ВЕСТНИК ТОМСКОГО ГОСУДАРСТВЕННОГО УНИВЕРСИТЕТА

2024 Управление, вычислительная техника и информатика № 69
Tomsk State University Journal of Control and Computer Science

ИНФОРМАТИКА И ПРОГРАММИРОВАНИЕ

INFORMATICS AND PROGRAMMING

Original article
UDC 004.023
doi: 10.17223/19988605/69/11

Quantum circuit optimization via local qubit reordering by quantum annealing

Mariia A. Maltseva

University of Trento, Trento, Italy;
Institute of Applied Mathematical Research, Karelian Research Centre of the Russian Academy of Science, Petrozavodsk, Russian Federation, mariia.maltseva@unitn.it

Abstract. Mapping quantum circuits to the nearest-neighbor topology architecture requires additional gates. In this paper, we propose a hybrid approach to optimize quantum circuits for the two-dimensional nearest-neighbor architecture based on graph partitioning and solving SAT problem using quantum annealing. The technique's applicability and solution quality are studied via real experiments.

Keywords: circuit optimization; graph partitioning; quantum annealing; Boolean satisfiability.

***For citation*:** Maltseva, M.A. (2024) Quantum circuit optimization via local qubit reordering by quantum annealing. *Vestnik Tomskogo gosudarstvennogo universiteta. Upravlenie, vychislitelnaja tehnika i informatika – Tomsk State University Journal of Control and Computer Science*. 69. pp. 103–111. doi: 10.17223/19988605/69/11

Научная статья
doi: 10.17223/19988605/69/11

Оптимизация квантовой цепи путем локального переупорядочивания кубитов методом квантового отжига

Мария Алексеевна Мальцева

Университет Тренто, Тренто, Италия;
Институт прикладных математических исследований Карельского научного центра Российской академии наук, Петрозаводск, Россия, mariia.maltseva@unitn.it

Аннотация. Отображение квантовых цепей на архитектуру с топологией ближайшего соседа требует введения дополнительных вентилей. В статье предложен гибридный подход к оптимизации квантовых цепей для двумерной архитектуры ближайшего соседа на основе построения разрезов графа и решения SAT-задачи на системе квантового отжига. Применимость метода и качество решения исследованы с помощью натурных экспериментов.

Ключевые слова: оптимизация квантовой цепи; разрез графа; квантовый отжиг; выполнимость булевых формул.

***Для цитирования*:** Мальцева М.А. Оптимизация квантовой цепи путем локального переупорядочивания кубитов методом квантового отжига // Вестник Томского государственного университета. Управление, вычислительная техника и информатика. 2024. № 69. С. 103–111. doi: 10.17223/19988605/69/11

Introduction

Quantum computing is a cutting-edge field that harnesses the principles of quantum mechanics to perform calculations exponentially faster than classical counterpart and solve complex problems that are intractable to it. Similar to classical circuits in traditional computing, quantum circuits (QCs) are the fundamental blocks of quantum algorithms and consist of quantum gates placed sequentially and operating on quantum bits (qubits). However, the noisy nature of current quantum systems, together with hardware constraints and limited available resources, poses the significant challenges to the implementation of QCs. QC optimization is therefore a crucial task, and various techniques have been proposed to perform it.

In general, there is a wide variety of QC optimization criteria. In [1], the authors aim to reduce the number of two-qubit controlled-NOT (CNOT) gates because of their proneness to errors. For that, the problem is converted to tree search problem and then solved by A^* graph-traversal heuristic search allowing to find optimal path between any two nodes of the graph.

QC depth is another popular QC optimization criteria, and it means the largest number of single-qubit and two-qubit gates on any path from input to output of a circuit [2]. In [3], the authors propose to construct depth-optimal QC via meet-in-the-middle algorithm being a heuristic graph-traversal path-finding search.

There is a separate group of topology-aware QC optimization techniques. As QC is a sequence of quantum gates operating on qubits, logical qubits used in a gate description should be mapped to physical qubits embedded into the architecture. Commonly, architecture should satisfy the Nearest Neighbor (NN) condition implying that physical qubits can interact only with their neighbors [4] and making the process of logical qubits mapping non-trivial. However, it allows to reduce impact of noise on quantum operations [5], but introduces significant overhead because of auxiliary SWAP gates (SWAPs) applied to make interacting qubits adjacent. Thus, QC optimization in NN architecture is significant to minimize the number of additional cost-expensive SWAPs (SWAP count) [6].

Several approaches to optimize QC by minimizing SWAP count have been proposed. They differ by the way of QC representation and optimization technique used. For instance, circuit is commonly studied as a NN-compliant circuit, and heuristics are used to minimize SWAP count. In [6], graph bipartitioning applied to qubit line adjacency graph is proposed as a SWAP count minimization technique in linear (one-dimensional) NN architecture. In [7], QC is represented as a graph of interactions between each control-target pair as an edge with weight equal to the number of times this pair appears in the circuit and SWAP count is minimized by qubit lines rearrangement in two-dimensional (2D) NN architecture using Harmony Search algorithm being a sort of gradient-based metaheuristic.

Despite global qubit lines rearrangement, qubit lines can be rearranged locally, implying that qubits used in the description of only the current gate are reordered. This kind of qubit lines rearrangement has been proposed in [8, 9] with SWAP count minimization performed by heuristics.

We are focusing on optimizing QCs for the 2D NN architecture intensively studied recently [9–12] and follow the approach proposed in [9] where the authors consider a circuit as a gate dependency graph and divide it into NN-compliant subcircuits by local reordering of qubits. For that, they propose to partition the graph using Boolean satisfiability. Further, to connect 2D placements of qubits obtained for each subcircuit, A^* search is used. The novelty of the present paper is that we propose hybrid quantum-quantum QC optimization approach with quantum computing techniques replacing the classical counterpart. The choice of such a revolutionary type of computing is the main advantage of our approach. More specifically, quantum annealing implemented in D-Wave quantum machine is proposed to solve the problem under study. Being a hardware implemented heuristic, it returns the solution in constant time, opposed to traditional classical heuristics. It makes quantum annealing beneficial for solving QC optimization problem. The details are described further.

The structure of the paper is as follows. In Section 1, the problem is formulated accurately. Section 2 is devoted to the theoretical fundamentals of quantum theory. In Section 3, the approach of QC optimization is proposed. Section 4 illustrates encouraging optimization results of several QCs.

1. Problem statement

In this paper, we study possibilities of QC optimization via local qubit line mapping in 2D NN architecture. For such an architecture, gate set is commonly restricted and consists of single-qubit gates and CNOT. In combination, the aforementioned gates form a universal gate set meaning that any unitary operation can be approximated by a QC composed of only these gates [13].

To introduce a 2D architecture, a 2D grid is defined and each qubit has four neighbors (two horizontal

$$\begin{bmatrix} q_1 & q_2 & q_3 \\ q_4 & q_5 & q_6 \end{bmatrix}$$

and two vertical) at most here. For instance, in grid qubit q_1 has two neighboring qubits which are q_2 and q_4.

In 2D NN architecture, two-qubit gates act only on adjacent qubits, and SWAPs are used to rearrange non-adjacent interacting qubit lines to perform given operations. Count of additional SWAPs depends on the initial qubit lines placement. Therefore, finding a 2D placement of qubit lines that minimizes the SWAP count is essential for optimizing the circuit.

More precisely, we have to find a sequence of qubit lines placements $\left\{A^k\right\}_{k=1}^{m}$ meaning that qubit lines placement is changed for, say, m times, where A^k is a 2D grid (matrix) with $A^k_{ij} = q$, if qubit q is positioned at (i, j). For convenience, we also define inverse mapping $q \to (i_q^k, j_q^k)$ as a position of qubit q in 2D placement A^k. Such a sequence consists of qubit line placements each of which is valid only for one subcircuit constructed from a given circuit by gates recombination and becoming NN-compliant after qubit line reordering.

To map each intermediate qubit line placement A^k to the next one, additional SWAPs are needed. For a 2D grid, its count is calculated via the Manhattan distance and should be minimized, i.e.

$$s(k) := \sum_{q=1}^{n} \left| i_q^k - i_q^{k+1} \right| + \left| j_q^k - j_q^{k+1} \right| \to \min. \tag{1}$$

Then, the overall SWAP count within the whole circuit is as follows:

$$\sum_{k=1}^{m} s(k) = \sum_{k=1}^{m} \sum_{q=1}^{n} \left| i_q^k - i_q^{k+1} \right| + \left| j_q^k - j_q^{k+1} \right| \to \min, \tag{2}$$

where n is the number of qubits.

However, quantum gates composing a given circuit should be recombined according to the gates dependency graph $G = (V, E)$, where V is a set of gates of the circuit and E is the set of dependencies of the gates with Boolean weight $e_{g1,g2} = 1$, if g_2 follows g_1 and they are not commutative (meaning that interchange of them causes different result of circuit implementation), and $e_{g1,g2} = 0$, if g_1 and g_2 are commutative.

As such, the problem is a graph partitioning problem with goal function (2).

2. Scientific background

2.1 Quantum theory basic properties

Qubit is a basic unit of information in QC [14]. A qubit has the two basic Boolean states. Because of the quantum theory nature, qubit can also store a superposition (linear combination) of the two basic Boolean states $|0\rangle \equiv \begin{bmatrix} 1 \\ 0 \end{bmatrix}$ or $|1\rangle \equiv \begin{bmatrix} 0 \\ 1 \end{bmatrix}$, i.e. the qubit state $|\varphi\rangle$ can be expressed as follows:

$$|\varphi\rangle = \begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \alpha|0\rangle + \beta|1\rangle,$$

where $\alpha, \beta \in C$, such that $\alpha^2 + \beta^2 = 1$.

The next fundamental quantum property is entanglement [13]. It allows qubits to interact in pairs irregardless of the distance between them and makes them dependent on each other that way.

To perform computations in quantum computing, quantum gates are used. They act on qubits and change their states correspondingly in that way [13]. Quantum gates are used as basic blocks to compose more complex functionality that is represented by a QC [15].

Algebraically a quantum gate is represented as a unitary operator [13] meaning that its inverse matrix is equal to its adjoint matrix. Unitarity conserves being of a quantum system in one of the possible states [13].

There is a great variety of quantum gates. Some of them are single-qubit, i.e. aforementioned NOT gate inverting the qubit. Other gates operate on several qubits. For example, CNOT (controlled-NOT) gate operates on 2 qubits and entangles them by inverting the target qubit if the control qubit is in state $|1\rangle$. For example, if control qubit is the first one and target is the second one (CNOT(q_1, q_2)), then it is as follows

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}.$$

Three CNOT gates being in combination construct a two-qubit SWAP gate that interchanges input qubits [13].

2.2. Quantum Annealing

Quantum Annealing (QA) is analogous to traditional Simulated Annealing (SA) that is a local search optimization heuristic resembling the physical annealing process. Similar to the physical cooling, cooling scheme manages SA process and transition to the next candidate solution is done with temperature-dependent probability decreasing over time. The difference between SA and QA is that the latter is able to find global optimum thanks to the quantum tunneling enabling annealing process to pass through the energetic barriers [16].

The QA system aims to decrease the cost function of the problem, which can be represented in an Ising model [17]:

$$\varepsilon(\bar{z}) = \sum_{i \in V} h_i z_i + \sum_{(i,j) \in E} J_{ij} z_i z_j, \tag{3}$$

where $\bar{z} = \{-1,1\}^{|V|}$ is a set of spins describing qubits from V, $G = (V, E)$ is an undirected graph of allowed interactions between qubits, $J_{ij} = J_{ji}$ is interaction strength of the qubits i and j connected by an edge and h_i is the on-site energy of qubit i.

Note that QA is not a universal quantum computer. QA can solve only Quadratic Unconstrained Binary Optimization (QUBO) problems defined as follows [18]:

$$\min_{\bar{x} \in \{0,1\}^{|V|}} \bar{x}^T Q \bar{x}, \tag{4}$$

where $Q_{ii} = h_i$, $Q_{ij} = J_{ij}$ from the Ising model (3).

As for the hardware implementation, QA is developed by D-Wave and can be run online for a limited access time. IT-company Fujitsu provides an alternative quantum-inspired Digital annealer.

3. Solution approach

We introduce a hybrid quantum-quantum SWAP count optimization of a QC to be implemented in a 2D NN architecture following [9]. The key ingredients of the original approach involves Boolean satisfiability (that checks if there is at least one set of the variables' assignments under which the given Boolean formula evaluates to TRUE [19]) and A* heuristic graph-traversal path-finding search [20].

Algorithmic description of our approach is presented below. It differs from [9] by solving SAT-problem in QUBO format using QA (see Step 3.2-Step 3.4 below).

Step 1. Require: 2D grid with initial qubits placement; gate dependency graph G; set of gates Γ; partition $P = \varnothing$.

Step 2. While Γ not empty.

Step 3. Binary search to construct a subcircuit G^k (taking into account gate dependency graph G):

Step 3.1. Convert G^k to a CNF CNF_k;

Step 3.2. Convert a SAT-problem with CNF_k to a QUBO problem with coefficient matrix Q_k (by *qubovert* Python package);

Step 3.3. Solve QUBO problem defined by Q_k using QA;

Step 3.4. Extract the first solution (with the lowest energy) - qubit lines placement A^k;

Step 3.5. If CNF_k is TRUE;

Step 3.5.1. Partition $P = P \cup G^k$, and $\Gamma = \Gamma \setminus G^k$, and return to Step 2;

Step 3.6. Return to Step 3;

Step 4. Apply A* search to connect $\left\{A^k\right\}_{l=1}^{m}$ and calculate number of SWAP gates needed for that.

The approach is iterative and is based on binary search technique used at Step 3 to construct a subcircuit taking into account gate dependencies. To obtain CNF at Step 3.1, the following conditions should be formulated:

Condition 1. Interacting bits of all gates are adjacent.

Condition 2. Each qubit is assigned to only one cell on a 2D grid.

Condition 3. At most one qubit is assigned to each cell on a 2D grid.

We introduce Boolean variables x_{ijr} that are 1 if qubit q_r is assigned to cell (i, j) and 0 otherwise. For simplicity, 2D grid of 3 rows and 3 columns where cells are placed rowly is considered.

To express Condition 1 as a Boolean function, we consider a CNOT(q_1, q_2) as an example. If q_1 is assigned to (1,1), then q_2 should be assigned to either one of (0,1), (1,0), (1,2), or (2,1). This condition can be expressed as

$$\neg x_{111} \vee x_{012} \vee x_{102} \vee x_{122} \vee x_{212}. \tag{5}$$

Then, applying disjunction to the Boolean formulas of such conditions for the other cells of the grid, we obtain a formula for the condition such that q_1 and q_2 should be adjacent on the grid. After that, we apply conjunction of the obtained Boolean formulas for all the pairs of interacting qubits.

As for Condition 2, each qubit is assigned to at least one cell and at most one cell. The former part of the condition can be expressed as

$$\bigvee_{(i,j)} x_{ijr} = 1 \tag{6}$$

for each qubit q_r. The latter part prohibits assigning the qubit to two different cells:

$$\neg x_{i_1 j_1 r} \vee \neg x_{i_2 j_2 r} = 1 \tag{7}$$

for each qubit q_r and each pair of cells $(i_1, j_1) \neq (i_2, j_2)$.

Condition 3 prohibits assigning two different qubits to one cell meaning that

$$\neg x_{ijr_1} \vee \neg x_{ijr_2} = 1 \tag{8}$$

for each cell (i, j) and each pair of qubits q_{r1} and q_{r2}, $r_1 \neq r_2$.

At Step 3.3, QA solves QUBO problem constructed from CNF by *qubovert* automatically. If the solution is satisfiable, i.e. there is such a qubit placement A^k satisfied to (5)–(8) then G^k forms a NN-compliant subcircuit with qubit placement A^k, and binary search on the rest of circuit is continued.

After partitioning the gate dependency graph G into NN-compliant subcircuits with qubit placements $\left\{A^k\right\}_{l=1}^{m}$, A* search is used to connect them optimally (Step 4). A* search is a heuristic that allows to find optimal way from the starting node S of a graph to the target one T based on a cost function $f(N) = g(N) + h(N)$, where N is a current node during the search, $g(N)$ is a cumulative cost from S to N [9], and $h(N)$ is a heuristic that estimates the cost from the current qubit placement N to T. In our problem, a qubit placement A^k is

a node of a graph with A^1 as the starting node and A^m as the target one. Cumulative cost $g(A^k)$ is SWAP count needed to obtain A^k from A^1 and it is the sum of the Manhattan distances between the positions of each qubit in the A^k and A^1, i.e.

$$g(A^k) = \sum_{i=1}^{k-1} s(i),$$

where $s(i)$ is defined in (1). Heuristic cost $h(A^k)$ is calculated analogously.

4. Numerical experiments

Recall that in order to solve a SAT problem in QA, at first it should be converted into QUBO format (4). It can be done using the technique from [19] which requires preliminary setup to encode a SAT-problem to the Ising model. Alternative way is to use Python package *qubovert* that performs such a conversion automatically by using *sat* library. In this paper, we propose to follow the latter approach. To obtain a SAT-problem, each of the Conditions 1, 2, and 3 ((5) – (8)) are added to the resulting Boolean formula with some positive Lagrangian parameter λ tuned by hand. For the experiments performed, λ=100 for (5), (7) and (8), and λ=10 for (6).

To verify the applicability of the approach, we optimize one quantum gate from *Reversible logic synthesis benchmark page by D. Maslov* (available at https://reversiblebenchmarks.github.io/) and another one created by us. To perform the experiments using QA, we use a real quantum annealer, namely, *D-Wave Advantage_system5.4* with 5614 qubits and 40050 couplers, fixing the annealing time to 20μs and taking the best out of 2000 simulation runs. Due to the limited access time on it, we also optimize the circuits via SA and classical SAT-solver *PicoSAT* run on local Ubuntu machine (Ubuntu 23.04 with 31 GiB RAM using 11th Gen Intel(R) Core(TM) i7-11700 @ 2.50GHz). Note that numerical experiments are performed only for the circuits with no more than 5 qubits because of the high computational demand on *qubovert*. Along the lines, we also benchmark the resulting quantum part of the algorithm in terms of logical and physical qubits to demonstrate its computational complexity of QA.

4.1. Fredkin gate optimization

Fredkin gate is a gate from *Reversible logic synthesis benchmark page by D. Maslov*. It is frequently used in QC because of its small cost in some QC technologies. The circuit realization at elementary quantum gate set is illustrated in Fig. 1 and consists of 7 two-qubit quantum gates (enumerated as $C_1 \ldots C_7$) operating on different pairs of 3 qubits. In case when qubits are placed in 2×2 grid rowly with numbering starting at the left side, i.e. NN topology defined as $\begin{bmatrix} 0 & 1 \\ 2 & - \end{bmatrix}$, 6 SWAPs are needed to implement the circuit (as 1 SWAP is needed before each gate to make it NN-compliant, and 1 SWAP returns the qubits to their initial order after the gate). In Fig. 2, the dependency of the gates of Fredkin gate from Fig. 1 is illustrated.

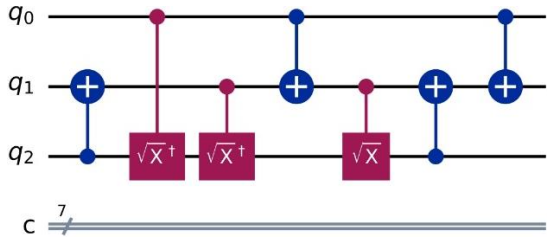

Fig. 1. Fredkin gate with initial SWAP count = 6

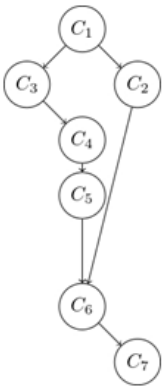

Fig. 2. Dependency of gates in Fredkin gate from Fig. 1

At first, we use SA due to the limited access time of QA on D-Wave. After applying the proposed approach, the circuit is divided into 2 NN-compliant subcircuits as follows from Fig. 3. The qubits are placed following $A^1 = \begin{bmatrix} 2 & 1 \\ 0 & - \end{bmatrix}$ and $A^2 = \begin{bmatrix} 1 & 2 \\ 0 & - \end{bmatrix}$ within the first and the second subcircuits, respectively. 1 SWAP is needed to make them consistent.

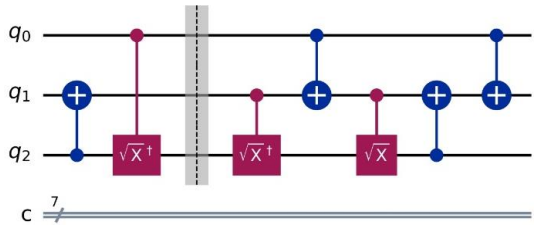

Fig. 3. The optimized Fredkin gate from Fig. 1

QA divides the circuit into subcircuits in the same way as SA does. The qubits are placed following $A^1 = \begin{bmatrix} - & 0 \\ 1 & 2 \end{bmatrix}$ and $A^2 = \begin{bmatrix} - & 0 \\ 2 & 1 \end{bmatrix}$ within the first and the second subcircuits, respectively. Similar to SA, 1 SWAP is needed to connect them. We should note that SA always returns positive result in case when a subcircuit is satisfiable unlike QA. Thus, we should repeat QA for several times (it is 15 in our case) to reduce the probability to exclude satisfiable subcircuit accidentally. In order to convert the subcircuit to QUBO, 19 logical qubits are used and 36 physical qubits are used for the first subcircuit whereas the second one holds 33 physical qubits. It implies that QA hardware (with more than 5000 qubits) used can solve bigger QUBO problems.

To validate the result obtained above, we replicate the classical approach proposed in [9] and use the classical SAT solver *PicoSAT* via *pyeda* Python package. In this experiment the circuit is divided in the same way as both in QA and SA (see Fig. 3). And the qubits are placed following $A^1 = \begin{bmatrix} - & 1 \\ 0 & 2 \end{bmatrix}$ and $A^2 = \begin{bmatrix} - & 2 \\ 0 & 1 \end{bmatrix}$ within the first and the second subcircuits, respectively. 1 SWAP is required to make them consistent.

4.2. Optimization of CNOT-based circuit

To verify the applicability of the proposed approach for a bigger QC, we construct one illustrated in Fig. 4. It contains 6 CNOT gates (enumerated as $C_1 \ldots C_6$) operating on some pairs of 5 qubits. Initially, it requires 6 SWAPs in order to implement the circuit in 2D NN architecture that is a 2D grid where qubits are placed rowly, i.e. $\begin{bmatrix} 0 & 1 & 2 \\ 3 & 4 & - \end{bmatrix}$. In Fig. 5, the dependency of the gates of circuit from Fig. 4 is depicted.

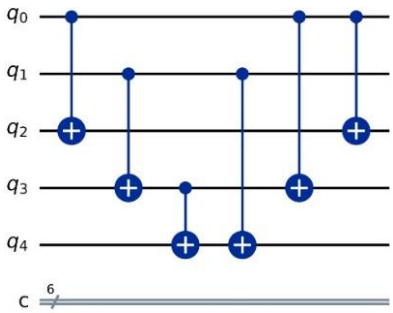

Fig. 4. The circuit to be optimized. Initial SWAP count = 6

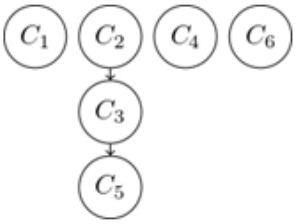

Fig. 5. Dependency of gates in circuit from Fig. 4

After applying SA, the number of SWAPs needed is 2 at best. During the optimization, the circuit was divided into 2 NN-compliant subcircuits: the first one contains C_1, C_2, C_3, C_5, and C_6, and the second subcir-

cuit is formed by C_4 gate (see Fig. 5). Note that C_4, C_5, and C_6 gates are commutative and it allows to place C_4 after C_6. The qubits are placed following $A^1 = \begin{bmatrix} - & 4 & 2 \\ 1 & 3 & 0 \end{bmatrix}$ and $A^2 = \begin{bmatrix} 3 & - & 2 \\ 1 & 4 & 0 \end{bmatrix}$ within the first and the second subcircuits, respectively.

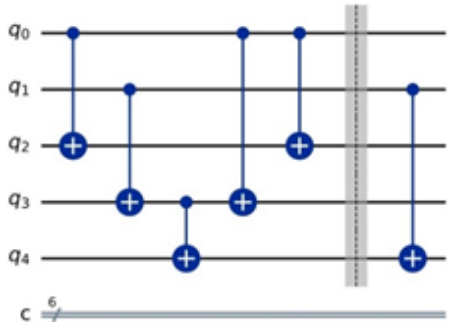

Fig. 6. The optimized circuit from Fig. 4

QA also divides the circuit from Fig. 4 in the same way as SA (see Fig. 6) where optimal qubits placements follow $A^1 = \begin{bmatrix} 4 & 3 & 0 \\ - & 1 & 2 \end{bmatrix}$ and $A^2 = \begin{bmatrix} 0 & 1 & 4 \\ - & 2 & 3 \end{bmatrix}$ within the first and the second subcircuits, respectively. In order to make these optimal qubits placements consistent, 5 SWAPs are needed.

To encode the first subcircuit into QUBO, 103 logical and 354 physical qubits are used and 111 logical and 304 physical qubits are used for the encoding of the second subcircuit into QUBO.

Here, QA returns suboptimal solution due to the dimension of the problem. As this CNOT-based circuit requires more logical and physical qubits than Fredkin gate investigated in Section 4.1, it seems that more simulation runs are needed to obtain better solution. However, its number is bounded by the access time provided by D-Wave machine that is also limited. We decided to increase the number of simulation runs up to 4000 and the result is encouraging. The proposed technique partitions the circuit into 2 subcircuits: the 1st one consists of the first four CNOT gates (C_1, C_2, C_3, C_4) with qubit placement $A^1 = \begin{bmatrix} 2 & 0 & - \\ 4 & 3 & 1 \end{bmatrix}$ and the other two gates construct the 2nd subcircuit with qubit placement $A^2 = \begin{bmatrix} 0 & - & 4 \\ 2 & 3 & 1 \end{bmatrix}$. Only 3 SWAPs are needed to make the subcircuits consistent.

By optimizing the circuit using *PicoSAT*, the circuit is divided in the same way as both in QA and SA (see Fig. 6). Optimal qubits placements of the subcircuits follows $A^1 = \begin{bmatrix} - & 4 & 2 \\ 1 & 3 & 0 \end{bmatrix}$ and $A^2 = \begin{bmatrix} - & 4 & 0 \\ 3 & 1 & 2 \end{bmatrix}$ within the first and the second subcircuits, respectively. 2 SWAPs are required to make them consistent.

Conclusion

In this work, we introduce hybrid QC optimization approach by local reordering of qubits using QA and Boolean satisfiability. Its promising nature is explained by the use of a breakthrough computing for solving such a problem, which provides a solution in constant time. Numerical experiments show that QA performs well on small scale: the solution obtained is comparable with one returned by classical SAT-solver. However, while solving a larger problem, QA may return suboptimal results, and making the simulation process longer improves the solution.

Author express thanks to Rumyantsev Alexander S. (Dr. Sci. in Physics and Mathematics) for the significant comments and advice while writing the paper.

References

1. Davis, M.G., Smith, E., Tudor, A., Sen, K., Siddiqi, I. & Iancu, C. (2020) Towards Optimal Topology Aware Quantum Circuit Synthesis. *2020 IEEE International Conference on Quantum Computing and Engineering (QCE).* pp. 223–234. DOI: 10.1109/QCE49297.2020.00036
2. Arabzadeh, M., Saheb Zamani, M., Sedighi, M. & Saeedi, M. (2013) Depth-optimized reversible circuit synthesis. *Quantum Information Processing.* 12(4). pp. 1677–1699. DOI: 10.1007/s11128-012-0482-8
3. Amy, M., Maslov, D., Mosca, M. & Roetteler, M. (2013) A Meet-in-the-Middle Algorithm for Fast Synthesis of Depth-Optimal Quantum Circuits. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems.* 32(6). pp. 818–830. DOI: 10.1109/tcad.2013.2244643
4. Wille, R., Lye, A. & Drechsler, R. (2014) Optimal SWAP gate insertion for nearest neighbor quantum circuits. *2014 19th Asia and South Pacific Design Automation Conference (ASP-DAC).* pp. 489–494. DOI: 10.1109/ASPDAC.2014.6742939
5. Datta, K., Kole, A., Sengupta, I. & Drechsler, R. (2022) Nearest Neighbor Mapping of Quantum Circuits to Two-Dimensional Hexagonal Qubit Architecture. *2022 IEEE 52nd International Symposium on Multiple-Valued Logic (ISMVL).* DOI: 10.1109/ismvl52857.2022.00013
6. Chakrabarti, A., Sur-Kolay, S. & Chaudhury, A. (2011) Linear Nearest Neighbor Synthesis of Reversible Circuits by Graph Partitioning. *arXiv*. DOI: 10.48550/ARXIV.1112.0564
7. Alfailakawi, M.G., Ahmad, I. & Hamdan, S. (2016) Harmony-Search Algorithm for 2D Nearest Neighbor Quantum Circuits Realization. Expert Systems *With Applications*. 61(C). pp. 16–27. DOI: 10.1016/j.eswa.2016.04.038
8. Bhattacharjee, A., Bandyopadhyay, C., Wille, R., Drechsler, R. & Rahaman, H. (2019) Improved Look-Ahead Approaches for Nearest Neighbor Synthesis of 1D Quantum Circuits. *2019 32nd International Conference on VLSI Design and 2019 18th International Conference on Embedded Systems (VLSID).* pp. 203–208. DOI: 10.1109/VLSID.2019.00054
9. Wakaki, H. & Yamashita, S. (2019) Mapping a Quantum Circuit to 2D Nearest Neighbor Architecture by Changing the Gate Order. *IEICE Transactions on Information and Systems.* E102.D. pp. 2127–2134. DOI: 10.1587/transinf.2018EDP7439
10. Shafaei, A., Saeedi, M. & Pedram, M. (2014) Qubit placement to minimize communication overhead in 2D quantum architectures. *Proceedings of the Asia and South Pacific Design Automation Conference (ASP-DAC).* pp. 495–500. DOI: 10.1109/ASPDAC.2014.6742940
11. Wille, R., Keszocze, O., Walter, M., Rohrs, P., Chattopadhyay, A. & Drechsler, R. (2016) Look-ahead schemes for nearest neighbor optimization of 1D and 2D quantum circuits. *2016 21st Asia and South Pacific Design Automation Conference (ASP-DAC).* pp. 292–297. DOI: 10.1109/ASPDAC.2016.7428026
12. Farghadan, A. & Mohammadzadeh, N. (2017) Quantum circuit physical design flow for 2D nearest-neighbor architectures: Quantum Circuit Design Flow. *International Journal of Circuit Theory and Applications.* 45. pp. 989–1000. DOI: 10.1002/cta.2335
13. Nielsen, M.A. & Chuang, I.L. (2010) *Quantum computation and quantum information.* Cambridge University Press. [Online] Available from: https://profmcruz.wordpress.com/wp-content/uploads/2017/08/quantum-computation-and-quantum-information-nielsen-chuang.pdf. (Accessed: 27th September 2024).
14. Pereira da Silva, R. (2023) Gate-based Quantum Computing: An Overview. *SSRN Electronic Journal.* DOI: 10.2139/ssrn.4347584
15. Racorean, O. (2015) Quantum Gates and Quantum Circuits of Stock Portfolio. *arxiv*. DOI: 10.2139/ssrn.2630341
16. Yarkoni, S., Raponi, E., Bäck, T. & Schmitt, S. (2022) Quantum annealing for industry applications: introduction and review. *Reports on Progress in Physics.* 85(10). pp. 1–43. DOI: 10.1088/1361-6633/ac8c54
17. Su, J. (2018). *Towards Quantum Computing: Solving Satisfiability Problem by Quantum Annealing*. UCLA Electronic Theses and Dissertations. [Online] Available from: https://escholarship.org/uc/item/8qp5200s. (Accessed: 26th August 2024).
18. Bonomi, A., De Min, T., Zardini, E., Blanzieri, E., Cavecchia, V. & Pastorello, D. (2022) Quantum annealing learning search implementations. *Quantum Information and Computation.* 22(3 & 4). pp. 181–208. DOI: 10.26421/qic22.3-4-1
19. Bian, Z., Chudak, F., Macready, W., Roy, A., Sebastiani, R. & Varotti, S. (2018) Solving SAT and MaxSAT with a Quantum Annealer: Foundations, Encodings, and Preliminary Results. *arXiv*. [Online] Available from: https://arxiv.org/abs/1811.02524. (Accessed: 26th August 2024).
20. Foead, D., Ghifari, A., Kusuma, M.B., Hanafiah, N. & Gunawan, E. (2021) A Systematic Literature Review of A* Pathfinding. *Procedia Computer Science.* 179. pp. 507–514. DOI: 10.1016/j.procs.2021.01.034

Information about the author:

Maltseva Mariia A. (Post-graduate Student, Department of Information Engineering and Computer Science, University of Trento, Trento, Italy; Junior research assistant, Institute of Applied Mathematical Research, Karelian Research Centre of the Russian Academy of Science, Petrozavodsk, Russian Federation). E-mail: mariia.maltseva@unitn.it

The author declares no conflicts of interests.

***Информация об авторе*:**

Мальцева Мария Алексеевна – аспирант факультета информационной инженерии и информатики Университета Тренто (Тренто, Италия); младший научный сотрудник Института прикладных математических исследований Карельского научного центра Российской академии наук (Петрозаводск, Россия). E-mail: mariia.maltseva@unitn.it

Автор заявляет об отсутствии конфликта интересов.

Received 28.08.2024; accepted for publication 02.12.2024

Поступила в редакцию 28.08.2024; принята к публикации 02.12.2024