhi, NCR (2019)
4. Buhrman, H., Röhrig, H.: Distributed quantum computing. In: Rovan, B., Vojtáš, P. (eds.) Mathematical Foundations of Computer Science 2003, pp. 1–20. Springer, Berlin (2003)
5. Cheng, X., Guan, Z., Ding, W.: Mapping from multiple-control Toffoli circuits to linear nearest neighbor quantum circuits. Quantum Inf. Process. **17**(7), 169 (2018)
6. Choi, B.S., Van Meter, R.: An $\Theta(\sqrt{}\{n\})$-depth quantum adder on a 2D NTC quantum computer architecture. J. Emerg. Technol. Comput. Syst. **8**(3), 1–22 (2012)
7. Denchev, V.S., Pandurangan, G.: Distributed quantum computing: A new frontier in distributed systems or science fiction? SIGACT News **39**(3), 77–95 (2008)
8. Ding, J., Yamashita, S.: Exact synthesis of nearest neighbor compliant quantum circuits in 2D architecture and its application to large-scale circuits. IEEE Trans. Comput. Aided Des. Integr. Circuits Syst. 1 (2019)
9. DiVincenzo, D.P.: IBM: the physical implementation of quantum computation. Fortschr. Phys. **48**(9–11), 771–783 (2000)
10. Dueck, G.W., Pathak, A., Rahman, M.M., Shukla, A., Banerjee, A.: Optimization of circuits for IBM's five-qubit quantum computers. In: 2018 21st Euromicro Conference on Digital System Design (DSD), pp. 680–684 (2018)
11. Farghadan, A., Mohammadzadeh, N.: Mapping quantum circuits on 3D nearest-neighbor architectures. Quantum Sci. Technol. **4**(3), 035001 (2019)
12. Hattori, W., Yamashita, S.: Quantum circuit optimization by changing the gate order for 2D nearest neighbor architectures. In: Kari, J., Ulidowski, I. (eds.) Reversible Computation. Lecture Notes in Computer Science, pp. 228–243. Springer, Berlin (2018)
13. Hirata, Y., Nakanishi, M., Yamashita, S., Nakashima, Y.: An efficient conversion of quantum circuits to a linear nearest neighbor architecture. Quantum Inf. Comput. **11**(1&2), 142–166 (2011)
14. Kok, P., Braunstein, S.L.: Entanglement swapping as event-ready entanglement preparation. Fortschr. Phys. Prog. Phys. **48**(5–7), 553–557 (2000)
15. Kole, A., Datta, K., Sengupta, I.: A heuristic for linear nearest neighbor realization of quantum circuits by SWAP gate insertion using N-gate lookahead. IEEE J. Emerg. Sel. Topics Circuits Syst. **6**(1), 62–72 (2016)
16. Kole, A., Datta, K., Sengupta, I.: A new heuristic for N-dimensional nearest neighbor realization of a quantum circuit. IEEE Trans. Comput. Aided Des. Integr. Circuits Syst. **37**(1), 182–192 (2018)
17. Kole, A., Datta, K., Sengupta, I., Wille, R.: Towards a cost metric for nearest neighbor constraints in reversible circuits. Rev. Comput. **9138**, 273–278 (2015)
18. Lin, C., Sur-Kolay, S., Jha, N.K.: PAQCS: physical design-aware fault-tolerant quantum circuit synthesis. IEEE Trans. Very Large Scale Int. Syst. **23**(7), 1221–1234 (2015)

19. Lye, A., Wille, R., Drechsler, R.: Determining the minimal number of swap gates for multi-dimensional nearest neighbor quantum circuits. In: The 20th Asia and South Pacific Design Automation Conference, pp. 178–183 (2015)
20. Matsuo, A., Yamashita, S.: Changing the gate order for optimal LNN conversion. In: De Vos, A., Wille, R. (eds.) Reversible Computation. Lecture Notes in Computer Science, pp. 89–101. Springer, Berlin (2012)
21. Mulderij, J.: Nearest neighbor compliance. Master's thesis, Delft University of Technology (2019)
22. Mulderij, J., Aardal, K., Chiscop, I., Phillipson, F.: A polynomial size model with implicit swap gate counting for exact qubit reordering. Submitted (2019)
23. Nielsen, M.A., Chuang, I.L.: Quantum Computation and Quantum Information: 10th Anniversary Edition. Cambridge University Press, Cambridge (2010). https://doi.org/10.1017/CBO9780511976667
24. Nielsen, M.A., Chuang, I.L., Grover, L.K.: Quantum computation and quantum information. Am. J. Phys. **70**(5), 558–559 (2002)
25. Pedram, M., Shafaei, A.: Layout optimization for quantum circuits with linear nearest neighbor architectures. IEEE Circuits Syst. Mag. **16**(2), 62–74 (2016)
26. Pham, P., Svore, K.M.: A 2D nearest-neighbor quantum architecture for factoring in polylogarithmic depth (2012). arXiv:1207.6655 [quant-ph]
27. Saeedi, M., Wille, R., Drechsler, R.: Synthesis of quantum circuits for linear nearest neighbor architectures. Quantum Inf. Process. **10**(3), 355–377 (2011)
28. Shafaei, A., Saeedi, M., Pedram, M.: Optimization of quantum circuits for interaction distance in linear nearest neighbor architectures. In: 2013 50th ACM/EDAC/IEEE Design Automation Conference (DAC), pp. 1–6 (2013)
29. Shafaei, A., Saeedi, M., Pedram, M.: Qubit placement to minimize communication overhead in 2D quantum architectures. In: 2014 19th Asia and South Pacific Design Automation Conference (ASP-DAC), pp. 495–500 (2014)
30. Shrivastwa, R.R., Datta, K., Sengupta, I.: Fast qubit placement in 2D architecture using nearest neighbor realization. In: 2015 IEEE International Symposium on Nanoelectronic and Information Systems, pp. 95–100 (2015)
31. Tan, Y., Cheng, X., Guan, Z., Liu, Y., Ma, H.: Multi-strategy based quantum cost reduction of linear nearest-neighbor quantum circuit. Quantum Inf. Process. **17**(3), 61 (2018)
32. Wehner, S., Elkouss, D., Hanson, R.: Quantum internet: a vision for the road ahead. Science (2018). https://doi.org/10.1126/science.aam9288
33. Wille, R., Burgholzer, L., Zulehner, A.: Mapping quantum circuits to IBM QX architectures using the minimal number of SWAP and H operations. In: Proceedings of the 56th Annual Design Automation Conference 2019 on DAC '19, pp. 1–6. ACM Press, Las Vegas (2019)
34. Wille, R., Große, D., Teuber, L., Dueck, G.W., Drechsler, R.: RevLib: an online resource for reversible functions and reversible circuits. In: 38th International Symposium on Multiple Valued Logic (ISMVL 2008), pp. 220–225 (2008)
35. Wille, R., Keszocze, O., Walter, M., Rohrs, P., Chattopadhyay, A., Drechsler, R.: Look-ahead schemes for nearest neighbor optimization of 1D and 2D quantum circuits. In: 2016 21st Asia and South Pacific design automation conference (ASP-DAC), pp. 292–297. IEEE, Macao (2016)
36. Wille, R., Lye, A., Drechsler, R.: Exact reordering of circuit lines for nearest neighbor quantum architectures. IEEE Trans. Comput. Aided Des. Integr. Circuits Syst. **33**(12), 1818–1831 (2014)
37. Yimsiriwattana, A., Lomonaco Jr, S.J.: Distributed quantum computing: a distributed shor algorithm. In: Quantum Information and Computation II, vol. 5436, pp. 360–372. International Society for Optics and Photonics (2004)
38. Zulehner, A., Bauer, H., Wille, R.: Evaluating the flexibility of A* for mapping quantum circuits. In: Thomsen, M.K., Soeken, M. (eds.) Reversible Computation, vol. 11497, pp. 171–190. Springer, Cham (2019)
39. Zulehner, A., Paler, A., Wille, R.: An efficient methodology for mapping quantum circuits to the IBM QX architectures. IEEE Trans. Comput. Aided Des. Integr. Circuits Syst. **38**(7), 1226–1236 (2019)