



OPEN Quantum annealing for nearest neighbour compliance problem

Sven Müller¹ & Frank Phillipson^{1,2}✉

Quantum Computing has emerged as a promising alternative, utilising quantum mechanics for faster computations. This paper explores the nearest neighbour compliance (NNC) Problem in Gate-based Quantum Computers, where quantum gates are constrained to operate on physically adjacent qubits. The NNC problem aims to optimise the insertion of SWAP-gates to ensure compliance with these constraints while minimising their count. This work introduces Quantum Annealing to tackle the NNC problem, proposing two Quadratic Unconstrained Optimisation Problem formulations. The formulations are tested on a contemporary Quantum Annealer, and their performance is compared with previous methods. It shows that the prospect of using Quantum Annealing is promising, however, the current state of the hardware makes that finding the embedding is the limiting factor.

Keywords Nearest neighbour compliance problem, Gate-based quantum computers, Quantum annealing, SWAP optimisation

Contemporary devices such as smartphones and computers, as we know them today, would not be possible without the continuous advances in microchips, or more specifically, semiconductors. They play a pivotal role in modern society by facilitating communication across vast distances and enabling seamless access to information, transforming how we connect, acquire knowledge, and make decisions. Subsequently, advances in semiconductor technologies are a key source of increased productivity which in turn has been a primary driver for economic growth^{1–3}.

Over the last 50 years, the number of transistors on a microchip has grown exponentially, following the predictions by Moore⁴, known as Moore's law. However, this growth has slowed down in recent years as the technology is soon to reach its fundamental physical limits^{5–7}.

In that context, Quantum Computing has emerged as an alternative paradigm to computing that harnesses the intricacies of quantum mechanics to perform certain computations more quickly and provide algorithms with an improved complexity scaling. Most prominently, there is Grover's algorithm⁸ to more efficiently search large databases and Shor's algorithm to factor large integers⁹.

Currently, there are two main realisations of Quantum Computers, the Quantum Annealer (QA) such as the Advantage Series by D-Wave Systems, and the Gate-based Quantum Computers (GQC) such as IBM's Quantum System Two. While the QA is purpose-built to solve a specific class of binary optimisation problems, the GQC is universal and can theoretically perform any computation or algorithm that a classical computer could. Specifically, the GQC performs computations by executing a quantum circuit that applies a sequence of quantum gates to a set of quantum bits (qubits)¹⁰.

The potential advantages that GQC offers, in theory, are accompanied by limitations of physical implementations of such computers. In the context of this work, two such limitations are of direct relevance. Firstly, qubits can not store information indefinitely due to an inevitable exchange of energy with their environment¹¹. This process is referred to as quantum decoherence and is a direct consequence of the imperfect isolation of quantum computers from their surroundings. Consequently, any computation performed on a GQC has to be completed before the quantum system loses its coherence which ultimately implies that it is desirable to physically implement any quantum algorithm with as few gates as possible. Secondly, in many common physical implementations of GQCs, qubits are only coupled with their nearest neighbours, that is, a limited set of adjacent qubits¹². This limitation gives rise to Nearest Neighbour Constraints which indicate that quantum gates can solely be applied to physically adjacent qubits.

To ensure the compliance of each two-qubit gate in a circuit with the nearest neighbour constraint, one can insert so-called SWAP-gates that swap the information contained by two adjacent qubits to effectively interchange the qubits' location in the coupling graph¹². Moreover, given the limited lifetime of the qubits due to decoherence, it is highly desirable to satisfy the constraints by inserting as few SWAP-gates into the circuit as

¹School of Business and Economics, Maastricht University, Minderbroedersberg 4, 6211 LK Maastricht, The Netherlands. ²Applied Cryptography and Quantum Algorithms, TNO, Anna van Buerenplein 1, 2595 DA The Hague, The Netherlands. ✉email: f.phillipson@maastrichtuniversity.nl

possible. The resulting optimisation problem is the “Nearest Neighbour Compliance Problem” (NNC)¹³, which is also known under other names like qubit routing and circuit compilation¹⁴.

Nearest Neighbour Constraints have been considered in a wide variety of contexts, including specific circuit types and qubit architectures¹⁵. In the latter case, which is of interest in this work, the meaning of adjacency is defined by a coupling graph. Again, various cases of coupling graphs have been considered where qubits are placed on either a linear array (see Fig. 1), two-dimensional grid, three-dimensional grid, or manufacturer-specific architectures. Moreover, different research areas either focus on a global or local perspective on the reordering of the qubits¹⁶. While the global perspective aims to reduce the problem’s difficulty by producing an, on average, optimal initial ordering of qubits, the local reordering problem searches for a permutation of qubits before each gate in the circuit, so that the total number of swaps between all consecutive permutations is minimised. Therefore, local reordering allows for overall better solutions¹⁷ at the cost of an often exponentially growing number of variables in the model, often resulting in intractable running times for exact methods on large instances.

Given the combinatorial and discrete nature of the NNC problem arising in QQC, another form of quantum computing, quantum annealing, is a natural candidate to tackle this difficult optimisation problem¹⁸. Even in its current, early stages, QA can solve some first real-world problems and is seen as a soon-to-be-competitive machine heuristic to solve quadratic problems formulated as a Quadratic Unconstrained Binary Optimisation Problem (QUBO). Therefore, an efficient QUBO formulation that is solved on a QA has the potential to be an efficient general-instance heuristic optimisation method, which does not yet exist in the NNC literature¹⁵. To the best of the author’s knowledge, Quantum Annealing has not yet been applied to solve the NNC problem as is done in this work. However, Quantum Annealing has been applied to a variety of relevant problems in recent years, including the Nurse Scheduling Problem¹⁹, Traffic Flow Optimisation²⁰, Portfolio Optimisation²¹, Energy System Optimisation^{22,23}, and Job Scheduling²⁴.

This work proposes two novel reformulations of the NNC problem as Quadratic Unconstrained Binary Optimization (QUBO) instances, which are subsequently tackled using a contemporary Quantum Annealer developed by D-Wave Systems. This builds upon the method introduced by¹³, which provided an exact Integer Linear Programming (ILP) formulation for the local-reordering NNC problem on a linear array, thus avoiding the factorial scaling in qubit count suffered by previous approaches. The primary innovation lies in proposing a quantum-compatible formulation for a problem already addressed classically. Currently, this approach faces two main limitations: it is only applicable to linear architectures, precluding direct usage on most existing quantum processors, and it yields results inferior to both exact and heuristic classical solutions, either in quality or runtime. Nonetheless, the authors suggest that these limitations might be mitigated with the advent of more potent and scalable Quantum Annealer processors. The central contribution of this work is the transformation of an existing ILP model, proposed in reference¹³, into two QUBO formulations, where one achieves variable reduction for improved efficiency.

The remainder of this paper is organised as follows. “Literature” section reviews relevant literature on the reordering and NNC problem. “Quantum annealing” section introduces Quantum Annealing as an optimisation method and the QUBO problem formulation. “Modelling approach” section reviews the ILP formulation for the NNC problem on a linear graph proposed by¹³ and proposes two QUBO formulations of the same problem. “Experimental results” section presents and analyses the results obtained from applying both formulations to instances from the²⁵ dataset and compares those to the results obtained by¹³ and others. Finally, “Conclusions and further research” section concludes and points out ideas for further research. This paper is based on a thesis written by the first author.

Literature

The NNC problem on a linear array has been studied comprehensively. The methods with which the problem has been approached can be subdivided into two groups, exact methods and heuristics. The use of heuristics is motivated since the size of most exact models scales exponentially in the number of qubits^{17,26}.

NNC: exact methods

A naive approach, formalized by²⁶, enumerates all possible permutations of qubits for all gates and then selects the solution that requires the minimum number of inserted SWAP-gates. Having $\mathcal{N}$ qubits and $\mathcal{K}$ gates in a circuit this entails generating all $(\mathcal{N}!)^{\mathcal{K}}$ solutions which is intractable even if the corresponding solution values can be computed efficiently.

Other approaches make use of graph theory by constructing Adjacent Transposition and Caley Graphs as in¹⁵ and²⁷ respectively. Both methods hinge on generating graphs where nodes correspond to qubit orders and subsequently identify a shortest path that corresponds to the solution of the NNC problem. While the shortest path can be found in polynomial time in terms of the size of the graph, the constructed graphs contain $O(\mathcal{N}!\mathcal{K})$ nodes. Therefore, these methods result again in the factorial-time scaling in terms of the number of qubits of the initial NNC problem.

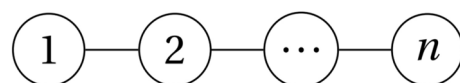


Fig. 1. Linear coupling graph with n qubits. Figure from:¹⁵.

In¹⁷, the authors propose a Pseudo-Boolean Optimisation (PBO) model to solve the NNC problem. The qubit permutations are encoded into binary variables x_{ij}^k , indicating if qubit i is located at location j before gate k . Moreover, the objective of counting the number of required SWAP-gates is then encoded by enumerating all permutations (and their corresponding costs) at each gate. Consequently, this method also fails to avoid the factorial scaling. However, their formulation paved the way for the following Integer Linear Programming (ILP) approach by¹³, which, for the first time, avoids the factorial scaling and produces a polynomial-size model. Ultimately, this ILP formulation will serve as the basis of the two polynomial-size QUBO formulations proposed in this work which will be presented in "Modelling approach" section).

NNC: heuristics

Numerous heuristics approaches have been proposed for the NNC problem. For relatively larger instances, these methods remain the only practical options. The proposed methods include: greedy searches where one qubit is moved toward another²⁸, a greedy heuristic similar to 2-opt²⁹, Sliding Window Search heuristics where only the next fixed number of gates are considered for the reordering^{30,31}, and a Fast Sliding Window Search where, additionally, only a restricted number of qubit orders are considered²⁸. Another Sliding Window approach uses the concept of the circuit tail³², in which only part of the tail is used to determine local optimal solutions, which are later on combined to obtain a complete strategy. Other attempts include the use of templates for circuits in the Multiple Control Toffoli (MTC) library³³. A divide and conquer technique, where the circuit is split into clusters that are considered separately, was developed in³⁴. Also³⁵ gives a decomposition approach. In³⁶, a solution method is proposed, based on a combination of MTC library templates and the heuristic method of³¹.

Quantum annealing

In order to solve the NNC problem, which arises within the context of Gate-based Quantum Computing, this work intends to exploit another form of Quantum Computing, namely Quantum Annealing, as the optimisation method of choice. A detailed explanation of QA, its advantages, limitations, and the quantum mechanics behind it is beyond the scope of this work. This section is limited to a brief description of Quantum Annealing as an optimisation method and a few examples of applications, while providing references for interested readers.

In contrast to Gate-based QC where universal computation is performed by a series of Quantum gates, Quantum Annealing is a hardware heuristic to specifically minimise an Ising Hamiltonian, or equivalently, solve Quadratic Unconstrained Optimisation (QUBO) Problems.

Definition 1 (*QUBO Problem*) Given a vector of n binary decision variables $x \in \{0, 1\}^n$ and a $n \times n$ coefficient matrix Q , the QUBO is defined as:

$$\min_x x^T Q x. \quad (1)$$

Note that $x_i^2 = x_i$. Generally, the QUBO problem is NP-hard³⁷.

The QUBO problem's (from now referred to as QUBO) objective function can equivalently be described by an Ising Hamiltonian (more commonly used in physics)¹⁸. This Hamiltonian effectively encodes the objective value, or energy, for any given solution. The challenge here is to reformulate the initial optimisation problem such that the minimum energy ground state of the Hamiltonian (that is, the QUBO's optimal solution), corresponds to the optimal solution of the original optimisation problem.

Now the optimisation process, as performed by one of the most advanced Quantum Annealers, such as the D-Wave Advantage Systems, can be outlined as follows. Initially, the quantum system in the annealer starts from a simple, well-known Hamiltonian. Then, the initial Hamiltonian is slowly perturbed by applying magnetic fields until the problem Hamiltonian is obtained. If this process was successful, the system remained within the ground state throughout and the optimal solution to the optimisation problem is encoded in the final state of the system³⁸. There are claims of analytical and numerical evidence suggesting that Quantum Annealing can outperform classical methods such as Simulated (Thermal) Annealing for some problems^{39,40}, potentially offering a quadratic speedup^{41,42}. A more comprehensive analysis is given by⁴³.

The Quantum Processing Unit (QPU) of the D-Wave Advantage System has around 5600 qubits which are arranged in a Pegasus Graph structure¹⁸. This QPU graph and the qubits therein are not to be confused with the Gated-based Quantum Computer whose qubit ordering is optimised in the linear NNC problem. Note that the connectivity of the Advantage QPU is (at most) 15, indicating that each qubit is connected to and can interact with 15 others¹⁸. Therefore, to encode the QUBO on the QA, each logical variable in the QUBO is mapped to at least one or more qubits on the QPU. Specifically, multiple qubits are needed to present a single logical variable if that variable is connected to, or interacts with more than 15 others in the QUBO. A variable i is said to (quadratically) interact with another variable j if the corresponding off-diagonal entry in the Q matrix is non-zero. This process of mapping the QUBO onto the QA hardware is called embedding.

While the limited connectivity and number of qubits available pose a hard limit on QUBO instance sizes that can be solved on the QA, there are other factors that adversely impact the annealer's performance as the instance size increases. First, Integrated Control Errors cause the representation of the QUBO on the hardware to be imprecise¹⁸. Secondly, the minimal energy gap between the ground-state and the first-excited state decreases as the problem size increases which in turn increases the probability to move away from the ground-state (optimal solution) during the anneal process⁴⁴.

Finally, the interested reader is referred to¹⁰ for the Quantum Mechanics involved, and to^{18,45,46} for more information on Quantum Annealing as a heuristic optimisation method.

Modelling approach

This section discusses the polynomial-size ILP formulation of the linear NNC problem by¹³ and then proposes one QUBO formulation based on their ILP (referred to as QUBO₁ in the remainder of this paper). Subsequently, another improved QUBO formulation with fewer variables and constraints is proposed afterwards (referred to as QUBO₂).

NNP problem

The ILP formulation of the linear NNC proposed in¹⁵ and the companion paper¹³ serves as the basis of the QUBO formulations proposed in this work. Consequently, their model and some of their key definitions, including a formal definition of the NNC problem on a linear array, are summarised in this and the next section.

- $Q = 1, \dots, n$ denotes the set of n qubits
- $L = 1, \dots, n$ denotes the set of n locations on the linear array which the qubits can occupy
- A permutation $\tau([n]) \in S_n$ is a bijective mapping of qubits to locations, $f: Q \mapsto L$. Here, $[n]$ denotes the vector $(1, \dots, n)$ and S_n denotes the permutation group (of all permutations).
- A SWAP-gate applied to a qubit ordering τ interchanges the location of two adjacent qubits (by swapping the qubit's states or information).
- Given two permutations τ_1, τ_2 , qubits i, j are said to be inverted if $\tau_1(i) < \tau_1(j)$ and $\tau_2(i) > \tau_2(j)$.

The efficient encoding of the objective in the ILP formulation by¹³ builds on recognising that the minimally required number of SWAP-gates to transform one permutation into another is equal to the total number of inversions between the two permutations, or equivalently the Kendall-tau distance:

Definition 2 (Kendall-tau distance) Given two permutations $\tau_1, \tau_2 \in S_n$, the Kendall-tau distance between the two permutations is

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

The Nearest Neighbour Constraints of the NNC problem require that each quantum gate can only act on qubits that are neighbours, or adjacent, to each other in the corresponding topology graph. Since the meaning of adjacency is only well-defined for pairs of qubits, it only makes sense to consider two-qubit gates. Therefore, multiple-qubit gates are decomposed into two-qubit gates as described in¹³. Notably, such a decomposition is always possible¹⁰. Moreover, Nearest Neighbour Constraints do naturally not apply to gates acting on individual qubits, so these gates can be ignored in the context of the NNC problem. Additionally, the function of a gate is not relevant in this context either, but only the two qubits i, j that a gate acts on. Consequently, a two-qubit gate $g = g_{i,j} = \{q_i, q_j\}$ is completely defined for this purpose by specifying these two qubits $q_i, q_j \in Q$. Finally, this allows the definition of a quantum circuit in terms of only two-qubit gates and the subsequent formulation of Nearest Neighbour Compliance:

Definition 3 (Quantum circuit) Let Q be a set of n qubits and $G = (g^1, \dots, g^m)$ a sequence of m two-qubit gates. Then, the tuple $QC = (Q, G)$ is a quantum circuit.

Definition 4 (Nearest Neighbour Compliance) Given a quantum gate $g_{i,j}^t \in G$ and a corresponding qubit order τ^t before that gate, the Nearest Neighbour Constraint requires qubits q_i, q_j to be adjacent, namely $|\tau^t(i) - \tau^t(j)| = 1$, or equivalently $(\tau^t(i) - \tau^t(j))^2 = 1$. Finally, given a sequence of qubit orders, where each qubit order τ^t corresponds to a quantum gate $g_{i,j}^t$ in the gate sequence G of the QC, the circuit QC is compliant with Nearest Neighbour Constraints if the above constraint holds for each pair of qubit order and gate $(\tau^t, g_{i,j}^t)$.

Ultimately, the formal definition of the NNC problem is as follows: Given a quantum circuit $QC = (Q, G)$ with $|Q| = n$ qubits and $|G| = m$ gates, we have to find a sequence of qubit orders $\tau = (\tau^1, \dots, \tau^m)$, one for each gate $g \in G$, that minimises the sum of Kendall-tau distances between consecutive qubit orderings

$$\sum_{t=1}^{m-1} I(\tau^t, \tau^{t+1}),$$

such that the quantum circuit is compliant with Nearest Neighbour Constraints. Note that the NNC problem on a linear graph is conjectured to be NP-hard²⁶, while the NNC problem on a general graph is a known NP-hard problem⁴⁷.

Building a first QUBO

In this subsection we will introduce the variables and build QUBO₁ based on the objective functions and the constraints as introduced in¹³.

Variables

To work of¹³ introduces the following variables:

$$x_i^t \in L = \{1, \dots, n\} \text{ indicates the location of qubit } i \text{ at gate } t. \quad (3)$$

$$y_{ij}^t = \begin{cases} 1 & \text{if qubit } i \text{ is located before qubit } j \text{ at gate } t: x_i^t < x_j^t, \\ 0 & \text{otherwise.} \end{cases} \quad (4)$$

Note that the y_{ij}^t variables keep track of the relative ordering of qubits and that $|y_{ij}^t - y_{ij}^{t+1}|$ determines whether qubit i and j are inverted between permutation t and $t + 1$. This key observation allows¹³ to count the total number of SWAP-gates while avoiding the explicit $n!$ scaling of the model in the number of variables and constraints. Moreover, describing an ordering of qubits by these y_{ij}^t variables, while efficiently being able to count the required SWAP-gates, will be the enabling observation for the significantly more efficient QUBO₂ introduced later in this section.

Objective function

Clearly, the objective is to minimise the sum of Kendall-tau distances between every two consecutive qubit orders, that is:

$$\min \sum_{\substack{i, j \in Q \\ i < j}} \sum_{t \in [m-1]} |y_{ij}^t - y_{ij}^{t+1}|. \quad (5)$$

This function is neither linear nor quadratic and thus had to be linearised by¹³ for their ILP. However, since the variables are binary and the QUBO allows quadratic interactions, we can equivalently write:

$$\min \sum_{\substack{i, j \in Q \\ i < j}} \sum_{t \in [m-1]} (y_{ij}^t - y_{ij}^{t+1})^2. \quad (6)$$

This expression will serve as the objective for both QUBO₁ and QUBO₂.

Ordering constraints

The following constraints relate the x -variables to the y -variables by enforcing the definition of the y -variables. Moreover, they also ensure that no two qubits can be located at the same location at the same time. For more details, see¹³.

$$x_j^t - x_i^t \leq M y_{ij}^t - 1 \quad \forall i, j \in Q, i < j, t \in [m], \quad (7)$$

$$x_i^t - x_j^t \leq M(1 - y_{ij}^t) - 1 \quad \forall i, j \in Q, i < j, t \in [m], \quad (8)$$

where $M = n + 1$. Due to the QUBO being an unconstrained problem, all constraints have to be incorporated into the objective function. This can be done by adding additional terms to the objective that either penalise infeasible solutions or favor feasible solutions by increasing or decreasing the objective value respectively.

In QUBO₁, all penalty terms are constructed such that they are precisely zero if the corresponding constraint is satisfied and strictly positive if they are not satisfied. The following penalty term (added to the initial objective) achieves exactly that:

$$+\lambda^o \cdot (x_j^t - x_i^t - M y_{ij}^t + 1 + s_{ij}^t)^2 \quad \forall i, j \in Q, i < j, t \in [m], \quad (9)$$

where $\lambda^o > 0$ is a constant and s_{ij}^t represents an integer-valued slack variable that can take on values in $[0, M - 2]$. It can easily be verified that there only exists a value for $s_{ij}^t \in [0, M - 2]$ that sets this term to zero if both Eq. (7) and Eq. (8) are satisfied. Finally, to encode the integer variable $s_{ij}^t \in [0, M - 2]$ in binary variables (as the QUBO requires), we can do as Section 2.4 of³⁷ and write:

$$s \mapsto \sum_{k=0}^{K-1} 2^k a_k + (U + 1 - 2^K) a^K, \quad (10)$$

using $K + 1$ binary variables a_k , where U is the maximum value that s can take, such that $2^K \leq U < 2^{K+1}$. The same holds for the encoding of the x_i^t -variable. Throughout the remainder of this paper, we will refrain from explicitly writing out this encoding when specifying (parts of) a QUBO for the sake of readability. Therefore, whenever an integer variable appears in a QUBO formulation, this encoding is implied.

Nearest neighbour constraints

To ensure that the Nearest Neighbour Constraint is satisfied at each gate¹³, also added the following constraints to their ILP:

$$-1 \leq x_i^t - x_j^t \leq 1 \quad \forall g_{ij}^t \in G. \quad (11)$$

The corresponding penalty terms in QUBO₁ are:

$$+\lambda^{nn} \cdot ((x_i^t - x_j^t)^2 - 1) \quad \forall g_{ij}^t \in G, \quad (12)$$

where $\lambda^{nn} > 0$. Note that this expression favours $x_i^t = x_j^t$, which implies that qubits i and j are located in the same position in the ordering at gate t , corresponding to an infeasible solution. However, the penalty terms of Eq. (9) penalise this behaviour. Therefore, we can resolve this issue by requiring $\lambda^o > \lambda^{nn}$.

An improved QUBO formulation

The QUBO₁ formulation introduces a lot of additional binary variables to model the slack and x -variables required in Eq. (9). This is undesirable due to the limited number of 5600 qubits available in the D-Wave Quantum Annealer to embed the QUBO and its variables. QUBO₂ reformulates the NNC problem only in terms of the y -variables to eliminate both the x -variables and the need to introduce slack variables as in Eq. (9).

For this, observe that x_i^t , the location of qubit i in the ordering at gate t , can be obtained by counting how many other qubits are located before qubit i , specifically:

$$x_i^t = \sum_{k=1}^n y_{ki}^t. \quad (13)$$

Since y_{ij}^t -variables are only defined for $i < j$, we need to write Eq. (13) more precisely as:

$$x_i^t = \sum_{k=1}^{i-1} y_{ki}^t + \sum_{k=i+1}^n (1 - y_{ik}^t). \quad (14)$$

Nonetheless, for the sake of readability, the notation of Eq. (13) will be used in the remainder of this paper.

Finally, recognising this, we can write QUBO₂ only in terms of the y variables. The initial objective function in Eq. (6) is unaffected. However, we need to specify a partially new set of constraints. Firstly, we need to ensure that the ordering of qubits described by the y -variables is feasible, i.e., each qubit takes exactly one location. Secondly, we need to specify the Nearest Neighbour Constraints as before.

Observe that the requirement of each qubit to take exactly one location is equivalent to requiring that any two qubits can not be located in the same location. In terms of the y -variables this can be expressed by requiring that no two qubits have an equal number of other qubits appearing before them in the ordering, that is:

$$\sum_{k=1}^n y_{ki}^t \neq \sum_{k=1}^n y_{kj}^t \quad \forall i, j \in Q, i \neq j, t \in [m]. \quad (15)$$

To incorporate these constraints into the objective of QUBO₂ without introducing slack variables, we take a different approach than previously. Rather than penalising infeasible solutions, we favour feasible solutions. In other words, we add additional terms to the objective that are strictly negative if Eq. (15) is satisfied and strictly zero if not. Adding the following terms achieves this:

$$-\lambda^o \cdot \left(\sum_{k=1}^n y_{ki}^t - \sum_{k=1}^n y_{kj}^t \right)^2 \quad \forall i, j \in Q, i \neq j, t \in [m]. \quad (16)$$

In QUBO₂, all added terms are zero for infeasible constraints, while taking on strictly negative values to favor feasible solutions.

Note that each individual term, corresponding to a specific constraint, is not always the same for all feasible solutions. However, it can be shown that the sum of all added terms (all constraints), needs to be identical for any feasible solution:

Proof Let τ be a feasible ordering of n qubits (permutation). Then, the location of the first qubit i and the last qubit j differs by $n - 1$. Moreover, there are exactly two pairs of qubits whose location differs by $n - 2$, three pairs differing by $n - 3$, and so on. Formally, the following holds for any feasible solution:

$$|\{(i, j) | 1 \leq i \neq j \leq n, |\tau^t(i) - \tau^t(j)| = k\}| = n - k \quad \forall k \in [n - 1], t \in [m]. \quad (17)$$

Finally, as $\tau^t(i) = \sum_{k=1}^n y_{ki}^t$,

$$\sum_{i,j \in [n], i \neq j} |\tau^t(i) - \tau^t(j)|^2 = \sum_{i,j \in [n], i \neq j} \left(\sum_{k=1}^n y_{ki}^t - \sum_{k=1}^n y_{kj}^t \right)^2 = \sum_{k=1}^{n-1} (n-k)^2 \quad \forall t \in [m]. \quad (18)$$

This is necessary to ensure that the added terms do not affect the ranking of the feasible solutions. In other words, we can only guarantee that the optimal solution to the NNC problem will correspond to the optimal solution of the QUBO if the sum of added terms is identical for any feasible solution of the NNC problem.

Finally, the Nearest Neighbour Constraints are logically identical with Eq. (12) from QUBO₁, we substitute the x -variables by Eq. (13):

$$+ \lambda^{nm} \cdot \left(\left(\sum_{k=1}^n y_{ki}^t - \sum_{k=1}^n y_{kj}^t \right)^2 - 1 \right) \quad \forall g_{ij}^t \in G, \quad (19)$$

where $\lambda^o > \lambda^{nm} > 0$.

Experimental results

This section analyses and discusses the results obtained from applying the proposed QUBO formulation to circuits from the revLib library²⁵ and others, in order to address the remaining research questions. The results are compared to the exact method of¹³ in terms of the time needed to obtain them and to other heuristics in terms of solution quality. With consideration of the limitations of the D-Wave Advantage System mentioned before, it is also of interest for which instance sizes solutions can be obtained and at which point the solution quality starts to decrease significantly.

Experimental setting

We use various instances from different sources. Quantum Fourier Transform (QFT) instances were obtained from¹⁰, all other instances from the RevLib Library containing reversible circuits²⁵. Moreover, circuits with multi-qubit gates are decomposed into two-qubit gates as described in¹³. Notably, this process is not optimised over and can affect the required number of SWAP gates.

The Quantum Annealer *D-Wave Advantage 5.3* by D-Wave is used to perform the Quantum Annealing, while a *Dell XPS 13 2-in-1 7390* with an *Intel i7-1065G7* with 4 cores at 1.30GHz and 16GB of RAM is used to do the pre-processing such as computing the embedding of the QUBO on the Annealer.

The parameters of the QUBOs were determined experimentally by extensive trial-and-error testing. The λ parameters determine how the constraints' contributions to the QUBO-objective are weighted relative to the initial objective. Choosing values too small for this parameter leads to sampling infeasible solutions, while setting values too high results in problem misrepresentation due to the limited range and fidelity with which the quantum annealer can represent the problem coefficients (see "Error Sources for Problem Representation" in¹⁸). It was found that setting $\lambda^o = 2$ and $\lambda^{nm} = 1$ for QUBO₁ performed relatively well. For QUBO₂, where feasible solutions could be obtained for a larger number of instances, it was found that $\lambda^o \in [0.2, 0.7]$ and $\lambda^{nm} = \lambda^o - 0.01$ yielded the best results. Notably, as the number of gates m and the instance size increases, larger values in this range were found to perform better on average.

For the Quantum Annealer, there are two parameters that were manually tuned by trial-and-error. First, the *annealing time*, which determines the length of the annealing process, was chosen between 100 to 450 microseconds. Second, the *number of samples* parameter determines how many solutions are sampled by the Quantum Annealer. Chosen values were between 100 and 1250 samples. Obviously, in both cases, larger values correspond to longer compute times on the Quantum Annealer. In addition, the improvements in the best objective value among samples diminished as the number of obtained samples increased. Thus, exceeding 1250 samples showed close to no improvements in the obtained solutions. Note that we generally set the *number of samples* parameter lower for small instances, since this was sufficient to find the optimal solution. Moreover, since adiabatic theory indicates that the larger problems require longer annealing times to obtain good solutions⁴⁴, we use a higher value for larger instances. Moreover, setting larger values for the *annealing time* generally increased the likelihood to obtain better solutions, again with diminishing returns. Again, it was found that larger values within the given ranges produced relatively better results, especially for larger instances. Nevertheless, the lower-end values were sufficient for small instances to obtain the optimal solution. The specific parameter configurations that were used to obtain the presented results for each instance are given in Table 1.

Lastly, we use the heuristic tool minorminer⁴⁸ to find an embedding for the problem graph in the hardware graph. To find a feasible embedding, it might be necessary to represent a single logical variable by a chain of qubits on the hardware. If a sampled solution violates the chain, i.e. not all variables in the chain take on the same value, we resolve these conflicts by a majority-vote mechanism where the more often occurring value is chosen for the logical variable.

Results

The feasibility of all solutions was validated by checking that all penalty terms (corresponding to the constraints) indeed do evaluate to zero. QUBO₁ failed to obtain feasible solutions for all instances except of *QFT_QFT3* and *peres_8*, where it did not obtain the optimal solution. A possible explanation of the poor performance is the inefficient encoding of the qubit permutations by the integer x -variables and the inefficient encoding of the constraints via slack variables that might significantly decrease the minimal energy gap. The hugely improved

Benchmarks	λ^o	NumReads	Annealing time (μ s)
4mod5-v0_19, alu-v0_27, alu-v1_29, alu-v2_33, mod5mils_65, QFT_QFT5, ex1_226, graycode6_47, graycode6_48, xor5_254	0.19	1250	450
4gt11-v1_85, 4gt11_83, 4gt11_84, 4mod5-v0_20, 4mod5-v1_22, 4mod5-v1_24, 4mod5-v1_25, mod5d1_63, mod5mils_71	0.19	250	250
3_17_13, 3_17_14, 3_17_15, ex-1_166, fredkin_5, fredkin_6, fredkin_7, ham3_102, ham3_103, miller_11, miller_12, peres_10, peres_8, peres_9, QFT_QFT3, toffoli_1, toffoli_2, decod24-v0_38, decod24-v0_39, decod24-v0_40, decod24-v1_42, decod24-v2_43, decod24-v2_44, decod24-v3_46, QFT_QFT4, rd32-v0_66, rd32-v0_67, rd32-v1_68, rd32-v1_69, toffoli_double_3, toffoli_double_4	0.25	150	150
4gt11_82, 4gt13-v1_93, 4gt13_92, 4mod5-v0_21, alu-v1_28, alu-v3_34, alu-v3_35, alu-v4_37, mod5d2_64, mod5d2_70, rd32_272, 4gt13_91, 4gt5_75, 4gt5_76, 4gt5_77, 4mod5-v0_18, 4mod5-v1_23, alu-v0_26, alu-v4_36, one-two-three-v2_100, one-two-three-v3_101, rd32_271	0.25	1250	450
4_49_17, aj-e11_168, decod24-v1_41, decod24-v3_45, hwb4_52, 4_49_16, aj-e11_165, mod10_171, mod10_176	0.35	1250	450
4gt10-v1_81, one-two-three-v1_99, 4mod5-bdd_287, alu-bdd_288, decod24-bdd_294, decod24-enable_124, decod24-enable_125, QFT_QFT6, QFT_QFT7	0.35	1250	450
hwb4_49, hwb4_50, hwb4_51, mini-alu_167	0.50	1250	450
4gt4-v0_72, 4gt4-v0_73, 4gt4-v1_74, 4gt4-v0_78, 4gt4-v0_79, 4gt4-v0_80, 4gt12-v0_87, 4gt12-v0_86, 4gt12-v1_89, 4gt13_90, mod8-10_178, one-two-three-v0_97, mod8-10_177, alu-v2_31, hwb5_55, rd32_273, alu-v2_30, 4gt12-v0_88, 4mod7-v0_94, 4mod7-v0_95, 4mod7-v1_96, alu-v2_32, one-two-three-v0_98 sf_275	0.70	1250	450

Table 1. Parameter settings used for the various circuit instances.

performance of QUBO₂, which encodes the problem only through the y -variables, provides anecdotal evidence for these hypotheses.

Consequently, only for QUBO₂ results are reported. The benchmark instances are grouped according to the number of qubits in the circuits, n , over multiple tables, starting with Table 2 for $n = 3, 4$, Tables 3 and 4 for $n = 5$ and Table 5 for $n = 6 - 10$. Moreover, tables are sorted by the number of two-qubit gates in the decomposed circuits, $|G| = m$. In each table, column *OPT* states the optimal minimum number of SWAP gates needed to make the circuit compliant (from¹³, assuming the decomposition procedure used there), column *SWAPSQ* presents the objective value obtained through a single run of QUBO₂ on the D-Wave Quantum Annealer, column *QUBO Time* shows the time to compute the QUBO expression, column *EmbedT* shows the time to (heuristically) find an embedding of the QUBO on the hardware graph of the annealer using “minorminer.find_embedding()” with default parameters⁴⁸, and column *QPU Time* shows the time to perform the actual annealing on the QPU following the definition of the *QPU access time* in¹⁸ (which varies depending on the NumReads and Annealing Time parameters). Note that all classical timings refer to elapsed time using a single core. The *TimeQ* column shows the overall time of the quantum algorithm (i.e. $TimeQ = QUBO\ Time + EmbedT + QPU\ Time$), column *TimeE* shows the running time of the exact method of¹³, and column *SWAPSH* shows the objective values obtained by other heuristic approaches. If *SWAPSQ* is reported to be -2 , this indicates that no feasible solutions could be obtained, while -3 indicates that no embedding was found by 10 attempts (the default setting) of the heuristic tool minorminer⁴⁸, which usually implies that the instance is too large to fit on the Annealer (yet there is no proof due to heuristic nature of minorminer). Consequently, a dash “-” in the *TimeQ* or *QPU Time* columns is the result of the fact that no feasible embedding was found and thus the problem could not be run on the annealer. Subscripts in the *SWAPSH* column indicate the source of the results, where: c: Kole et al.⁴⁹, d: Shafaei et al.⁵⁰, e: AlFailakawi et al.⁵¹, f: Kole et al.³¹ and g: Wille et al.³², while a dash “-” indicates that neither of c-g reported a result for this instance. Asterisks in column *SWAPSH* indicate that the number of gates after decomposition differs, or that the objective value of the heuristic is lower than that of the exact method. This is believed to be the consequence of decomposing multi-qubit gates differently.

Discussion on computation time

The time to perform the Quantum Annealing and to obtain a given number of samples, *QPU Time*, depends on the aforementioned parameter values *annealing time* and *number of samples*, rather than on the instance size. The choice of these parameters is instance-size-dependent within the practical upper bounds discussed previously. The *QPU Time* peaks at around 0.9 seconds and does not further increase even for large instances. Also the *QUBO time* is in almost all cases lower than 1 second. This means that for the larger instances the *Embedding time* is by far the dominating factor.

As concluded, *QPU Time* and *QUBO time* are in the same order of magnitude as the run time of the exact method for small instances and significantly smaller for larger instances ($n = 4$ and $m > 30$ or $n = 5$ and $m > 20$). Thus, Quantum Annealing as a heuristic has the potential to significantly reduce the running time to obtain solutions for large instances if the embedding can be computed efficiently or if no embedding is required at all. The latter would be true, for example, if the hardware graph of the Quantum Annealer has a fully-connected subgraph of sufficient size. Alternatively, one could pre-compute the embedding of the largest fully-connected graph that can be embedded on the hardware in advance, and use this embedding when possible.

However, in practice, this would limit the maximum number of variables too significantly to be a viable solution and the computation of the embedding can not be avoided. As the results indicate, the time to (heuristically) find an embedding exceeds the run time of the exact method for almost all instances with five or more qubits. Therefore, the limited connectivity of the hardware graph and the computational task to find an embedding pose two major limitations to the proposed method.

Finally, the authors want to note that the reported compute times for the computations that were performed on classical computers highly depend on the hardware configuration of the system and therefore may not be directly comparable to other reported run times.

Benchmark	n	m	OPT	SWAPSQ	QUBO Time	EmbedT	QPU Time	TimeQ	TimeE	SWAPSH
QFT_QFT3	3	3	1	1	0.00	0.05	0.05	0.10	0.02	—
peres_10	3	4	1	1	0.01	0.06	0.05	0.12	0.14	—
peres_8	3	4	1	1	0.01	0.05	0.05	0.11	0.06	—
toffoli_2	3	5	1	1	0.01	0.06	0.05	0.12	0.12	—
toffoli_1	3	5	1	1	0.01	0.10	0.05	0.16	0.10	—
peres_9	3	6	1	1	0.01	0.08	0.05	0.14	0.02	—
fredkin_7	3	7	1	1	0.02	0.09	0.06	0.17	0.16	—
ex-1_166	3	7	2	2	0.01	0.07	0.05	0.13	0.08	—
fredkin_5	3	7	1	1	0.01	0.09	0.05	0.15	0.15	—
ham3_103	3	8	2	2	0.01	0.05	0.05	0.11	0.04	—
miller_12	3	8	2	2	0.02	0.46	0.05	0.53	0.14	—
ham3_102	3	9	1	1	0.01	0.09	0.05	0.15	0.05	—
3_17_15	3	9	2	2	0.01	0.09	0.05	0.15	0.04	—
3_17_13	3	13	3	3	0.02	0.14	0.05	0.21	0.12	$4_c^*, 4_d, 3_e, 6_g$
3_17_14	3	13	3	3	0.02	0.15	0.05	0.22	0.15	—
fredkin_6	3	15	3	3	0.02	0.17	0.06	0.25	0.06	—
miller_11	3	17	4	4	0.02	0.17	0.05	0.24	0.15	—
QFT_QFT4	4	6	3	3	0.02	0.25	0.06	0.33	0.17	—
toffoli_double_3	4	7	1	1	0.02	0.24	0.05	0.31	0.11	—
rd32-v1_69	4	8	2	2	0.03	0.37	0.05	0.45	0.16	—
decod24-v1_42	4	8	2	2	0.03	0.27	0.05	0.35	0.12	—
rd32-v0_67	4	8	2	2	0.03	0.32	0.05	0.40	0.07	$2_c, 2_d$
decod24-v2_44	4	8	3	3	0.04	0.26	0.05	0.35	0.07	—
decod24-v0_40	4	8	3	3	0.03	0.25	0.06	0.34	0.06	—
decod24-v3_46	4	9	3	3	0.04	0.34	0.06	0.44	0.09	$3_c, 3_d$
toffoli_double_4	4	10	2	2	0.04	0.34	0.05	0.43	0.07	—
rd32-v1_68	4	12	3	3	0.05	0.55	0.05	0.65	0.24	—
rd32-v0_66	4	12	3	3	0.04	0.50	0.06	0.60	0.09	—
decod24-v0_39	4	15	5	5	0.04	0.59	0.05	0.68	0.53	—
decod24-v2_43	4	16	5	5	0.05	0.54	0.06	0.65	0.23	—
decod24-v0_38	4	17	4	4	0.07	0.95	0.06	1.08	0.57	—
decod24-v1_41	4	21	7	7	0.08	0.82	0.72	1.62	0.50	—
hwb4_52	4	23	8	8	0.11	1.44	0.71	2.26	0.97	$9_c, 10_d, 9_e, 9_f$
aj-e11_168	4	29	12	13	0.10	1.16	0.79	2.05	5.36	—
4_49_17	4	30	12	12	0.14	1.16	0.73	2.03	6.10	$12_c^*, 12_d, 16_e$
decod24-v3_45	4	32	13	15	0.10	2.23	0.78	3.11	6.25	—
mod10_176	4	42	15	19	0.16	6.15	0.76	7.07	7.94	—
aj-e11_165	4	44	18	26	0.17	2.34	0.72	3.23	9.36	$36_d, 33_g^*$
mod10_171	4	57	24	33	0.16	2.88	0.75	3.79	27.18	—
4_49_16	4	59	22	34	0.22	2.92	0.79	3.93	24.23	—
mini-alu_167	4	62	27	43	0.21	20.30	0.82	21.33	23.70	—
hwb4_50	4	63	23	35	0.23	9.63	0.83	10.69	17.61	—
hwb4_49	4	65	23	37	0.22	5.39	0.81	6.42	21.64	—
hwb4_51	4	75	28	64	0.25	6.25	0.82	7.32	75.09	—

Table 2. Benchmark circuit instances with three or four qubits.

Discussion on solution quality

All instances with three qubits were solved to optimality in run times comparable to the exact method of¹³. The same holds for four-qubit instances with at most 23 gates. For instances with 29 to 42 gates, the deviation from optimality remains small with at most four additional SWAP-gates. Then, the deviation rises to around 50% and peaks at 100% of OPT for the largest four-qubit instance with 75 gates. Nonetheless, it can be argued that it is more insightful to compare the total number of gates in the circuit, m +SWAPSQ, to the total minimum m +OPT. In this case, the relative deviation rises first to 20% and then peaks at 50%. Figure 2 illustrates this measure for the majority of instances.

Benchmark	n	m	OPT	SWAPSQ	QUBO Time	EmbedT	QPU Time	TimeQ	TimeE	SWAPSH
4mod5-v1_25	5	7	1	1	0.11	1.04	0.12	1.27	0.26	—
4gt11_84	5	7	1	1	0.06	1.02	0.11	1.19	0.06	$1_c, 1_d, 1_e$
4gt11-v1_85	5	7	1	1	0.17	1.62	0.11	1.90	0.09	—
4mod5-v0_20	5	8	2	2	0.08	0.97	0.11	1.16	0.08	—
4mod5-v1_22	5	9	1	1	0.11	0.94	0.10	1.15	0.08	—
QFT_QFT5	5	10	6	8	0.12	1.17	0.68	1.97	0.41	$7_c, 6_d$
mod5d1_63	5	11	2	2	0.09	1.63	0.10	1.82	0.12	—
4mod5-v0_19	5	12	3	3	0.15	1.11	0.80	2.06	0.84	—
4gt11_83	5	12	3	3	0.14	3.02	0.11	3.27	0.15	—
4mod5-v1_24	5	12	3	3	0.13	1.73	0.12	1.98	0.28	—
mod5mils_65	5	12	4	4	0.12	2.55	0.75	3.42	0.26	—
mod5mils_71	5	12	2	2	0.14	1.75	0.11	2.00	0.15	—
alu-v2_33	5	13	4	4	0.16	2.02	0.78	2.96	0.45	—
alu-v1_29	5	13	4	4	0.13	1.73	0.76	2.62	0.61	—
alu-v0_27	5	13	4	5	0.15	3.23	0.70	4.08	0.48	—
mod5d2_70	5	14	5	5	0.16	0.99	0.73	1.88	0.43	—
alu-v3_35	5	14	5	6	0.14	1.89	0.78	2.81	0.38	—
alu-v4_37	5	14	5	6	0.15	4.72	0.73	5.60	0.37	—
alu-v1_28	5	14	4	4	0.16	1.75	0.74	2.65	0.26	—
4gt13-v1_93	5	15	5	7	0.14	4.73	0.77	5.64	0.69	$7_c^*, 6_d, 4_e^*$
4gt13_92	5	15	6	8	0.17	2.02	0.74	2.93	0.53	—
4gt11_82	5	16	6	8	0.15	3.32	0.75	4.22	0.89	—
4mod5-v0_21	5	17	8	9	0.17	2.67	0.74	3.58	2.84	—
rd32_272	5	18	7	8	0.18	3.68	0.80	4.66	0.94	—
alu-v3_34	5	18	4	5	0.16	3.65	0.74	4.55	0.40	—
mod5d2_64	5	19	6	13	0.22	1.62	0.78	2.62	1.81	—
alu-v0_26	5	21	8	12	0.17	11.54	0.74	12.45	3.56	—
4gt5_75	5	21	6	11	0.18	2.06	0.74	2.98	1.10	$9_c^*, 12_d$
4mod5-v0_18	5	23	8	13	0.20	5.02	0.82	6.04	3.35	—
4mod5-v1_23	5	24	9	16	0.19	4.29	0.80	5.28	5.06	$9_c, 9_d, 15_e$
one-two-three-v2_100	5	24	7	15	0.24	4.81	0.80	5.85	5.37	—
one-two-three-v3_101	5	24	7	11	0.16	7.04	0.78	7.98	2.96	—
rd32_271	5	26	11	24	0.22	3.33	0.78	4.33	7.37	—
4gt5_77	5	28	10	21	0.26	6.68	0.79	7.73	6.20	—
4gt5_76	5	29	10	19	0.21	6.48	0.84	7.53	5.45	—
alu-v4_36	5	30	9	24	0.27	8.60	0.83	9.70	6.34	$15_c^*, 18_d, 17_e$
4gt13_91	5	30	8	32	0.24	5.15	0.81	6.20	4.46	—
4gt13_90	5	34	12	61	0.60	3.15	0.82	4.57	6.77	—
4gt10-v1_81	5	34	13	50	0.34	23.97	0.82	25.13	12.38	$18_c^*, 20_d, 16_e, 24_g^*$
one-two-three-v1_99	5	36	15	45	0.62	16.11	0.84	17.57	17.27	—
4gt4-v0_80	5	36	19	84	0.39	10.37	0.82	11.58	43.45	$34_d, 33_f$
4mod7-v0_94	5	38	12	86	0.33	2.36	0.75	3.44	12.83	—
alu-v2_32	5	38	16	94	0.38	6.70	0.83	7.91	22.05	—
4mod7-v0_95	5	38	14	84	0.48	12.24	0.82	13.54	14.59	$19_c^*, 21_d, 22_e$
4mod7-v1_96	5	38	14	86	0.33	11.01	0.87	12.21	13.49	—

Table 3. Benchmark circuit instances with five qubits and less than 40 gates.

Tables 3 and 4 show the results for the benchmark instances with 5 qubits. A similar trend as previously can be identified here. For smaller values of m , an optimal or near-optimal solution is obtained by the proposed method, while for larger values of m the obtained objective values start diverging. This divergence again starts at around 20 gates, but is significantly more pronounced for instances with 5 qubits. Nonetheless, up until 24 gates the obtained results are comparable to other heuristic solutions that were reported. However, for the largest five-qubit instance for which a feasible solution was obtained, *4gt12-v1_89* with 44 gates, the obtained number of SWAP-gates is more than five times the optimum and more than four times the best heuristic value. For instances with more gates, no feasible solutions could be obtained.

Benchmark	n	m	OPT	SWAPSQ	QUBO Time	EmbedT	QPU Time	TimeQ	TimeE	SWAPSH
one-two-three-v0_98	5	40	15	81	0.40	10.28	0.82	11.50	15.67	—
4gt12-v0_88	5	41	20	87	0.40	17.67	0.83	18.90	34.01	—
4gt12-v1_89	5	44	22	116	0.74	53.92	0.82	55.48	52.36	$35_d, 26_e, 32_f$
sf_275	5	46	18	—2	0.45	14.39	0.80	15.64	21.42	—
4gt4-v0_79	5	49	22	—2	1.94	12.75	0.84	15.53	80.16	—
4gt4-v0_78	5	53	26	—2	0.21	11.18	0.75	12.14	167.03	—
4gt4-v0_72	5	53	24	—2	0.21	12.13	0.74	13.08	49.70	—
4gt12-v0_87	5	54	22	—2	0.22	28.55	0.75	29.52	45.88	—
4gt4-v1_74	5	57	29	—2	0.23	26.05	0.75	27.03	84.87	—
4gt12-v0_86	5	58	26	—2	0.42	26.04	0.76	27.22	108.35	—
mod8-10_178	5	68	37	—2	0.46	74.41	0.75	75.62	389.47	—
one-two-three-v0_97	5	71	32	—2	0.51	16.25	0.75	17.51	76.80	—
4gt4-v0_73	5	89	40	—2	0.59	88.54	0.76	89.89	699.65	—
mod8-10_177	5	93	48	—2	0.59	64.22	0.76	65.57	3650.26	72_d
alu-v2_31	5	100	49	—2	0.64	62.95	0.76	64.35	2906.35	—
hwb5_55	5	101	48	—2	0.67	239.71	0.76	241.14	2264.00	$59_e, 63_d, 60_e, 66_g$
rd32_273	5	104	50	—2	0.71	130.15	0.76	131.62	4631.70	—
alu-v2_30	5	112	55	—2	0.76	162.17	0.76	163.69	13558.87	—

Table 4. Benchmark circuit instances with five qubits and more than 40 gates.

Benchmark	n	m	OPT	SWAPSQ	QUBO Time	EmbedT	QPU Time	TimeQ	TimeE	SWAPSH
graycode6_47	6	5	0	0	0.13	2.17	0.72	3.02	0.02	—
graycode6_48	6	5	0	2	0.12	1.77	0.74	2.63	0.02	—
QFT_QFT6	6	15	11	65	0.57	12.75	0.83	14.15	7.43	$11_c, 12_d$
decod24-enable_124	6	21	5	64	0.47	25.56	0.84	26.87	1.86	—
decod24-enable_125	6	21	5	71	0.63	13.66	0.82	15.11	1.83	—
decod24-bdd_294	6	24	7	—2	0.87	7.34	0.84	9.05	9.37	—
mod5adder_129	6	71	34	—3	1.03	313.86	—	—	534.38	—
mod5adder_128	6	77	36	—3	1.11	1006.92	—	—	1103.51	$45_c^*, 51_d, 46_g^*$
decod24-enable_126	6	86	37	—3	1.11	1776.86	—	—	1954.28	—
xor5_254	7	5	3	8	0.32	3.44	0.70	4.46	0.61	—
ex1_226	7	5	3	4	0.17	4.51	0.80	5.48	0.25	—
QFT_QFT7	7	21	16	—2	1.12	42.98	0.86	44.96	28.26	$28_c, 26_d, 18_g$
4mod5-bdd_287	7	23	7	—2	0.94	32.21	0.86	34.01	4.30	—
alu-bdd_288	7	28	8	—2	1.51	115.55	0.88	117.94	20.65	—
ham7_106	7	49	28	—3	1.17	1804.92	—	—	495.43	—
ham7_105	7	65	34	—3	0.95	1010.51	—	—	1613.33	—
ham7_104	7	83	42	—3	2.10	1012.42	—	—	3238.82	56_c^*
QFT_QFT8	8	28	23	—3	1.24	1010.09	—	—	334.60	$32_c, 33_d, 31_g$
rd53_139	8	36	11	—3	1.56	1010.33	—	—	76.29	—
rd53_138	8	44	11	—3	1.94	1019.52	—	—	100.86	—
rd53_137	8	66	35	—3	2.89	1026.47	—	—	6271.11	—
QFT_QFT9	9	36	30	—3	2.69	1470.68	—	—	1482.53	$52_c, 54_d, 49_g$
QFT_QFT10	10	45	39	—3	5.15	2484.26	—	—	39594.99	64_g
mini_alu_305	10	57	23	—3	5.37	1028.15	—	—	1711.75	—
sys6-v0_144	10	62	19	—3	8.82	1057.60	—	—	887.71	—
rd73_141	10	64	21	—3	5.49	1016.72	—	—	845.05	—
parity_247	18	16	14	—3	25.77	1062.65	—	—	5762.29	—

Table 5. Benchmark circuit instances with six or more qubits.

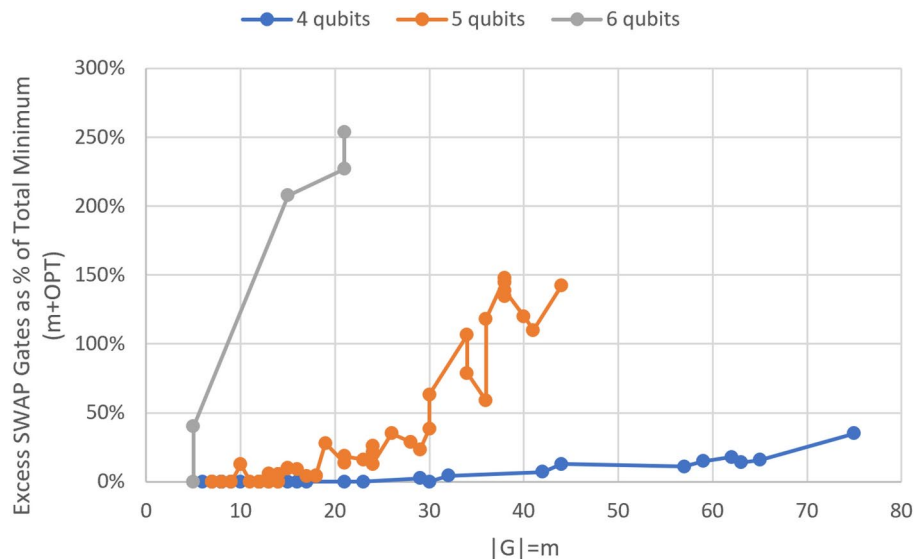


Fig. 2. For circuit instances with 4,5, and 6 qubits respectively, the y-axis plots the difference between the solution's objective value obtained by our method and the optimal objective value ($SWAPSQ - OPT$), as a percentage of the minimum number of gates to implement a given circuit instance ($m + OPT$), against the number of gates m in the non-compliant circuit instance on the x-axis.

For benchmark instances with 6 or more qubits, this trend continues again more extremely, while also starting earlier at instances with as little as 15 gates. Moreover, for six-qubit instances with more than 71 gates and seven-qubit instances with more than 49 gates, it was not possible anymore to find an embedding due to the size of the resulting QUBOs.

To compare these trends graphically, panel (A) of Fig. 3 plots the absolute deviation from OPT (excess) against the number of gates m , for instances with 4, 5, and 6 qubits. Moreover, it is expected that these trends become more pronounced as the number of qubits n increases since the number of possible permutations at each gate scales with $n!$. Similarly, the number of variables in $QUBO_2$ is $n(n-1)/2 \cdot m$ and therefore scales with n^2 . Consequently, it is better to compare instances with the same number of QUBO variables rather than with the same number of gates if the number of qubits differs. Therefore, panel (B) plots the excess number of SWAP gates against the number of variables in $QUBO_2$. While the previously observed differences in the trends of 4, 5, and 6 qubits are now less pronounced, they still do show clearly. They indicate that even for instances, for which the size of the resulting QUBO is identical in terms of the number of variables, the instance corresponding to a circuit with more qubits is more difficult to solve.

In order to better understand why this is the case, consider the two instances *decod24-v0_39*, in Table 2 and *QFT_QFT5*, in Table 3. While the former is a circuit with $n = 4$ qubits and $m = 15$ gates, and the latter is a circuit with $n = 5$ qubits and $m = 10$ gates, the QUBOs of both instances have 90 y -variables. However, there are $n(n-1)/2 y_{ij}^t$ -variables for each gate t , which will be (almost) fully connected due cross-terms from the ordering constraints, while only one y_{ij}^{t+1} -variable at gate t will be connected to exactly one y_{ij}^{t+1} -variable at gate $t+1$ due to the objective function. Therefore, the number of interactions in the QUBO corresponding to a circuit with more qubits will be larger even if the number of QUBO variables is identical. Finally, panel (C) plots the number of excess SWAP gates against the number of interactions in $QUBO_2$ of the benchmark instances. It can indeed be observed that the four- and five-qubit instances follow a now almost identical exponential trend. For the six-qubit instances, the number of data points is most likely too low to observe the same trend graphically. Conclusively, these results indicate that the quality of the solution obtained with the proposed method inversely scales with the number of interactions in the $QUBO_2$ formulation and therefore scales more poorly as the number of qubits n increases (compared to the number of gates m). In general, the observed exponential decay of the obtained objective values with increasing instance sizes is believed to be the consequence of the previously discussed physical limitations of the *D-Wave Advantage 5.3* Quantum Annealer.

Conclusions and further research

In this paper we proposed a polynomial-size QUBO formulation of the NNC problem. Both QUBO formulations proposed in this work are based on the polynomial-size ILP model by¹³. It was recognised that the number of additionally introduced slack variables in $QUBO_1$ do scale polynomial in n and m . $QUBO_2$ reduces the number of variables relative to the ILP formulation by reformulating the NNC problem purely in terms of the relative ordering variables y , which can be seen as the main contribution of this work. Therefore, as the number of constraints scales polynomially too, both QUBO formulations of the NNC problem, proposed in 4, are of polynomial size in n and m .

For all benchmark instances where the number of variables in $QUBO_2$, namely $n(n-1)/2 \cdot m$, did not exceed 450, feasible solutions were obtained. The quality of the obtained solutions was competitive with both

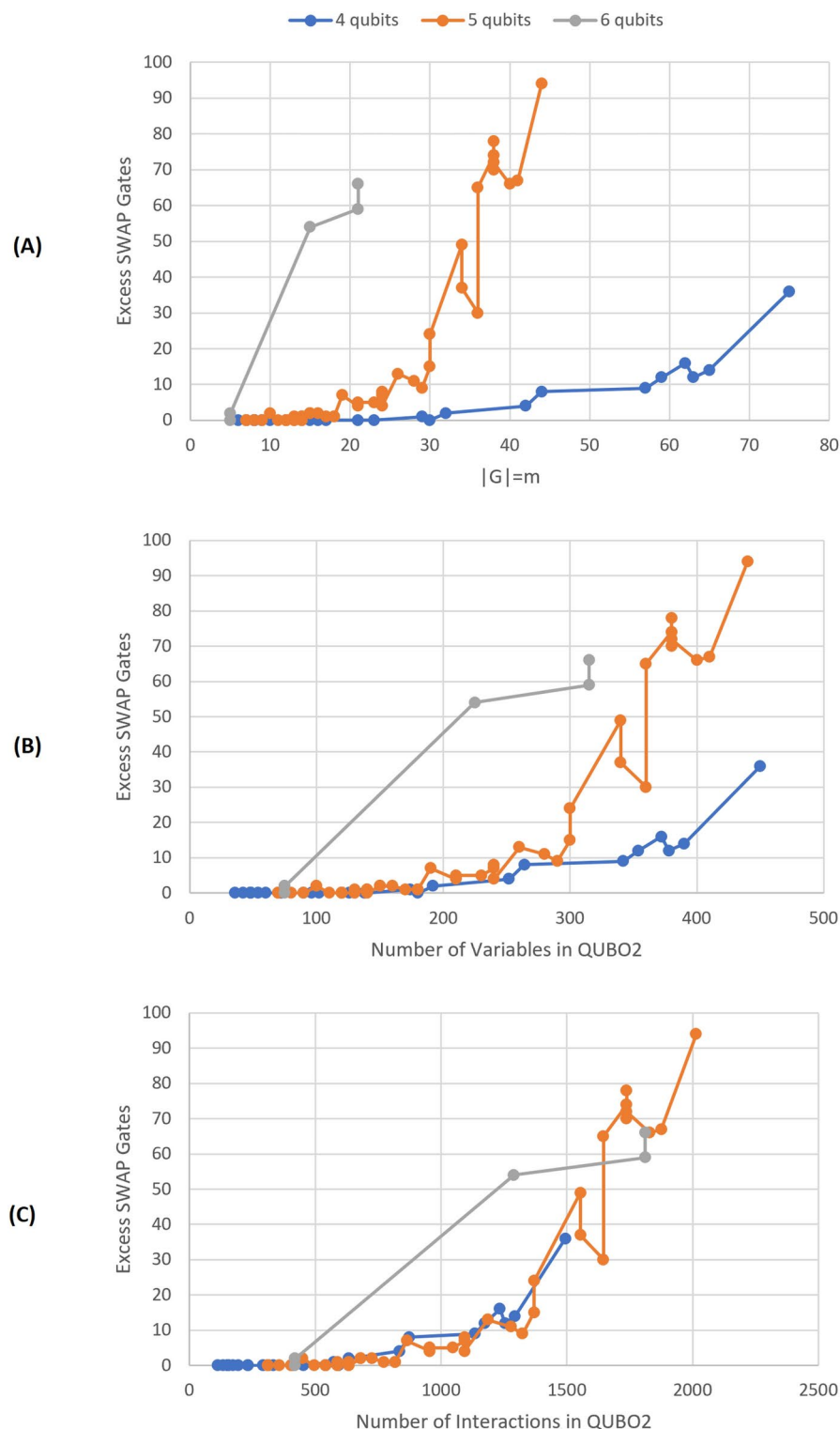


Fig. 3. For circuit instances with 4, 5, and 6 qubits respectively, the y-axis plots the difference between the solution's objective value obtained by our method and the optimal objective value ($SWAPSQ - OPT$), against (A) the number of gates m in the non-compliant circuit instance, (B) the number of binary variables in QUBO₂, (C) the number of interactions (quadratic terms with non-zero coefficients) in QUBO₂ on the x-axis.

heuristic and exact methods for smaller instances. However, as the problem size increased, the required time to compute an embedding of the problem on the Quantum Annealer limited any upside in terms of running time, even compared to the exact method by¹³. Moreover, the solution quality increasingly worsened for large instances, so that the proposed method did not remain competitive with other heuristic approaches.

Lastly, it was graphically demonstrated that the obtained objective values' absolute deviation from optimality seems to increase exponentially with the problem size, specifically, with the number of interactions in the resulting QUBO₂ formulation of an instance.

Concluding, we can state that while the linear NNC problem can be effectively solved for smaller instances, the physical limitations of contemporary state-of-the-art Quantum Annealers are restricting the scalability to larger instances where neither an advantage in terms of solution quality or run time compared to existing methods could be observed.

Given the relatively strong performance on small instances and the discussed connectivity of the QUBO₂ formulation, the authors see the use of decomposition methods such as the method proposed by⁵² as an avenue of further research. Their method identifies strongly connected components, which should exist in the QUBO₂ formulation, and then splits the overall problem into sub-problems accordingly. This might significantly increase the solution quality while also reducing the overall run time due to a reduction in the time needed to find embeddings.

Moreover, further research to identify minor-embeddings more efficiently by exploiting the problem-specific structure seems of interest, given that this step is the major contributor to the overall runtime.

Data availability

We used various instances from different sources. Quantum Fourier Transform (QFT) instances were obtained from [Nielsen, M.A., Chuang, I.L.: Quantum Computation and Quantum Information. Cambridge university press, Cambridge, UK (2010)]. All other instances come from the RevLib Library containing reversible circuits as can be found in [Wille, R., Große, D., Teuber, L., Dueck, G.W., Drechsler, R.: RevLib: An online resource for reversible functions and reversible circuits. In: International Symp. on Multi-Valued Logic, pp. 220–225 (2008)]. RevLib is available at <http://www.revlib.org> The datasets used and analysed during the current study are available at the corresponding author.

Received: 30 January 2024; Accepted: 23 September 2024

Published online: 07 October 2024

References

- Whelan, K. Computers, obsolescence, and productivity. *Rev. Econ. Stat.* **84**(3), 445–461 (2002).
- Kendrick, J.W., et al. Productivity trends in the united states. Productivity trends in the United States. (1961)
- Anderson, R.G. How well do wages follow productivity growth? Economic Synopses **2007**(2007-03-02) (2007)
- Moore, G. E. Cramming more components onto integrated circuits. *IEEE Solid State Circuits Soc. Newsl.* **38**(8), 114 (1965).
- Cross, T. After Moore's law. Technology quarterly—The economist (2016)
- Kumar, S. Fundamental limits to Moore's law. [arXiv:1511.05956](https://arxiv.org/abs/1511.05956) (2015)
- Markoff, J. Smaller, faster, cheaper, over: The future of computer chips. The New York Times (2015)
- Grover, L. K. Quantum mechanics helps in searching for a needle in a haystack. *Phys. Rev. Lett.* **79**(2), 325 (1997).
- Shor, P.W. Algorithms for quantum computation: discrete logarithms and factoring. In *Proceedings 35th Annual Symposium on Foundations of Computer Science* 124–134 (1994).
- Nielsen, M. A. & Chuang, I. L. *Quantum Computation and Quantum Information* (Cambridge University Press, 2010).
- Horowitz, M. & Grumblin, E. *Quantum Computing: Progress and Prospects* (The National Academies Press, 2019).
- Ding, Y. & Chong, F. T. *Quantum Computer Systems: Research for Noisy Intermediate-scale Quantum Computers* (Morgan & Claypool Publishers, 2020).
- Mulderij, J., Aardal, K.I., Chiscop, I. & Phillipson, F. A polynomial size model with implicit swap gate counting for exact qubit reordering. In *International Conference on Computational Science* 72–89 (Springer, 2023)
- Ito, T., Kakimura, N., Kamiyama, N., Kobayashi, Y. & Okamoto, Y. Algorithmic theory of qubit routing. In *Algorithms and Data Structures Symposium* 533–546 (Springer, 2023).
- Mulderij, J. Nearest neighbor compliance. Master's thesis, Delft University of Technology (2019)
- Van Houte, R., Mulderij, J., Attema, T., Chiscop, I. & Phillipson, F. Mathematical formulation of quantum circuit design problems in networks of quantum computers. *Quantum Inf. Process.* **19**, 1–22 (2020).
- Wille, R., Lye, A. & Drechsler, R. Exact reordering of circuit lines for nearest neighbor quantum architectures. *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.* **33**(12), 1818–1831 (2014).
- DWave D-Wave System Documentation. <https://docs.dwavesys.com/docs/latest/>
- Ikeda, K., Nakamura, Y. & Humble, T. S. Application of quantum annealing to nurse scheduling problem. *Sci. Rep.* **9**(1), 12837 (2019).
- Neukart, F. et al. Traffic flow optimization using a quantum Annealer. *Front. ICT* **4**, 29 (2017).
- Phillipson, F. & Bhatia, H. S. Portfolio optimisation using the D-Wave quantum annealer. In *International Conference on Computational Science* 45–59 (Springer, 2021)
- Ajagekar, A. & You, F. Quantum computing for energy systems optimization: Challenges and opportunities. *Energy* **179**, 76–89 (2019).
- Phillipson, F., Bontekoe, T. & Chiscop, I. Energy storage scheduling: A qubo formulation for quantum computing. In *Innovations for Community Services: 21st International Conference, I4CS 2021, Bamberg, Germany, May 26–28, 2021, Proceedings* 21 251–261 (Springer, 2021)
- Kuroski, K., Weglarz, J., Subocz, M., Różycki, R. & Waligóra, G. Hybrid quantum annealing heuristic method for solving job shop scheduling problem. In *Computational Science—ICCS 2020: 20th International Conference, Amsterdam, The Netherlands, June 3–5, 2020, Proceedings, Part VI* 502–515 (Springer, 2020)
- Wille, R., Große, D., Teuber, L., Dueck, G. W. & Drechsler, R. RevLib: An online resource for reversible functions and reversible circuits. In *International Symp. on Multi-Valued Logic*, 220–225 (2008). RevLib is available at <http://www.revlib.org>
- Hirata, Y., Nakanishi, M., Yamashita, S. & Nakashima, Y. An efficient conversion of quantum circuits to a linear nearest neighbor architecture. *Quantum Inf. Comput.* **11**(1), 142 (2011).
- Matsuo, A. & Yamashita, S. Changing the gate order for optimal LNN conversion. In *Reversible Computation: Third International Workshop, RC 2011, Gent, Belgium, July 4–5, 2011. Revised Papers* 3 89–101 (Springer, 2012)
- Hirata, Y., Nakanishi, M., Yamashita, S. & Nakashima, Y. An efficient conversion of quantum circuits to a linear nearest neighbor architecture. *Quantum Inf. Comput.* **11**(1), 142 (2011).
- AlFailakawi, M., AlTerkawi, L., Ahmad, I. & Hamdan, S. Line ordering of reversible circuits for linear nearest neighbor realization. *Quantum Inf. Process.* **12**, 3319–3339 (2013).

30. Bhattacharjee, A., Bandyopadhyay, C., Wille, R., Drechsler, R. & Rahaman, H. Improved look-ahead approaches for nearest neighbor synthesis of 1D quantum circuits. In *2019 32nd International Conference on VLSI Design and 2019 18th International Conference on Embedded Systems (VLSID)*, 203–208 (IEEE, 2019)
31. Kole, A., Datta, K. & Sengupta, I. A heuristic for linear nearest neighbor realization of quantum circuits by swap gate insertion using n -gate lookahead. *IEEE J. Emerg. Sel. Top. Circuits Syst.* **6**(1), 62–72 (2016).
32. Wille, R., Keszocze, O., Walter, M., Rohrs, P., Chattopadhyay, A. & Drechsler, R. Look-ahead schemes for nearest neighbor optimization of 1D and 2D quantum circuits. In *2016 21st Asia and South Pacific Design Automation Conference (ASP-DAC)* 292–297 (IEEE, 2016)
33. Cheng, X., Guan, Z. & Ding, W. Mapping from multiple-control Toffoli circuits to linear nearest neighbor quantum circuits. *Quantum Inf. Process.* **17**, 1–26 (2018).
34. Pedram, M. & Shafaei, A. Layout optimization for quantum circuits with linear nearest neighbor architectures. *IEEE Circuits Syst. Mag.* **16**(2), 62–74 (2016).
35. Wagner, F., Bärman, A., Liers, F. & Weissenböck, M. Improving quantum computation by optimized qubit routing. *J. Optim. Theory Appl.* **197**(3), 1161–1194 (2023).
36. Tan, Y.-Y., Cheng, X.-Y., Guan, Z.-J., Liu, Y. & Ma, H. Multi-strategy based quantum cost reduction of linear nearest-neighbor quantum circuit. *Quantum Inf. Process.* **17**, 1–14 (2018).
37. Lucas, A. Ising formulations of many np problems. *Front. Phys.* **2**, 5 (2014).
38. Fuente Ruiz, A. Quantum annealing. CoRR (2014) [arXiv:1404.2465](https://arxiv.org/abs/1404.2465)
39. Morita, S. & Nishimori, H. Mathematical foundation of quantum annealing. *J. Math. Phys.* [SPACE] <https://doi.org/10.1063/1.2995837> (2008).
40. Santoro, G. E. & Tosatti, E. TOPICAL REVIEW: Optimization using quantum mechanics: quantum annealing through adiabatic evolution. *J. Phys. Math. General* **39**(36), 393–431. <https://doi.org/10.1088/0305-4470/39/36/R01> (2006).
41. Sinitsyn, N. A. & Yan, B. Topologically protected Grover's oracle for the Partition Problem (2023)
42. Brooke, J., Bitko, D. & Rosenbaum Aeppli, G. Quantum annealing of a disordered magnet. *Science* **284**(5415), 779–781 (1999).
43. Heim, B., Rønnow, T. F., Isakov, S. V. & Troyer, M. Quantum versus classical annealing of Ising spin glasses. *Science* **348**(6231), 215–217. <https://doi.org/10.1126/science.aaa4170> (2015).
44. Rajak, A., Suzuki, S., Dutta, A. & Chakrabarti, B. K. Quantum annealing: an overview. *Phil. Trans. R. Soc. A* **381**(2241), 20210417 (2023).
45. Johnson, M. W. et al. Quantum annealing with manufactured spins. *Nature* **473**(7346), 194–198 (2011).
46. Kadowaki, T. & Nishimori, H. Quantum annealing in the transverse Ising model. *Phys. Rev. E* **58**(5), 5355–5363. <https://doi.org/10.1103/physreve.58.5355> (1998).
47. Siraichi, M. Y., Santos, V. F. d., Collange, C. & Pereira, F. M. Q. Qubit allocation. In *Proceedings of the 2018 International Symposium on Code Generation and Optimization* 113–125 (2018)
48. DWave Dwavesystems/Minorminer: Minorminer is a heuristic tool for minor embedding: Given a minor and target graph, it tries to find a mapping that embeds the minor into the target. <https://github.com/dwavesystems/minorminer>
49. Kole, A., Datta, K. & Sengupta, I. A new heuristic for n -dimensional nearest neighbor realization of a quantum circuit. *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.* **37**(1), 182–192 (2017).
50. Shafaei, A., Saeedi, M. & Pedram, M. Optimization of quantum circuits for interaction distance in linear nearest neighbor architectures. In *Proceedings of the 50th Annual Design Automation Conference* 1–6 (2013)
51. AlFailakawi, M., AlTerkawi, L., Ahmad, I. & Hamdan, S. Line ordering of reversible circuits for linear nearest neighbor realization. *Quantum Inf. Process.* **12**, 3319–3339 (2013).
52. Billionnet, A. & Jaumard, B. A decomposition method for minimizing quadratic pseudo-Boolean functions. *Oper. Res. Lett.* **8**(3), 161–163 (1989).

Author contributions

This work is based on the Thesis written by SM under supervision of FP. FP wrote the article based on this Thesis. All authors reviewed the manuscript.

Declaration

Competing interests

The authors declare no competing interests.

Additional information

Correspondence and requests for materials should be addressed to F.P.

Reprints and permissions information is available at www.nature.com/reprints.

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Open Access This article is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License, which permits any non-commercial use, sharing, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if you modified the licensed material. You do not have permission under this licence to share adapted material derived from this article or parts of it. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by-nc-nd/4.0/>.

© The Author(s) 2024