Exploiting Symmetries in Optimal Quantum Circuit Design

Frank de Meijer * Dion Gijswijt † Renata Sotirov ‡

Abstract

A physical limitation in quantum circuit design is the fact that gates in a quantum system can only act on qubits that are physically adjacent in the architecture. To overcome this problem, SWAP gates need to be inserted to make the circuit physically realizable. The nearest neighbour compliance problem (NNCP) asks for an optimal embedding of qubits in a given architecture such that the total number of SWAP gates to be inserted is minimized. In this paper we study the NNCP on general quantum architectures. Building upon an existing linear programming formulation, we show how the model can be reduced by exploiting the symmetries of the graph underlying the formulation. The resulting model is equivalent to a generalized network flow problem and follows from an in-depth analysis of the automorphism group of specific Cayley graphs. As a byproduct of our approach, we show that the NNCP is polynomial time solvable for several classes of symmetric quantum architectures. Numerical tests on various architectures indicate that the reduction in the number of variables and constraints is on average at least 90%. In particular, NNCP instances on the star architecture can be solved for quantum circuits up to 100 qubits and more than 1000 quantum gates within a very short computation time. These results are far beyond the computational capacity when solving the instances without the exploitation of symmetries.

Keywords: Quantum computing, nearest neighbour constraints, symmetry reduction, Cayley graphs, fixed point subspace, generalized network flow problem

1 Introduction

The most commonly used model for quantum computation is that of the gated quantum computer, where a calculation is performed by executing so-called quantum circuits. A quantum circuit acts on multiple quantum bits, i.e., qubits, which are the physical particles embedded in a quantum system. Whereas classical bits exclusively take the Boolean values zero or one, qubits can be in a superposition state, which upon measurement are displayed as zero or one with a certain probability. A quantum circuit sequentially acts on the qubits via quantum gates, which are unitary transformations that sequentially adjust the state of one or more qubits to perform an operation. Quantum circuits extend on the gate model for classical computing, and hence, a quantum computer can perform any computation that a classical computer can perform [50]. However, based on quantum phenomena such as superposition and entanglement, a quantum system is able to perform a much broader spectrum of operations. For an extensive overview of the advances and applications of quantum computing, see e.g., [47].

Given the current state of technology, most physical implementations of quantum gates operate on only one or two qubits at a time [28, 50, 54]. In this setting, gates that act on more than two qubits therefore need to be realized as a sequence of gates of size at most two, which, fortunately, is possible for any quantum gate [50]. For instance, the set of one-qubit gates and two-qubit controlled-NOT gates is universal [4], meaning that this set is sufficient to perform any quantum computation.

*Corresponding author. Delft Institute of Applied Mathematics, Delft University of Technology, The Netherlands, f.j.j.demeijer@tudelft.nl

†Delft Institute of Applied Mathematics, Delft University of Technology, The Netherlands, d.c.gijswijt@tudelft.nl

‡CentER, Department of Econometrics and OR, Tilburg University, The Netherlands, r.sotirov@uvt.nl

The qubits in a quantum system are physically embedded in a certain design, i.e., the quantum architecture. This architecture is commonly represented as a coupling graph, where the vertices represent the qubits and an edge is drawn between two qubits whenever the qubits can communicate in the quantum system. With "communicate", we refer to the possibility to apply a gate to the two qubits and consequently affect their simultaneous state. Among the special coupling graphs considered in the literature are the linear array, see e.g., [6, 9, 30, 35, 49], the two-dimensional grid, see e.g., [1, 5, 10], the three-dimensional grid [16], the IBM QX architecture, see e.g., [65], but also general coupling graphs [7, 58, 63, 40, 11, 32].

A physical limitation of the architecture is that two-qubit gates can only be applied when the qubits are physically adjacent to each other in the coupling graph. These restrictions are known as nearest neighbour constraints and have been subject of interest in the design of quantum realizations of specific circuits, see e.g., [18], or the design of quantum architectures itself, see [49] and the references therein. Instead of research on quantum realizations that comply with the nearest neighbour constraints, we can also disregard these constraints at first and alter existing quantum circuits to make them feasible, which will be the followed approach in this paper.

A quantum circuit can be made compliant with respect to the nearest neighbour constraints by the insertion of SWAP gates. A SWAP gate acts on two adjacent qubits by interchanging their location in the coupling graph[1]. If the coupling graph is connected, any quantum circuit can be made compliant by the insertion of a finite number of SWAP gates and there are often many ways to do so. However, due to a qubit's interaction with its environment [13], quantum systems currently still suffer from physical instability of qubit states after some period of time. This raises the desire for quantum circuits with as few gates as possible. We therefore prefer to add the minimum number of SWAP gates in order to make a circuit compliant.

Given a quantum circuit and a coupling graph, the nearest neighbour compliance problem (NNCP) asks for an optimal sequential allocation of the qubits over the quantum architecture such that the total number of SWAP gates to be inserted is minimized. With "sequential", we refer to the decision variables to not only concern the initial allocation, but also the actual SWAP operations that take place over time. The NNCP was proven to be $\mathcal{NP}$-hard via a reduction from the token swapping problem [58].

Most research on the NNCP has been on heuristic methods, such as greedy methods [1, 30], harmony search [1], optimal linear arrangement [52] and receding horizon methods [30, 35, 55, 67]. Exact approaches to tackle the NNCP include exhaustive search [12, 30], explicit cost enumeration [68] and linear programming (LP) based methods on the adjacent transposition graph [46, 48]. All these methods embrace an implicit factorial scaling in the number of qubits, due to the inherited total number of possible assignments of the qubits. Recently, also polynomial sized models have been considered that are based on mixed-integer linear programming [49, 62]. The construction considered in [49] is based on the linear array coupling graph, while the models in [62] consider ordering problems for distributed quantum computing. Other research focuses on a related version of the NNCP, where an initial qubit ordering has to be realized that minimizes the (approximated) number of SWAP operations, without actually considering the exact insertions into the quantum circuit, see [56, 37, 36].

Building upon the LP formulation considered in [46, 48], a main feature of our approach concerns the exploitation of symmetries in the model. The literature on symmetry reduction methods in mathematical optimization is extensive, and we refer the reader to [41, 45] for comprehensive overviews in this direction. It is well-known that symmetries in integer linear programming (ILP) problems lead to poor behaviour of numerical algorithms, due to the costly duplication of computational effort in branching approaches. To reduce this negative effect, symmetries need to be broken, e.g., by perturbation, symmetry-breaking inequalities (e.g., [44]) or specialized branching techniques (e.g., [57]). The literature on symmetry reduction for integer linear programs (ILPs) can be distinguished between problem-based approaches, whose symmetry groups are known a priori (see e.g., [39]), or generic techniques. The latter class on one hand contains methods based on

[1]Strictly speaking, a SWAP gate does only interchange the state of the involved qubits, while the actual hardware entities remain unchanged in the architecture.

branching tree reductions, such as isomorphism pruning [42, 43] and orbital branching [51]. Alternative methods mainly consider symmetry-handling constraints to restrict the feasible region of an optimization problem by eliminating symmetric solutions. Two well-known streams in this direction are the utilization of orbitopes [34] and fundamental domains [21]. Branching tree reductions and symmetry-handling constraints can also be combined, see e.g., [14].

When considering symmetry reduction methods for linear programs, a major research line considers the study of symmetric polyhedra, see [45, Section 6] and the references therein. Another research line considers the exploitation of symmetries in the simplex algorithm [60, 61]. Bödi et al. [8] consider the exploitation of symmetries in linear programs by restricting to the subspace of fixed points under a linear map induced by the symmetries in the program. This approach can be generalized to convex programs and is closely related to the invariant-based symmetry reduction approaches applied to conic and semidefinite programs, see e.g., Gatermann and Parrilo [26], to which our reduction method also belongs.

Main results and outline

In this paper we consider the nearest neighbour compliance problem on general coupling graphs. Following the linear programming (LP) formulation derived in [46], we analyse the group symmetry of the underlying graph, which is a sequence of connected Cayley graphs. By exploiting these symmetries, we reduce the LP model in the number of variables and constraints, leading to a symmetry-reduced formulation for solving the NNCP. We show the theoretical and practical strength of our approach for several classes of symmetric coupling graphs for which the reduction is most significant, namely the graphs that embrace a large automorphism group.

The LP formulation of [46] can be viewed as a single-pair shortest path problem on a directed graph that we refer to as the graph $X = (V, A)$. As a first step in our approach, we consider the automorphism group of the subgraphs of X. Each subgraph is a Cayley graph of the symmetric group $\mathbb{S}_n$ generated by the edges in the coupling graph of the quantum architecture. We derive the full automorphism group of such Cayley graphs in the case that it is normal, and review some conditions on the coupling graph under which normality holds. Afterwards, we extend these automorphism results of the subgraphs to derive the automorphism group of X. In particular, we derive an explicit group description of a subgroup G_X of the automorphism group of X, which is the full automorphism group of X when normality holds. We also study the orbit and orbital structure of the group action of G_X on X. The results on the group structure of these Cayley graphs are in itself interesting, as such graphs are of main importance in interconnection networks [25, 29].

By averaging over each orbital of the action of G_X on X via the Reynolds operator, we show how the LP formulation can be reduced following the approach of [8]. We show that the resulting reduced LP formulation is equivalent to a generalized network flow problem on an auxiliary graph following from our construction. For symmetric coupling graphs, this reduced LP formulation is significantly smaller in size. As a byproduct of our approach, we show that the NNCP is polynomial time solvable for coupling graph whose automorphism group scales factorially in the number of qubits, e.g., the star graph or complete bipartite graphs with one of the sizes fixed. The construction of the reduced LP formulation follows completely from the algebraic analysis of X and does not rely on the use of any external algebraic software.

Although the ingredients of our approach are presented generally, we explicitly show how the reduced LP can be constructed for three special graph types: the cycle graph, the star graph and the biclique graph. For each of these classes, we show how the orbital structure unfolds by analyzing a specific subgroup of the automorphism group of the coupling graphs.

Finally, we test our symmetry-reduced formulation on real and randomly generated quantum circuits defined on the above-mentioned coupling graphs. Our numerical tests confirm that the effort spent in the algebraic analysis pays off, as computation times to solve an instance are several orders of magnitude smaller compared to the nonreduced model. Whereas the model from [46] can only solve instances up to 8 qubits, the largest instances we solve contain up to 100 (resp. 40) qubits and several hundreds of quantum gates on the star (resp. biclique) coupling graph. Observe that such instances are far out of reach for the nonreduced model, as this would require the use of at

least $100! \approx 9.33 \cdot 10^{157}$ constraints and even more variables.

This paper is structured as follows. Section 2 formally introduces the NNCP and reviews the shortest path formulation of [46]. In Section 3 we analyse the automorphism group of the graph underlying the formulation, as well as its orbit and orbital structure. These algebraic properties are exploited in Section 4, where we present our symmetry-reduced NNCP formulation. In Section 5 we apply our approach to several specific types of coupling graphs. Computational results are discussed in Section 6.

Notation

A directed graph is given by a pair (V, A), where V is a vertex set and $A \subseteq V \times V$ an arc set. For $i \in V$ and $A' \subseteq A$, we let $\delta^+(i, A')$ (resp. $\delta^-(i, A')$) denote the set of arcs in A' that leave (resp. enter) vertex i. In case $A' = A$, we just write $\delta^+(i)$ (resp. $\delta^-(i)$).

The set of integers $\{1, \dots, k\}$ is denoted by $[k]$. For a subset S of a finite set T, we denote by $\mathbb{1}_S \in \{0,1\}^T$ the indicator vector of S in T.

For a group G, we denote by id_G (or simply id) its identity element. When G acts on a set X, we denote by $\mathrm{Orb}(x) := \{g(x) : g \in G\} \subseteq X$ the orbit of $x \in X$ under the action of G. The set of orbits of X under G is denoted by the quotient X/G. For any $g \in G$, we let $X^g := \{x : g(x) = x\}$ be the set of fixed points of g.

The symmetric group on a finite set Y is denoted by $\mathrm{Sym}(Y)$. When $Y = [n]$, its symmetric group is shortened to $\mathbb{S}_n$. A permutation $\tau \in \mathbb{S}_n$ can be written in one-line notation, i.e., as an ordered array $(\tau(1), \tau(2), \dots, \tau(n))$ of images of $\{1, \dots, n\}$ under τ. Alternatively, τ can be written in the usual cycle notation of permutations. Permutations that only consist of a 2-cycle are called transpositions. For $S \subseteq [n]$ and $\tau \in \mathbb{S}_n$, we define the sets $\tau(S) := \{\tau(s) : s \in S\}$ and $\tau^{-1}(S) := \{\tau^{-1}(s) : s \in S\}$. Moreover, we let $\mathbb{S}_n(S) := \{\tau \in \mathbb{S}_n : \tau(S) = S\}$ denote the setwise stabilizer of S under $\mathbb{S}_n$.

2 Nearest neighbour compliance problem

A given quantum circuit can be made feasible with respect to the adjacent interaction constraints by inserting SWAP gates. Although these do not interfere with the functionality of the quantum circuit, the total number of gates is favoured to be as small as possible. The nearest neighbour compliance problem (NNCP) aims at finding an embedding of the qubits over a given architecture such that the number of SWAP gates needed to make the final circuit feasible with respect to the adjacent interaction constraints is minimized.

In this section we formally introduce the nearest neighbour compliance problem as a shortest path problem.

2.1 Mathematical formulation of the NNCP

We make two model assumptions about the quantum circuits under consideration. First, quantum gates that act on a single qubit always comply with the adjacent interaction constraints and are therefore not taken into consideration. Second, it only makes sense to talk about adjacency in the context of two-qubit quantum gates. If a quantum gate acts on more than two qubits, we first decompose it into two-qubit gates. This is always possible [50] and there exist a large variety of ways for doing this. Throughout this paper, we assume without loss of generality that quantum circuits consist of a sequence of two-qubit gates.

Let $Q = [n]$ denote the set of qubits of the quantum system. The qubits need to be embedded in a certain topology, that we refer to as the architecture of the quantum system. This architecture is fixed and can be modeled as a graph (L, E). Here $L = [n]$ denotes a set of physical locations and $E \subseteq L^{(2)}$ is the adjacency structure of the architecture. That is, if $\{i, j\} \in E$, then locations i and j are physically adjacent to each another and can therefore directly share information. The graph is denoted as the coupling graph of the quantum system and denoted by $\mathrm{Coup}(E) := (L, E)$.

We assume that (L, E) is connected, which implies that all pairs of locations can indirectly share information.

Each qubit in Q needs to be assigned to a physical location in L. A bijection $\tau : L \to Q$ is called a qubit order. To present a qubit order, we use one-line notation with respect to the images in Q. For example, the order

$$\tau = (\tau(1), \tau(2), \tau(3), \tau(4)) = (2, 3, 1, 4)$$

corresponds to the assignment where qubit 2 is on location 1, qubit 3 on location 2, qubit 1 on location 3 and qubit 4 on location 4. The set of all qubit orders on n qubits is equal to $\mathbb{S}_n$.

A SWAP gate interchanges the qubits on two locations in the embedding. It can also be modeled as an element $\sigma \in \mathbb{S}_n$, where σ is a transposition. Using cycle notation, the SWAP gate $\sigma = (i\ j)$ applied on the qubit order τ interchanges the qubits $\tau(i)$ and $\tau(j)$. Applying this SWAP gate can be seen as a right action of σ on $\mathbb{S}_n$, i.e.,

$$\begin{aligned}\tau \circ \sigma &= (\tau(1), \tau(2), \ldots, \tau(i), \ldots, \tau(j), \ldots, \tau(n)) \circ (i\ j) \\ &= (\tau(1), \tau(2), \ldots, \tau(j), \ldots, \tau(i), \ldots, \tau(n)),\end{aligned}$$

for all $\tau \in \mathbb{S}_n$. To simplify notation, we omit the $\circ$ in group actions and just write $\tau\sigma$ in the sequel.

Remark 2.1. Although both elements of $\mathbb{S}_n$, τ represents a qubit order, while σ represents a SWAP gate. To discriminate between these objects, we always use one-line notation for qubit orders and cycle notation for SWAP gates throughout the paper.

A SWAP gate can only be applied to qubits on locations that are adjacent in $\mathrm{Coup}(E)$. Whenever there is an edge $\{i, j\} \in E$, the SWAP gate $(i\ j)$ acts on adjacent locations. Let

$$T := \{(i\ j) \in \mathbb{S}_n\ :\ \{i, j\} \in E\} \tag{1}$$

denote the set of transpositions that correspond to a SWAP gate in the quantum system.

Given two qubit orders $\tau_1, \tau_2 \in \mathbb{S}_n$, we are interested in the minimum number of SWAP gates that need to be applied to τ_1 to obtain τ_2 by only using SWAP gates from T. Let $J_T : \mathbb{S}_n \times \mathbb{S}_n \to \mathbb{Z}_+$ be defined as

$$J_T(\tau_1, \tau_2) := \min\{k\ :\ \tau_2 = \tau_1\sigma_1\sigma_2 \ldots \sigma_k,\ \sigma_1, \ldots, \sigma_k \in T\},$$

which forms a metric on all qubit orders and depends on the quantum architecture T. Observe that this metric is left-invariant, i.e., $J_T(\tau_1, \tau_2) = J_T(\pi\tau_1, \pi\tau_2)$ for all $\pi \in \mathbb{S}_n$, implying that $J_T(\tau_1, \tau_2)$ equals the length of the shortest sequence of transpositions of T needed to generate $\tau_2^{-1}\tau_1$. It is known that finding such minimum-length sequence is in general *PSPACE*-complete [33]. For special types of coupling graphs, however, the metric J_T is computationally tractable, e.g., when $\mathrm{Coup}(E)$ is a path or the complete graph. For these cases, J_T coincides with the Kendall tau distance and the Cayley distance, respectively.

Let $i, j \in Q$ be two qubits such that $i \neq j$. Then the unordered pair $g_{ij} = \{i, j\}$ is a two-qubit quantum gate that acts on qubits i and j. Whenever the specific qubits on which the gate acts are irrelevant, we sometimes omit the subscripts. A finite sequence $C = (g^1, \ldots, g^m)$ of gates $g^1, \ldots, g^m$ is called a gate sequence of size m. Given a set of qubits Q and a gate sequence C, the tuple $\Gamma = (Q, C)$ is called a quantum circuit.

We say that a qubit order τ complies with a gate g_{ij} if qubits i and j are adjacent in τ with respect to the coupling graph $\mathrm{Coup}(E)$, i.e., if $\tau^{-1}(g_{ij}) = \{\tau^{-1}(i), \tau^{-1}(j)\} \in E$. We now formulate the NNCP.

Definition 2.2 (NNCP)**.** *Let $\Gamma = (Q, C)$ be a quantum circuit with n qubits and m gates, and let $\mathrm{Coup}(E) = (L, E)$ be the coupling graph of the underlying architecture. Then, the nearest neighbor compliance problem asks for a sequence of qubit orders τ^k, $k \in [m]$, each one corresponding to an order prior to applying a gate of C, such that $\sum_{k=1}^{m-1} J_T(\tau^k, \tau^{k+1})$ is minimized and such that τ^k complies with g^k for all $k \in [m]$.*

The NNCP as presented in Definition 2.2 is known to be $\mathcal{NP}$-hard in general [58].

We end this section by introducing the notion of the so-called gate graph, which captures the underlying qubit dependencies imposed by the gates in the circuit.

Definition 2.3. *Let $\Gamma = (Q, C)$ be a quantum circuit. The gate graph (Q, U) is an undirected graph that has vertex set Q and edge set $U = \{g : g \in C\}$.*

The gate graph (Q, U) will be exploited in Section 3.2.

2.2 The NNCP as a shortest path problem

In this section we show how the NNCP can be modeled as a shortest path problem in a directed graph following the construction of [46, 48].

Let an instance of the NNCP as defined in Section 2.1 be given. Of key importance in the reduction to a shortest problem is the notion of a Cayley graph.

Definition 2.4 (Cayley graph)**.** *Let G be a finite group and let S be a subset of G such that $\mathrm{id}_G \notin S$ and $S = S^{-1} := \{s^{-1} : s \in S\}$. The Cayley graph $\mathrm{Cay}(G, S)$ on G with respect to S is defined as the (directed) graph with vertex set G and arc set $\{(g, gs) : g \in G, s \in S\}$.*

Observe that $\mathrm{Cay}(G, S)$ as in Definition 2.4 contains an arc if and only if it also contains the reversed arc. Although this suggests that any $\mathrm{Cay}(G, S)$ is undirected, we stick to the setting of two reversed directed arcs, since we will employ the Cayley graphs as subgraphs of a larger directed graph.

Let $H := \mathrm{Cay}(\mathbb{S}_n, T)$, where T is given by (1). More precisely, the vertex and arc set of H are given by $V(H) := \mathbb{S}_n$ and $A(H) := \{(\tau, \tau\sigma) : \tau \in \mathbb{S}_n, \sigma \in T\}$, respectively. Each vertex in $V(H)$ represents a qubit order, while an arc in $A(H)$ represents a SWAP gate that translates a qubit order into another qubit order with respect to the coupling graph. Now, we define the subgraphs H^k for $k \in [m]$ as disjoint copies of H, one for each gate in the circuit.

The m subgraphs H^k are merged to obtain a graph $X = (V, A)$. The vertex set V of X consists of the union of all V^k, $k \in [m]$, as well as a source s and sink t, i.e., $V = \{s\} \cup V^1 \cup \cdots \cup V^m \cup \{t\}$. Since the subgraphs $H^1, \ldots, H^m$ are identical, we use superscripts to indicate to which subgraph a vertex belongs. For example, τ^k and τ^{k+1} correspond to the same qubit order in subgraph k and $k+1$, respectively.

The arc set A of X contains the union of all A^k, $k \in [m]$. Moreover, the arcs between different subgraphs are introduced by the following sets:

$$\begin{aligned} D^0 &:= \{(s, \tau^1) : \tau^1 \in V^1\} \\ D^k &:= \{(\tau^k, \tau^{k+1}) : \tau^k \in V^k, \tau^{k+1} \in V^{k+1},\ (\tau^k)^{-1}(g^k) \in E\}, \quad k \in [m-1] \\ D^m &:= \{(\tau^m, t) : \tau^m \in V^m,\ (\tau^m)^{-1}(g^m) \in E\}. \end{aligned} \tag{2}$$

These sets can be interpreted as follows. The set D^0 contains an arc from s to all nodes in H^1. For all $k \in [m-1]$, D^k contains the connecting arcs from H^k to H^{k+1}. Suppose the gate g^k acts on qubits i and j. Then we include an arc from a qubit order τ^k in H^k to the same qubit order τ^{k+1} in H^{k+1} if and only if i and j are adjacent in τ^k with respect to $\mathrm{Coup}(E)$. That is, whenever $(\tau^k)^{-1}(g^k) = \{(\tau^k)^{-1}(i), (\tau^k)^{-1}(j)\} \in E$. Similarly, D^m contains all arcs from τ^m with this property to the sink node t. Now, the arc set A of X is given by

$$A = A^1 \cup \cdots \cup A^m \cup D^0 \cup D^1 \cup \cdots \cup D^m.$$

We set the cost of each arc in $A^k, k \in [m]$, equal to one, as traversing these arcs corresponds to applying one SWAP gate. The cost of the arcs in $D^k, k = 0, \ldots, m$, is equal to zero, as no SWAP gates are applied when moving from a subgraph to the next.

This construction implies the following result.

Theorem 2.5 ([48])**.** *Any (s,t)-path in X corresponds to a sequence $(\tau^1, \ldots, \tau^m)$ of qubit orders that all comply with the adjacent interaction constraints. A shortest (s,t)-path in X corresponds to an optimal solution of the NNCP.*

There are many algorithms in the literature for solving the shortest path instance, e.g., Dijkstra's algorithm with Fibonacci heaps [20]. Alternatively, we can solve it as a linear programming (LP) problem. For all $k \in [m]$ and $e \in A^k$, let x_e denote a variable that is one if arc e is used on a path, and zero otherwise. Similarly, for all $k \in \{0\} \cup [m]$ and $e \in D^k$, let y_e denote a variable that is one if arc e is used on a path, and zero otherwise. Then the shortest (s,t)-path in X can be found by solving the following LP:

$$
\begin{aligned}
&\min \sum_{k=1}^{m} \sum_{e \in A^k} x_e \\
&\text{s.t.} \sum_{e \in D^0} y_e = 1, \quad \sum_{e \in D^m} y_e = 1 \\
&\text{(SPP)} \qquad \sum_{e \in \delta^-(\tau, D^{k-1})} y_e + \sum_{e \in \delta^-(\tau, A^k)} x_e = \sum_{e \in \delta^+(\tau, D^k)} y_e + \sum_{e \in \delta^+(\tau, A^k)} x_e \quad \forall \tau \in V^k, k \in [m] \\
&\qquad 0 \le x_e \le 1 \quad \forall e \in A^k,\ k \in [m], \\
&\qquad 0 \le y_e \le 1 \quad \forall e \in D^k,\ k \in \{0\} \cup [m].
\end{aligned}
$$

3 Symmetries in $X = (V, A)$

The graph X constructed in Section 2.2 contains $\Theta(mn!)$ vertices and $\Theta(|E|mn!)$ arcs. The bottleneck in solving the NNCP to optimality is clearly the factorial scaling in the number of qubits. Fortunately, for many structured quantum system architectures, the problem can be reduced by exploiting the symmetries in X. In this section we study these symmetries in terms of the automorphism group of X.

In Section 3.1 and 3.2 we study the automorphism group of Cayley graphs generated by transpositions and the automorphism group of X, respectively. In Section 3.3 we study the orbit and orbital structure induced by this group action on X. The results in this section are the key ingredients of the symmetry reduction explained in Section 4.

3.1 Automorphism group of $\mathbf{Aut(Cay(\mathbb{S}_n, T))}$

For a directed graph X with vertex set V and arc set A, a permutation $\rho \in \mathrm{Sym}(V)$ is called an automorphism of X if $(\rho(i), \rho(j)) \in A$ if and only if $(i,j) \in A$. We also say that such ρ acts on X. The automorphism group of X is the group of all automorphisms of X and is denoted by $\mathrm{Aut}(X)$.

In order to determine the automorphism group of the graph X introduced in Section 2.2, we start by considering the automorphism group of the subgraphs H^k, $k \in [m]$. Recall that all H^k are identical and equal to $\mathrm{Cay}(\mathbb{S}_n, T)$, where T is a set of transpositions, see (1). Hence, the goal of this subsection is to study $\mathrm{Aut}(\mathrm{Cay}(\mathbb{S}_n, T))$.

There exist several works in the literature on the automorphism group of Cayley graphs generated by transpositions. As indicated by Feng [17], we can show that $\mathbb{S}_n$ acts on $\mathrm{Cay}(\mathbb{S}_n, T)$ by left multiplication. That is, for any $a \in \mathbb{S}_n$ the mapping $\tau \mapsto a\tau$ defines an automorphism of $\mathrm{Cay}(\mathbb{S}_n, T)$. All such automorphisms form a subgroup of $\mathrm{Aut}(\mathrm{Cay}(\mathbb{S}_n, T))$. We can also show that the group $\mathrm{Aut}(\mathrm{Coup}(E))$ acts on $\mathrm{Cay}(\mathbb{S}_n, T)$ by right multiplication via the mapping $\tau \mapsto \tau b^{-1}$, which is an automorphism of $\mathrm{Cay}(\mathbb{S}_n, T)$ for all $b \in \mathrm{Aut}(\mathrm{Coup}(E))$. To verify this, let (τ_1, τ_2) be an arc in $\mathrm{Cay}(\mathbb{S}_n, T)$. Then $\tau_2 = \tau_1 \sigma_1$ for some $\sigma_1 \in T$. The image of this arc under the action of an element $b \in \mathrm{Aut}(\mathrm{Coup}(E))$ is

$$(\tau_1 b^{-1},\, \tau_2 b^{-1}) = (\tau_1 b^{-1},\, \tau_1 \sigma_1 b^{-1}) = (\tau_1 b^{-1},\, \tau_1 b^{-1} b \sigma_1 b^{-1}).$$

It is well-known that if a permutation maps i to j, then the conjugate of this permutation by b maps $b(i)$ to $b(j)$. Therefore, if $\sigma_1 = (i\ j)$, then $\sigma_2 := b\sigma_1 b^{-1} = (b(i)\ b(j))$. Since b is an automorphism of $\mathrm{Coup}(E)$, $\sigma_2 \in T$, which implies that $(\tau_1 b^{-1},\, \tau_2 b^{-1})$ is again an arc of $\mathrm{Cay}(\mathbb{S}_n, T)$.

Since $\tau \mapsto \tau b^{-1}$ is bijective, it follows that $\mathrm{Aut}(\mathrm{Coup}(E))$ indeed acts on $\mathrm{Cay}(\mathbb{S}_n, T)$ by right multiplication.

We now show how both group actions are combined in order to obtain a subgroup of $\mathrm{Aut}(\mathrm{Cay}(\mathbb{S}_n, T))$. Let us define the mapping $\theta : \mathbb{S}_n \times \mathrm{Aut}(\mathrm{Coup}(E)) \to \mathrm{Aut}(\mathrm{Cay}(\mathbb{S}_n, T))$ given by

$$\theta(a, b) := (\tau \mapsto a\tau b^{-1}). \tag{3}$$

Indeed, $\theta(a, b)$ is the composition of an action by left multiplication by an element $a \in \mathbb{S}_n$ and a right multiplication by an element $b \in \mathrm{Aut}(\mathrm{Coup}(E))$ (in arbitrary order). So, for all (a, b) in its domain, $\theta(a, b)$ is indeed an automorphism of $\mathrm{Cay}(\mathbb{S}_n, T)$. We can show that the map θ is a group homomorphism that is injective.

Theorem 3.1. *For $n \geq 3$, the mapping θ is a group homomorphism from $\mathbb{S}_n \times \mathrm{Aut}(\mathrm{Coup}(E))$ to $\mathrm{Aut}(\mathrm{Cay}(\mathbb{S}_n, T))$ that is injective.*

Proof. We start by showing that θ is indeed a group homomorphism. Let $(a_1, b_1), (a_2, b_2) \in \mathbb{S}_n \times \mathrm{Aut}(\mathrm{Coup}(E))$. Then, for all $\tau \in \mathbb{S}_n$:

$$\begin{aligned} \theta\left((a_1, b_1)(a_2, b_2)\right)(\tau) &= \theta\left((a_1 a_2, b_1 b_2)\right)(\tau) = a_1 a_2 \tau (b_1 b_2)^{-1} = a_1 a_2 \tau b_2^{-1} b_1^{-1} \\ \theta((a_1, b_1))\theta((a_2, b_2))(\tau) &= \theta(a_1, b_1)(a_2 \tau b_2^{-1}) = a_1 a_2 \tau b_2^{-1} b_1^{-1}. \end{aligned}$$

Hence, θ is a group homomorphism. To prove injectivity, assume that $(a_1, b_1), (a_2, b_2) \in \mathbb{S}_n \times \mathrm{Aut}(\mathrm{Coup}(E))$ are such that $\theta((a_1, b_1)) = \theta((a_2, b_2))$. Then, $a_1 \tau b_1^{-1} = a_2 \tau b_2^{-1}$ for all $\tau \in \mathbb{S}_n$. In particular, this must hold for $\tau = \mathrm{id}$, from which it follows that $a_1 b_1^{-1} = a_2 b_2^{-1}$, and hence, $a_2 = a_1 b_1^{-1} b_2$. Substituting this into $a_1 \tau b_1^{-1} = a_2 \tau b_2^{-1}$, yields

$$a_1 \tau b_1^{-1} = a_1 b_1^{-1} b_2 \tau b_2^{-1} \quad \forall \tau \in \mathbb{S}_n, \qquad \text{or equivalently,} \quad \tau b_1^{-1} b_2 = b_1^{-1} b_2 \tau \quad \forall \tau \in \mathbb{S}_n.$$

This implies that $b_1^{-1} b_2 \in Z(\mathbb{S}_n) := \{g \in \mathbb{S}_n \,:\, gh = hg \quad \forall h \in \mathbb{S}_n\}$. It is well-known that the center $Z(\mathbb{S}_n)$ is trivial for $n \geq 3$, hence $b_1 = b_2$. From this, it simply follows that also $a_1 = a_2$, hence θ is injective. □

Theorem 3.1 shows that the image of $\mathbb{S}_n \times \mathrm{Aut}(\mathrm{Coup}(E))$ under θ is a subgroup of $\mathrm{Aut}(\mathrm{Cay}(\mathbb{S}_n, T))$, which is isomorphic to $\mathbb{S}_n \times \mathrm{Aut}(\mathrm{Coup}(E))$ by the injectivity of θ.

The map θ is in general not a bijection, which means that $\mathbb{S}_n \times \mathrm{Aut}(\mathrm{Coup}(E))$ is not the full automorphism group of $\mathrm{Cay}(\mathbb{S}_n, T)$. However, in many of the cases that are interesting for our application, the subgroup turns out to be the full automorphism group. We now present a series of sufficient conditions for this to be true.

We call the Cayley graph $\mathrm{Cay}(\mathbb{S}_n, T)$ normal if the subgroup of all automorphisms by left multiplication by elements of $\mathbb{S}_n$, i.e., $\{(\tau \mapsto a\tau) \,:\, a \in \mathbb{S}_n\}$, is a normal subgroup of $\mathrm{Aut}(\mathrm{Cay}(\mathbb{S}_n, T))$.

Theorem 3.2 ([24])**.** *The graph $\mathrm{Cay}(\mathbb{S}_n, T)$ is normal if and only if $\mathrm{Aut}(\mathrm{Cay}(\mathbb{S}_n, T)) \cong \mathbb{S}_n \times \mathrm{Aut}(\mathrm{Coup}(E))$.*

The following theorem states some known sufficient conditions for $\mathrm{Cay}(\mathbb{S}_n, T)$ to be (non)normal. Recall that the girth of a graph is the length of its shortest cycle. Trees have infinite girth.

Theorem 3.3 ([22, 23])**.** *The graph $\mathrm{Cay}(\mathbb{S}_n, T)$ is normal if $\mathrm{Coup}(E)$ is a graph with girth at least 5. The graph $\mathrm{Cay}(\mathbb{S}_n, T)$ is nonnormal if $\mathrm{Coup}(E)$ is the 4-cycle C_4 or the complete graph K_n.*

Proof. The first part of the statement was proven by Ganesan [22], although normality in case $\mathrm{Coup}(E)$ is a tree was first shown by Feng [17]. The nonnormality results implied by $\mathrm{Coup}(E)$ being C_4 or K_n are obtained by Ganesan [22] and Ganesan [23], respectively. □

In [25] it is conjectured that the two latter cases from Theorem 3.3 are the only connected coupling graphs for which its corresponding Cayley graph $\mathrm{Cay}(\mathbb{S}_n, T)$ is nonnormal. If this conjecture is true, it follows from Theorem 3.2 that $\mathbb{S}_n \times \mathrm{Aut}(\mathrm{Coup}(E))$ is the full automorphism group for almost all quantum architectures. In case $\mathrm{Coup}(E)$ is C_4 or K_n, the automorphism group of $\mathrm{Cay}(\mathbb{S}_n, T)$ is known, see [22, Section 3] and [23, Theorem 1.1], respectively.

3.2 Automorphism group of X

Now that we established either the full automorphism group of $\mathrm{Cay}(\mathbb{S}_n, T)$ or a subgroup of it, we focus on the automorphism group of the entire graph X. Indeed, we need to take the arc structure in-between the subgraphs H^k into account. We start by showing how these arcs restrict the automorphism group of a single subgraph, after which we combine these results to obtain $\mathrm{Aut}(X)$.

Each H^k corresponds to a gate g^k acting on two qubits in Q. The set of outgoing arcs D^k consists of arcs leaving qubit orders τ where $\tau^{-1}(g^k) \in E$, see (2). Since this arc structure needs to be preserved, the automorphisms of interest must setwise fix the qubit orders with this property. For all $k \in [m]$, let

$$F^k := \{\tau \in \mathbb{S}_n \,:\, \tau^{-1}(g^k) \in E\}. \tag{4}$$

Instead of the automorphism group of $\mathrm{Cay}(\mathbb{S}_n, T)$, we are only interested in its subgroup that setwise fixes F^k. That is,

$$\mathrm{Aut}(\mathrm{Cay}(\mathbb{S}_n, T), F^k) := \left\{\rho \in \mathrm{Aut}(\mathrm{Cay}(\mathbb{S}_n, T)) \,:\, \rho(F^k) = F^k\right\}.$$

For each $S \subseteq [n]$, let $\mathbb{S}_n(S) = \{\tau \in \mathbb{S}_n \,:\, \tau(S) = S\}$, which is clearly a subgroup of $\mathbb{S}_n$. Now, if $\mathrm{Coup}(E) = K_n$, it follows that $F^k = \mathbb{S}_n$ and $\mathrm{Aut}(\mathrm{Cay}(\mathbb{S}_n, T), F^k) = \mathrm{Aut}(\mathrm{Cay}(\mathbb{S}_n, T))$. The following results establish a characterization of $\mathrm{Aut}(\mathrm{Cay}(\mathbb{S}_n, T), F^k)$ when $\mathrm{Coup}(E) \neq K_n$.

Theorem 3.4. *Let* $\mathrm{Coup}(E)$ *be connected.* $\mathrm{Aut}(\mathrm{Cay}(\mathbb{S}_n, T), F^k)$ *has a subgroup that is isomorphic to* $\mathbb{S}_n(g^k) \times \mathrm{Aut}(\mathrm{Coup}(E))$. *If* $\mathrm{Cay}(\mathbb{S}_n, T)$ *is normal, then this subgroup equals* $\mathrm{Aut}(\mathrm{Cay}(\mathbb{S}_n, T), F^k)$.

Proof. Let θ be the group homomorphism defined in (3). We now consider its restriction to the subgroup $\mathbb{S}_n(g^k) \times \mathrm{Aut}(\mathrm{Coup}(E))$, which we denote by θ_r. Then its image $\theta_r(\mathbb{S}_n(g^k) \times \mathrm{Aut}(\mathrm{Coup}(E)))$ is clearly a subgroup of $\mathrm{Aut}(\mathrm{Cay}(\mathbb{S}_n, T))$. Since θ is injective by Theorem 3.1, so is θ_r, and thus $\theta_r(\mathbb{S}_n(g^k) \times \mathrm{Aut}(\mathrm{Coup}(E)))$ is isomorphic to $\mathbb{S}_n(g^k) \times \mathrm{Aut}(\mathrm{Coup}(E))$.

We now prove that the set $\theta_r(\mathbb{S}_n(g^k) \times \mathrm{Aut}(\mathrm{Coup}(E)))$ is a subgroup of $\mathrm{Aut}(\mathrm{Cay}(\mathbb{S}_n, T), F^k)$. Let $a \in \mathbb{S}_n(g^k)$ and $b \in \mathrm{Aut}(\mathrm{Coup}(E))$. Then $\theta_r(a, b)$ is the mapping $\tau \mapsto a\tau b^{-1}$. Now, let $\tau \in F^k$, i.e., $\tau^{-1}(g^k) \in E$. Using the fact that $a(g^k) = g^k$ and b maps pairs in E to pairs in E, we obtain

$$(a\tau b^{-1})^{-1}(g^k) = (b\tau^{-1}a^{-1})(g^k) \in E,$$

which implies $a\tau b^{-1} \in F^k$. So, $\theta_r(a, b) \in \mathrm{Aut}(\mathrm{Cay}(\mathbb{S}_n, T), F^k)$, from where it follows that $\theta_r(\mathbb{S}_n(g^k) \times \mathrm{Aut}(\mathrm{Coup}(E)))$ is a subgroup of $\mathrm{Aut}(\mathrm{Cay}(\mathbb{S}_n, T), F^k)$.

Next, we show that if $\mathrm{Cay}(\mathbb{S}_n, T)$ is normal, then it is actually the full automorphism group. It suffices to show that any element in $\mathrm{Aut}(\mathrm{Cay}(\mathbb{S}_n, T), F^k)$ is of the form $\theta_r(a, b)$ for some $a \in \mathbb{S}_n(g^k)$ and $b \in \mathrm{Aut}(\mathrm{Coup}(E))$. Let $\rho \in \mathrm{Aut}(\mathrm{Cay}(\mathbb{S}_n, T), F^k)$. By Theorem 3.1, we know that $\rho : \tau \mapsto a\tau b^{-1}$ for some $a \in \mathbb{S}_n, b \in \mathrm{Aut}(\mathrm{Coup}(E))$. Suppose $a \notin \mathbb{S}_n(g^k)$. Let g^k be the pair $\{i, j\}$. Then there exist k_1, k_2 such that $a(k_1) = i$ and $a(k_2) = j$, with $\{k_1, k_2\} \neq \{i, j\}$. Now, we select two pairs of vertices $e \in E$ and $f \notin E$ as follows. If $|\{k_1, k_2, i, j\}| = 3$, take e and f such that they share one vertex, otherwise take e and f disjoint. The only cases in which such selection is not possible, is when the subgraph induced by any three distinct vertices is a clique or for each edge in E the graph resulting from deleting the edge is a clique. The only connected coupling graphs that satisfy either of these properties are C_4 and K_n. However, by Theorem 3.3, $\mathrm{Coup}(E)$ cannot be these graphs due to the normality of the Cayley graph.

Now, take any $\hat{\tau} \in \mathbb{S}_n$ such that

$$\hat{\tau}(e) = \{i, j\} \quad \text{and} \quad \hat{\tau}(f) = \{k_1, k_2\}.$$

As $\hat{\tau}^{-1}(\{i, j\}) = e \in E$, it follows that $\hat{\tau} \in F^k$. However,

$$\rho(\hat{\tau})^{-1}(\{i, j\}) = (a\hat{\tau}b^{-1})^{-1}(\{i, j\}) = b\hat{\tau}^{-1}a^{-1}(\{i, j\}) = b\hat{\tau}^{-1}(\{k_1, k_2\}) = b(f) \notin E,$$

since b maps non-edges to non-edges in $\mathrm{Coup}(E)$. We conclude that $\rho(\hat{\tau}) \notin F^k$, which implies that $\rho \notin \mathrm{Aut}(\mathrm{Cay}(\mathbb{S}_n, T), F^k)$. Since this is a contradiction, each automorphism in $\mathrm{Aut}(\mathrm{Cay}(\mathbb{S}_n, T), F^k)$ is in $\theta_r(\mathbb{S}_n(g^k) \times \mathrm{Aut}(\mathrm{Coup}(E)))$. □

Let G^k_{sub} denote the subgroup of $\text{Aut}(\text{Cay}(\mathbb{S}_n, T), F^k)$ that is isomorphic to $\mathbb{S}_n(g^k) \times \text{Aut}(\text{Coup}(E))$. Suppose X consists of one subgraph. Then, X has vertex set $\{s\} \cup V^1 \cup \{t\}$. One can verify that in that case $\text{id}_{\{s\}} \times G^1_{\text{sub}} \times \text{id}_{\{t\}}$ is a subgroup of $\text{Aut}(X)$, which is the entire automorphism group in case $\text{Cay}(\mathbb{S}_n, T)$ is normal. Now, suppose X has two subgraphs. Then, H^1 corresponds to gate g^1 and H^2 corresponds to a possibly different gate g^2. In the sequel, we study how this affects the automorphism group of X.

To that end, we need two intermediate results. For a set $S \subseteq [n]$, let $C(\mathbb{S}_n(S))$ denote the centralizer subgroup of $\mathbb{S}_n(S)$ which is defined as

$$C(\mathbb{S}_n(S)) = \{\tau \in \mathbb{S}_n \,:\, \tau\pi = \pi\tau \text{ for all } \pi \in \mathbb{S}_n(S)\}. \tag{5}$$

When $n \leq 2$, we know that $\mathbb{S}_n$ is abelian and thus $C(\mathbb{S}_n(S)) = \mathbb{S}_n$. Otherwise, we show that the centralizer subgroup is contained in $\mathbb{S}_n(S)$.

Lemma 3.5. *Let $n \geq 3$. Then, we have $C(\mathbb{S}_n(S)) \subseteq \mathbb{S}_n(S)$ for all $S \subseteq [n]$.*

Proof. Since $\mathbb{S}_n(S) = \mathbb{S}_n([n]\setminus S)$, we may assume that $|S| \geq 2$. Now, let $\tau \in C(\mathbb{S}_n(S))$ and assume for the sake of contradiction that $\tau \notin \mathbb{S}_n(S)$. Then there exist distinct $i, j \in S$ such that $\tau(i) \notin S$. Now, consider the transposition $(i\ j)$. We have $(i\ j)\tau(i) = \tau(i)$, while $\tau(i\ j)(i) = \tau(j)$. Hence, τ and $(i\ j)$ do not commute, while $(i\ j) \in \mathbb{S}_n(S)$. Therefore, $\tau \notin C(\mathbb{S}_n(S))$, which is a contradiction. □

Exploiting Lemma 3.5, we can show the following result for general sets F of the form (4).

Theorem 3.6. *Let $i, j \in [n]$, $n \geq 3$, and let $F = \{\tau \in \mathbb{S}_n \,:\, \{\tau^{-1}(i), \tau^{-1}(j)\} \in E\}$. Let $a, b \in \mathbb{S}_n$ and suppose that $a\tau b^{-1} = \tau$ for all $\tau \in F$. Then $a = b = \text{id}$.*

Proof. Observe that for all $\tau_1, \tau_2 \in F$ we have:

$$\tau_1 b \tau_1^{-1} = a = \tau_2 b \tau_2^{-1}.$$

Now, let us fix an edge $e \in E$. We can write any element $\pi \in \mathbb{S}_n(e)$ in the form $\pi = \tau^{-1}\tau'$ for some $\tau, \tau' \in F$. To verify this, observe that since $e \in E$ there exist elements in F that map e to $\{i, j\}$. By combining two such elements τ and τ', the composition $\tau^{-1}\tau'$ always maps e back to e. On the complement $[n] \setminus e$ we find all possible permutations in F, so we can always find $\tau, \tau' \in F$ such that $\tau^{-1}\tau'$ acts like π on the set $[n] \setminus e$.

Let $\tau_1, \tau_2 \in F$ be such that $\pi = \tau_1^{-1}\tau_2$. Then we know $\tau_1 b \tau_1^{-1} = \tau_2 b \tau_2^{-1}$, which can be rewritten as $\pi^{-1} b \pi = b$. As $\pi \in \mathbb{S}_n(e)$ was chosen arbitrarily, it follows that $\pi^{-1} b \pi = b$ for all $\pi \in \mathbb{S}_n(e)$, and thus $b \in C(\mathbb{S}_n(e))$. We now apply Lemma 3.5 with $S = e$. Since $n \geq 3$, it follows that $b \in \mathbb{S}_n(e)$.

By repeating this argument for all $e \in E$, it follows that $b \in \bigcap_{e \in E} \mathbb{S}_n(e)$. As $\text{Coup}(E)$ is connected, we conclude that $b = \text{id}$, from which it immediately follows that $a = \text{id}$ as well. □

Let $\rho \in \text{Aut}(X)$ where X consists of two subgraphs. The case $n \leq 2$ leads to a trivial NNCP instance. Therefore, we may assume that $n \geq 3$. Then the restriction of ρ to H^1 is an element of G^1_{sub}. In particular, each $\tau^1 \in F^1$ is mapped to $\rho(\tau^1)$ (here the superscript 1 is added to indicate that τ^1 is a vertex of H^1). In order to maintain the arc structure of D^1, it follows that the restriction of ρ to H^2 should not only be an element of G^2_{sub}, it should also pointwise fix the elements $\rho(\tau^2)$ for all $\tau^2 \in F^1$. Applying the result of Theorem 3.6, the restriction of ρ to H^2 should be the same automorphism as the restriction to H^1. On top of that, this restriction must also be in G^2_{sub}. Thus, ρ is of the form $(\text{id}_{\{s\}}, \pi, \pi, \text{id}_{\{t\}})$ with $\pi \in G^1_{\text{sub}} \cap G^2_{\text{sub}}$. Extending this argument to larger k, let us define the following groups:

$$G_{\text{sub}} := \bigcap_{k=1}^{m} G^k_{\text{sub}} \cong \bigcap_{k=1}^{m} \mathbb{S}_n(g^k) \times \text{Aut}(\text{Coup}(E)), \tag{6}$$

$$G_X := \left\{ (\text{id}_{\{s\}}, \rho, \ldots, \rho, \text{id}_{\{t\}}) \in \text{id}_{\{s\}} \times \prod_{k=1}^{m} \text{Aut}(H^k) \times \text{id}_{\{t\}} \,:\, \rho \in G_{\text{sub}} \right\}. \tag{7}$$

By construction, G_X is a subgroup of $\mathrm{Aut}(X)$. If follows from the results above that it is the full automorphism group whenever $\mathrm{Cay}(\mathbb{S}_n, T)$ is normal.

To get rid of of the intersection in the definition of G_{sub}, we exploit the notion of the gate graph (Q, U) of a quantum circuit Γ, see Definition 2.3. If g^{k_1} is in C with $g^{k_1} = \{i, j\}$, this implies that the set $\{i, j\}$ must be setwise fixed by all permutations in the group $\mathbb{S}_n(g^{k_1})$. If also $g^{k_2} \in C$ with $g^{k_2} = \{j, \ell\}$, there is no other option than fixing i, j and ℓ elementwise in the group intersection $\mathbb{S}_n(g^{k_1}) \cap \mathbb{S}_n(g^{k_2})$. From this observation, we can partition all qubits in Q based on whether they belong to a connected component of size one, two or at least three in the gate graph (Q, U). This leads to the introduction of the fixing pattern of Γ.

Definition 3.7. *Let $\Gamma = (Q, C)$ be a quantum circuit on n qubits. We define the fixing pattern of Γ as the partition $\mathcal{F} := \{S_1, \ldots, S_l\}$ of Q such that each S_i is either:*

- *a single qubit contained in a connected component of the gate graph (Q, U) of size at least 3;*
- *a pair of qubits $\{i, j\}$ that forms a connected component in the gate graph (Q, U);*
- *the set of all singletons in the gate graph (Q, U), which we denote by the free set in $\mathcal{F}$.*

Moreover, we define f as the size of the free set, p as the number of pairs and c $(= n - 2p - f)$ to be the number of qubits in a connected component of size at least 3 in (Q, U).

Observe that $\mathcal{F}$ can be easily constructed by a scan of the connected components of (Q, U). The extreme cases are $\mathcal{F} = \{Q\}$ if Γ contains no gates, whereas $\mathcal{F} = \{\{1\}, \ldots, \{n\}\}$ if (Q, U) is connected. The group $\cap_{k=1}^{m} \mathbb{S}_n(g^k)$ consists of all permutations that setwise fix the elements in $\mathcal{F}$. To simplify notation, we define

$$\mathbb{S}_n(\mathcal{F}) := \{a \in \mathbb{S}_n \,:\, a(S_i) = S_i \text{ for all } i \in [l]\}.$$

We know that $G_{\mathrm{sub}} \cong \mathbb{S}_n(\mathcal{F}) \times \mathrm{Aut}(\mathrm{Coup}(E))$, which implies that

$$G_X \cong \mathbb{S}_n(\mathcal{F}) \times \mathrm{Aut}(\mathrm{Coup}(E)). \tag{8}$$

It follows from a simple counting argument that $|\mathbb{S}_n(\mathcal{F})| = 2^p f!$.

3.3 Orbit and orbital structure of group action on X

The elements of G_X act on the vertices and arcs of X. In this section we study this group action in terms of its induced orbit and orbital structure, which will become of key importance in the symmetry reduction explained in Section 4.

Each automorphism in G_X maps the vertex set of X to itself. Given a vertex $\tau \in V$, the orbit of τ is the set of vertices to which τ is mapped to by the elements in G_X, i.e., all vertices $\rho(\tau)$ with $\rho \in G_X$. The set of orbits forms a partition of V, which is written as the quotient V/G_X.

Similarly, G_X acts on the arc set A by $\rho((\tau_1, \tau_2)) = (\rho(\tau_1), \rho(\tau_2))$ for all $\rho \in G_X$. We denote the set of orbitals by A/G_X. Note that arcs in the same orbital have their initial vertices in the same orbit. It is therefore natural to first understand the orbit structure of the action of G_X on V.

Let $\mathrm{Orb}(\tau)$ denote the orbit of vertex $\tau \in V$. It follows from the construction of G_X that $\mathrm{Orb}(s) = \{s\}$ and $\mathrm{Orb}(t) = \{t\}$. Moreover, the subgraphs H^k, $k \in [m]$, are invariant under the action of G_X on X. For that reason, we can restrict ourselves to identifying the orbits within each subgraph H^k under the action of G_{sub}. Since all subgraphs are identical, this provides the orbit description for the entire graph G_X.

Similar as before, we use τ to denote a vertex, as each vertex represents a qubit order in $\mathbb{S}_n$. For all $k \in [m]$ and all $\tau \in V^k$, we obtain

$$\mathrm{Orb}(\tau) = \{\rho(\tau) \,:\, \rho \in G_{\mathrm{sub}}\} = \left\{a\tau b^{-1} \,:\, a \in \mathbb{S}_n(\mathcal{F}),\ b \in \mathrm{Aut}(\mathrm{Coup}(E))\right\}. \tag{9}$$

We also define the stabilizer subgroup with respect to τ under the action of G_{sub} as

$$\begin{aligned}\text{Stab}(\tau) &:= \{\rho \in G_{\text{sub}} \,:\, \rho(\tau) = \tau\} \\ &\cong \left\{(a,b) \in \mathbb{S}_n(\mathcal{F}) \times \text{Aut}(\text{Coup}(E)) \,:\, a\tau b^{-1} = \tau\right\}.\end{aligned} \tag{10}$$

The condition given in (10) for (a,b) to act as a stabilizer can be rewritten as $a = \tau b \tau^{-1}$. Thus, a pair $(a,b) \in \mathbb{S}_n(\mathcal{F}) \times \text{Aut}(\text{Coup}(E))$ corresponds to an element in $\text{Stab}(\tau)$ if and only if the permutation $\tau b \tau^{-1}$ is in $\mathbb{S}_n(\mathcal{F})$ and $a = \tau b \tau^{-1}$. This implies that for all $S_i \in \mathcal{F}$ we must have $\tau b \tau^{-1}(S_i) = S_i$, or equivalently, $b(\tau^{-1}(S_i)) = \tau^{-1}(S_i)$. Hence, b setwise fixes the inverse fixing pattern in $\mathcal{F}$ with respect to τ. Let us define the subgroup B_τ of $\text{Aut}(\text{Coup}(E))$ that consists of all such elements, i.e.,

$$B_\tau := \left\{b \in \text{Aut}(\text{Coup}(E)) \,:\, b\left(\tau^{-1}(S_i)\right) = \tau^{-1}(S_i) \quad \forall i \in [l]\right\}. \tag{11}$$

Since for each $b \in B_\tau$, there exists exactly one element $a \in \mathbb{S}_n(\mathcal{F})$ such that $a\tau b^{-1} = \tau$, we know

$$\text{Stab}(\tau) \cong \{(a,b) \,:\, b \in B_\tau,\, a = \tau b \tau^{-1}\}, \tag{12}$$

in particular, we have $|\text{Stab}(\tau)| = |B_\tau|$.

As B_τ is a subgroup of $\text{Aut}(\text{Coup}(E))$, it acts on the edge set of $\text{Coup}(E)$. The orbital of an edge $\{i,j\} \in E$ under this group action is the set of all edges $\{b(i), b(j)\}$ with $b \in B_\tau$. We denote by the quotient E/B_τ the set of orbitals under this group action.

We can show that if τ_1 and τ_2 belong to the same orbit, then the subgroups B_{τ_1} and B_{τ_2} are conjugate subgroups. Moreover, the quotients of their actions on E have the same cardinality.

Lemma 3.8. *Let τ_1 and τ_2 be two qubit orders with $\tau_2 = a\tau_1 b^{-1}$ for some $a \in \mathbb{S}_n(\mathcal{F})$ and $b \in \text{Aut}(\text{Coup}(E))$. Then,*

(i) $B_{\tau_2} = bB_{\tau_1}b^{-1}$;

(ii) there exists a bijection from E/B_{τ_1} to E/B_{τ_2} given by left multiplication with b.

Proof. (i) Exploiting the fact that $a^{-1}(S_i) = S_i$ for all $i \in [l]$, we obtain

$$\begin{aligned}B_{\tau_2} &= \left\{b_2 \in \text{Aut}(\text{Coup}(E)) \,:\, b_2\left(\tau_2^{-1}(S_i)\right) = \tau_2^{-1}(S_i) \quad \forall i \in [l]\right\} \\ &= \left\{b_2 \in \text{Aut}(\text{Coup}(E)) \,:\, b_2\left((a\tau_1 b^{-1})^{-1}(S_i)\right) = (a\tau_1 b^{-1})^{-1}(S_i) \quad \forall i \in [l]\right\} \\ &= \left\{b_2 \in \text{Aut}(\text{Coup}(E)) \,:\, b_2 b\tau_1^{-1}a^{-1}(S_i) = b\tau_1^{-1}a^{-1}(S_i) \quad \forall i \in [l]\right\} \\ &= \left\{b_2 \in \text{Aut}(\text{Coup}(E)) \,:\, b^{-1}b_2 b\left(\tau_1^{-1}(S_i)\right) = \tau_1^{-1}(S_i) \quad \forall i \in [l]\right\} \\ &= \left\{bb_1b^{-1} \in \text{Aut}(\text{Coup}(E)) \,:\, b_1\left(\tau_1^{-1}(S_i)\right) = \tau_1^{-1}(S_i) \quad \forall i \in [l]\right\} \\ &= bB_{\tau_1}b^{-1}.\end{aligned}$$

(ii) This fact follows directly from (i), by observing that

$$b\ \text{Orb}_{B_{\tau_1}}(i) = \left\{bb_1b^{-1}(b(i)) \,:\, b_1 \in B_{\tau_1}\right\} = \{b_2(b(i)) \,:\, b_2 \in B_{\tau_2}\} = \text{Orb}_{B_{\tau_2}}(b(i)).$$

One easily verifies that left multiplication by b gives a bijection.

□

As a consequence of the well-known orbit-stabilizer theorem, we establish the following relation between $\text{Orb}(\tau)$ and $\text{Stab}(\tau)$:

$$|\text{Orb}(\tau)| = \frac{|G_{\text{sub}}|}{|\text{Stab}(\tau)|} = \frac{2^p f! \cdot |\text{Aut}(\text{Coup}(E))|}{|B_\tau|}. \tag{13}$$

Of course, $\text{Orb}(\tau)$ does not depend on the particular choice of the representative τ in the orbit.

To increase our understanding of $\mathrm{Orb}(\tau)$, we rewrite (9) as follows:

$$\mathrm{Orb}(\tau) = \mathbb{S}_n(\mathcal{F})\tau\mathrm{Aut}(\mathrm{Coup}(E)) = \bigcup_{\tilde{\tau}\in\mathbb{S}_n(\mathcal{F})\tau} \tilde{\tau}\mathrm{Aut}(\mathrm{Coup}(E)). \tag{14}$$

In other words, if $\mathbb{S}_n(\mathcal{F})$ is trivial, then the orbit partition of V^k is given by the left cosets of $\mathrm{Aut}(\mathrm{Coup}(E))$ in G_{sub}. Otherwise, each orbit is the union of several left cosets of $\mathrm{Aut}(\mathrm{Coup}(E))$ in G_{sub}, where the union is determined by the elements in the right cosets of $\mathbb{S}_n(\mathcal{F})$ in G_{sub}.

Of particular importance in the symmetry reduction is the number of orbits in each subgraph. We let V^k/G_X denote the set of orbits of vertices in V^k under the action of G_X, although we formally refer to the action of G_X restricted to V^k. We allow for this slight abuse of notation, in order to simplify the terminology in Section 4.

Theorem 3.9. *The number of orbits of V^k under G_X is $|V^k/G_X| = \dfrac{\sum_{\tau\in\mathbb{S}_n}|B_\tau|}{2^p f!\cdot|\mathrm{Aut}(\mathrm{Coup}(E))|}$.*

Proof. Let $(V^k)^\rho$ denote the set of vertices in V^k that are (pointwise) fixed by $\rho \in G_X$. Then, Burnside's lemma implies that $|V^k/G_X| = \frac{\sum_{\rho\in G_X}|(V^k)^\rho|}{|G_X|}$. The sum in the numerator counts for every group element the number of vertices that are fixed. Alternatively, we can also sum over all vertices and count the number of group elements that stabilize the vertex. This leads to

$$|V^k/G_X| = \frac{\sum_{\tau\in\mathbb{S}_n}|\mathrm{Stab}(\tau)|}{|\mathbb{S}_n(\mathcal{F})\times\mathrm{Aut}(\mathrm{Coup}(E))|} = \frac{\sum_{\tau\in\mathbb{S}_n}|B_\tau|}{2^p f!\cdot|\mathrm{Aut}(\mathrm{Coup}(E))|}.$$

□

We now shift our focus to the analysis of the orbital structure of the arcs of X under the action of G_X. Recall that A consists of two types of arcs: arcs within a subgraph (the sets A^k, $k\in[m]$) and the arcs between the subgraphs (the sets $D^k, k\in\{0\}\cup[m]$). Since the sets $A^1,\dots,A^k$ are identical and each set is invariant under the group action G_X, we can restrict our focus to the action of G_{sub} on a single subgraph. The orbital of an arc $(\tau,\tau\sigma)\in A^k$ corresponding to transposition $\sigma=(i\ j)\in T$ is given by

$$\begin{aligned}\mathrm{Orb}((\tau,\tau\sigma)) &:= \{(\rho(\tau),\rho(\tau\sigma)) \,:\, \rho\in G_{\mathrm{sub}}\}\\ &= \left\{(a\tau b^{-1}, a\tau\sigma b^{-1} \,:\, a\in\mathbb{S}_n(\mathcal{F}),\ b\in\mathrm{Aut}(\mathrm{Coup}(E))\right\}\\ &= \left\{(a\tau b^{-1}, a\tau b^{-1}(b(i)\ b(j)) \,:\, a\in\mathbb{S}_n(\mathcal{F}),\ b\in\mathrm{Aut}(\mathrm{Coup}(E))\right\},\end{aligned}$$

where the last line follows from the fact that $b(i\ j)b^{-1} = (b(i)\ b(j))$. This expression of $\mathrm{Orb}((\tau,\tau\sigma))$ implies that all arcs within the same orbital start at vertices within the same orbit and end at vertices within the same orbit (where the start- and end-orbits can differ). Moreover, the transpositions to which the arcs in $\mathrm{Orb}((\tau,\tau\sigma))$ correspond are related via $\mathrm{Aut}(\mathrm{Coup}(E))$, as the following lemma illustrates.

Lemma 3.10. *Let $\tau\in\mathbb{S}_n$. There exists a bijection between the orbitals starting from $\mathrm{Orb}(\tau)$ and the orbitals in E/B_τ.*

Proof. It suffices to consider the orbital partition of the arcs leaving τ, i.e., $\delta^+(\tau,A^k)$. If B_τ is trivial, the stabilizer subgroup of τ in G_{sub} is trivial, implying that no two arcs in $\delta^+(\tau,A^k)$ belong to the same orbital. In that case, E/B_τ is just the partition of E under the identity map. If B_τ is nontrivial and $b\in B_\tau$ maps the edge corresponding to σ_1 to a different edge corresponding to σ_2, then the distinct arcs $(\tau,\tau\sigma_1)$ and $(\tau,\tau\sigma_2)$ belong to the same orbital under G_{sub}. If $b\in B_\tau$ maps the edge corresponding to σ_1 to itself, then the orbital containing $(\tau,\tau\sigma_1)$ has a smaller cardinality. These three cases are depicted in Figure 1. We conclude that the arcs $(\tau,\tau\sigma_1)$ and $(\tau,\tau\sigma_2)$ belong to the same orbital if and only if the edges corresponding to σ_1 and σ_2 in $\mathrm{Coup}(E)$ belong to the same orbital in E/B_τ. □

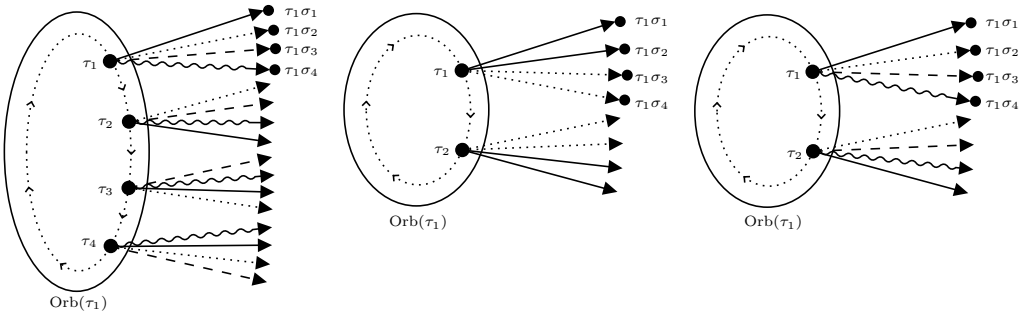


Figure 1: Graphical overview of orbital structure within a subgraph H^k. Each line type (solid, dotted, dashed and curled) corresponds to another orbital. Case I (left): B_{τ_1} is trivial. Case II (middle): B_{τ_1} is nontrivial and the orbital of σ_1 under B_{τ_1} contains σ_2. Case III (right): B_{τ_1} is nontrivial, but the orbital of σ_1 under B_{τ_1} only consists of σ_1.

The following result regards the cardinality of the set of orbitals of A^k under the action of G_X restricted to A^k. By slight abuse of notation, we again denote this set by the quotient A^k/G_X.

Theorem 3.11. *The number of orbitals of A^k under G_X is $|A^k/G_X| = \frac{\sum_{\tau\in\mathbb{S}_n} |B_\tau|\cdot|E/B_\tau|}{2^p f!\cdot|\mathrm{Aut}(\mathrm{Coup}(E))|}$.*

Proof. Since the arcs belonging to an orbital all start from vertices in the same orbit, it suffices to enumerate over all orbits and count the number of orbitals starting from that orbit. It follows from Lemma 3.10 that the number of distinct orbitals starting from $\mathrm{Orb}(\tau)$ is $|E/B_\tau|$, where the choice of τ to represent $\mathrm{Orb}(\tau)$ does not affect this quantity, see Lemma 3.8. We now sum over all $\mathrm{Orb}(\tau) \in V^k/G_X$:

$$\begin{aligned}
|A^k/G_X| &= \sum_{\mathrm{Orb}(\tau)\in V^k/G_X} |E/B_\tau| \\
&= \sum_{\mathrm{Orb}(\tau)\in V^k/G_X} \frac{2^p f!\cdot|\mathrm{Aut}(\mathrm{Coup}(E))|}{|B_\tau|}\cdot\frac{|B_\tau|\cdot|E/B_\tau|}{2^p f!\cdot|\mathrm{Aut}(\mathrm{Coup}(E))|} \\
&= \sum_{\mathrm{Orb}(\tau)\in V^k/G_X} |\mathrm{Orb}(\tau)|\cdot\frac{|B_\tau|\cdot|E/B_\tau|}{2^p f!\cdot|\mathrm{Aut}(\mathrm{Coup}(E))|} \\
&= \frac{\sum_{\tau\in\mathbb{S}_n}|B_\tau|\cdot|E/B_\tau|}{2^p f!\cdot|\mathrm{Aut}(\mathrm{Coup}(E))|}.
\end{aligned}$$

In the third equality we used (13), as well as the fact that the sum of $|\mathrm{Orb}(\tau)|\cdot|B_\tau|\cdot|E/B_\tau|$ over all orbits equals the sum of $|B_\tau|\cdot|E/B_\tau|$ over all vertices, since $|B_\tau|\cdot|E/B_\tau|$ is constant for all τ within an orbit, see Lemma 3.8. □

To study the orbital representation of D^k under the action of G_X, we distinguish between the case $k = 0$ and $k \in [m]$. For $k = 0$, D^k contains all arcs between s and V^1. Therefore, each orbital of D^0 under G_X consists of all arcs starting from s and ending at vertices in an orbit of V^1. The arcs in D^k, $k \in [m]$, correspond to ordered pairs (τ^k, τ^{k+1}), where τ represents the same qubit order in H^k and H^{k+1}. Such an arc exists in D^k whenever $\tau^k \in F^k$, see (4). The orbital of (τ^k, τ^{k+1}) is the set

$$\begin{aligned}
\mathrm{Orb}((\tau^k,\tau^{k+1})) &= \{(\rho(\tau^k),\rho(\tau^{k+1}) \;:\; \rho\in G_{\mathrm{sub}}\} \\
&= \{(a\tau^k b^{-1}, a\tau^{k+1}b^{-1}) \;:\; a\in\mathbb{S}_n(\mathcal{F}), b\in\mathrm{Aut}(\mathrm{Coup}(E))\}.
\end{aligned}$$

Let D^k/G_X denote the set of orbitals of the group action of G_X restricted to D^k. Since τ^k and τ^{k+1} represent the same qubit orders in H^k and H^{k+1}, respectively, all arcs within $\mathrm{Orb}((\tau^k, \tau^{k+1}))$ start and end at vertices in the same orbit. This leads to the following result.

Theorem 3.12. *The number of orbitals of D^0 under G_X is $|D^0/G_X| = \frac{\sum_{\tau\in\mathbb{S}_n}|B_\tau|}{2^p f!\cdot|\mathrm{Aut}(\mathrm{Coup}(E))|}$. For $k\neq 0$, the number of orbitals of D^k under G_X is $|D^k/G_X| = \frac{\sum_{\tau\in F^k}|B_\tau|}{2^p f!\cdot|\mathrm{Aut}(\mathrm{Coup}(E))|}$.*

Proof. The first part follows directly from Theorem 3.9. For the second part, observe that we have $D^k = \{(\tau^k,\tau^{k+1}) : \tau^k \in V^k,\ \tau^{k+1}\in V^{k+1},\ \tau^k\in F^k\}$, where F^k is defined in (4). The cardinality of D^k/G_X is equal to the number of orbits of F^k under the action of G_X restricted to the vertices in F^k. The cardinality of F^k/G_X can be derived similarly as in the proof of Theorem 3.9, leading to

$$|D^k/G_X| = |F^k/G_X| = \frac{\sum_{\tau\in F^k}|B_\tau|}{2^p f!\cdot|\mathrm{Aut}(\mathrm{Coup}(E))|}.$$

□

The results of Theorems 3.9, 3.11 and 3.12 are summarized in Table 1. Moreover, we simplify the cardinalities of the quotients for the special case where B_τ is trivial for all $\tau\in\mathbb{S}_n$.

Quotient	**Order**	**Order when B_τ is trivial for all $\tau\in\mathbb{S}_n$**
V^k/G_X, $k\in[m]$	$\frac{\sum_{\tau\in\mathbb{S}_n}\lvert B_\tau\rvert}{2^p f!\cdot\lvert\mathrm{Aut}(\mathrm{Coup}(E))\rvert}$	$\frac{n!}{2^p f!\cdot\lvert\mathrm{Aut}(\mathrm{Coup}(E))\rvert}$
A^k/G_X, $k\in[m]$	$\frac{\sum_{\tau\in\mathbb{S}_n}\lvert B_\tau\rvert\cdot\lvert E/B_\tau\rvert}{2^p f!\cdot\lvert\mathrm{Aut}(\mathrm{Coup}(E))\rvert}$	$\frac{n!\cdot\lvert E\rvert}{2^p f!\cdot\lvert\mathrm{Aut}(\mathrm{Coup}(E))\rvert}$
D^0/G_X	$\frac{\sum_{\tau\in\mathbb{S}_n}\lvert B_\tau\rvert}{2^p f!\cdot\lvert\mathrm{Aut}(\mathrm{Coup}(E))\rvert}$	$\frac{n!}{2^p f!\cdot\lvert\mathrm{Aut}(\mathrm{Coup}(E))\rvert}$
D^k/G_X, $k\in[m]$	$\frac{\sum_{\tau\in F^k}\lvert B_\tau\rvert}{2^p f!\cdot\lvert\mathrm{Aut}(\mathrm{Coup}(E))\rvert}$	$\frac{2\lvert E\rvert(n-2)!}{2^p f!\cdot\lvert\mathrm{Aut}(\mathrm{Coup}(E))\rvert}$

Table 1: Overview of the orders of quotients V^k/G_X, A^k/G_X and D^k/G_X in terms of the cardinality of B_τ.

4 Symmetry reduction for the NNCP

In this section we show how the automorphism results derived in Section 3 can be exploited to reduce the size of the NNCP introduced in Section 2.2.

In Section 4.1 we exploit the subgroup G_X, see (8), in order to reduce the linear programming formulation (SPP) in terms of the number of variables and constraints. In Section 4.2 we show how this reduced LP can be rewritten as a generalized network flow problem. The backward reconstruction of optimal qubit orders from the reduced model is the topic of Section 4.3.

4.1 Reduced LP formulation

The elements in G_X act on the vertex and arc set of G. For any arc $e\in A$ and any $\rho\in G_X$, let $\rho(e)$ denote the ordered pair to which e is mapped to by ρ, which is again in A since ρ is an automorphism. Now, let $x\in\prod_{k=1}^m\mathbb{R}^{A^k}$ and $y\in\prod_{k=0}^m\mathbb{R}^{D^k}$ be feasible for (SPP). We define the Reynolds operator ψ that maps x (resp. y) to the average of the images of x (resp. y) under the action of G_X on A. That is,

$$\psi(x) := \frac{1}{|G_X|}\sum_{\rho\in G_X} x^\rho \quad\text{and}\quad \psi(y) := \frac{1}{|G_X|}\sum_{\rho\in G_X} y^\rho, \tag{15}$$

where x^ρ and y^ρ are defined as $x^\rho_e = x_{\rho(e)}$ and $y^\rho_e = y_{\rho(e)}$ for all arcs e. As A^k for all $k \in [m]$ and D^k for all $k \in \{0\} \cup [m]$ are invariant under the action of G_X on A, it follows that $\psi(x) \in \prod_{k=1}^m \mathbb{R}^{A^k}$ and $\psi(y) \in \prod_{k=0}^m \mathbb{R}^{D^k}$. We can now prove the following result, which was proven for general linear programs by Bödi et al. [8].

Theorem 4.1. *Let* $(x, y) \in \prod_{k=1}^m \mathbb{R}^{A^k} \times \prod_{k=0}^m \mathbb{R}^{D^k}$ *be feasible (resp. optimal) for* (SPP)*. Then,* $(\psi(x), \psi(y))$ *is also feasible (resp. optimal) for* (SPP)*.*

Proof. As the flow conservation constraints hold for (x, y) and ρ preserves the arc structure of X, the pair (x^ρ, y^ρ) also satisfies these constraints for all $\rho \in G_X$. It follows that (x^ρ, y^ρ) is feasible for (SPP) for all $\rho \in G_X$. Observe that the pair $(\psi(x), \psi(y))$ is a convex combination of (x^ρ, y^ρ) over the elements of G_X. Because the feasible set of (SPP) is convex, it follows that $(\psi(x), \psi(y))$ is also feasible for (SPP).

The objective function of (SPP) can be written as $f(x, y) := \sum_{e \in A} x_e$. Since arcs are mapped to arcs by all $\rho \in G_X$, we have $f(x^\rho, y^\rho) = f(x, y)$. We then obtain:

$$f(\psi(x), \psi(y)) = \sum_{e \in A} \psi(x)_e = \frac{1}{|G_X|} \sum_{\rho \in G_X} \sum_{e \in A} x^\rho_e = \frac{1}{|G_X|} |G_X| \sum_{e \in A} x_e = f(x, y).$$

Thus, if (x, y) is optimal for (SPP), then so is $(\psi(x), \psi(y))$. □

An implication of Theorem 4.1 is that we may restrict the feasible set of (SPP) to the subspace

$$\mathcal{H}_{G_X} := \left\{ (\psi(x), \psi(y)) \,:\, (x, y) \in \prod_{k=1}^m \mathbb{R}^{A^k} \times \prod_{k=0}^m \mathbb{R}^{D^k} \right\}, \tag{16}$$

which is also denoted as the fixed point subspace in [8]. By construction of the Reynolds operator (15), the entries in $\psi(x)$ belonging to the same orbital are equal. Therefore, the subspace $\mathcal{H}_{G_X}$ is spanned by the incidence vectors of orbitals of X. In Section 3.3 we derived the orbital structure of the action of G_X on X. Recall that A^k/G_X denotes (the index set of) the collection of orbitals of A^k under the action of G_X. Now, if we denote the ith orbital of A^k by W^k_i, we obtain

$$A^k = \bigsqcup_{i \in A^k/G_X} W^k_i \quad \text{for all } k \in [m]. \tag{17}$$

In a similar fashion, the arc sets $D^k, k \in \{0\} \cup [m]$ can be partitioned into its collection of orbitals. If Z^k_i denotes the ith orbital of D^k, then

$$D^k = \bigsqcup_{i \in D^k/G_X} Z^k_i \quad \text{for all } k \in \{0\} \cup [m]. \tag{18}$$

Now, the subspace $\mathcal{H}_{G_X}$ can be rewritten as:

$$\mathcal{H}_{G_X} = \prod_{k=1}^m \left(\text{Span}\{ \mathbb{1}_{W^k_i} \,:\, i \in A^k/G_X \} \right) \times \prod_{k=0}^m \left(\text{Span}\{ \mathbb{1}_{Z^k_i} \,:\, i \in D^k/G_X \} \right), \tag{19}$$

which implies that the characteristic vectors of the orbitals form a basis for $\mathcal{H}_{G_X}$.

Also the orbits of each of the vertex sets V^k under the action of G_X induce a partition of V^k. Let V^k/G_X denote (the index set of) the collection of orbits of V^k under G_X. The uth orbit of V^k is denoted by O^k_u, with $u \in V^k/G_X$. Then,

$$V^k = \bigsqcup_{u \in V^k/G_X} O^k_u \quad \forall k \in [m]. \tag{20}$$

To write the symmetry-reduced equivalent of (SPP) explicitly, we need some further terminology. Let the out-degree $d^+(\tau, W_i^k)$ (resp. in-degree $d^-(\tau, W_i^k)$) denote the number of arcs in orbital W_i^k that start (resp. end) at vertex τ, i.e.,

$$d^+(\tau, W_i^k) := \left|\left\{(\tau, \tau\sigma) \in W_i^k \,:\, \sigma \in T\right\}\right| \quad \text{and} \quad d^-(\tau, W_i^k) := \left|\left\{(\tau\sigma, \tau) \in W_i^k \,:\, \sigma \in T\right\}\right|,$$

for all $i \in A^k/G_X$ and $k \in [m]$. Since $d^+(\tau_1, W_i^k) = d^+(\tau_2, W_i^k)$ for all orbitals i when τ_1 and τ_2 belong to the same orbit, it makes sense to define $d^+(W_i^k)$ ($:= d^+(\tau, W_i^k)$ for any $(\tau, \tau\sigma) \in W_k^i$) as the orbital out-degree in W_i^k. In a similar fashion we define $d^-(W_i^k)$.

From Lemma 3.10 we know that there is a single case in which $d^+(\tau, W_i^k) > 1$. Namely, two distinct arcs $(\tau, \tau\sigma_1)$ and $(\tau, \tau\sigma_2)$ with $\sigma_1 = (i\ j)$ are both in the same orbital W_i^k if and only if there exists a $b \in B_\tau$ such that $\sigma_2 = (b(i)\ b(j))$. This corresponds to case II in Figure 1. Hence, we have

$$\begin{aligned} d^+(\tau, W_i^k) &= |\{b(\{i,j\}) \,:\, b \in B_\tau\}| \text{ for some } (\tau, \tau(i\ j)) \in W_i^k, \\ d^-(\tau, W_i^k) &= |\{b(\{i,j\}) \,:\, b \in B_\tau\}| \text{ for some } (\tau(i\ j), \tau) \in W_i^k. \end{aligned}$$

Indeed, these equal the number of elements in an orbital of $\mathrm{Coup}(E)$ under the action of B_τ. Moreover, we also define $d^+(Z_i^0)$ (resp. $d^-(Z_i^m)$) as the number of arcs in orbital Z_i^0 (resp. Z_i^m) starting from s (resp. ending at t). For these degrees one can verify that $d^+(Z_i^0) = |Z_i^0|$ and $d^-(Z_i^m) = |Z_i^m|$.

For any vertex τ, we let $\delta^+(\tau, A^k/G_X)$ (resp. $\delta^-(\tau, A^k/G_X)$) denote the set of orbitals that contain an arc starting (resp. ending) at vertex τ. That is,

$$\begin{aligned} \delta^+(\tau, A^k/G_X) &:= \left\{i \in A^k/G_X \,:\, (\tau, \tau\sigma) \in W_i^k \text{ for some } \sigma \in T\right\}, \\ \delta^-(\tau, A^k/G_X) &:= \left\{i \in A^k/G_X \,:\, (\tau\sigma, \tau) \in W_i^k \text{ for some } \sigma \in T\right\}. \end{aligned}$$

Similar definitions hold for $\delta^+(\tau, D^k/G_X)$ and $\delta^-(\tau, D^k/G_X)$. Again, observe that if τ_1 and τ_2 belong to the same orbit O_u^k, then $\delta^+(\tau_1, A^k/G_X) = \delta^+(\tau_2, A^k/G_X)$. For that reason, it makes sense to define $\delta^+(O_u^k, A^k/G_X)$, which is equal to $\delta^+(\tau, A^k/G_X)$ for any $\tau \in O_u^k$. In a similar fashion, we define $\delta^-(O_u^k, A^k/G_X)$, $\delta^+(O_u^k, D^k/G_X)$ and $\delta^-(O_u^k, D^k/G_X)$ for all $u \in V^k/G_X$ and $k \in [m]$.

The symmetry reduced equivalent formulation of (SPP) is obtained by replacing every variable x_e in H^k by a variable λ_i^k corresponding to the orbital W_i^k to which arc e belongs. Similarly, we replace every variable y_e in D^k by a variable θ_i^k corresponding to the orbital Z_i^k to which arc e belongs. As a consequence, the flow conservation constraint corresponding to vertices that belong to the same orbit becomes equivalent, hence we only keep one per orbit. The remaining linear programming problem we denote by (RSPP) and is given by

$$\begin{aligned} \min \quad & \sum_{k=1}^{m} \sum_{i \in A^k/G_X} |W_i^k| \lambda_i^k \\ \text{s.t.} \quad & \sum_{i \in D^0/G_X} d^+(Z_i^0)\theta_i^0 = 1, \quad \sum_{i \in D^m/G_X} d^-(Z_i^m)\theta_i^m = 1 \\ & \sum_{\substack{i \in \delta^-(O_u^k,\\ D^{k-1}/G_X)}} \theta_i^{k-1} + \sum_{\substack{i \in \delta^-(O_u^k,\\ A^k/G_X)}} d^-(W_i^k)\lambda_i^k = \\ & \qquad \sum_{\substack{i \in \delta^+(O_u^k,\\ D^k/G_X)}} \theta_i^k + \sum_{\substack{i \in \delta^+(O_u^k,\\ A^k/G_X)}} d^+(W_i^k)\lambda_i^k \qquad \forall u \in V^k/G_X,\ k \in [m] \\ & 0 \le \lambda_i^k \le 1 \quad \forall i \in A^k/G_X,\ k \in [m] \\ & 0 \le \theta_i^k \le 1 \quad \forall i \in D^k/G_X,\ k \in \{0\} \cup [m]. \end{aligned} \tag{RSPP}$$

Observe that $|W_i^k|, d^+(Z_i^0)$ and $d^-(Z_i^m)$ for all appropriate k and i are proportional to the size of an orbit in one of the subgraphs, which is in turn proportional to $|\mathrm{Aut}(\mathrm{Coup}(E))|$, see (13). For highly

symmetric coupling graphs, the order of this automorphism group becomes very large, leading to extreme coefficient values in (RSPP). This may lead to numerical instability when solving such program.

To improve practical performance, we apply a scaling operation prior to solving the program. We first multiply both sides of the flow conservation constraints by $|\mathrm{Aut}(\mathrm{Coup}(E))|$ for all $u \in V^k/G_X$ and $k \in [m]$. After that, we apply the following substitution:

$$\begin{aligned} \overline{\lambda}_i^k &:= |\mathrm{Aut}(\mathrm{Coup}(E))|\lambda_i^k \qquad \text{for all } i \in A^k/G_X, k \in [m], \\ \overline{\theta}_i^k &:= |\mathrm{Aut}(\mathrm{Coup}(E))|\theta_i^k \qquad \text{for all } i \in D^k/G_X, k \in \{0\} \cup [m]. \end{aligned}$$

This leads to the equivalent linear program (RSPP′). Observe that the new upper bounds on $\overline{\lambda}_i^k$ and $\overline{\theta}_i^k$ are omitted in this program. Indeed, the out-degree of s and in-degree of t needs to be 1, which implicitly enforces an upper bound of 1 on all $\overline{\theta}_i^0$ and $\overline{\theta}_i^m$. Since all variables and coefficient values are nonnegative and flow conservation holds throughout the program, we can without loss of generality omit the upper bounds on the variables. Hence, the coefficients of this program no longer depend on $|\mathrm{Aut}(\mathrm{Coup}(E))|$.

$$\text{(RSPP}') \qquad \begin{aligned} \min \quad & \sum_{k=1}^{m} \sum_{i \in A^k/G_X} \frac{|W_i^k|}{|\mathrm{Aut}(\mathrm{Coup}(E))|} \overline{\lambda}_i^k \\ \text{s.t.} \quad & \sum_{i \in D^0/G_X} \frac{d^+(Z_i^0)}{|\mathrm{Aut}(\mathrm{Coup}(E))|} \overline{\theta}_i^0 = 1, \quad \sum_{i \in D^m/G_X} \frac{d^-(Z_i^m)}{|\mathrm{Aut}(\mathrm{Coup}(E))|} \overline{\theta}_i^m = 1 \\ & \sum_{\substack{i \in \delta^-(O_u^k, \\ D^{k-1}/G_X)}} \overline{\theta}_i^{k-1} + \sum_{\substack{i \in \delta^-(O_u^k, \\ A^k/G_X)}} d^-(W_i^k)\overline{\lambda}_i^k = \\ & \qquad \sum_{\substack{i \in \delta^+(O_u^k, \\ D^k/G_X)}} \overline{\theta}_i^k + \sum_{\substack{i \in \delta^+(O_u^k, \\ A^k/G_X)}} d^+(W_i^k)\overline{\lambda}_i^k \qquad \forall u \in V^k/G_X,\ k \in [m] \\ & 0 \le \overline{\lambda}_i^k \quad \forall i \in A^k/G_X,\ k \in [m] \\ & 0 \le \overline{\theta}_i^k \quad \forall i \in D^k/G_X,\ k \in \{0\} \cup [m] \end{aligned}$$

Recall that the NNCP is in general $\mathcal{NP}$-hard [58]. Based on the LP formulation (RSPP′), we are able to unfold some special cases where the problem turns out to be polynomial time solvable. The condition that provides the key to this complexity result is the order of the automorphism group of the coupling graph.

Since all permutations in B_τ should setwise stabilize the sets $\tau^{-1}(S_i)$ for all $i \in [l]$, it follows that B_τ is a subgroup of $\mathbb{S}_n(\mathcal{G})$, where $\mathcal{G} := \{\tau^{-1}(S_1), \ldots, \tau^{-1}(S_l)\}$. The order of $\mathbb{S}_n(\mathcal{G})$ is $2^p f!$, which implies that $|B_\tau| \le 2^p f!$. This leads to the following complexity result.

Theorem 4.2. *The NNCP is polynomial time solvable on coupling graphs with automorphism groups of order $\Omega((n-b)!)$, where n is the number of vertices in the coupling graph and b is a constant independent of n.*

Proof. The number of variables in (RSPP) equals $m|A^1/G_X| + |D^0/G_X| + m|D^1/G_X|$. Based on Table 1 and the inequalities $|B_\tau| \le 2^p f!$, $|E/B_\tau| \le |E|$ and $|F^k| \le 2|E|(n-2)!$ for all $\tau \in \mathbb{S}_n$ and $k \in [m]$, we have

$$\begin{aligned} & \frac{m \sum_{\tau \in \mathbb{S}_n} |B_\tau| \cdot |E/B_\tau|}{2^p f! \cdot |\mathrm{Aut}(\mathrm{Coup}(E))|} + \frac{\sum_{\tau \in \mathbb{S}_n} |B_\tau|}{2^p f! \cdot |\mathrm{Aut}(\mathrm{Coup}(E))|} + \frac{m \sum_{\tau \in F^1} |B_\tau|}{2^p f! \cdot |\mathrm{Aut}(\mathrm{Coup}(E))|} \\ \le \quad & \frac{m\ 2^p f! |E| n!}{2^p f! \cdot |\mathrm{Aut}(\mathrm{Coup}(E))|} + \frac{2^p f! n!}{2^p f! \cdot |\mathrm{Aut}(\mathrm{Coup}(E))|} + \frac{m\ 2|E|(n-2)! 2^p f!}{2^p f! \cdot |\mathrm{Aut}(\mathrm{Coup}(E))|} \end{aligned}$$

$$= \quad O\left(\frac{m\ |E|\ n!}{|\mathrm{Aut}(\mathrm{Coup}(E))|}\right).$$

Whenever $|\mathrm{Aut}(\mathrm{Coup}(E))| = \Omega((n-b)!)$, the number of variables in (RSPP) is $O\left(m|E|n^b\right)$. Since b does not depend on the input, the number of variables in the reduced instance is polynomial in n, m and $|E|$. □

The implication of Theorem 4.2 does not solely restrict to trivial NNCP classes, such as the ones with a coupling graph that is complete. An example of a less trivial class of coupling graphs having a sufficiently large automorphism group are the bicliques, i.e., the complete bipartite graphs.

Corollary 4.3. *The NNCP is polynomial time solvable on the biclique $K_{N,M}$ with N of fixed size. In particular, the NNCP on the star $K_{1,N}$ is polynomial time solvable.*

4.2 Reduced combinatorial formulation

Similar to (SPP) being an LP formulation of a shortest path problem, we show in this section that (RSPP) and (RSPP′) also have a combinatorial interpretation. Such combinatorial approaches often have the potential to induce efficient algorithms that are favoured over solving their LP formulation. In order to simplify notation, we work with (RSPP) in this section, although the construction for (RSPP′) is similar.

To view (RSPP) as a combinatorial problem, we consider the so-called quotient graph of X under the action of G_X. In its most general form, a quotient graph of a graph X is induced by an equivalence relation on the vertices of X. We below provide the formal definition for the particular case where the equivalence relation is induced by an automorphism group of X.

Definition 4.4 (Quotient graph implied by automorphisms)**.** *Let $X = (V, A)$ be a directed graph and let G be a subgroup of* $\mathrm{Aut}(X)$*. Then the quotient graph of X under G is the graph $\mathcal{X} = (\mathcal{V}, \mathcal{A})$ with $\mathcal{V} := V/G$ and $\mathcal{A} := A/G \subseteq \mathcal{V} \times \mathcal{V}$.*

Since all arcs within an orbital of X start at vertices in the same orbit and end at vertices in the same orbit, the quotient graph is well-defined. Observe that $\mathcal{X}$ can contain loops and multi-arcs, even if X is simple.

Let $\mathcal{X} = (\mathcal{V}, \mathcal{A})$ be the quotient graph of X under G_X. Since the source vertex s and the sink vertex t are in isolated orbits, the vertices s and t are again in $\mathcal{V}$. By abuse of notation, we again denote these vertices as $s, t \in \mathcal{V}$. Since the constraints and variables in (RSPP) correspond to orbits and orbitals of X under G_X, respectively, the problem (RSPP) is an optimization problem on the quotient graph $\mathcal{X}$. Now, for all $(j, \ell) \in \mathcal{A}$ we define the following flow variable $f_{j\ell}$:

$$f_{j\ell} := \begin{cases} d^+(Z_i^0)\theta_i^0 & \text{if } (j,\ell) \text{ corresponds to } Z_i^0, \\ \theta_i^k & \text{if } (j,\ell) \text{ corresponds to } Z_i^k,\ k \in [m], \\ d^+(W_i^k)\lambda_i^k & \text{if } (j,\ell) \text{ corresponds to } W_i^k,\ k \in [m]. \end{cases} \tag{21}$$

Moreover, we define for all $(j, \ell) \in \mathcal{A}$ a cost vector

$$w_{j\ell} := \begin{cases} \frac{|W_i^k|}{d^+(W_i^k)} & \text{if } (j,\ell) \text{ corresponds to } W_i^k,\ k \in [m], \\ 0 & \text{otherwise,} \end{cases} \tag{22}$$

and an upper bound vector

$$u_{j\ell} := \begin{cases} d^+(Z_i^0) & \text{if } (j,\ell) \text{ corresponds to } Z_i^0, \\ 1 & \text{if } (j,\ell) \text{ corresponds to } Z_i^k,\ k \in [m], \\ d^+(W_i^k) & \text{if } (j,\ell) \text{ corresponds to } W_i^k,\ k \in [m]. \end{cases} \tag{23}$$

Finally, for all $(j,\ell) \in \mathcal{A}$ we define a multiplier $p_{j\ell}$:

$$p_{j\ell} := \begin{cases} \frac{d^-(W_i^k)}{d^+(W_i^k)} & \text{if } (j,\ell) \text{ corresponds to } W_i^k,\ k \in [m], \\ d^-(Z_i^m) & \text{if } (j,\ell) \text{ corresponds to } Z_i^m, \\ 1 & \text{otherwise.} \end{cases} \tag{24}$$

We now substitute $f_{j\ell}, w_{j\ell}$ and $p_{j\ell}$ for all orbitals $(j,\ell) \in \mathcal{A}$ into (RSPP). This yields an equivalent linear programming problem that has the structure of a minimum cost generalized network flow problem:

$$\text{(GNFP)} \qquad \begin{aligned} \min \quad & \sum_{(j,\ell)\in\mathcal{A}} w_{j\ell} f_{j\ell} \\ \text{s.t.} \quad & \sum_{(j,\ell)\in\delta^+(s)} f_{j\ell} = 1, \quad \sum_{(j,\ell)\in\delta^-(t)} p_{j\ell} f_{j\ell} = 1 \\ & \sum_{(j,\ell)\in\delta^+(v)} f_{j\ell} = \sum_{(j,\ell)\in\delta^-(v)} p_{j\ell} f_{j\ell} \qquad \forall v \in \mathcal{V} \setminus \{s,t\} \\ & 0 \le f_{j\ell} \le u_{j\ell} \qquad \forall (i,j) \in \mathcal{A}. \end{aligned}$$

A generalized flow is a flow starting from a sink s, conserving the flow at each vertex and ending at a source t, where along each arc (j,ℓ) only a fraction of $p_{j\ell}$ of flow is moved from j to ℓ. This fraction, called the multiplier, can also be larger than one, which means that the flow is increased along the arc. The problem (GNFP) aims to send a generalized flow of one from s to t that has a minimal cost with respect to the cost vector w. The minimum cost generalized network flow problem is solvable in weakly polynomial time by the algorithm of Wayne [64]. This is the only known combinatorial algorithm for this problem in the literature.

In the special case where B_τ is trivial for all $\tau \in \mathbb{S}_n$, the problem (GNFP) can be solved more efficiently. In that case we have $d^+(W_i^k) = d^-(W_i^k) = 1$ for all orbitals W_i^k, hence $p_{j\ell} = 1$ for all W_i^k. Now, for all $(j,\ell) \in \delta^-(t)$ we replace $p_{j\ell} f_{j\ell}$ by a new variable, say $g_{j\ell}$, that is upper-bounded by $d^-(Z_i^m)$. After these modifications, the resulting problem equals the LP formulation of a shortest path problem, for which strongly-polynomial time algorithms exist [20].

4.3 Backward reconstruction of optimal solutions

By construction, solving (RSPP) or (GNFP) provides the optimal cost of a shortest path in X. However, because of the reduction, the solutions of (RSPP) or (GNFP) do no longer correspond itself to paths. Let (λ, θ) be an optimal solution to (RSPP) (in case of solving (GNFP), we can obtain (λ, θ) from the flow variable f by (21)). Now, we define $(x,y) \in \prod_{k=1}^m \mathbb{R}^{A^k} \times \prod_{k=0}^m \mathbb{R}^{D^k}$ as follows:

$$x := \left(\sum_{i \in A^k/G_X} \lambda_i^k \mathbb{1}_{W_i^k} \right)_{k=1}^m \quad \text{and} \quad y := \left(\sum_{i \in D^k/G_X} \theta_i^k \mathbb{1}_{Z_i^k} \right)_{k=1}^m. \tag{25}$$

It follows from the construction that the pair (x,y) corresponds to an optimal solution of (SPP). Hence, it is a convex combination of characteristic vectors of (s,t)-paths in X. Let $X^{\sup}$ denote the subgraph of X induced by the support of (x,y). Then, $X^{\sup}$ is an acyclic graph. Namely, if there would exist a cycle in $X^{\sup}$, due to the orientation of the arcs in X, it can only consist of arcs within one subgraph. Since these arcs all have a positive cost, the solution (x,y) can be improved by excluding the cycle from it. By a similar argument, it follows that any (s,t)-path in $X^{\sup}$ must be optimal. Namely, if there is an (s,t)-path in the support with a strictly larger cost than the optimum, we can improve the solution (x,y) by excluding this path from it.

These observations imply that any (s,t)-path in $X^{\sup}$ is optimal. Finding such path can be done without actually constructing $X^{\sup}$. Starting from s, we select an arbitrary arc from an

orbital in D^0/G_X that is in the support of θ^0. This arc leads to a new vertex τ. From the orbit where τ belongs to, we again select an orbital leaving this orbit that has a support in the optimal solution (λ, θ). Within this orbital, there is at least one arc starting from τ and we select such an arc arbitrarily if there are multiple. We continue doing this, which will eventually lead to the sink vertex t. It follows from the discussion above that this (s,t)-path provides an optimal qubit ordering for the NNCP.

5 Special coupling graphs

Of key importance in the formulation discussed in Section 4 are the orbit and orbital representation of the subgraphs, which rely on the subgroups B_τ. These objects heavily depend on the specific coupling graph. In this section we demonstrate how these objects are obtained for three specific structured coupling graphs: the cycle graph, the biclique graph and the star graph.

Table 2 provides an overview of certain important characteristics of each of the considered coupling graphs. Details are provided in the subsections below.

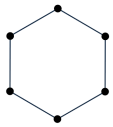
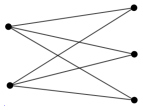
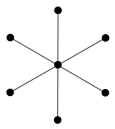
Architecture	n	$\lvert E\rvert$	**Graph structure**	**Aut(Coup(E))**	$\lvert B_\tau\rvert$
Cycle C_N	N	N	 Example of C_6	$\mathcal{D}_{2n}$	1 (unless the NNCP instance is trivial, see Theorem 5.1)
Biclique $K_{M,N}$	$M+N$	MN	 Example of $K_{2,3}$	$\mathbb{S}_M \times \mathbb{S}_N$ (if $M \neq N$)	$2^{p-\hat{p}} f_1! f_2!$, where $\hat{p}$, f_1 and f_2 follow from Theorem 5.2
Star $K_{1,N}$	$N+1$	N	 Example of $K_{1,6}$	$\mathbb{S}_{n-1}$	$2^{p-\hat{p}} f_1! f_2!$, where $\hat{p}$, f_1 and f_2 follow from Theorem 5.2

Table 2: Summary of NNCP symmetry reduction characteristics for a set of special coupling graphs.

5.1 Cycle graph C_N

Let $C_N = (L, E)$ be the cycle on N vertices, i.e., $L = [N]$ and $E = \{\{i, i+1\} : i \in [N-1]\} \cup \{N, 1\}$. Then $n = |L| = N$. It is well-known that the automorphism group of C_N is given by $\mathcal{D}_{2n}$, the dihedral group of order $2n$, see e.g., Godsil and Royle [27]. This group consists of all reflections and rotations of the regular polygon of order n. It follows from Theorem 3.3 that $\mathrm{Cay}(\mathbb{S}_n, T)$ is normal when $N \geq 5$ and, as a consequence, its full automorphism group is isomorphic to $\mathbb{S}_n \times \mathcal{D}_{2n}$. The Cayley graph $\mathrm{Aut}(\mathrm{Cay}(\mathbb{S}_n, T))$ with T induced by C_N is in the literature known as the modified bubble-sort graph, see e.g., [38].

The first step in studying the orbit and orbital structure of X under G_X is the identification of B_τ. It can be proven that B_τ is trivial under a very mild condition. Recall that c is the number of qubits in a connected component of size at least three in (Q, U), see Definition 3.7.

Theorem 5.1. *Suppose $c \geq 3$. Then B_τ is trivial for all $\tau \in \mathbb{S}_n$.*

Proof. Let $\tau \in \mathbb{S}_n$. If the gate graph (Q, U) contains a connected component of size at least three, then the fixing pattern $\mathcal{F}$ contains at least three single-element sets, say $\{i\}, \{j\}$ and $\{\ell\}$. Since B_τ is the subgroup of $\mathcal{D}_{2n}$ that setwise stabilizes the sets $\tau^{-1}(S_1), \ldots, \tau^{-1}(S_l)$, it follows that any $b \in B_\tau$ must pointwise fix $\tau^{-1}(i)$, $\tau^{-1}(j)$ and $\tau^{-1}(\ell)$. However, the only element in $\mathcal{D}_{2n}$ that fixes more than two elements is the identity element. Thus, B_τ is trivial. □

Observe that the condition of Theorem 5.1 is not restrictive. Namely, when $c < 3$, the quantum circuit does not have overlapping quantum gates. This implies that a trivial qubit assignment is possible without the need of any inserted SWAP gates, making the NNCP instance trivial.

5.2 Biclique graph $K_{M,N}$ and star graph $K_{1,N}$

The biclique graph $K_{M,N}$ is given by $L = [M] \sqcup [N]$ and $E = \{\{i, j\} : i \in [M], j \in [N]\}$. The induced partition of the vertex set L we denote by the sets L_M and L_N. We assume here that $M < N$. Any independent setwise permutation of vertices in L_M and L_N forms an automorphism of the graph, hence $\mathrm{Aut}(\mathrm{Coup}(E)) \cong \mathbb{S}_M \times \mathbb{S}_N$. The corresponding Cayley graph $\mathrm{Cay}(\mathbb{S}_n, T)$ is in the literature known as the generalized star graph, see e.g., [25]. With respect to the structure of the subgroups B_τ, we prove the following result.

Theorem 5.2. *Let $\tau \in \mathbb{S}_n$. Let $\mathcal{F}'$ denote the fixing pattern obtained from $\mathcal{F}$ by replacing any $S \in \mathcal{F}$ with $|S| \geq 2$ by*

$$S_1 = \{i \in S : \tau^{-1}(i) \in L_M\} \quad \text{and} \quad S_2 = \{i \in S : \tau^{-1}(i) \in L_N\}.$$

Then $B_\tau \cong \mathbb{S}_n(\mathcal{F}')$. Moreover, let $\hat{p}$ denote the number of pairs $\{i, j\}$ for which $\tau^{-1}(i) \in L_M$ and $\tau^{-1}(j) \in L_N$, let f_1 denote the number of elements in the free set that are mapped to L_M by τ^{-1}, and let $f_2 = f - f_1$. Then,

$$|B_\tau| = 2^{p-\hat{p}} f_1! f_2!.$$

Proof. Let $\mathcal{G} := \{\tau^{-1}(S_1), \ldots, \tau^{-1}(S_l)\}$ be the partition of $[n]$ defined by $\mathcal{F}$ shifted over τ^{-1}. Then, B_τ is the subgroup of $\mathbb{S}_n(\mathcal{G})$ which are also automorphisms of $K_{M,N}$. Since any automorphism of $K_{M,N}$ setwise fix the vertices in L_M and L_N, we obtain B_τ by splitting each set of $\mathcal{G}$ into its subset in L_M and its subset in L_N, leading to the partition $\mathcal{G}'$. The partition $\mathcal{F}'$ is exactly $\mathcal{G}'$ shifted over τ, leading to $B_\tau \cong \mathbb{S}_n(\mathcal{F}')$. The second part of the statement follows from counting the number of elements in $\mathbb{S}_n(\mathcal{F}')$. □

The special case where $M = 1$ is commonly known as the star graph $K_{1,N}$. Its induced Cayley graph is studied in [38]. Since we consider this coupling graph extensively in the numerical results of Section 6, we add this case explicitly to Table 2.

6 Computational results

In this section we test our symmetry-reduced NNCP formulation on a set of instances for the coupling graphs discussed in Section 5. We compare the result against the nonreduced shortest path formulation (SPP).

We first describe the design of our numerical tests in Section 6.1, after which we discuss the results on real and random instances in Section 6.2 and 6.3, respectively.

6.1 Design of computational experiments

For our experiments we consider both realistic as well as randomly generated quantum circuits on different coupling graphs. As described in Section 2, we are justified to make two assumptions on the quantum circuits under consideration, imposing a preprocessing strategy in case these assumptions are not met:

1. Single-qubit gates can be ignored for the NNCP, since these do always comply with the adjacent interaction constraints. Without loss of generality, we therefore remove the single-qubit gates from the circuits in the preprocessing phase.

2. All gates that act on more than two qubits are decomposed into gates that act on one or two qubits. Nielsen and Chuang [50] have shown that these gates are universal, and that any quantum gate can therefore be decomposed into one- or two-qubit gates. There exists a large number of different decomposition strategies, leading to possibly different quantum gates (with the same functionality, however). As the choice of the optimal decomposition strategy is outside the scope of our research, we always choose the same strategy, namely the method considered in [48, 49].

The quantum circuits that we consider in this paper consist of general one- or two-qubit gates, multiple-control Toffoli gates up to size five, Peres gates and multiple-control Fredkin gates up to size four. In Appendix A we consider the decomposition of these gates into one- or two qubit gates, following the approach from [48, 49]. After that, we remove all single-qubit gates from the circuit. The preprocessed circuit that remains, will be the quantum circuit $\Gamma = (Q, C)$ that we take as an input to our approach.

We consider the following two instance classes:

- **Real data**: Realistic quantum circuits that we consider are obtained from the RevLib library [66]. This dataset consists of quantum gates of (well-known) reversible functions considered in the quantum literature. Due to the assumptions of the preprocessing phase, we only consider instances consisting of the above-mentioned gates, see Appendix A for an overview. This leads to a set of 84 instances with $n \in \{5, \ldots, 17\}$ and $m \in \{7, \ldots, 112\}$.

- **Random data**: We also consider synthetic quantum gates in order to also test our approach on circuits consisting of more qubits and gates. We apply two strategies:

  - *Random Class I*: Given n and m, we create a random circuit on n qubits consisting of m two-qubit gates. Each gate acts on two qubits that are chosen uniformly at random from $[n]$ without replacement, independently from the other gates.

    For each combination of $n \in \{20, 30, \ldots, 100\}$ and $m = \{2n, 4n\}$, we consider 5 randomly generated instances of this type. This leads to a test set of 90 instances.

  - *Random Class II*: Given n and m, we first create a random circuit on n qubits consisting of m gates selected from: Toffoli gate (on 3, 4 or 5 qubits), Fredkin gate (on 3 or 4 qubits), Peres gate, or a general two-qubit gate. The latter class includes the CNOT, SWAP and controlled-V or -$V^\dagger$ gates. Each gate type is selected with equal probability, and the qubits on which each gate acts, is chosen uniformly at random from $[n]$ without replacement. After that, we apply the preprocessing approach explained above to convert each circuit to an equivalent circuit of two-qubit gates. This leads to quantum gates with possibly more realistic patterns than Random Class I.

    For each combination of $n \in \{20, 30, \ldots, 100\}$ and $m \in \{n, 2n\}$, we consider 5 randomly generated instances of this type, leading to a test set of 90 instances. After the preprocessing step, the values of m increase and are within $117 \leq m \leq 1872$.

We solve the NNCP for each quantum circuit on the following coupling graphs:

- **Cycle graph**: The undirected cycle C_N on $N = n$ qubits, see Section 5.1.

- **Star graph**: The star graph $K_{1,N}$ with $N = n - 1$, see Section 5.2.

- **Biclique graph**: The biclique graph $K_{M,N}$ with $M = 2$ and $N = n - 2$, see Section 5.2.

For each combination of quantum circuit and coupling graph, we solve the unreduced LP-formulation (SPP) and the reduced scaled formulation (RSPP′). The unreduced formulation is

implemented by a full construction of the graph $X = (V, A)$. The reduced formulation is implemented based on the results from Sections 3-5. We emphasize that it does not rely on the use of algebraic software, nor does it require a construction of the full graph X. Preliminary experiments have shown that the performance between the nonscaled and scaled formulations, (RSPP) and (RSPP′), respectively, is very similar. However, as the size of the coefficients in (RSPP) grows with the order of the automorphism group of Coup(E), the LP formulation becomes unstable for the star and biclique graphs when $n \geq 11$ or $n \geq 12$, respectively. Therefore, we only use the more robust scaled version (RSPP′) in our tests.

Experiments are carried out on a PC with an Intel(R) Core(TM) i7-8700 CPU, 3.20GHz and 8 GB RAM. Our methods are implemented in Julia 1.8.4 using JuMP v1.6.0 [15] to model the mathematical optimization problems. We use the LP solver of Mosek 10.0 [3] to solve our models in the default configuration. The maximum computation time (including the construction time of the program) is set to 2 hours.

6.2 Results on RevLib instances

Table 3, 4 and 5 show the results for the RevLib instances on the cycle, star and biclique graph, respectively. The columns 'n' and 'm' show the number of qubits and quantum gates in the preprocessed circuit. The column 'OPT' shows the optimal value of the NNCP instance, i.e., the minimum number of inserted SWAP gates in order to make the quantum circuit compliant. The columns 'time ($RSPP'$)' and 'time (SPP)' show the computation time (i.e., clocktimes) in seconds to solve the reduced model (RSPP′) and the base model (SPP), respectively. The values are rounded to three decimals. The columns '#var ($RSPP'$)' and '#const ($RSPP'$)' denote the total number of variables and constraints after the symmetry reduction. The column 'reduction #var (%)' shows the relative reduction in the number of variables compared to the base model, i.e., $\frac{\#\text{var } (SPP) - \#\text{var } (RSPP')}{\#\text{var } (SPP)} \cdot 100\%$, rounded to two decimal places. The final column shows the same relative reduction for the number of constraints. Whenever a given instance is not solvable (including construction) within the time limit of 2 hours, or whenever an instance leads to a shortage of memory, we report a '-' in the tables.

It turned out that the 62 instances with $n = 5$ are very easy to compute for both models (SPP) and ($RSPP'$). For that reason, results on these instances are not depicted in Tables 3, 4 and 5. The total relative reduction in the number of variables and constraints on the instances with $n = 5$ turns out to be at least 90% and 89.8%, respectively.

For the cycle graph, one can clearly see that the bottleneck in the computational limit is the number of qubits n. It follows from Table 3 that our approach is able to solve instances up to roughly 8 qubits, while the base model can only solve instances up to 7 qubits. The total computation time of (RSPP′) is often negligible and below 30 seconds for the instances that can be solved. For the base model the total computation times are significantly higher, with a maximum difference of about a factor 100. This can be explained by the large reduction in the total number of variables and constraints, which are both above 91% for all instances.

For the star graph, we conclude from Table 4 that the reduced model can easily handle the full set of RevLib instances. The computation times are negligible for almost all instances and always below 0.2 seconds. This can be explained by the order of Aut(Coup(E)) being factorial in n, implying that the model (RSPP′) scales linearly in both m and n. The relative reductions with the base model are enormous, i.e., above 99% in terms of the number of variables and constraints on all instances. For the unreduced model, the largest instance we can solve has $n = 8$ and $m = 36$, which could not be solved on the cycle coupling graph. This can be explained by the fact that the star graph on n vertices has one edge less than the cycle graph on n vertices, resulting in the Cayley graph containing significantly fewer edges. The computational frontier, however, is reached already at the next instance, for which the base model runs into memory issues.

Finally, the results on the biclique coupling graph look very similar to the results on the star graph, see Table 5. The total relative reduction between the models is extremely large, leading to all instances to be solvable within 0.25 seconds using (RSPP′). The computation times are slightly larger than for the star graph, which can be explained by the smaller size of the automorphism group of the biclique. For the unreduced formulation we can only solve up to $n = 7$, while the reduction

in computation time for the largest instance solvable by (SPP) is about a factor 4700.

Benchmark	n	m	**OPT**	**time** ($RSPP'$)	**time** (SPP)	**#var** ($RSPP'$)	**#const** ($RSPP'$)	**reduction #var (%)**	**reduction #const (%)**
graycode6_47	6	5	0	0.000	0.172	1980	3602	91.67	91.62
graycode6_48	6	5	0	0.016	0.172	1980	3602	91.67	91.62
decod24-enable_124	6	21	5	0.047	0.937	8124	15122	91.67	91.65
decod24-enable_125	6	21	4	0.047	0.906	8124	15122	91.67	91.65
decod24-bdd_294	6	24	8	0.062	1.203	9276	17282	91.67	91.66
mod5adder_129	6	71	27	0.157	4.563	27324	51122	91.67	91.66
mod5adder_128	6	77	32	0.172	4.250	29628	55442	91.67	91.66
decod24-enable_126	6	86	34	0.188	5.500	33084	61922	91.67	91.66
xor5_254	6	5	3	0.016	0.188	1980	3602	91.67	91.62
ex1_226	6	5	3	0.016	0.187	1980	3602	91.67	91.62
4mod5-bdd_287	7	23	8	0.469	36.203	61080	115922	92.86	92.86
alu-bdd_288	7	28	7	0.641	51.031	74280	141122	92.86	92.86
ham7_106	7	49	20	1.172	91.672	129720	246962	92.86	92.86
ham7_105	7	65	32	1.485	135.625	171960	327602	92.86	92.86
ham7_104	7	83	38	1.984	181.734	219480	418322	92.86	92.86
rd53_137	7	66	33	3.750	146.811	174600	23762	92.24	92.86
rd53_139	8	36	14	22.672	-	754200	90722	93.75	93.75
rd53_138	8	44	20	26.266	-	921240	110882	93.75	93.75
mini_alu_305	10	57	-	-	-	-	-	-	-
sys6-v0_144	10	62	-	-	-	-	-	-	-
rd73_141	10	64	-	-	-	-	-	-	-
parity_247	17	16	-	-	-	-	-	-	-

Table 3: Results on the 'RevLib' instances on the cyclic coupling graph. We compare the performance of the base model (SPP) with the reduced model ($RSPP$). Times are clocktimes given in seconds.

Benchmark	n	m	**OPT**	**time** ($RSPP'$)	**time** (SPP)	**#var** ($RSPP'$)	**#const** ($RSPP'$)	**reduction #var (%)**	**reduction #const (%)**
graycode6_47	6	5	2	0.000	0.125	166	32	99.17	99.11
graycode6_48	6	5	2	0.000	0.125	166	32	99.17	99.11
decod24-enable_124	6	21	4	0.016	0.609	678	128	99.17	99.15
decod24-enable_125	6	21	5	0.000	0.594	678	128	99.17	99.15
decod24-bdd_294	6	24	8	0.000	0.672	774	146	99.17	99.16
mod5adder_129	6	71	19	0.000	2.343	2278	428	99.17	99.16
mod5adder_128	6	77	18	0.000	2.953	2470	464	99.17	99.16
decod24-enable_126	6	86	19	0.016	2.718	2758	518	99.17	99.16
xor5_254	6	5	0	0.000	0.109	166	32	99.17	99.11
ex1_226	6	5	0	0.000	0.125	166	32	99.17	99.11
4mod5-bdd_287	7	23	5	0.000	14.875	1019	163	99.86	99.86
alu-bdd_288	7	28	11	0.000	16.141	1239	198	99.86	99.86
ham7_106	7	49	20	0.015	28.328	2163	345	99.86	99.86
ham7_105	7	65	18	0.000	36.157	2867	457	99.86	99.86
ham7_104	7	83	18	0.015	57.516	3659	583	99.86	99.86
rd53_137	7	66	10	0.000	38.521	2911	464	99.86	99.86
rd53_139	8	36	15	0.047	7031.828	2096	290	99.98	99.98
rd53_138	8	44	12	0.000	-	2560	354	99.98	99.98
mini_alu_305	10	57	16	0.016	-	5254	572	100.00	100.00
sys6-v0_144	10	62	26	0.015	-	5714	622	100.00	100.00
rd73_141	10	64	27	0.000	-	5898	642	100.00	100.00
parity_247	17	16	0	0.000	-	4401	274	100.00	100.00

Table 4: Results on the 'RevLib' instances on the star coupling graph. We compare the performance of the base model (SPP) with the reduced model ($RSPP$). Times are clocktimes given in seconds.

Benchmark	n	m	OPT	time ($RSPP'$)	time (SPP)	#var ($RSPP'$)	#const ($RSPP'$)	reduction #var (%)	reduction #const (%)
graycode6_47	6	5	1	0.015	0.250	655	77	97.92	97.86
graycode6_48	6	5	1	0.000	0.235	655	77	97.92	97.86
decod24-enable_124	6	21	4	0.016	1.500	2703	317	97.92	97.90
decod24-enable_125	6	21	4	0.000	1.297	2703	317	97.92	97.90
decod24-bdd_294	6	24	5	0.015	1.485	3087	362	97.92	97.91
mod5adder_129	6	71	15	0.032	5.031	9103	1067	97.92	97.91
mod5adder_128	6	77	14	0.031	5.016	9871	1157	97.92	97.91
decod24-enable_126	6	86	16	0.031	5.844	11023	1292	97.92	97.91
xor5_254	6	5	1	0.000	0.266	655	77	97.92	97.86
ex1_226	6	5	1	0.000	0.265	655	77	97.92	97.86
4mod5-bdd_287	7	23	4	0.016	72.110	5081	485	99.58	99.58
alu-bdd_288	7	28	5	0.016	83.844	6181	590	99.58	99.58
ham7_106	7	49	8	0.031	143.265	10801	1031	99.58	99.58
ham7_105	7	65	14	0.031	217.359	14321	1367	99.58	99.58
ham7_104	7	83	8	0.078	282.453	18281	1745	99.58	99.58
rd53_137	7	66	10	0.047	223.981	14541	1388	98.14	99.58
rd53_139	8	36	8	0.063	-	12556	1010	99.93	99.93
rd53_138	8	44	10	0.062	-	15340	1234	99.94	99.94
mini_alu_305	10	57	14	0.218	-	41997	2567	100.00	100.00
sys6-v0_144	10	62	13	0.141	-	45677	2792	100.00	100.00
rd73_141	10	64	14	0.095	-	47149	2882	100.00	100.00
parity_247	17	16	1	0.108	-	65896	2178	100.00	100.00

Table 5: Results on the 'RevLib' instances on the biclique coupling graph. We compare the performance of the base model (SPP) with the reduced model ($RSPP$). Times are clocktimes given in seconds.

6.3 Results on random instances

From Table 4 and 5 we observe that the RevLib instances can be easily solved by our symmetry reduced formulation. To test the performance on larger instances, we consider the random data set, consisting of quantum circuits with up to 100 qubits and 1837 quantum gates. For the cycle coupling graph, we have seen that we could only solve instances up to $n = 8$. Therefore, we do not include the cycle coupling graph anymore for the random data set. For the same reason, we do no longer consider the base model (SPP).

Table 6 and 7 show the performance of our symmetry-reduced NNCP formulation on Random Class I and Random Class II for both the star and biclique coupling graph. Next to the total solution time, which is given in the column 'time ($RSPP$)', we show in the column 'time constr.' the time that is required to construct the LP-instance. Each row in the tables corresponds to the average value over 5 randomly generated instances of that type. In Figure 2 we plot the averaged total computation time, i.e., construction and solution time, compared to n and m for both coupling graphs and random classes.

For the star coupling graph, we see that we can easily solve all instances from Random Class I within on average 25 seconds, while at most 90 seconds are needed to construct the model. For Random Class II, we can solve up to $n = 100$, however, when m is too large, the PC runs out of memory. For the biclique coupling graph on Random Class I, we can solve instances up to $n = 40$ within the time span of 2 hours, whereas for Random Class II the instances with large m cannot be solved anymore.

The sum of solution and construction times on the biclique graphs is significantly higher than on the star graphs, see Figure 2. The tables reveal that the solution times on the former are an order of magnitude 2 higher. This can be explained by the difference in the order of $\mathrm{Aut}(\mathrm{Coup}(E))$, as explained in Section 6.2. The construction times, however, heavily deviate among the instances on the star and the biclique coupling graph. Indeed, the smaller automorphism group increases the number of orbits. For each of these orbits, one needs to evaluate the orbitals of the group action of B_τ on E. Hence, the negative effects of having a smaller number of symmetries and a larger number of edges, strengthen one another and result in large construction times when n and m increase.

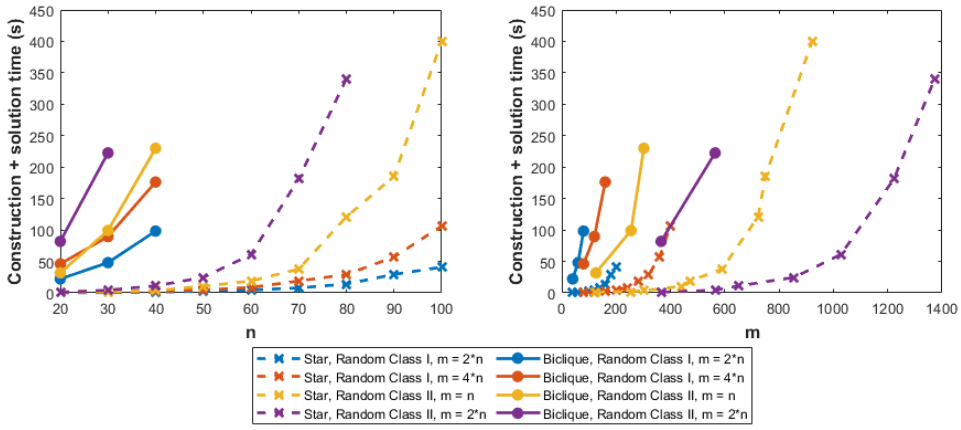


Figure 2: Overview of total average computation times (construction + solution time) of random instances with respect to n and m. Each data point displays the average over 5 randomly generated instances of that type.

When comparing Random Class I and II, we do not observe significant structural differences. It seems to be primarily the magnitude of n and m that influences the complexity of the instance. Due to the construction, m grows more rapidly with respect to n for Random Class II than for Random Class I. This effect can be observed from Figure 2, where we observe that for fixed n, an instance from Random Class II on average requires more computation time.

The largest quantum circuit that we can successfully solve contains 100 qubits and 1047 quantum gates. Observe that the unreduced model of this instance would embrace subgraphs of 100! vertices, hence solving this model is infeasible.

Random Class I				
n	m	**OPT**	**time** ($RSPP'$)	**time constr.**
20	40	29.6	0.031	0.088
20	80	65.2	0.056	0.134
30	60	52.4	0.106	0.274
30	120	101.2	0.243	0.551
40	80	72.4	0.282	0.820
40	160	144.4	0.631	1.556
50	100	91.2	0.569	1.971
50	200	184.6	1.312	3.336
60	120	112.8	1.025	3.349
60	240	222.4	2.447	6.162
70	140	132.0	1.769	6.135
70	280	264.6	4.662	14.194
80	160	151.0	3.313	10.704
80	320	304.2	7.775	21.305
90	180	172.0	4.809	24.524
90	360	343.8	14.681	42.293
100	200	191.8	9.106	31.802
100	400	385.0	21.066	85.295

Random Class II				
n	m	**OPT**	**time** ($RSPP'$)	**time constr.**
20	125.6	35.0	0.119	1.425
20	365.6	80.8	0.334	0.712
30	255	59.6	0.544	1.189
30	564.6	126.4	1.350	2.940
40	302.2	76.6	1.150	2.482
40	652.4	159.8	3.150	8.081
50	441.4	104.6	3.272	7.223
50	854.8	203.8	8.091	16.474
60	471	122.8	5.737	13.121
60	1027.4	247.6	16.903	44.061
70	589.4	145.2	10.838	26.888
70	1223	292.8	31.875	149.451
80	722.4	171.6	23.634	97.156
80	1372.8	333.6	32.600	307.844
90	750	184.8	22.312	162.915
90	1602.8	-	-	-
100	921.2	218.8	36.966	363.073
100	1709.6	-	-	-

Table 6: Results on the random instances on the star coupling graph. Each row shows the average values over 5 randomly generated instances. Times are clocktimes given in seconds.

Random Type I				
n	m	OPT	time ($RSPP'$)	time constr.
20	40	20.6	1.588	12.029
20	80	43.4	2.590	14.217
30	60	38.4	9.675	374.035
30	120	71.2	18.322	390.226
40	80	54.2	44.053	3872.272
40	160	109.4	66.884	3989.450

Random Type II				
n	m	OPT	time ($RSPP'$)	time constr.
20	125.6	27.6	4.188	17.270
20	365.6	66.8	15.113	30.584
30	255	50.0	49.175	413.975
30	564.6	107.2	115.350	829.710
40	302.2	61.7	168.276	2800.738
40	652.4	-	-	-

Table 7: Results on the random instances on the biclique coupling graph. Each row shows the average values over 5 randomly generated instances. Times are clocktimes given in seconds.

7 Conclusions

In this paper we study an exact method for solving the NNCP in the gated quantum computing model by exploiting symmetries in the underlying formulation.

Starting from the shortest path formulation introduced by [46], see (SPP), we study the algebraic structure of the underlying graph in Section 3. This graph is composed of a series of Cayley graphs of the symmetric group $\mathbb{S}_n$ generated by the transpositions in the coupling graph of the quantum system. We show that $\mathbb{S}_n \times \mathrm{Aut}(\mathrm{Coup}(E))$ is a subgroup of the automorphism group of such Cayley graph, which turns out to be the full automorphism group in case the Cayley graph is normal as shown by [24]. Although the automorphism groups of specific Cayley graphs generated by transpositions has been studied before in the literature, we do not make any assumption on the underlying coupling graph apart from being connected. Next, we show how these subgroups are merged into a subgroup G_X of the automorphism group of the entire graph, see (8). One component of this subgroup is determined by the algebraic structure of the coupling graph, while the other component relies on a so-called fixing pattern $\mathcal{F}$ following from the quantum gates in the circuit, see Definition 3.7. The orbit and orbital structures of the action of this group on the graph are also studied, leading in particular to an overview of the cardinalities of the corresponding quotients, see Table 1.

By exploiting the convexity of (SPP), we reduce the symmetries in the formulation by averaging over all symmetric solutions using the Reynolds operator, see (15). This leads to a more compact equivalent formulation (RSPP) and its scaled variant (RSPP$'$). We show that this formulation is equivalent to a generalized network flow problem (GNFP). Due to the in-depth analysis on the orbit and orbital structure, these formulations can be explicitly constructed from scratch without the need to first construct the exponentially large Cayley graphs. A direct theoretical implication of our approach are the complexity results of Theorem 4.2 and Corollary 4.3, which reveal a class of polynomial time solvable special cases of the NNCP.

The gain of using our approach compared to the base model (SPP) is most vibrant in case the fixing pattern is less restrictive and the coupling graph is (highly) symmetric. We test our approach on three types of coupling graphs, for which we explicitly derive the key ingredients of our formulation, see Table 2. Our numerical results show that the gain in efficiency due to the exploitation of symmetries is very large. For each of the 84 real and 180 random instances, the total reduction in the number of variables and constraints is at least 90% and 89.8%, respectively, and this number grows with n and m. The computation times are significantly reduced compared to the unreduced model, resulting in solving NNCP instances that are much larger than the ones considered so far in the literature. The largest instance we can solve contains 100 qubits and 1047 quantum gates.

Given that we are only at the beginning of the quantum era, related optimization problems such as the NNCP are likely to remain important in the near future. Based on the successful implementation of our symmetry-reduced solution approach, it would be interesting to consider the NNCP on other quantum architectures having a large symmetry group.

References

[1] M.G. Alfailakawi, I. Ahmad, and S. Hamdan. Harmony-search algorithm for 2D nearest neigbor quantum circuits realization. *Expert Syst. with Appl.*, 61:16–27, 2016.

[2] N. Alhagi. Synthesis of reversible functions using various gate libraries and design specifications. Technical report, Portland State University, 2000.

[3] MOSEK ApS. *MOSEK Optimization Suite 10.0.40*, 2022.

[4] A. Barenco, C.H. Bennett, R. Cleve, D.P. DiVincenzo, P.W. Margolus, P.W. Shor, T. Sleator, J. Smolin, and H. Weinfurter. Elementary gates for quantum computation. *Phys. Rev.*, 52:3457–3467, 1995.

[5] A. Bhattacharjee, C. Bandyopadhyay, R. Wille, R. Drechsler, and H. Rahaman. A novel approach for nearest neighbor realization of 2D quantum circuits. In *2018 IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, pages 305–310, Hong Kong, 2018. IEEE.

[6] A. Bhattacharjee, C. Bandyopadhyay, R. Wille, R. Drechsler, and H. Rahaman. 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)*, pages 203–208, Delhi, NCR, India, 2019. IEEE.

[7] D. Bhattacharjee and A. Chattopadhyay. Depth-optimal quantum circuit placement for arbitrary topologies. arXiv:1703.08540, 2017.

[8] R. Bödi, K. Herr, and M. Joswig. Algorithms for highly symmetric linear and integer programs. *Math. Program.*, 137(1-2):65–90, 2013.

[9] X. Cheng, Z. Guan, and W. Ding. Mapping from multiple-control Toffoli circuits to linear nearest neighbor quantum circuits. *Quantum Inf. Process.*, (17):169, 2018.

[10] B.S. Choi and R. Van Meter. An $\theta(\sqrt{n})$-depth quantum adder on a 2D ntc quantum computer architecture. *J. Emerg. Technol. Coput. Syst.*, 8:1–22, 2012.

[11] A. Cowtan, S. Dilkes, R. Duncan, A. Krajenbrink, W. Simmons, and S. Sivarajah. On the qubit routing problem. arXiv:1902.08091, 2019.

[12] J. Ding and S. Yamashita. Exact synthesis of nearest neighbor compliant quantum circuits in 2-d architecture and its application to large-scale circuits. *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, 39(5):1045–1058, 2019.

[13] D.P. DiVincenzo and IBM. The physical implementation of quantum computation. *Fortschr. der Phys.*, (48):771–783, 2000.

[14] J. Van Doornmalen and C. Hojny. A unified framework for symmetry handling. *arXiv:2211.01295*, 2022.

[15] I. Dunning, J. Huchette, and M. Lubin. JuMP: A modeling language for mathematical optimization. *SIAM Rev.*, pages 295–320, 2017.

[16] A. Farghadan and N. Mohammadzadeh. Mapping quantum circuits on 3D nearest-neighbor architectures. *Quantum Sci. Technol.*, 4:035001, 2019.

[17] Y.Q. Feng. Automorphism groups of Cayley graphs on symmetric groups with generating transposition sets. *Journal of Combinatorial Theory, Series B*, 96:67–72, 2006.

[18] A.G. Fowler, S.J. Devitt, and L.C.L. Hollenberg. Implementation of Shor's algorithm on a linear nearest neighbour qubit array. *Quantum Inf. Comput.*, 4:237–251, 2004.

[19] E. Fredkin and T. Toffoli. Conservative logic. *Int. J. Theor. Physics*, 21:219–253, 1982.

[20] M.L. Fredman and R.E. Tarjan. Fibonacci heaps and their uses in improved network optimization algorithms. *J. ACM*, 34(3):596–615, jul 1987.

[21] E.J. Friedman. Fundamental domains for integer programs with symmetries. In *Combinatorial Optimization and Applications: First International Conference, COCOA 2007, Xi'an, China, August 14-16, 2007. Proceedings 1*, pages 146–153. Springer, 2007.

[22] A. Ganesan. Automorphism groups of Cayley graphs generated by connected transposition sets. *Discrete Mathematics*, 313:2482–2485, 2013.

[23] A. Ganesan. Automorphism group of the complete transposition graph. *Journal of Algebraic Combinatorics*, 42:793–801, 2015.

[24] A. Ganesan. On the automorphism group of Cayley graphs generated by transpositions. *Australasian Journal of Combinatorics*, 64(3):432–436, 2016.

[25] A. Ganesan. Cayley graphs and symmetric interconnection networks. arXiv:1703.08109, 2017.

[26] K. Gatermann and P.A. Parrilo. Symmetry groups, semidefinite programs, and sums of squares. *J. Pure Appl. Algebra*, 192(1-3):95–128, 2004.

[27] C. Godsil and G.F. Royle. *Algebraic Graph Theory*. Number 207 in Graduate Texts in Mathematics. Springer, 2001.

[28] H. Häffner, C.F. Roos, and R. Blatt. Quantum computing with trapped ions. *Phys. Rep.*, 469(4):155–203, 2008.

[29] M.C. Heydemann. *Cayley graphs and interconnection networks*, pages 167–224. Springer Netherlands, Dordrecht, 1997.

[30] Y. Hirata, M. Nakanishi, S. Yamashita, and Y. Nakashima. An efficient conversion of quantum circuits to a linear nearest neighbour architecture. *Quantum Inf. and Comput.*, 11:142–166, 2011.

[31] W.N. Hung, X. Song, G. Yang, J. Yang, and M. Perkowski. Optimal synthesis of multiple output boolean functions using a set of quantum gates by symbolic reachability analysis. *IEEE Trans. Comput.-Aid. Des.*, 25:1652–1663, 2006.

[32] T. Itoko, R. Raymond, T. Imamichi, and A. Matsuo. Optimization of quantum circuit mapping using gate transformation and commutation. *Integration*, 70:43–50, 2020.

[33] M.R. Jerrum. The complexity of finding minimum-length generator sequences. *Theor. Comput. Sci.*, 36:265–289, 1985.

[34] V. Kaibel and M. Pfetsch. Packing and partitioning orbitopes. *Math. Program.*, 114(1):1–36, 2008.

[35] A. Kole, K. Datta, and I. Sengupta. A heurstic for linear nearest neighbor realization of quantum circuits by SWAP gate insertion using n-gate lookahead. *IEEE J. Emerg. Sel. Top. Circuits Syst.*, 6:62–72, 2016.

[36] A. Kole, K. Datta, and I. Sengupta. 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, 2017.

[37] A. Kole, K. Datta, I. Sengupta, and R. Wille. Towards a cost metric for nearest neighbor constraints in reversible circuits. In *Reversible Computation (RC): 7th International Conference*, pages 273–278, Grenoble, France, 2015.

[38] S. Lakshmivarahan, J.S. Jho, and S.K. Dhal. Symmetry in interconnection networks based on Cayley graphs of permutation groups: A survey. *Parallel Computing*, (19):361–407, 1993.

[39] J. Lee and F. Margot. On a binary-encoded ilp coloring formulation. *INFORMS J. Comput.*, 19(3):406–415, 2007.

[40] G. Li, Y. Ding, and Y. Xie. Tackling the qubit mapping problem for NISQ-era quantum devices. In *Proceedings of the 24th International Conference on Architectural Support for Programming Languages and Operating Systems*, pages 1001–1014, 2019.

[41] L. Liberti. Symmetry in mathematical programming. In *Mixed Integer Nonlinear Programming*, pages 263–283. Springer, 2012.

[42] F. Margot. Pruning by isomorphism in branch-and-cut. *Math. Program.*, 94:71–90, 2002.

[43] F. Margot. Exploiting orbits in symmetric ilp. *Math. Program.*, 98:3–21, 2003.

[44] F. Margot. Small covering designs by branch-and-cut. *Math. Program.*, 94:207–220, 2003.

[45] F. Margot. *Symmetry in integer linear programming*, pages 647–686. Springer, 2009.

[46] A. Matsuo and S. Yamashita. Changing the gate order for optimal lnn conversion. In A. De Vos and R. Wille, editors, *Reversible Computation*, pages 89–101, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

[47] A. Montanaro. Quantum algorithms: an overview. *npj Quantum Inf.*, 2(15023), 2016.

[48] J. Mulderij. Nearest neighbor compliance in quantum circuit design. Master's thesis, Delft University of Technology, 2019.

[49] J. Mulderij, K.I. Aardal, I. Chiscop, and F. Phillipson. A polynomial size model with implicit swap gate counting for exact qubit reordering. arXiv:2009.08748, 2020.

[50] M.A. Nielsen and I. Chuang. *Quantum computation and quantum information.* Cambridge University Press, New York, USA, 2010.

[51] J. Ostrowski, J. Linderoth, F. Rossi, and S. Smriglio. Orbital branching. *Math. Program.*, 126:147–178, 2011.

[52] M. Pedram and A. Shafaei. Layout optimization for quantum circuits with linear nearest neighbor architectures. *IEEE Circuits and Syst. Mag.*, 16:62–74, 2016.

[53] A. Peres. Reversible logic and quantum computers. *Phys. Rev. A, Gen. Phys.*, 32:3266–3276, 1985.

[54] Qiskit contributors. Qiskit: An open-source framework for quantum computing, 2023.

[55] A. Shafaei, M. Saeedi, and M. Pedram. Optimization of quantum circuits for interaction distance in linear nearest neighbor architectures. In *2013 50th ACM/EDAC/IEEE Design Automation Conference (DAC)*, pages 1–6, 2013.

[56] A. Shafaei, M. Saeedi, and M. Pedram. Qubit placement to minimize communication overhead in 2d quantum architectures. In *2014 19th Asia and South Pacific Design Automation Conference (ASP-DAC)*, pages 495–500, 2014.

[57] H.D. Sherali and J.C. Smith. Improving discrete model representations via symmetry considerations. *Manage. Sci.*, 47(10):1396–1407, 2001.

[58] M.Y. Siraichi, V.F. dos Santos, S. Collange, and F.M.Q. Pereira. Qubit allocation. In *Proceedings of the 2018 International Symposium on Code Generation and Optimization (CGO 2018)*, pages 113–125. ACM, 2018.

[59] T. Toffoli. Reversible computing. Technical report, MIT Lab for Computer Science, 1980. Technical memo MIT/LCS/TM-151.

[60] A.W. Tucker. Solving a matrix game by linear programming. *IBM Journal of Research and Development*, 4(5):507–517, 1960.

[61] A.W. Tucker. Combinatorial theory underlying linear programs. *Recent Adv. Math. Program.*, pages 1–16, 1963.

[62] R. Van Houte, J. Mulderij, T. Attema, I. Chiscop, and F. Phillipson. Mathematical formulation of quantum circuit design problems in networks of quantum computers. *Quantum Inf. Process.*, 19:1–22, 2020.

[63] D. Venturelli, M. Do, E. Rieffel, and J. Frank. Temporal planning for compilation of quantum approximate optimization circuits. In *Scheduling and Planning Applications Workshop (SPARK)*, page 58, 2017.

[64] K.D. Wayne. A polynomial combinatorial algorithm for generalized minimum cost flow. *Mathematics of Operations Research*, 27:445–459, 2002.

[65] R. Wille, L. Burgholzer, and A. Zulehner. 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*, pages 1–6, Las Vegas, NV, USA, 2019.

[66] R. Wille, D. Große, L. Teuber, G. W. Dueck, and R. Drechsler. RevLib: An online resource for reversible functions and reversible circuits. In *Int'l Symp. on Multi-Valued Logic*, pages 220–225, 2008. RevLib is available at http://www.revlib.org.

[67] R. Wille, O. Keszocze, M. Walter, P. Rohrs, A. Chattopadhyay, and R. Drechsler. 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)*, pages 292–297, Macao, 2016. IEEE.

[68] R. Wille, A. Lye, and R. Drechsler. Exact reordering of circuit lines for nearest neighbor quantum architectures. *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, 33(12):1818–1831, 2014.

A Quantum gates and their two-qubit decompositions

Since the NNCP is only well-defined when a quantum circuit consists solely of one- or two-qubit gates, we have to decompose all gates that act on more than two gates. As indicated in Section 6, this task can be completed in lots of ways and performing this decomposition optimally can be seen as a research problem in itself. In this paper we apply the decomposition method used in [49], although the authors of [49] already indicated that this method might be open for improvement.

The quantum circuits that we consider in our experiments consist of the following types of quantum gates: one-qubit gates, two-qubit gates, three-qubit Peres gates, three- and four-qubit Fredkin gates and three-, four- and five-qubit Toffoli gates. Commonly used one-qubit gates are the Hadamard gate and the Pauli-gates, e.g., the Pauli-X-gate. When applying the Hadamard gate to a qubit in any state, it brings the qubit in a superposition state where it has an equal probability to be 0 or 1 upon measurement. The Hadamard gate in a quantum circuit is depicted as $-\boxed{H}-$. The Pauli-X-gate is also known as the NOT gate and can be seen as its quantum analog. The NOT-gate is depicted as $-\oplus-$.

The most commonly used two-qubit gates are depicted in Figure 3. The controlled-NOT gate, also known as CNOT or Feynman gate, acts on a control qubit and a target qubit. If the control qubit is in state $|1\rangle$, a NOT-gate is applied to the target qubit, otherwise nothing happens. The SWAP gate swaps the states of the two qubits where it acts on. The controlled-V and controlled-$V^\dagger$ act similarly to the controlled-NOT gate, with the only difference that the unitary operation V or $V^\dagger$ is applied to the target qubit. The operation V and $V^\dagger$ are the square root of the NOT-gate and its Hermitian conjugate, respectively. That is, if two controlled-V gates are placed in succession, the result is similar to a controlled-NOT gate, while the identity gate is obtained when applying a controlled-V and a controlled-$V^\dagger$ gate in succession.

(a) Controlled-NOT (b) SWAP (c) Controlled-V (d) Controlled-$V^\dagger$

Figure 3: Overview of commonly used two-qubit quantum gates.

A Toffoli gate [59] is the multiple-control NOT gate. Acting on several control qubits and a single target qubit, a NOT gate is applied to the target qubit if all the control qubits are in state $|1\rangle$. The three-qubit Toffoli gate is depicted in Figure 4, along with a possible decomposition into two-qubit gates, following the approach of [4].

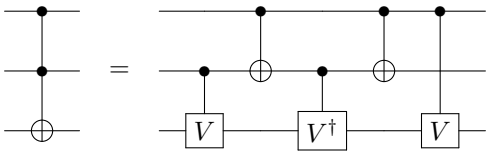


Figure 4: Decomposition of multiple-control Toffoli gate with two controls and a single target qubit.

The Peres gate [53] is obtained from a combination of a two-qubit controlled-NOT gate and standard controlled-NOT gate. Following the approach from [31], the Peres gate can be decomposed into four two-qubit gates, as shown in Figure 5.

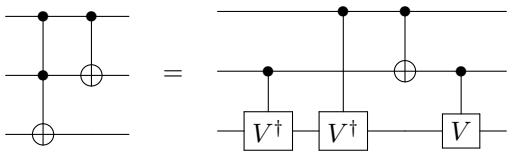


Figure 5: Decomposition of Peres gate on three qubits.

The Fredkin gate [19] operates on three qubits as a controlled-SWAP gate. If the state of the control qubit is $|1\rangle$, then a SWAP gate on the two target qubits is performed. The decomposition into two-qubit gates that we adapt here is the same as the one considered in [49, 66], see Figure 6

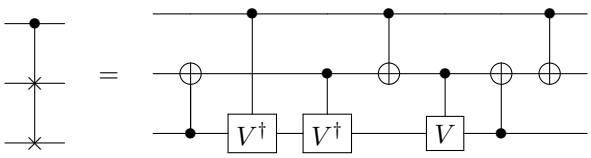


Figure 6: Decomposition of Fredkin gate (controlled swap gate) with one control qubit.

Finally, we consider the four- and five qubit variants of the Fredkin and Toffoli gate. The functionality of these gates is similar to their three-qubit implementation, only the number of control qubits is larger. The four-qubit Fredkin gate can be decomposed as shown by [2], see Figure 7. Fredkin gates on a larger number of qubits do not appear in our experiments.

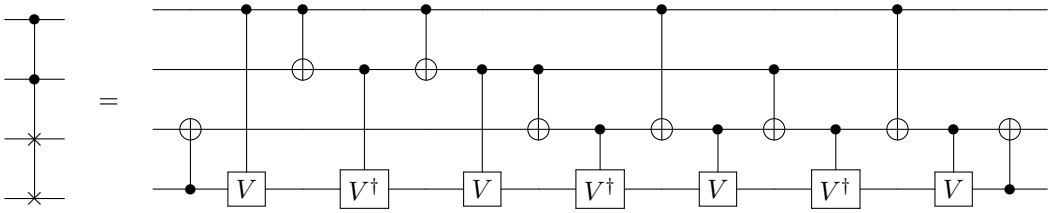


Figure 7: Decomposition of Fredkin gate (controlled swap gate) with two control qubits.

Finally, the four- and five-qubit Toffoli gates are shown in Figure 8 and 9. The decompositions shown here follow from the construction derived in [4]. Toffoli gates on more than five qubits do not appear in our experiments.

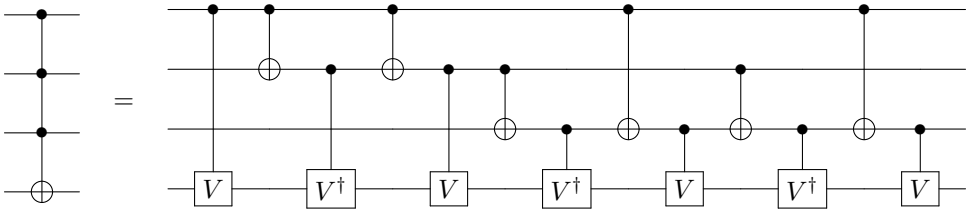


Figure 8: Decomposition of multiple-control Toffoli gate with three controls and a single target qubit.

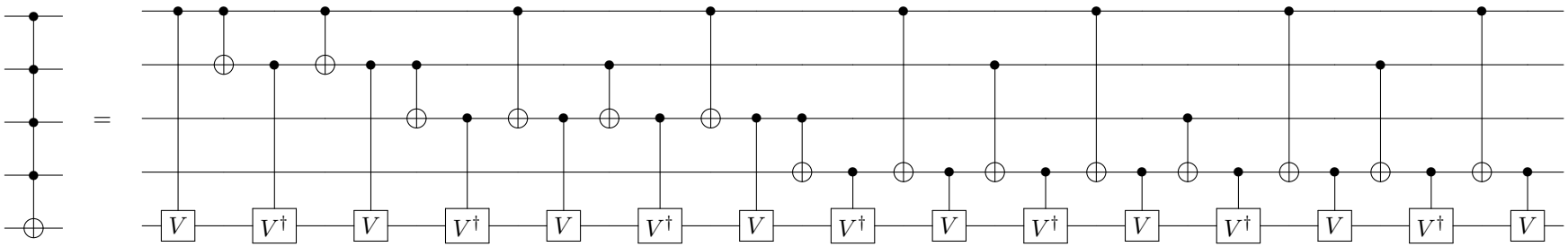


Figure 9: Decomposition of multiple-control Toffoli gate with four controls and a single target qubit.