A polynomial size model with implicit SWAP gate counting for exact qubit reordering

J. Mulderij[1,2][0000-0003-2688-9808], K.I. Aardal[2][0000-0001-5974-6219], I. Chiscop[1][0000-0002-1249-8518], and F. Phillipson[1,3][0000-0003-4580-7521]

[1] TNO, The Netherlands
[2] Delft University of Technology, The Netherlands
[3] Maastricht University, The Netherlands
`frank.phillipson@tno.nl`

Abstract. Due to the physics behind quantum computing, quantum circuit designers must adhere to the constraints posed by the limited interaction distance of qubits. Existing circuits need therefore to be modified via the insertion of SWAP gates, which alter the qubit order by interchanging the location of two qubits' quantum states. We consider the Nearest Neighbor Compliance problem on a linear array, where the number of required SWAP gates is to be minimized. We introduce an Integer Linear Programming model of the problem of which the size scales polynomially in the number of qubits and gates. Furthermore, we solve 131 benchmark instances to optimality using the commercial solver CPLEX. The benchmark instances are substantially larger in comparison to those evaluated with exact methods before. The largest circuits contain up to 18 qubits or over 100 quantum gates. This formulation also seems to be suitable for developing heuristic methods since (near) optimal solutions are discovered quickly in the search process.

Keywords: Integer Linear Programming · Nearest Neighbor Architectures · NNC · Optimization · Quantum Circuits · SWAP gate

1 Introduction

The rules that govern physical interactions in a quantum setting allow quantum computing to provide algorithms with a better complexity scaling than their classical counterparts for many naturally arising problems. Exploiting the properties of phenomena such as superposition and entanglement, one can search in a database [18], factor integers [44] or estimate a phase [37] more efficiently than previously possible.

The many advantages of quantum computing come at the price of physical limitations in circuit design. First, relevant coherence times (what is relevant depends on the technology) indicate that information on qubits is perturbed or even lost after some time due to a qubit's interaction with its environment [11]. It is therefore, for a fixed number of qubits, desirable to do calculations with as few gates as possible. A second limitation is induced by nearest neighbor constraints,

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36030-5_7

where 2-qubit quantum gates can only be used when the qubits are physically adjacent. The nearest neighbor constraints have been considered in proposals for a range of potential technological realizations of quantum computers such as ion traps [4, 30, 36], nitrogen-vacancy centers in diamonds [36, 53], quantum dots emitting linear cluster states linked by linear optics [9, 20], laser manipulated quantum dots in a cavity [24] and superconducting qubits [12, 38, 32]. They are also considered in realizations of specific types of circuits and architectures, such as surface codes [47], Shor's algorithm [14], the Quantum Fourier Transform (QFT) [46], circuits for modular multiplication and exponentiation [33], quantum adders on the 2D NTC architecture [8], factoring [40], fault-tolerant circuits [31], error correction [15], and more recently, IBM QX architectures [13, 48, 54, 55].

Up to now, the design of quantum circuits consists of manual work in elementary cases and for specific circuits. As the complexity of the algorithms increases, however, manual synthesis will no longer be feasible. When constructing a circuit from scratch, using only the set of elementary gates, even without considering nearest neighbor constraints, one is solving specific instances of the PSPACE-complete Minimum Generator Sequence problem [23], where the group consists of all unitary matrices and the elementary gate operations form the set of generators. Here one tries to find the shortest sequence of generators to map an input to a given output. A lot of work was done in this area using boolean satisfiability [17], template matching [34, 41] and methods for reversible circuits [51, 3] as all quantum gates perform unitary operations [37]. Other methods consider already designed circuits that do not comply with nearest neighbor constraints. In these approaches, SWAP gates, which swap the information of two adjacent qubits, are inserted into the circuit. The goal herein is to minimize the number of required SWAP gates to make the whole circuit compliant. Within this branch of research there are two approaches to the topic, global and local reordering. Global reordering determines the initial layout of the qubits such that there are as few SWAP gates as possible required in the remainder of the circuit. In order to elude the micromanagement that local reordering is concerned with, the global reordering problem is generally approximated with the NP-complete [16] problem of Optimal Linear Arrangement (OLA) on the interaction graph of the circuit with edge weights taking the Nearest Neighbor Cost [29]. Here the gate sequence is either disregarded [43] or encoded in the weights [28].

The local reordering problem allows for any change in the qubit order before each gate, resulting in a vast feasible region, even for small instances. The more general problem of SWAP minimization where qubits are placed on a coupling graph (two qubits can share a gate if their corresponding nodes share an edge) is shown to be NP-Complete [45] via a reduction from the NP-complete token swapping problem [6, 25]. The problem we consider, where the graph is a simple path, is widely believed to be NP-complete (as conjectured in [21]) but to the best of the authors' knowledge, no formal proof is given yet. Many heuristics have been developed including receding horizon [21, 27, 42, 50], greedy [1, 21], harmony search [1] and OLA on parts of the circuit [39]. Only a few works have dared to approach the problem with exact methods, all of which embody an explicit factorial scaling in the amount of variables or processed nodes, either through the use of the adjacent transposition graph [35], exhaustive searches [10, 21] or explicit cost enumeration for each permutation [52]. The exact approaches have delivered small benchmark instances to compare the heuristics' results to. The size of these benchmark instances typically does not exceed circuits of about 5 qubits and 16 gates due to the vast scaling of the number of variables in the optimization model.

In this work we will provide an exact Integer Linear Programming (ILP) formulation of the Nearest Neighborhor Compliance (NNC) problem that does not entail a factorial scaling in the number of qubits, by implicitly counting the number of required SWAP gates at each reordering step. The power of the commercial solver CPLEX is used to optimally solve the problem for 123 instances from the *RevLib* library [49] and 8 QFT circuits. The considered benchmark instances include the largest circuits to be exactly solved up to this point. They include the QFT for 10 qubits and even a circuit with 18 qubits. The evaluation of the bigger benchmark instances finally allows for heuristics to be compared to exact solutions on larger circuits.

The remainder of this paper is structured as follows. In Section 2 we introduce basic concepts of quantum computing. In Section 3 the problem of NNC is formulated. Next, in Section 4, the proposed mathematical model is introduced. The results are presented and discussed in Section 5. Finally, conclusions are drawn in Section 6.

2 Background

In this section we will first introduce some basic concepts of quantum computing. A more detailed explanation can be found in [37]. Then, a description of decomposing multi-qubit gates is given.

2.1 Building blocks of QC

The quantum version of the classical basic unit of computation, the bit, is the quantum bit (qubit). The qubit has the special property that it does not have to take value 0 or 1, but it can be in a superposition of the computational basis states $|0\rangle \equiv [1,0]^T$ and $|1\rangle \equiv [0,1]^T$. The state of a qubit $|\phi\rangle$ is denoted by a vector in $\mathbb{C}^2$ where in general we write

$$|\phi\rangle = \alpha|0\rangle + \beta|1\rangle, \tag{1}$$

where $\alpha, \beta \in \mathbb{C}$. When information about the state's value is extracted by the means of measurement, the state collapses to a single value. If, for example, the measurement is done in the standard basis, one would obtain $|0\rangle$ with probability $|\alpha|^2$ and $|1\rangle$ with probability $|\beta|^2$. Necessarily, $|\alpha|^2+|\beta|^2 = 1$. In n-qubit systems, the combined state is the tensor product of individual states, which is an element of $\mathbb{C}^{2^n}$. Calculations are done by executing quantum circuits, which consist of a set of qubits and a list of quantum gates. The initial qubit states are the input of the calculation. The gates operate, in order, on specified qubits. Afterwards, a measurement is performed on one or more of the qubits to determine the probabilistic outcome of the calculation. Quantum gates are inherently reversible and are denoted by linear operators in the form of invertible matrices. Their action on the combined qubit state is simply the matrix vector product.

Below we will introduce some of the most common quantum gates, starting with the controlled NOT (CNOT) gate, see Fig. 1.

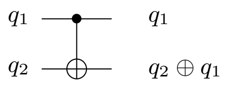

Fig. 1. A CNOT gate. Qubit q_1 is the control qubit and q_2 is the target qubit.

The controlled CNOT gate is one of the most commonly used gates. It is also used to construct the SWAP gate by placing three CNOT gates consecutively such as in Fig. 2.

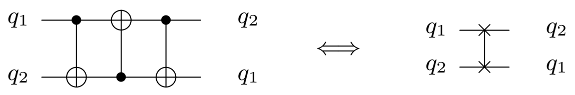

Fig. 2. A decomposed and composite SWAP gate. The operations are equivalent, the gates interchange the states of two qubits.

The SWAP gate will be the main tool used to overcome the physical constraints that limit quantum circuit design. We will, for the remainder of this text, not make a distinction between interchanging two qubits and interchanging the quantum states of two qubits.

2.2 Decomposing multi-qubit gates

Many circuits make use of composite gates that resemble entire circuits themselves. They are often performed on more than two qubits at once, take for example the quantum Fourier Transform (QFT), which can act on any number of qubits. In order to describe what it means for a gate to act on adjacent qubits, it only makes sense to consider 2-qubit gates. To achieve this without losing the meaning of the circuit, we have to do a modification in the following two cases:

1. Gates that only act on a single qubit are ignored for the rest of this research. These gates are of no interest in this context.
2. Gates that act on more than two qubits are decomposed into 2-qubit gates. The fact that this is always possible can be found in [37].

The second point can be implemented in a great variety of ways and doing this “optimally" is outside the scope of this work. We therefore make two straightforward design choices: 1) We only consider circuits using multiple-control Toffoli gates, Peres gates and multiple-control Fredkin gates up to a certain size; 2) We always decompose a given circuit in the same way. There is clearly room for improvement here, but the search space we consider is large enough as it is. We ignore all single-qubit gates during the modification to a nearest neighbor compliant circuit. The normal Toffoli gate's decomposition, with two control qubits can be found in [5] in the section “Three-Bit Networks”. In the same work, the decomposition of a 3-control Toffoli gate is shown in the section “n-Bit Networks”. The decomposition of the 4-control Toffoli is the direct extension of the previous decompositions. The Peres gate is decomposed as in the circuit “peres_8.real” from RevLib [49]. The Fredkin gate is decomposed as in the circuit “fredkin_5.real", also from RevLib. The two-qubit controlled Fredkin gate is decomposed into a controlled-NOT gate, a Toffoli gate and another controlled-NOT gate as shown by [3] in Fig. 2.4c. Larger composed gates do not make an appearance in the circuits that are considered in this work.

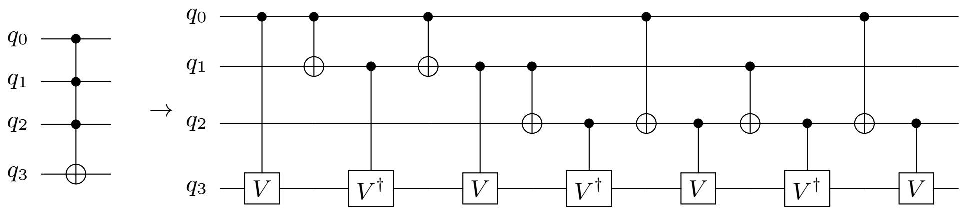

Fig. 3. The decomposition of a Toffoli gate with 3 control qubits and one target qubit into only 2-qubit gates. Here we have $V^4 = X$, where X is the usual Pauli-X gate.

3 Problem Definition

In this section, some basic definitions will be introduced in order to formalize the NNC problem.

For the NNC problem, the actual operation corresponding to a gate that is being used has no influence on the problem. Only the qubits on which the gate acts matter. Some definitions are introduced below. Denote the set Q of n qubits as the set of integers $Q = \{1, \ldots, n\}$. Since all the qubits have one physical location in a one dimensional array, the locations are numbered as $L = (1, \ldots, n)$ and are in a fixed order. To keep track of the location of each qubit before every gate, the notion of a qubit order will be introduced.

Definition 1. *Let $\mathcal{S}_n$ be the permutation group and $[n]$ the vector $(1, \ldots, n)$. Then a qubit order is a permutation denoted by the vector $\tau([n])$ with $\tau \in \mathcal{S}_n$, which maps the qubits to locations. We call τ^t the qubit order before gate t.*

Now that the qubit orders are defined, one needs a way of altering such an order. This is done via the previously mentioned SWAP gates.

Definition 2. *A SWAP gate is an adjacent transposition* $\tau \in \mathcal{S}_n$ *that permutes a qubit order,* $\tau \circ (q_1, \dots, q_i, q_{i+1}, \dots, q_n) = (q_1, \dots, q_{i+1}, q_i, \dots, q_n)$*, by interchanging the positions of two adjacent qubits.*

The number of SWAP gates that one minimally requires to "move" from one qubit order to another is inherently equal to the Kendall tau distance between the corresponding permutations.

Definition 3. *Given two permutations* $\tau_1, \tau_2 \in \mathcal{S}_n$ *for some fixed* n*, the Kendall tau distance between* τ_1 *and* τ_2 *is defined as*

$$I(\tau_1, \tau_2) \equiv |\{(i,j) \mid 1 \le i,j \le n, \tau_1(i) < \tau_1(j), \tau_2(i) > \tau_2(j)\}|. \tag{2}$$

The nearest neighbor interaction constraints can only be formulated once the concept of quantum gates has been properly introduced in this setting.

Definition 4. *Let* $q_i, q_j \in Q$ *be two qubits such that* $i \neq j$*. Let* g_{ij} *be an unordered pair* $g = \{q_i, q_j\}$*. Then we say that* g_{ij} *is a quantum gate, or simply a gate, that acts on qubits* q_i *and* q_j*. When the specific qubits do not matter in the context, the subscripts may be omitted. When multiple gates are present and their order is important, this will be reflected with a superscript as* g^t*.*

Please note that this definition only allows for quantum gates that act on pairs of qubits. If a gate (in the more general sense) acts on more qubits, we assume it to be decomposed, whilst if it only works on one qubit, the gate can be ignored.

To describe an entire quantum circuit, multiple gates are needed and their order is important. To this end, a gate sequence is introduced.

Definition 5. *Let* $g^1, \dots, g^m$ *be* m *gates. Let* G *be the finite sequence of gates* $G = (g^1, \dots, g^m)$*, then we say* G *is a gate sequence of size* m*.*

We also assume the gate sequence to be given and fixed. Allowing changes in the gate order when some commutative rules are satisfied, as was done in [35, 22, 19], is beyond the scope of this work.

Now we can introduce the concept of a quantum circuit more formally.

Definition 6. *Let* Q *be the set of qubits and* G *be a gate sequence. Let* QC *be a tuple of the set of qubits and the gate sequence* $QC = (Q, G)$*. Then we say that* QC *is a quantum circuit.*

At the core of the problem are the nearest neighbor (NN) constraints. Formalizing these requires a number of the above definitions. These constraints are what make the problem difficult.

Definition 7. *Given are a gate g_{ij}^t and a qubit order τ^t before that gate. We say that the gate complies with the NN constraints if $|\tau(i) - \tau(j)| = 1$, i.e., if the qubits on which the gate acts are adjacent in the qubit order. If, given a qubit order for each gate, all the gates in a quantum circuit's gate sequence comply with the NN constraints, we say that the quantum circuit complies with the NN constraints.*

Now that all these concepts have been formalized, we can continue with defining the problem of NNC.

Problem 1 (Nearest Neighbor Compliance Problem).
Input: A quantum circuit $QC = (Q, G)$ with $|Q| = n$ qubits and $|G| = m$ gates and an integer $k \in \mathbb{Z}_{\geq 0}$.
Question: Do there exist qubit orders $\tau^t, t \in [m]$, one before each gate of QC, such that the sum of the Kendall tau distances between consecutive qubit orders satisfies $\sum_{t=1}^{m-1} I(\tau^t, \tau^{t+1}) \leq k$ and such that the quantum circuit complies with the NN constraints?

In the minimization version of the problem, which we model in the next section, we seek to find the smallest integer k such that Problem 1 is still answered affirmatively. Considering the problem in this way, we do not require the qubits to end up in the same qubit order as they started out in. We also do not allow for changes in the gate order and do not optimize over different ways of decomposing multi-qubit quantum gates. The objective function in the minimization problem simply counts the number of required SWAP gates.

Note that calculating the Kendall tau distance between two permutations can be naively done in $\mathcal{O}(n^2)$ time, following the steps of the bubble sort algorithm [26]. A faster computation of the distance, in $\mathcal{O}(n\sqrt{\log n})$ time, can be found in [7].

We will however not be concerned with explicitly listing the Kendall tau distances for all $n!$ permutations. In order to avoid the listing, the metric should be implicitly calculated in the model. The objective function, variables and constraints that allow us to do so, will be introduced in the next section.

4 Mathematical Model

In this section the proposed ILP formulation of the NNC minimization problem will be discussed in detail. First, the variables and constraints are presented and explained. Finally, the complete model is given, along with a linearization of the constraints.

Given a quantum circuit $QC = (Q, G)$, we introduce integer variables $x_i^t \in L = \{1, \ldots, n\}$ for the location of each qubit $q_i \in Q$ before each gate $g^t \in G$. Since the goal is to avoid the explicit $n!$ scaling in the number of variables and constraints, we make use of the Kendall tau metric to count the number of required SWAP gates when going from one qubit order τ^t to the next τ^{t+1}. To accomplish this, keeping track of the pairwise order of the qubits is essential. We introduce binary variables to do precisely this,

$$y_{ij}^t = \begin{cases} 1 & \text{if location } x_i^t \text{ is before location } x_j^t \text{ in qubit order } \tau^t \\ 0 & \text{else.} \end{cases} \tag{3}$$

Keeping track of changes in the y variables when moving from one qubit order to the next allows us to count the amount of SWAP gates needed. The x and y variables are related through the following big-M type constraints,

$$x_i^t - x_j^t \leq M y_{ij}^t - 1 \qquad \forall i,j \in Q, i<j, t \in [m] \tag{4}$$
$$x_j^t - x_i^t \leq M(1 - y_{ij}^t) - 1 \qquad \forall i,j \in Q, i<j, t \in [m] \tag{5}$$

where M is a big enough constant, $M = (n+1)$ being sufficient in this case. Note that these constraints also enforce two important features:

1. No two qubits can be at the same location at the same time.
2. The definition of the y variables is enforced by the constraints.

For fixed i, j and t, one of the two constraints is always trivially satisfied due to the large value of M. The -1 term in the right-hand side even ensures that the location indices differ by at least one from each other. This allows us, later on, to relax the x variables to be continuous without losing the property that feasible solutions have integer x variables.

To make sure that the result also complies with the NN constraints, the following constraints need to be added:

$$x_i^t - x_j^t \leq 1 \qquad \forall g_{ij}^t \in G \tag{6}$$
$$x_i^t - x_j^t \geq -1 \qquad \forall g_{ij}^t \in G. \tag{7}$$

For each gate that acts on qubits q_i and q_j, the qubit order that is assumed just before the gate, it is required to have the qubits in adjacent locations.

The objective is to minimize the total amount of absolute changes in the y variables,

$$\min \sum_{\substack{i,j \in Q \\ i<j}} \sum_{t \in [m-1]} |y_{ij}^t - y_{ij}^{t+1}|. \tag{8}$$

Note that the objective function exactly computes the Kendall tau distance between every two consecutive qubit orders. Currently, the objective function is not linear, so extra binary variables k_{ij}^t are added to complete the model. These substitute $|y_{ij}^t - y_{ij}^{t+1}|$ in the objective function and are constrained in the following way

$$y_{ij}^t - y_{ij}^{t+1} \leq k_{ij}^t \qquad \forall i,j \in Q, i<j, t \in [m-1] \tag{9}$$
$$y_{ij}^t - y_{ij}^{t+1} \geq -k_{ij}^t \qquad \forall i,j \in Q, i<j, t \in [m-1]. \tag{10}$$

Now the k variables can be substituted into Expression (8), which, together with the constraints, result in the ILP model:

$$\begin{aligned}
\min \quad & \sum_{\substack{i,j\in Q\\ i<j}} \sum_{t\in[m-1]} k_{ij}^t \\
\text{subject to} \quad & (4),(5),(6),(7),(9),(10) \\
& x_i^t \in \{1,\dots,n\} && \forall i \in Q, t \in [m] \\
& y_{ij}^t \in \{0,1\} && \forall i,j \in Q, i<j, t\in[m] \\
& k_{ij}^t \in \{0,1\} && \forall i,j \in Q, i<j, t\in[m-1]
\end{aligned} \tag{11}$$

The number of variables in this formulation is equal to

$$\#\text{ variables} = n^2 m - \frac{n^2-n}{2}, \tag{12}$$

and the number of constraints in the ILP is equal to

$$\#\text{ constraints} = 2(n^2-n)m - n^2 + n + 2m, \tag{13}$$

which is polynomial in the number of qubits and gates. In order to improve running times in practice, it helps to relax variables to take continuous values. We state the following about this relaxation:

Proposition 1. *Allowing the x- and k-variables to take continuous values does not change the optimal value.*

Proof. The x-variables must take values that are pairwise separated from each other by at least 1 due to constraints $(4),(5)$. There are n variables that all have to take a value in a connected interval of length n, all spaced at least 1 from each other. This can only be done if the x's are all integer and all integer values are taken. The k-variables are constrained by $(9),(10)$. Since the y-variables are binary, their difference is also binary (or -1, in which case $k=0$ is allowed). Since we are minimizing over the k-variables, their value will always assume the smallest possible allowed value by the constraints, which is integer.

Even though relaxing these variables does not impact the objective value of optimal solutions, it reduces the number of integer-restricted variables which improves the running time in practice.

5 Experimental Results

In this section, the exact solutions provided by the proposed method are compared to the previous best exact approaches in terms of instances they can solve, as well as state-of-the-art heuristic approaches in terms of attained objective value.

5.1 Experimental setup

The mathematical model as described in the previous section has been implemented in Python and solved with the commercial solver CPLEX 12.7 through the Python API. All but the quantum fourier transform instances, which were constructed following the circuit of [37], were obtained form the RevLib [49] website. The evaluations were conducted using up to 16 threads of 2.4 GHz each, working with 16 GB of RAM. All instances were solved to optimality.

The benchmark instances are subdivided over three tables, according to the amount of qubits addressed. In the first column of each table, the name of the circuit is provided, and in the second column, n denotes the number of qubits in the circuit. In the third column, $|G|$ denotes the number of 2-qubit gates present in the circuit after gate decomposition and the removal of single-qubit gates. The optimal value of the local reordering problem, i.e., the minimum number of needed SWAP gates to make the circuit nearest neighbor compliant, is provided in the fourth column. The column "Time" denotes the run time in seconds. The column entitled "Time E" denotes the running time of other exact methods, also in seconds. Exact running times with subscript a are from [52], subscript b from [35]. Heuristic solution's objective values are presented in the last column, denoted by "# SWAPS H". Here the subscript c indicates the results are from [28], subscript d from [42], subscript e from [2], subscript f from [27] and subscript g from [50]. An asterisk as superscript indicates that for the other exact solution methods, either the objective value differs, or the number of gates differs or they both differ. For the heuristic results, the asterisk indicates that the number of gates differs or the objective value of the heuristic is lower than that of the proposed exact method. These anomalies are believed to find their roots in differing gate decomposition methods, resulting in slightly different instances.

5.2 Results

The running time required to solve the instance is heavily dependent on three factors:

1. The number of qubits in the quantum circuit,
2. The number of gates in the quantum circuit,
3. The minimal number of required SWAP gates.

The number of qubits and gates is expected to heavily influence the running time. The number of qubits is the term that influences the run time the most.

Table 1. Benchmark instances with three or four qubits

Benchmark	n	$\|G\|$	$\|S\|$	Time	Time E	# SWAPS H
QFT_QFT3	3	3	1	0.02	-	-
peres_10	3	4	1	0.14	0.1_a	-
peres_8	3	4	1	0.06	0.1_a	-
toffoli_2	3	5	1	0.12	0.2_a	-
toffoli_1	3	5	1	0.1	0.1_a	-
peres_9	3	6	1	0.02	2463_a	-
fredkin_7	3	7	1	0.16	-	-
ex-1_166	3	7	2	0.08	0.1_a	-
fredkin_5	3	7	1	0.15	$0.1_a, 0.1_b^*$	-
ham3_103	3	8	2	0.04	-	-
miller_12	3	8	2	0.14	$745.6_a, 0.1_b$	-
ham3_102	3	9	1	0.05	0.1_a^*	-
3_17_15	3	9	2	0.04	$630.2_a, 0.1_b^*$	-
3_17_13	3	13	3	0.12	0.1_a^*	$4_c^*, 4_d, 3_e, 6_g$
3_17_14	3	13	3	0.15	0.1_a^*	-
fredkin_6	3	15	3	0.06	4.6_a	-
miller_11	3	17	4	0.15	0.1_a^*	-
QFT_QFT4	4	6	3	0.17	-	-
toffoli_double_3	4	7	1	0.11	$0.9_a, 0.1_b^*$	-
rd32-v1_69	4	8	2	0.16	0.1_a	-
decod24-v1_42	4	8	2	0.12	$7.7_a, 0.1_b^*$	-
rd32-v0_67	4	8	2	0.07	1.6_a	$2_c, 2_d$
decod24-v2_44	4	8	3	0.07	0.1_b^*	-
decod24-v0_40	4	8	3	0.06	0.1_b^*	-
decod24-v3_46	4	9	3	0.09	$0.1_a, 0.1_b^*$	$3_c, 3_d$
toffoli_double_4	4	10	2	0.07	200_a^2	-
rd32-v1_68	4	12	3	0.24	0.4_a^*	-
rd32-v0_66	4	12	0	0.09	0.4_a^*	-
decod24-v0_39	4	15	5	0.53	0.5_a	-
decod24-v2_43	4	16	5	0.23	0.1_a^*	-
decod24-v0_38	4	17	4	0.57	19.2_a	-
decod24-v1_41	4	21	7	0.5	-	-
hwb4_52	4	23	8	0.97	-	$9_c, 10_d, 9_e, 9_f$
aj-e11_168	4	29	12	5.36	-	-
4_49_17	4	30	12	6.1	-	$12_c^*, 12_d, 16_e$
decod24-v3_45	4	32	13	6.25	-	-
mod10_176	4	42	15	7.94	-	-
aj-e11_165	4	44	18	9.36	-	$36_d, 33_g^*$
mod10_171	4	57	24	27.18	-	-
4_49_16	4	59	22	24.23	-	-
mini-alu_167	4	62	27	23.7	-	-
hwb4_50	4	63	23	17.61	-	-
hwb4_49	4	65	23	21.64	-	-
hwb4_51	4	75	28	75.09	-	-

This is due to the fact that the number of feasible solutions scales factorially in the number of qubits. Surprisingly, the run time also scales quite badly with the number of required SWAP gates. During the Branch & Bound tree search, the upper bound determined by CPLEX, which is the best feasible solution found up to that point, converges to the optimal value (or close to it) rather quickly. The best known lower bound, however, takes a long time to improve. When the number of required SWAP gates increases, the time needed to improve the lower bound all the way to the optimal value increases as well. This phenomenon is analyzed for two of the benchmark instances that require a lot of SWAP gates:

1. **mod8-10_177** The search method found a feasible solution with an objective value within 10% of the optimal value in $2.4 \cdot 10^6$ iterations, found an

optimal solution in $4.0 \cdot 10^7$ iterations, and proved optimality by a matching lower bound after $1.7 \cdot 10^8$ iterations.

2. **decod24-enable_126** The search method found a feasible solution with an objective value within 10% of the optimal value in $5.3 \cdot 10^6$ iterations, found an optimal solution in $1.0 \cdot 10^7$ iterations, and proved optimality by a matching lower bound after $5.8 \cdot 10^7$ iterations.

If a 10% optimality gap would suffice, only less than 2% of the total number of iterations would be needed in the first case, and 10% in the second case. This observation indicates that running an incomplete Branch and Bound algorithm might be an interesting and easy-to-implement heuristic algorithm.

The 131 evaluated benchmark instances are listed in the tables below. The improvement in computation time with respect to previous exact methods is significant. The results show exact solutions that are obtained for much larger circuits than previously held possible. The largest instance with respect to the number of qubits has as much as 18 qubits. Furthermore, for the first time, NNC has been solved to optimality for circuits with more than 100 quantum gates.

Table 2. Benchmark instances with six or more qubits. No times of other exact methods are known.

Benchmark	n	$\|G\|$	$\|S\|$	Time	# SWAPS H
graycode6_47	6	5	0	0.02	-
graycode6_48	6	5	0	0.02	-
QFT_QFT6	6	15	11	7.43	$11_c, 12_d$
decod24-enable_124	6	21	5	1.86	-
decod24-enable_125	6	21	5	1.83	-
decod24-bdd_294	6	24	7	9.37	-
mod5adder_129	6	71	34	534.38	-
mod5adder_128	6	77	36	1103.51	$45_c^*, 51_d, 46_g^*$
decod24-enable_126	6	86	37	1954.28	-
xor5_254	7	5	3	0.61	-
ex1_226	7	5	3	0.25	-
QFT_QFT7	7	21	16	28.26	$28_c, 26_d, 18_g$
4mod5-bdd_287	7	23	7	4.3	-
ham7_106	7	49	28	495.43	-
ham7_105	7	65	34	1613.33	-
ham7_104	7	83	42	3238.82	56_c^*
QFT_QFT8	8	28	23	334.6	$32_c, 33_d, 31_g$
rd53_139	8	36	11	76.29	-
rd53_138	8	44	11	100.86	-
rd53_137	8	66	35	6271.11	-
QFT_QFT9	9	36	30	1482.53	$52_c, 54_d, 49_g$
QFT_QFT10	10	45	39	39594.99	64_g
g mini_alu_305	10	57	23	1711.75	-
sys6-v0_144	10	62	19	887.71	-
rd73_141	10	64	21	845.05	-

6 Conclusion

In this paper we consider the local reordering scheme for nearest neighbor architectures of quantum circuits. We propose a new mathematical model that counts the number of required SWAP gates implicitly, by using specific properties of the constraints. The implicit counting improves upon previous exact approaches in which costs were explicitly determined for each permutation, leading to a factorial scaling of the model size, and therefore, a high running time. The presented innovations result in a great improvement in the model size, such that the resulting ILP only contains $\mathcal{O}(n^2m)$ variables and constraints.

The benchmark instances with available exact solutions known in the literature were no larger than circuits with five qubits and no more than twenty gates, due to the excessive running times. The proposed method can handle quantum circuits with five qubits and 112 gates or up to eighteen qubits and sixteen gates. In total 131 benchmark instances are evaluated, most of which have not been solved to optimality.

Because the implicit counting is based on counting inversions in permutations, the formulation is not easily translated to the popular higher dimensional cases where qubits are placed on a 2D or 3D grid. To the authors' best knowledge there is no known polynomial time algorithm that, in 2- or 3-dimensional grids, solves the subproblem of calculating the minimum number of required SWAP gates when transforming one qubit order into another. Such a method could have great impact on exact solution methods in the higher-dimensional setting.

Practical experience with the Branch & Bound tree search indicates that finding a (near) optimal feasible solution does not consume the most computation time. This means that solving the ILP heuristically, with a restriction in running time or iteration count for example, could make for a good heuristic solution method.

For future research we propose to look at algorithms dedicated to specific hardware topologies and benchmarks on larger number of qubits and gates, potentially with approximate, nearly-optimal approach and comparison with existing transpilers.

References

1. AlFailakawi, M.G., et al.: Harmony-search algorithm for 2d nearest neighbor quantum circuits realization. Expert Syst. with Appl. **61**, 16–27 (2016)
2. AlFailakawi, M.G., et al.: Line ordering of reversible circuits for linear nearest neighbor realization. Quantum Inf. Process. **12**(10), 3319–3339 (2013)
3. Alhagi, N.: Synthesis of Reversible Functions Using Various Gate Libraries and Design Specifications. Tech. rep., Portland State University (2000)
4. Amini, J.M., et al.: Toward scalable ion traps for quantum information processing. New J. Phys. **12**(3), 033031 (2010)
5. Barenco, A., et al.: Elementary gates for quantum computation. Phys. Rev. A **52**(5), 3457–3467 (1995)
6. Bonnet, E., et al.: Complexity of Token Swapping and Its Variants. Algorithmica **80**(9), 2656–2682 (2018)
7. Chan, T.M., Pătraşcu, M.: Counting Inversions, Offline Orthogonal Range Counting, and Related Problems. In: Discrete Algorithms, pp. 161–173. (2010)
8. 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)

9. Devitt, S.J., Fowler, A.G., et al.: Architectural design for a topological cluster state quantum computer. New J. Phys. **11**(8), 083032 (2009)
10. Ding, J., Yamashita, S.: Exact Synthesis of Nearest Neighbor Compliant Quantum Circuits in 2d architecture and its Application to Large-scale Circuits. IEEE Trans. on Comput.-Aided Des. of Integr. Circuits and Syst. pp. 1–1 (2019)
11. DiVincenzo, D.P., IBM: The Physical Implementation of Quantum Computation. Fortschr. der Phys. **48**(9-11), 771–783 (2000)
12. DiVincenzo, D.P., Solgun, F.: Multi-qubit parity measurement in circuit quantum electrodynamics. New J. Phys. **15**(7), 075001 (2013)
13. Dueck, G.W., Pathak, A., et al.: Optimization of Circuits for IBM's five-qubit Quantum Computers. pp. 680–684 (2018)
14. Fowler, A.G., et al.: Implementation of Shor's Algorithm on a Linear Nearest Neighbour Qubit Array. arXiv:quant-ph/0402196 (2004).
15. Fowler, A.G., et al. : Quantum Error Correction on Linear Nearest Neighbor Qubit Arrays. Phys. Rev. A **69**(4), 042314 (2004)
16. Garey, M.R., Johnson, D.S.: Computers and Intractability; A Guide to the Theory of NP-Completeness. W. H. Freeman & Co., New York, NY, USA (1979)
17. Große, D., et al.: Exact Multiple-Control Toffoli Network Synthesis With SAT Techniques. Comput.-Aided Des. of Integr. Circuits and Syst. **28**(5), 703–715 (2009)
18. Grover, L.K.: Quantum Mechanics Helps in Searching for a Needle in a Haystack. Phys. Rev. Lett. **79**(2), 325–328 (1997)
19. Hattori, W., Yamashita, S.: Quantum Circuit Optimization by Changing the Gate Order for 2d Nearest Neighbor Architectures. In: Lecture Notes in Computer Science, pp. 228–243. Springer (2018)
20. Herrera-Martí, D.A., et al.: A Photonic Implementation for the Topological Cluster State Quantum Computer. Phys. Rev. A **82**(3), 032332 (2010)
21. Hirata, Y., et al.: An Efficient Conversion of Quantum Circuits to a Linear Nearest Neighbor Architecture. Quantum Inf. and Comput. **11**(1&2), 25 (2011)
22. Itoko, T., et al.: Quantum circuit compilers using gate commutation rules. In:Design Automation, pp. 191–196. ACM Press, Tokyo, Japan (2019)
23. Jerrum, M.R.: The Complexity of Finding Minimum-Length Generator Sequences. Theor. Comput. Sci. **36**, 25 (1985)
24. Jones, N.C., et al.: Layered Architecture for Quantum Computing. Phys. Rev. X **2**(3), 031007 (2012)
25. Kawahara, J., et al.: The Time Complexity of the Token Swapping Problem and Its Parallel Variants. In: WALCOM: Algorithms and Computation, pp. 448–459. (2017)
26. Knuth, D.E.: The Art of Computer Programming, Volume 3: Sorting and Searching, Second Edition, vol. 3, 2nd edn. (1974)
27. Kole, A., et al.: A Heuristic for Linear Nearest Neighbor Realization of Quantum Circuits by SWAP Gate Insertion Using$N$-Gate Lookahead. IEEE J. on Emerg. and Sel. Top. in Circuits and Syst. **6**(1), 62–72 (2016)
28. Kole, A., et al.: A New Heuristic for $N$ -Dimensional Nearest Neighbor Realization of a Quantum Circuit. IEEE Trans. on Comput.-Aided Des. of Integr. Circuits and Syst. **37**(1), 182–192 (2018)
29. Kole, A., et al.: Towards a Cost Metric for Nearest Neighbor Constraints in Reversible Circuits. Rev. Comput. **9138**, 273–278 (2015)
30. Kumph, M., Brownnutt, M., Blatt, R.: Two-dimensional arrays of radio-frequency ion traps with addressable interactions. New J. Phys. **13**(7), 073043 (2011)

31. Lin, C., et al.: PAQCS: Physical Design-Aware Fault-Tolerant Quantum Circuit Synthesis. IEEE Trans. on Very Large Scale Int. Syst. **23**(7), 1221–1234 (2015)
32. Linke, N.M., et al.: Experimental comparison of two quantum computing architectures. Proc Natl Acad Sci USA **114**(13), 3305–3310 (2017)
33. Markov, I.L., Saeedi, M.: Constant-Optimized Quantum Circuits for Modular Multiplication and Exponentiation. arXiv:1202.6614 [quant-ph] (2012).
34. Maslov, D., et al.: Quantum Circuit Simplification Using Templates. In: Design, Automation and Test in Europe, pp. 1208–1213. IEEE, Munich, Germany (2005)
35. Matsuo, A., Yamashita, S.: Changing the Gate Order for Optimal LNN Conversion. In: Reversible Computation, pp. 89–101. Springer (2012)
36. Nickerson, N.H., et al.: Topological quantum computing with a very noisy network and local error rates approaching one percent. Nat. Commun. **4**(1) (2013)
37. Nielsen, M.A., et al.: Quantum Computation and Quantum Information. Am. J. of Phys. **70**(5), 558–559 (2002)
38. Ohliger, M., Eisert, J.: Efficient measurement-based quantum computing with continuous-variable systems. Phys. Rev. A **85**(6), 062318 (2012)
39. Pedram, M., Shafaei, A.: Layout Optimization for Quantum Circuits with Linear Nearest Neighbor Architectures. IEEE Circuits and Syst. Mag. **16**(2), 62–74 (2016)
40. Pham, P., Svore, K.M.: A 2d Nearest-Neighbor Quantum Architecture for Factoring in Polylogarithmic Depth. arXiv:1207.6655 [quant-ph] (2012).
41. Saeedi, M., et al.: Synthesis of quantum circuits for linear nearest neighbor architectures. Quantum Inf. Process. **10**(3), 355–377 (2011)
42. Shafaei, A., et al.: Optimization of quantum circuits for interaction distance in linear nearest neighbor architectures. In: Design Automation Conf., pp. 1–6 (2013)
43. Shafaei, A., et al.: Qubit placement to minimize communication overhead in 2d quantum architectures. In: Design Automation Conference, pp. 495–500 (2014)
44. Shor, P.: Algorithms for quantum computation: discrete logarithms and factoring. In: Foundations of Computer Science, pp. 124–134. (1994)
45. Siraichi, M.Y., et al.: Qubit Allocation. In: Code Generation and Optimization, pp. 113–125. ACM, New York, NY, USA (2018).
46. Takahashi, Y., et al.: The Quantum Fourier Transform on a Linear Nearest Neighbor Architecture. Quantum Info. Comput. **7**(4), 383–391 (2007)
47. Versluis, R., et al.: Scalable quantum circuit and control for a superconducting surface code. Phys. Rev. Applied **8**(3), 034021 (2017)
48. Wille, R., et al.: Mapping Quantum Circuits to IBM QX Architectures Using the Minimal Number of SWAP and H Opers. In: Design Autom. Conf., pp. 1–6. (2019)
49. Wille, R., et al.: RevLib: An Online Resource for Reversible Functions and Reversible Circuits. In: Multiple Valued Logic, pp. 220–225 (2008)
50. Wille, R., et al.: Look-ahead schemes for nearest neighbor optimization of 1d and 2d quantum circuits. In: Design Automation Conference, pp. 292–297. (2016)
51. Wille, R., et al.: Considering nearest neighbor constraints of quantum circuits at the reversible circuit level. Quantum Inf. Process. **13**(2), 185–199 (2014)
52. Wille, R., et al.: Exact Reordering of Circuit Lines for NN Quantum Architectures. IEEE Comput.-Aided Des. of Integr. Circuits and Syst. **33**(12), 1818–1831 (2014)
53. Yao, N.Y., et al.: Quantum Logic between Remote Quantum Registers. Phys. Rev. A **87**(2), 022306 (2013)
54. Zulehner, A., et al.: Evaluating the Flexibility of A* for Mapping Quantum Circuits. In: Reversible Computation, vol. 11497, pp. 171–190. (2019)
55. Zulehner, A., et al.: An Efficient Methodology for Mapping Quantum Circuits to the IBM QX Architectures. IEEE Trans. on Comput.-Aided Des. of Integr. Circuits and Syst. **38**(7), 1226–1236 (2019)

Table 3. Benchmark instances with five qubits

Benchmark	n	$\|G\|$	$\|S\|$	Time	Time E	# SWAPS H
4mod5-v1_25	5	7	1	0.26	11705.3_a	-
4gt11_84	5	7	1	0.06	16.6_a	$1_c, 1_d, 1_e$
4gt11-v1_85	5	7	1	0.09	-	-
4mod5-v0_20	5	8	2	0.08	45.5_a	-
4mod5-v1_22	5	9	1	0.08	548.8^*_a	-
QFT_QFT5	5	10	6	0.41	1.6_a	$7_c, 6_d$
mod5d1_63	5	11	2	0.12	-	-
4mod5-v0_19	5	12	3	0.84	55.3^*_a	-
4gt11_83	5	12	3	0.15	9^*_a	-
4mod5-v1_24	5	12	3	0.28	-	-
mod5mils_65	5	12	4	0.26	-	-
mod5mils_71	5	12	2	0.15	-	-
alu-v2_33	5	13	4	0.45	-	-
alu-v1_29	5	13	4	0.61	-	-
alu-v0_27	5	13	4	0.48	-	-
mod5d2_70	5	14	5	0.43	-	-
alu-v3_35	5	14	5	0.38	-	-
alu-v4_37	5	14	5	0.37	-	-
alu-v1_28	5	14	4	0.26	-	-
4gt13-v1_93	5	15	5	0.69	489.3^*_a	$7^*_c, 6_d, 4^*_e$
4gt13_92	5	15	6	0.53	-	-
4gt11_82	5	16	6	0.89	-	-
4mod5-v0_21	5	17	8	2.84	-	-
rd32_272	5	18	7	0.94	-	-
alu-v3_34	5	18	4	0.4	-	-
mod5d2_64	5	19	6	1.81	-	-
alu-v0_26	5	21	8	3.56	-	-
4gt5_75	5	21	6	1.1	-	$9^*_c, 12_d$
4mod5-v0_18	5	23	8	3.35	-	-
4mod5-v1_23	5	24	9	5.06	-	$9_c, 9_d, 15_e$
one-two-three-v2_100	5	24	7	5.37	-	-
one-two-three-v3_101	5	24	7	2.96	-	-
rd32_271	5	26	11	7.37	-	-
4gt5_77	5	28	10	6.2	-	-
4gt5_76	5	29	10	5.45	-	-
alu-v4_36	5	30	9	6.34	-	$15^*_c, 18_d, 17_e$
4gt13_91	5	30	8	4.46	-	-
4gt13_90	5	34	12	6.77	-	-
4gt10-v1_81	5	34	13	12.38	-	$18^*_c, 20_d, 16_e, 24^*_g$
one-two-three-v1_99	5	36	15	17.27	-	-
4gt4-v0_80	5	36	19	43.45	-	$34_d, 33_f$
4mod7-v0_94	5	38	12	12.83	-	-
alu-v2_32	5	38	16	22.05	-	-
4mod7-v0_95	5	38	14	14.59	-	$19^*_c, 21_d, 22_e$
4mod7-v0_95	5	38	14	14.59	-	$19^*_c, 21_d, 22_e$
4mod7-v1_96	5	38	14	13.49	-	-
one-two-three-v0_98	5	40	15	15.67	-	-
4gt12-v0_88	5	41	20	34.01	-	-
4gt12-v1_89	5	44	22	52.36	-	$35_d, 26_e, 32_f$
sf_275	5	46	18	21.42	-	-
4gt4-v0_79	5	49	22	80.16	-	-
4gt4-v0_78	5	53	26	167.03	-	-
4gt4-v0_72	5	53	24	49.7	-	-
4gt12-v0_87	5	54	22	45.88	-	-
4gt4-v1_74	5	57	29	84.87	-	-
4gt12-v0_86	5	58	26	108.35	-	-
mod8-10_178	5	68	37	389.47	-	-
one-two-three-v0_97	5	71	32	76.8	-	-
4gt4-v0_73	5	89	40	699.65	-	-
mod8-10_177	5	93	48	3650.26	-	72_d
alu-v2_31	5	100	49	2906.35	-	-
hwb5_55	5	101	48	2264.0	-	$59_c, 63_d, 60_e, 66_g$
rd32_273	5	104	50	4631.7	-	-
alu-v2_30	5	112	55	13558.87	-	-