Reinforcement Learning for Variational Quantum Circuit Design

Simone Foderà[1,†], Gloria Turati[2,*], Riccardo Nembrini[2], Maurizio Ferrari Dacrema[2] and Paolo Cremonesi[2]

[1]Ludwig Maximilians Universität, Germany
[2]Politecnico di Milano, Italy

Abstract

Variational Quantum Algorithms (VQAs) are widely used for solving optimization problems in the Noisy Intermediate-Scale Quantum (NISQ) era. However, designing effective quantum circuits (ansatzes) that are compatible with the limitations of current quantum hardware remains a significant challenge. In this work, we introduce a Reinforcement Learning (RL) agent that autonomously generates ansatzes for VQAs. The RL agent is trained on several optimization problems, including Maximum Cut, Maximum Clique, and Minimum Vertex Cover, across different graph topologies. Our results show that the agent is able to generate effective quantum circuits, with approximation ratios that favorably compare to commonly used ansatzes. Additionally, we identify a novel family of ansatzes, termed "R_{yz}-connected", particularly effective on Maximum Cut problems. These findings highlight the potential of RL techniques in designing efficient quantum circuits for a broad class of applications in quantum computing.

Keywords

Variational Quantum Algorithms, Reinforcement Learning, Ansatz, Quantum Computing

1. Introduction

Quantum computing has gained attention for its potential to tackle problems that are computationally challenging for classical computers. Variational Quantum Algorithms (VQAs) [1], which employ a parametric quantum circuit and a classical optimizer to adjust the circuit parameters, represent a promising paradigm in the Noisy Intermediate-Scale Quantum (NISQ) [2] era. However, a key challenge in these algorithms lies in identifying an appropriate ansatz for the specific problem being addressed. Approaches for selecting suitable ansatzes include leveraging problem-specific properties, such as symmetries [3–5], or using adaptive methods that dynamically modify circuits by adding and removing gates during execution [6–8]. However, identifying which problem properties can be exploited is non-trivial and adaptive methods rely on carefully designed heuristics and may require numerous circuit executions to converge to a suitable ansatz.

Reinforcement Learning (RL) is a Machine Learning paradigm where an agent learns to perform actions to achieve a desired goal. By receiving a reward for each action taken, the agent iteratively refines its strategy to maximize the cumulative reward, ultimately achieving the desired objective. The RL paradigm is highly flexible and has already been successfully deployed for many tasks in quantum computing [9–12], including circuit design [13, 14]. One crucial difference between the RL paradigm and the existing adaptive approaches is that it is possible with RL to learn how to build circuits without relying on manually designed heuristics or domain-specific knowledge. Furthermore, RL can be effective in scenarios with very large solution spaces, such as the one of possible quantum circuits.

AIQ x QIA 2024: 2nd International Workshop on AI for Quantum and Quantum for AI, November 25th - 28th 2024, Bolzano, Italy
*Corresponding author.
†This study was conducted while the author was a Master's student at Politecnico di Milano.
simone.fodera@lmu.de (S. Foderà); gloria.turati@polimi.it (G. Turati); riccardo.nembrini@polimi.it (R. Nembrini); maurizio.ferrari@polimi.it (M. Ferrari Dacrema); paolo.cremonesi@polimi.it (P. Cremonesi)
0009-0008-8180-0670 (S. Foderà); 0000-0001-5367-4058 (G. Turati); 0000-0002-1915-6107 (R. Nembrini); 0000-0001-7103-2788 (M. Ferrari Dacrema); 0000-0002-1253-8081 (P. Cremonesi)

In our work we present a RL-based algorithm designed to search for variational quantum circuits for solving optimization problems. We train the agent on instances of the Maximum Cut, Maximum Clique, and Minimum Vertex Cover problems and analyze the solution quality and the properties of the circuits obtained. Our findings show that the agent is capable of building circuits leading to satisfactory results, particularly on the Maximum Cut problem.

Moreover, during training on the Maximum Cut problem, the agent discovered effective circuits with a regular structure, which we denote as R_{yz}-connected, characterized by circuits with only R_{yz} gates connecting all the qubits. We test a specific member of this family, referred to as Linear circuit, by comparing it to state-of-the-art quantum algorithms on a range of Quadratic Unconstrained Binary Optimization (QUBO) problems with varying underlining graph topologies and sizes. Our findings reveal that the Linear circuit is able to find high-quality solutions for the Maximum Cut problem, but it is less effective for the other tasks. Finally, we discuss how R_{yz}-connected could be implemented on the hardware.

In summary, our contributions are the following:

- we introduce a Reinforcement Learning agent designed to autonomously generate Variational Quantum Circuits for solving optimization problems;
- we show that the RL-generated circuits achieve high approximation ratios across a variety of optimization problems;
- we identify and analyze a novel family of circuits discovered by the agent, termed R_{yz}-connected, which generalizes effectively and can be applied to other instances of the same problem.

Overall, our study shows the potential of RL-based approaches for constructing effective variational quantum circuits. Moreover, this methodology holds promise for broader applications within quantum computing, including designing more general circuits or optimizing circuit properties.

2. Variational Quantum Algorithms

Variational Quantum Algorithms (VQAs) [1] represent a class of hybrid quantum algorithms that can be used to tackle certain optimization problems and are characterized by the presence of a parametric quantum circuit, denoted as ansatz, and a classical optimizer. The goal of the classical optimizer is to adjust the circuit parameters by minimizing a predefined cost function, in such a way that the execution of the circuit with optimized parameters yields the solution to the given optimization problem. In recent years, VQAs have known large diffusion due to their potential to tackle complex problems using near-term quantum devices. Indeed, the variational approach enables the utilization of shallower circuits, making these algorithms more resilient to noise and qubit decoherence.

One of the most widely utilized VQAs is the Variational Quantum Eigensolver (VQE) [15, 16], which allows to determine the ground state of a given Hamiltonian H, and is largely applied in quantum chemistry [17, 18]. Specifically, this algorithm employs a suitable ansatz capable of generating the final parametric state $|\psi(\theta)\rangle$, where θ denotes a parameter vector. The objective is to identify the circuit parameters that minimize the cost function $\langle\psi(\theta)|H|\psi(\theta)\rangle$, also denoted as $\langle H\rangle$, through a variational approach. Notice that, when using real quantum hardware, we do not have direct access to the expectation $\langle\psi(\theta)|H|\psi(\theta)\rangle$, but this quantity can be estimated by executing the circuit a number n_{shots} of times and computing the quantity:

$$\langle H\rangle^* = \langle\psi(\theta)|H|\psi(\theta)\rangle^* = \frac{1}{n_{\text{shots}}}\sum_{i=1}^{n_{\text{shots}}}\langle\tilde{\psi}_i|H|\tilde{\psi}_i\rangle, \tag{1}$$

where $|\tilde{\psi}_i\rangle$ denotes the basis state obtained after performing the measurement at the i-th execution of the circuit.

Another widely adopted VQA is the Quantum Approximate Optimization Algorithm (QAOA) [19, 20], often employed to address combinatorial optimization problems [21–27]. Similarly to VQE, QAOA

aims to minimize the expectation value of a designated cost Hamiltonian on the circuit's final state. However, QAOA employs a predefined circuit comprising an initial layer of Hadamard gates followed by p layers, each implementing two parametric operators: the cost operator, which depends on the cost Hamiltonian, and the mixer operator, which allows to further explore the solution space. QAOA can be considered as a discretized form of continuous-time quantum evolution.

One variant of QAOA is the multi-angle QAOA (ma-QAOA) [28], which shares the same objective and circuit structure as QAOA, but uses distinct parameters for each parametric gate associated with the cost and mixer operators. Whereas ma-QAOA enhances space exploration capability using the same circuit structure and depth as QAOA, it also increases the computational load on the classical optimizer due to the higher number of parameters, rendering a careful trade-off necessary.

Another variant, QAOA+ [29], extends the ansatz used by QAOA with $p = 1$ by introducing an additional problem-independent layer consisting of R_{zz} gates, and a mixer layer with R_x gates. This results in a circuit with Hadamard, cost and mixer layer, and the two newly introduced components. Notably, this structure is not repeated. Since each gate in the additional layers is independently parameterized, the QAOA+ ansatz results in a deeper circuit with $2n-1$ additional parameters compared to QAOA with $p = 1$. Similar to ma-QAOA, the higher number of parameters enables a broader exploration of the solution space, but at the same time increases the computational burden on the classical optimizer.

However, one critical challenge in the application of VQAs is the identification of suitable circuits for addressing specific problems [30–34]. An ideal ansatz should have a limited number of gates and depth to mitigate noise and decoherence, utilize the native gates of the hardware to simplify implementation and reduce circuit depth, as well as effectively sample the correct solution to the optimization problem. In particular, one significant obstacle to finding the correct solution to the optimization problem lies in the phenomenon of barren plateaus [35–41]. Barren plateaus occur when the gradient of the cost function exponentially vanishes as the system size increases, resulting in a flat landscape in which the classical optimizer struggles to escape local optima and may not be able to find good parameters for the circuit. To address this challenge, strategies for selecting suitable ansatzes have been explored. These include leveraging problem-specific properties [3–5, 19] and employing adaptive algorithms [6–8].

Adaptive Variational Quantum Algorithms are a family of methods which dynamically construct the quantum circuit by iteratively adding and removing gates throughout the algorithm execution. This approach allows to explore various gate configurations and choose the most suitable circuit structures for a given problem. One category of adaptive VQAs, proposed in the literature, includes ADAPT-VQE [42], qubit-ADAPT-VQE [43], QEB-ADAPT-VQE [44], and Overlap-ADAPT-VQE [45], adaptive variants of VQE. These algorithms aim to find the ground state of a Hamiltonian operator representing the energy of a molecule, and their distinguishing feature lies in the construction of the ansatz, which is achieved by adding gates selected from a pool which depends on the specific chemistry problem being addressed. Additionally, there exists an adaptive version of QAOA [46], which, at each iteration, determines the most appropriate mixer for the following layer of the circuit. Other adaptive VQAs adopt a genetic approach [47–49], employ more generalized Machine Learning techniques for circuit construction [50–52] or make use of Reinforcement Learning as described in Section 3.1.

3. Reinforcement Learning

Our objective is to use a Machine Learning model to design new circuits capable of finding the ground state of a Hamiltonian. This problem requires exploring a large solution space of potential circuits, and is strongly influenced by the specific Hamiltonian under consideration. Consequently, creating an exhaustive dataset of quantum circuits for *supervised* learning is impractical. Thus, we choose to use a Reinforcement Learning (RL) approach, where an agent interacts with an environment to achieve a specific goal. This enables us to generate data, i.e. quantum circuits, and evaluate their performance during training. Because of the complexity of this topic, in this section we only describe the fundamental concepts of RL and the selected algorithm. For a comprehensive overview of this

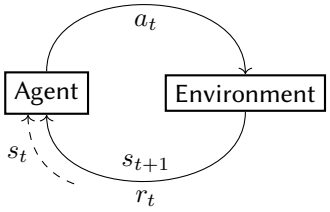


Figure 1: Interaction between agent and environment. At time step t the agent observes the environment's state s_t and acts with action a_t on the environment, which gives reward r_t to the agent. At the next time step, the agent will observe the new state s_{t+1}.

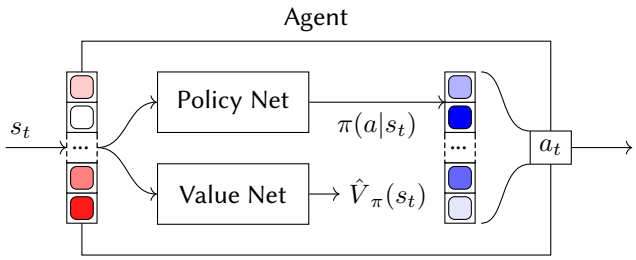


Figure 2: State s_t is processed by the agent's neural networks. The value network outputs an estimate $\hat{V}_\pi(s_t)$ of the value function (4), while the policy network outputs a probability distribution $\pi(a|s_t)$ on the actions. Action a_t is sampled from this probability distribution.

technology, the interested reader may refer to [53].

In RL, the agent interacts with the environment in discrete time steps. At each step t, the environment is in a particular *state* s_t, which is observed by the agent. Based on this observation, the agent performs an *action* a_t, which may have an effect on the environment, causing the transition to a new state s_{t+1}. The agent receives a *reward* r_t, a value depending on the quality of the action taken (see Fig. 1). Starting from an initial state and cyclically interacting with the environment until a termination condition is reached, the agent completes an *episode*. Let us consider a practical example, where the agent is a robot whose goal is to move an object from one place to another in an environment constituted by a table. In this scenario, the actions correspond to the movements that the robot's mechanical arms can perform, the state of the environment is represented by the current position of the object on the table, and the reward is a function which increases as the distance between the object and its target location decreases. An episode starts with the object in an initial position on the table and ends when the robot has successfully placed the object at the target position.

Spanning through multiple episodes, the objective of the agent is to learn which actions to perform in order to obtain the highest cumulative reward, or *return*. Specifically, the return at time step t is the weighted sum of rewards starting from t until the end of the episode:

$$g_t = \sum_{k=0}^{t_e} \gamma^k r_{t+k+1} \tag{2}$$

where t_e is the number of remaining steps until the end of the episode and $\gamma \in (0, 1)$ represents the discount factor. This factor serves two purposes: facilitating convergence and balancing short-term versus long-term rewards.

When choosing a new action in each state, the agent follows a *policy* π, which is a probability distribution on the set of all actions, conditioned on the state:

$$\pi(a|s) = P(a_t = a \mid s_t = s) \tag{3}$$

Various methods exist to learn an optimal policy. In this work we use *Proximal Policy Optimization* (PPO), which is currently a state-of-the-art Deep Reinforcement Learning algorithm introduced by

OpenAI [54]. PPO makes use of two neural networks with the same structure, respectively called *policy* and *value network* (Fig. 2). Both networks receive an appropriate representation of the environment's state as input. The policy network then outputs a probability distribution on the possible actions at time step t, from which action a_t is then sampled. On the other hand, the value network outputs a single value estimating the *value function*, which quantifies the quality of the state in terms of expected return when following the policy π:

$$V_\pi(s) = \mathbb{E}_\pi[\, g_t \mid s_t = s \,] \tag{4}$$

While the policy network effectively chooses the actions to perform, the value network is used to estimate the so-called *advantage function*:

$$A_\pi(s, a) = Q_\pi(s, a) - V_\pi(s) \tag{5}$$

Here, Q_π represents the *state-action value function*, which is the expected return obtained by starting in state s, taking action a and then following the policy π:

$$Q_\pi(s, a) = \mathbb{E}_\pi[\, g_t \mid s_t = s,\ a_t = a \,] \tag{6}$$

Thus, the advantage measures the potential benefit of taking action a in state s, independently of the quality of that state. In order to learn an optimal policy, PPO performs gradient ascent with the goal of maximizing an objective function that incorporates both the advantage and a loss that helps the value network in learning how to better estimate the real value function. Moreover, a clipping mechanism is implemented to avoid a too large update to the policy during training. For further details on the loss and implementation, we refer to the original articles [54, 55].

3.1. Reinforcement Learning for Quantum Computing

In recent years, RL techniques have been applied to various challenging tasks in quantum computing. Some studies focus on optimizing circuit parameters [9–11], while others tackle circuit learning tasks consisting in generating quantum circuits to transform an initial state into a target state [56–60]. This task is particularly useful because it can be applied to the crucial step of quantum circuit compiling. Additionally, RL has been used to learn optimization strategies aimed at reducing circuit depth and gate count [61]. Furthermore, RL algorithms have been applied to the task of designing quantum circuits tailored for Machine Learning [13] and optimization [14] problems. Specifically, [13] discusses the use of RL in designing parameterized quantum circuits for classification tasks, while [14] employs RL to find suitable ansatzes for VQE to determine the ground state energy of specific molecules. It is worth noting that the approach taken by [14] tackles our same task of finding circuits to generate the ground state of a Hamiltonian. However, the RL agent is specifically tailored for chemistry problems, and the architecture may not be directly applicable to other domains. In contrast, our work introduces a novel RL algorithm designed for more general optimization problems and differs in many crucial components: architecture for the agent, state representations, rewards, and available actions.

4. Agent Design and Training

In this study we propose a RL-based algorithm called *Reinforcement Learning for Variational Quantum Circuits* (RLVQC), designed to learn how to build new quantum circuits aimed at finding the ground state of the Hamiltonian associated with a given optimization problem. In this section we describe the different components of RLVQC, including environment, actions, and reward. Then, we present the experimental setup and specify some implementation details useful to reproduce the experiments.

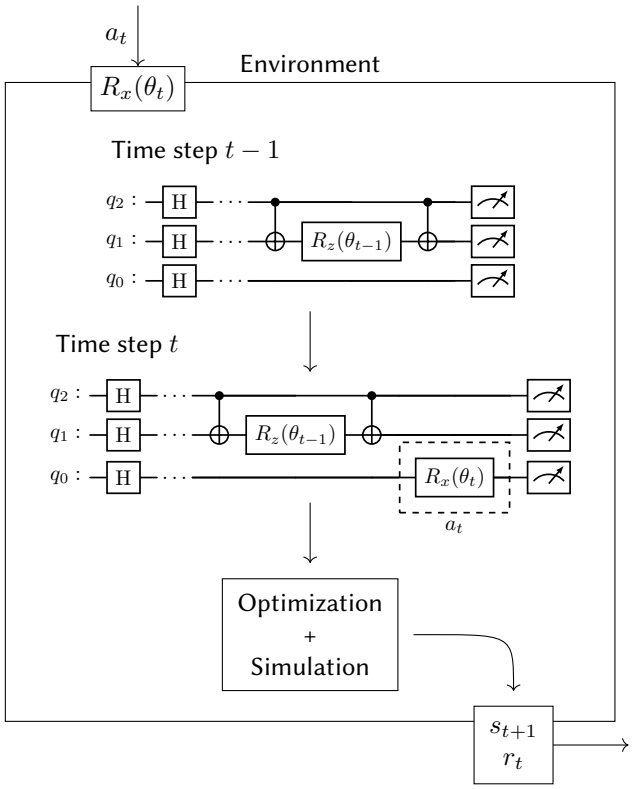


Figure 3: When the environment receives action a_t, the corresponding gate is added to the circuit. Then, its parameters are optimized and the circuit is simulated to obtain the next state s_{t+1}, which is sent back to the agent with the corresponding reward r_t.

4.1. Environment, Actions, Reward

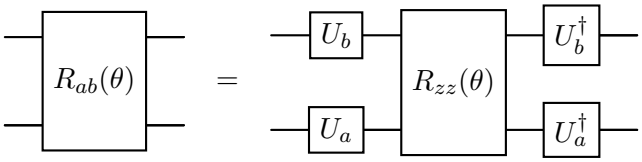
In RLVQC, the environment is represented by a parametric circuit with n qubits. At each step t, the agent performs an action consisting in adding a new gate to the circuit. The environment configuration at the beginning of each episode is represented by a circuit with a single layer of Hadamard gates. The action set $\mathcal{A}$ comprises the gates that the agent can insert into the circuit. Specifically, $\mathcal{A}$ is the union of the following sets:

- $\mathcal{S} = \{R_a^i(\theta) \mid a \in \{x, y, z\},\ i = 0, ..., n-1\}$
- $\mathcal{D} = \{R_{ab}^{ij}(\theta) \mid a, b \in \{x, y, z\},\ i, j = 0, ..., n-1,\ i < j\}$

where R_a^i is a single rotation gate applied to qubit i and R_{ab}^{ij} is a double rotation applied to a pair of different qubits i and j. Formally, denoting as σ_a the Pauli-a matrix for all $a, b \in \{x, y, z\}$, the double rotations are defined as:

$$R_{ab}(\theta) = e^{-i\frac{\theta}{2}\sigma_a \otimes \sigma_b} \tag{7}$$

A specific double rotation $R_{ab}(\theta)$ can be decomposed in simpler gates as follows:



where U_a and U_b are gates which map the a and b axis into the z axis respectively. Thus, $U_x = H$, $U_y = R_x(\frac{\pi}{2})$, and $U_z = I$. These double rotations are able to generate entanglement between qubits.

The gates in the set $\mathcal{A}$ are chosen because they exhibit identity-like behavior when their parameters are set to 0. This property is advantageous because starting from a circuit with optimized parameters

and adding a new gate with parameters set to 0 allows the optimization process to begin from a favorable initial point rather than a random one. This approach potentially reduces computational time and mitigates the risk of converging to a local minimum [62].

At each time step t, the agent chooses a gate as an action to be applied to the environment, which is updated by adding that gate with parameter $\theta_t = 0$ to the circuit. Notice that all the gates are independently parametrized. Subsequently, the circuit parameters are optimized using the classical optimizer COBYLA[1], which has been shown to be effective in noise-free scenarios without demanding an excessive computational cost [63, 64]. COBYLA is run for a maximum of 1000 iterations with the goal of minimizing the cost function given by the expectation of the problem Hamiltonian on the final state of the circuit, defined in (1). Such expectation is estimated by simulating the circuit 1000 times[2]. The choice of estimating the expectation instead of computing it exactly is made to reflect the use of an actual quantum computer. Indeed, on real quantum devices, we do not have direct access to the quantum state, but we can only estimate it by performing multiple measurements.

After COBYLA converges, the circuit using the optimized parameters is executed an additional 1000 times to obtain the estimated probability distribution of the final state of the circuit, in the form of a vector of 2^n real-valued elements. This probability distribution represents the next state observed by the agent. Notice that the design of an effective representation for the environment state is not trivial and depends on several factors related to the specific task and the agent's architecture. For this reason, designing new state representations specifically tailored for quantum systems can be considered an important and wide research area that goes beyond the scope of this paper.

The reward at time step t is defined to align with our goal of minimizing the expectation value of the Hamiltonian and the circuit depth while the agent learns to maximize the return (2). Specifically, the reward is expressed as:

$$r_t = -\langle H \rangle_t^* - \beta \cdot d_t, \tag{8}$$

where $\langle H \rangle_t^*$ is an estimate of the current expectation value as defined in (1) and d_t denotes the current circuit depth, weighted by the β hyperparameter, which is fixed at 0.015 in our experiments. This reward is straightforward and directly tied to the quality of the circuit built at time step t: the first term incentivizes the minimization of $\langle H \rangle^*$, which is our primary goal, while the second term drives the agent to prefer shallower circuits, which are more resilient to noise and qubit decoherence. It is worth noting that in this phase the circuit depth is computed after expressing the circuits in terms of the basis gates $\{H, R_x, R_y, R_z, R_{zz}\}$. In particular, while the double rotation R_{zz} is directly used, the others are represented in terms of it. Finally, we remark that designing an effective reward function that combines different goals is again a critical aspect in RL which allows for great flexibility. Moreover, it is important to highlight that the suitability of a particular reward function may vary depending on the nature of the task and specific requirements. A complete visual overview of the RL pipeline is represented in Fig. 3.

4.2. Training Details

RLVQC uses the version of PPO implemented as described by OpenAI [54, 55], with default hyperparameters. This implementation employs two multi-layer fully-connected neural networks, each sharing the same structure but with separate learnable parameters, for the policy and value network. With the problem size denoted as n, the input layer of each neural network comprises 2^n neurons, matching the size of the vector representing the environment state (see Section 4.1). The output layer for the policy network has size $|\mathcal{A}|$, which depends on the number of problem variables. Agent training comprises 64 *epochs*, with each epoch being a collection of 384 action steps[3] after which the parameters of the PPO's neural networks are updated. Each epoch consists of multiple episodes, each starting from a circuit with only Hadamard gates and ending when a termination condition is met. Specifically, an episode

[1]We use the SciPy implementation of COBYLA, keeping its default hyperparameters. The documentation is available here: https://docs.scipy.org/doc/scipy/reference/optimize.minimize-cobyla.html

[2]Circuit simulation is performed using Qiskit QASM Simulator. The documentation is available here: https://qiskit.github.io/qiskit-aer/stubs/qiskit_aer.QasmSimulator.html

[3]This number is obtained by having multiple processes performing 64 action steps each in our default implementation.

terminates either when a maximum of $2 \cdot n$ actions have been performed or earlier, if the reward does not improve with new actions. The latter mechanism is controlled by the *patience* hyperparameter. Patience is initially set to 3 and decreases every time a step ends with a worse reward than the highest found during the current episode. Otherwise, it is increased, but only up to its initial value. Stopping the episode earlier avoids spending time steps on actions that are unlikely to improve the reward. Throughout training, we keep track of the circuit with the highest reward, which is then analyzed after training is completed.

4.3. Problem Instances

In our experiments, we evaluate RLVQC on the task of finding the solution of optimization problems that can be formulated as *Quadratic Unconstrained Binary Optimization* (QUBO) problems [65]. The general formulation of a QUBO problem is the following:

$$\min_{x \in \{0,1\}^n} x^T \mathbf{Q} x \tag{9}$$

where $x = \{x_1, \ldots, x_n\} \in \{0,1\}^n$ and $\mathbf{Q}$ denotes an upper triangular or symmetrical $n \times n$ matrix. QUBO problems can be expressed as equivalent Ising problems through a change of variable [66]. Leveraging this formulation, the objective becomes to identify the ground state of a Hamiltonian operator, a task which can be accomplished using our algorithm RLVQC.

We address three diverse optimization problem types: Maximum Cut, Maximum Clique, and Minimum Vertex Cover, all formulated as QUBO problems [65, 67]. It is important to highlight that Minimum Vertex Cover and Maximum Clique involve constraints. To incorporate these constraints into the QUBO formulation, penalty terms are directly integrated into the cost functions. Each penalty term is a function that yields a value of 1 when a constraint is violated and 0 when it is satisfied, multiplied by a weight coefficient. These coefficients are chosen to ensure that all the feasible solutions have a lower expectation than the infeasible ones, thereby guiding the algorithm towards favoring feasible solutions over infeasible ones. For each problem type, we consider three different graph topologies:

- 3-regular, a graph where each vertex is connected to exactly three neighbors;
- 2D-grid, a two-dimensional integer lattice graph where each vertex can have up to 4 neighbours;
- Star graph, characterized by one central node connected to all other nodes, with no further connections among the non-central nodes.

For each problem type, we explore graphs with $n = 8$ and $n = 14$ vertices across each topology, resulting in a total of 18 trained agents.

4.4. Evaluation Metrics

Our main goal is to assess whether it is possible to use a RL agent to discover ansatzes capable of sampling high-quality solutions to optimization problems. We aim to evaluate both the accuracy of the solutions found and the characteristics of the resulting circuits. To this purpose, the most relevant metric we use is the *approximation ratio* (A.R.), defined as:

$$\text{A.R.} = \frac{\langle H \rangle^* - \langle H \rangle_{\max}}{\langle H \rangle_{\min} - \langle H \rangle_{\max}}, \tag{10}$$

where $\langle H \rangle_{\min}$ and $\langle H \rangle_{\max}$ represent the minimum and maximum values of the achievable expectations, respectively, and $\langle H \rangle^*$ is an estimate of the expectation, as defined in (1). The intuition behind this metric is that the ratio $\frac{\langle H \rangle^*}{\langle H \rangle_{\min}}$ approaches 1 when $\langle H \rangle^*$ closely approximates $\langle H \rangle_{\min}$, indicating that the circuit allows to sample with high probability states whose energy values are close to the minimum. To guarantee that both the numerator and the denominator of the ratio have negative values, we subtract $\langle H \rangle_{\max}$ from both quantities. This ensures that the final approximation ratio falls between 0 and 1,

with closer proximity to 1 when the solution quality is higher. It is worth noting that in the case of the Maximum Cut problem, $\langle H \rangle_{\max} = 0$.

However, the approximation ratio alone does not indicate if the constraints in Minimum Vertex Cover and Maximum Clique are satisfied. Hence, in Table 1, we include a threshold corresponding to the lowest approximation ratio of feasible solutions. Circuits with approximation ratios below this threshold are more likely to sample infeasible solutions. Notice that, for the unconstrained Maximum Cut problem, the feasibility threshold is 0.

Finally, we analyze the composition of the final circuit, reporting the number of single and two-qubit gates, as well as the circuit depth. These metrics are computed after expressing the circuit in terms of the gates $\{Cx, R_x, R_y, R_z\}$, where Cx denotes the CNOT gate, and and applying some simplifications using the Qiskit transpiler with optimization level 1, which is the default setting.

5. Results and Discussion

In this section, we discuss the results obtained by executing the RLVQC algorithm according to the experimental protocol outlined in Section 4, using the metrics detailed in Section 4.4. To establish a baseline for comparison, we also run QAOA with $p = 1$ on the same problem instances and analyze the same set of metrics. Since QAOA is a stochastic algorithm which strictly depends on the initial parameters, we perform 10 executions of the algorithm for each problem instance. We choose random initial parameters for each run, and then average the approximation ratios obtained. This comparative analysis is presented in Section 5.1.

Notably, during training on the Maximum Cut problem, the agent discovered circuits with a common structure and high approximation ratios. This observation led us to identify a novel family of ansatzes, which we denote as R_{yz}-connected circuits, as discussed in Section 5.2. In particular, we conduct an analysis of the most intuitive circuit within this family, which we refer to as *Linear* circuit, by examining its performance across a broader set of graph topologies and different problem types. Finally, in Section 5.3, we discuss the straightforward implementability of the R_{yz}-connected circuits.

5.1. RLVQC Results

In Table 1 we report the results relative to the solutions and circuits found with RLVQC and QAOA with $p = 1$ in terms of the metrics described in Section 4.4. Specifically, the table shows the approximation ratio (A.R.) of the solutions, the approximation ratio threshold (A.R. Thresh.) for feasibility, the number of single and two-qubit gates, and the circuit depth.

By examining Table 1, it can be noticed that the approximation ratios found by RLVQC is significantly influenced by the type of optimization problem being addressed. RLVQC consistently outperforms QAOA in Maximum Cut and Minimum Vertex Cover instances, with one exception observed in the case of the star-topology graph instances with $n = 14$ vertices for both problem types, where RLVQC performs worse. Moreover, for the Minimum Vertex Cover instance with star-topology and $n = 14$, RLVQC falls below the threshold for feasibility in terms of approximation ratio. The particularly strong performance of RLVQC on Maximum Cut instances, especially on instances with $n = 8$ qubits where RLVQC consistently achieves an approximation ratio of 0.99, prompted us to conduct a deeper analysis of these circuits. Interestingly, we observed that the agent exclusively added R_{yz} rotations to these circuits during their construction, without performing any single-qubit actions. This observation suggests that circuits containing R_{yz} rotations may be particularly effective for addressing Maximum Cut problems. In contrast, for Maximum Clique instances, RLVQC consistently yields lower approximation ratios compared to QAOA, except for the instance built from a 2D-grid-topology graph with $n = 8$ vertices. However, neither algorithm attains satisfactory results, as the threshold for feasibility is never met for any graph topology.

Upon analyzing the count of single and two-qubit gates, as well as the circuit depth, we observe that RLVQC generally employs a number of gates comparable to QAOA but exhibits a higher depth. Notably, there are cases where the RLVQC circuit has a gate count that is significantly lower than in

Size	Problem	Graph	A.R. RLVQC	A.R. QAOA	A.R. Thresh.	Single-qubit RLVQC	Single-qubit QAOA	Two-qubit RLVQC	Two-qubit QAOA	Depth RLVQC	Depth QAOA
8	Maximum Cut	3-regular	0.99	0.75	0.00	37	36	14	24	38	12
		2D-grid	0.99	0.63	0.00	37	34	14	20	38	12
		Star graph	0.99	0.70	0.00	37	31	14	14	38	24
	Maximum Clique	3-regular	0.75	0.81	0.91	62	48	22	32	40	15
		2D-grid	0.83	0.78	0.94	47	50	20	36	43	15
		Star graph	0.70	0.80	0.95	44	53	18	42	38	21
	Minimum Vertex Cover	3-regular	0.99	0.83	0.77	63	44	18	24	43	12
		2D-grid	0.96	0.85	0.73	28	42	6	20	16	12
		Star graph	0.93	0.83	0.69	31	39	8	14	20	24
14	Maximum Cut	3-regular	0.92	0.74	0.00	67	63	26	42	56	12
		2D-grid	0.76	0.65	0.00	64	61	24	38	52	12
		Star graph	0.50	0.71	0.00	49	55	14	26	32	42
	Maximum Clique	3-regular	0.64	0.79	0.98	80	126	26	140	48	34
		2D-grid	0.72	0.78	0.99	78	128	24	144	35	34
		Star graph	0.63	0.76	0.99	97	134	34	156	55	39
	Minimum Vertex Cover	3-regular	0.85	0.78	0.74	87	77	28	42	54	12
		2D-grid	0.87	0.87	0.74	72	75	18	38	33	12
		Star graph	0.65	0.82	0.71	55	69	14	26	32	42

Table 1
Comparative analysis of the performance of RLVQC and QAOA with $p = 1$ on instances of different problem types built from graph with various topologies and $n = 8$ and $n = 14$ vertices. The table reports the approximation ratios relative to the circuits with the highest reward found by RLVQC and the average approximation ratio reached by QAOA across 10 executions of the algorithm, along with the threshold for feasibility. Additionally, the table shows the number of single-qubit and two-qubit gates, as well as the circuit depth.

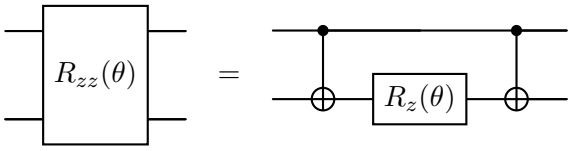
QAOA, yet the depth increases. An example is given by Maximum Clique with $n = 14$, where the count of two-qubit gates amounts to less than 25% of the two-qubit gates used by QAOA. Finally, it is important to notice that, while circuits produced by RLVQC tend to have higher depth compared to QAOA, adjusting the β hyperparameter in the reward function (8) allows for further penalizing deep circuits. This adjustment may guide RLVQC toward the preference of low-depth circuits.

In summary, RLVQC successfully found solutions with approximation ratios comparable to QAOA, or superior in the case of the Maximum Cut problem. It is noteworthy that these results are obtained using a relatively straightforward design for the RL agent, environment and reward. Therefore, there is a considerable potential to further improve these results by better understanding how to design these components for quantum computing tasks.

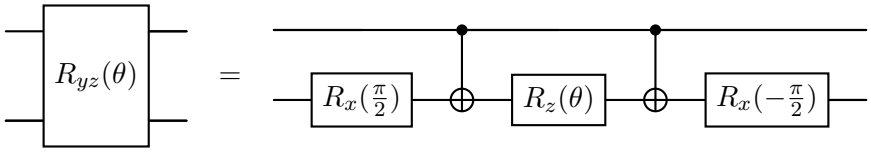
5.2. R_{yz}-connected Ansatzes

Here we describe the family of R_{yz}-connected ansatzes discovered during training on the Maximum Cut problem. This family is composed of circuits with an initial layer of Hadamard gates, followed by $n-1$ R_{yz} rotations, which are applied such that each new R_{yz} adds only one qubit to the "chain" of connected qubits. More formally, calling C_j the graph representing the connectivity of the circuit after applying the first j R_{yz} rotations, the C_j graphs associated to each R_{yz}-connected ansatz are connected for all $j = 1, \ldots, n-1$.

Recall that a R_{zz} gate is equivalent to two Cx gates with a R_z rotation between them on the target qubit, as shown below:



Using the more general definition of double rotations provided in (7) and the above expansion of the R_{zz} gate, we have that R_{yz} rotations can be decomposed as follows:



A notable property of R_{yz}-connected circuits is that two basis states that have all bits flipped w.r.t. to one another have the same probability of being measured. This characteristic makes the circuits especially well-suited for problems that possess this symmetry property, such as the Maximum Cut problem.

Among the elements of the R_{yz}-connected family, we focus on a specific circuit, which we denote as *Linear* circuit, where each R_{yz} gate connects one qubit to the next one (see Fig. 4). It is worth noting that each block of the figure represents the decomposition of a R_{yz} gate as explained in Section 5.2. The first $R_x(\frac{\pi}{2})$ gate is omitted since it behaves as the identity when applied after a Hadamard gate.

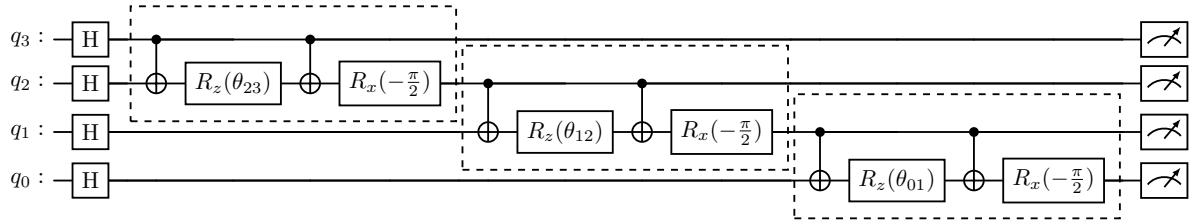


Figure 4: Linear circuit, a specific element of the family of R_{yz}-connected ansatzes found during training on Maximum Cut.

The choice of analyzing this particular R_{yz}-connected ansatz is motivated by its uniform application of R_{yz} gates across all qubits, making it the most straightforward configuration within the family.

The primary objective of our analysis is to evaluate the quality of solutions obtained by the Linear circuit on the Maximum Cut problem, but across a broader range of underlying graph topologies. We maintain the use of 3-regular, 2D-grid, and star graphs, and further include cycle graphs, where each vertex is connected to exactly two other vertices, forming a closed loop. Additionally, we introduce Erdős–Rényi graphs generated from a set of vertices, with edges independently connected based on fixed probabilities of 0.2, 0.5, and 0.8. Subsequently, we evaluate the performance of the Linear circuit on other problem types, specifically Maximum Clique and Minimum Vertex Cover, using the same underlying topologies. The algorithms are tested on instances with 8, 14, and 16 vertices, which represents the largest tractable size with our available hardware. However, we present results only for the 16-vertex instances, as the findings for smaller instances are analogous. We compare the performance of the Linear circuit with state-of-the-art quantum algorithms, including QAOA with depths $p = 1$ and $p = 2$, QAOA+, and ma-QAOA (see Section 2). Each algorithm is executed 10 times on each problem instance and the results are averaged. We choose random initial parameters for each execution.

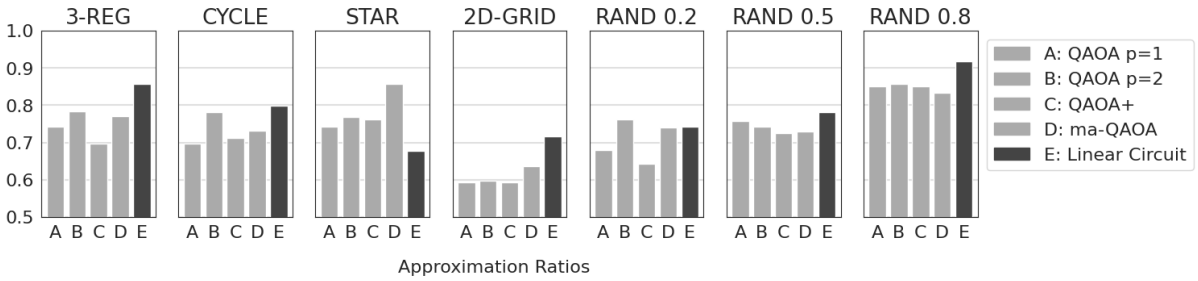


Figure 5: Approximation Ratio obtained by the tested quantum algorithms on Maximum Cut instances on graphs with 16 vertices.

Fig. 5 represents the average approximation ratios obtained by executing the algorithms on the Maximum Cut problem across diverse graph topologies, as described above. A notable observation is that the Linear circuit often achieves the best approximation ratio. However, there are exceptions, such as the Erdős–Rényi random graph with an edge probability of 0.2, where the Linear circuit performs slightly worse than QAOA with $p = 2$. Moreover, on the star-topology graph, the approximation ratio obtained by the Linear circuit is significantly lower than that obtained by all other algorithms. We believe that further investigation into the cause of this behavior can be useful.

In contrast, when applying the same algorithms to Maximum Clique and Minimum Vertex Cover problems, we observe that the approximation ratios of the Linear circuit are lower than those achieved by the other algorithms. Among the algorithms tested, QAOA with $p = 2$ consistently achieves the highest approximation ratio across all graph topologies, except for the Erdős–Rényi graphs with edge probabilities of 0.5 and 0.8, where QAOA with $p = 1$ and ma-QAOA perform the best, respectively. These findings support our observation that R_{yz}-connected ansatzes are suitable for problems where the cost of a solution remains the same if all bits are flipped, which include Maximum Cut, but not Maximum Clique and Minimum Vertex Cover problems.

Overall, we can conclude that RLVQC was able to find a reasonable ansatz, consistently capable of finding higher approximation ratios than other quantum state-of-the-art algorithms on the Maximum Cut problem across various underlining graph topologies.

5.3. Implementability of R_{yz}-connected Ansatzes

In this subsection we discuss the ease of implementability of the R_{yz}-connected ansatzes on specific real quantum devices. In modern technologies, R_z rotations can be implemented with negligible error and time for any rotation angle [68], while implementing a R_x rotation requires a specific application time. Since any arbitrary rotation can be achieved by combining R_z and $R_x(\frac{\pi}{2})$ rotations, superconducting computers are often calibrated to perform only R_z and $R_x(\frac{\pi}{2})$ rotations, and not the full set of R_x rotations. Our R_{yz}-connected ansatzes are very good in such a framework, since we can observe that the R_{yz} rotations can be decomposed into R_z and $R_x(-\frac{\pi}{2})$ rotations, and Cx gates. In particular, the only single-qubit parametric gates are the R_z rotations, which can be executed virtually without error. Additionally, to implement the $R_x(-\frac{\pi}{2})$ rotations, we can substitute each of these gates with a $R_x(\frac{\pi}{2})$. R_{yz}-connected ansatzes maintain their performance after this substitution, allowing us to exploit the calibration of the hardware.

Moreover, the R_{yz}-connected family offers advantages when mapping the logic circuit to the quantum device topology. Specifically, we can select a particular element of the R_{yz}-connected ansatzes based on the connectivity of the available hardware. This approach reduces the number of SWAP gates required to connect qubits that are connected in the logical circuit but not in the hardware. This approach results in resource savings and reduces errors caused by an increased number of gates and circuit depth.

6. Conclusions and Future Directions

In this study, we introduced RLVQC, a Reinforcement Learning algorithm designed to learn how to construct quantum circuits to solve optimization problems through a quantum variational approach. Overall, RLVQC demonstrated its ability to build circuits able to generate solutions with approximation ratios comparable to those obtained by QAOA, and particularly high in the case of the Maximum Cut problem.

Furthermore, we analyzed the R_{yz}-connected ansatzes, a specific family of circuits discovered by the agent during its training on the Maximum Cut problem. Our investigation revealed that the R_{yz}-connected ansatzes can achieve high approximation ratios, superior to those obtained with other state-of-the-art algorithms on Maximum Cut instances.

These results suggest that approaches employing Reinforcement Learning agents to construct variational circuits hold promise. Given the flexibility of RL, there is potential for future research aimed at improving each of the components of RLVQC, including the representation of the environment's state, the agent's network architecture, the action set, and the reward function. Each of these elements may be designed, for example, to take advantage of the peculiarities of a given problem, a specific hardware or even adapt to the requirements set by the task at hand. More generally, RL looks promising for researchers aiming to address challenges characterized by large solution spaces within the quantum computing domain.

Acknowledgments

We acknowledge the financial support from ICSC - "National Research Centre in High Performance Computing, Big Data and Quantum Computing", funded by European Union – NextGenerationEU. We also acknowledge the support and computational resources provided by E4 Computer Engineering S.p.A.

References

[1] M. Cerezo, A. Arrasmith, R. Babbush, S. C. Benjamin, S. Endo, K. Fujii, J. R. McClean, K. Mitarai, X. Yuan, L. Cincio, P. J. Coles, Variational quantum algorithms, Nature Reviews Physics 3 (2021) 625–644. URL: http://dx.doi.org/10.1038/s42254-021-00348-9. doi:10.1038/s42254-021-00348-9.

[2] J. Preskill, Quantum computing in the nisq era and beyond, Quantum 2 (2018) 79. URL: http://dx.doi.org/10.22331/q-2018-08-06-79. doi:10.22331/q-2018-08-06-79.

[3] J. J. Meyer, M. Mularski, E. Gil-Fuster, A. A. Mele, F. Arzani, A. Wilms, J. Eisert, Exploiting symmetry in variational quantum machine learning, PRX Quantum 4 (2023) 010328. URL: https://link.aps.org/doi/10.1103/PRXQuantum.4.010328. doi:10.1103/PRXQuantum.4.010328.

[4] I. N. M. Le, O. Kiss, J. Schuhmacher, I. Tavernelli, F. Tacchino, Symmetry-invariant quantum machine learning force fields, 2023. arXiv:2311.11362.

[5] D. Wierichs, R. D. P. East, M. Larocca, M. Cerezo, N. Killoran, Symmetric derivatives of parametrized quantum circuits, 2023. URL: https://arxiv.org/abs/2312.06752. arXiv:2312.06752.

[6] G. Turati, M. Ferrari Dacrema, P. Cremonesi, Benchmarking adaptative variational quantum algorithms on qubo instances, in: 2023 IEEE International Conference on Quantum Computing and Engineering (QCE), volume 01, 2023, pp. 407–413. doi:10.1109/QCE57702.2023.00053.

[7] D. Claudino, J. Wright, A. J. McCaskey, T. S. Humble, Benchmarking adaptive variational quantum eigensolvers, Frontiers in Chemistry 8 (2020). URL: https://www.frontiersin.org/articles/10.3389/fchem.2020.606863. doi:10.3389/fchem.2020.606863.

[8] A. Mukherjee, N. F. Berthusen, J. C. Getelina, P. P. Orth, Y.-X. Yao, Comparative study of adaptive variational quantum eigensolvers for multi-orbital impurity models, Communications Physics 6 (2023) 4. URL: https://doi.org/10.1038/s42005-022-01089-6. doi:10.1038/s42005-022-01089-6.

[9] M. M. Wauters, E. Panizon, G. B. Mbeng, G. E. Santoro, Reinforcement-learning-assisted quantum optimization, Phys. Rev. Res. 2 (2020) 033446. URL: https://link.aps.org/doi/10.1103/PhysRevResearch.2.033446. doi:10.1103/PhysRevResearch.2.033446.

[10] S. Khairy, R. Shaydulin, L. Cincio, Y. Alexeev, P. Balaprakash, Learning to optimize variational quantum circuits to solve combinatorial problems, Proceedings of the AAAI Conference on Artificial Intelligence 34 (2020) 2367–2375. URL: http://dx.doi.org/10.1609/aaai.v34i03.5616. doi:10.1609/aaai.v34i03.5616.

[11] J. Yao, M. Bukov, L. Lin, Policy gradient based quantum approximate optimization algorithm, 2020. arXiv:2002.01068.

[12] R. Nembrini, M. F. Dacrema, P. Cremonesi, An application of reinforcement learning for minor embedding in quantum annealing, in: Proceedings of the International Workshop on AI for Quantum and Quantum for AI (AIQxQIA 2024) co-located with the 23rd International Conference of the Italian Association for Artificial Intelligence (AIxIA 2024), CEUR Workshop Proceedings, CEUR-WS.org, 2024.

[13] M. Pirhooshyaran, T. Terlaky, Quantum circuit design search, Quantum Machine Intelligence 3 (2021) 25. URL: https://doi.org/10.1007/s42484-021-00051-z. doi:10.1007/s42484-021-00051-z.

[14] M. Ostaszewski, L. M. Trenkwalder, W. Masarczyk, E. Scerri, V. Dunjko, Reinforcement learning for optimization of variational quantum circuit architectures, in: M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, J. W. Vaughan (Eds.), Advances in Neural Information Processing Systems, volume 34, Curran Associates, Inc., 2021, pp. 18182–18194. URL: https://proceedings.neurips.cc/paper_files/paper/2021/file/9724412729185d53a2e3e7f889d9f057-Paper.pdf.

[15] A. Peruzzo, J. McClean, P. Shadbolt, M.-H. Yung, X.-Q. Zhou, P. J. Love, A. Aspuru-Guzik, J. L. O'Brien, A variational eigenvalue solver on a photonic quantum processor, Nature Communications 5 (2014). URL: http://dx.doi.org/10.1038/ncomms5213. doi:10.1038/ncomms5213.

[16] J. Tilly, H. Chen, S. Cao, D. Picozzi, K. Setia, Y. Li, E. Grant, L. Wossnig, I. Rungger, G. H. Booth, J. Tennyson, The variational quantum eigensolver: A review of methods and best practices, Physics Reports 986 (2022) 1–128. URL: https://www.sciencedirect.com/science/article/pii/S0370157322003118. doi:https://doi.org/10.1016/j.physrep.2022.08.003, the Variational Quantum Eigensolver: a review of methods and best practices.

[17] Y. Cao, J. Romero, J. P. Olson, M. Degroote, P. D. Johnson, M. Kieferová, I. D. Kivlichan, T. Menke, B. Peropadre, N. P. D. Sawaya, S. Sim, L. Veis, A. Aspuru-Guzik, Quantum chemistry in the age of quantum computing, Chemical Reviews 119 (2019) 10856–10915. URL: https://doi.org/10.1021/acs.chemrev.8b00803. doi:10.1021/acs.chemrev.8b00803. arXiv:https://doi.org/10.1021/acs.chemrev.8b00803, pMID: 31469277.

[18] J. R. McClean, J. Romero, R. Babbush, A. Aspuru-Guzik, The theory of variational hybrid quantum-classical algorithms, New Journal of Physics 18 (2016) 023023. URL: https://dx.doi.org/10.1088/1367-2630/18/2/023023. doi:10.1088/1367-2630/18/2/023023.

[19] E. Farhi, J. Goldstone, S. Gutmann, A quantum approximate optimization algorithm, 2014. arXiv:1411.4028.

[20] K. Blekos, D. Brand, A. Ceschini, C.-H. Chou, R.-H. Li, K. Pandya, A. Summer, A review on quantum approximate optimization algorithm and its variants, 2023. arXiv:2306.09198.

[21] G. E. Crooks, Performance of the quantum approximate optimization algorithm on the maximum cut problem, 2018. arXiv:1811.08419.

[22] M. Willsch, D. Willsch, F. Jin, H. D. Raedt, K. Michielsen, Benchmarking the quantum approximate optimization algorithm, Quantum Information Processing 19 (2020). URL: https://doi.org/10.1007%2Fs11128-020-02692-8. doi:10.1007/s11128-020-02692-8.

[23] J. Cook, S. Eidenbenz, A. Bärtschi, The quantum alternating operator ansatz on maximum k-vertex cover, in: 2020 IEEE International Conference on Quantum Computing and Engineering (QCE), 2020, pp. 83–92. doi:10.1109/QCE49297.2020.00021.

[24] C. Y.-Y. Lin, Y. Zhu, Performance of qaoa on typical instances of constraint satisfaction problems with bounded degree, 2016. arXiv:1601.01744.

[25] M. Radzihovsky, J. Murphy, M. Swofford, A qaoa solution to the traveling salesman problem using pyquil, 2019.

[26] S. Brandhofer, D. Braun, V. Dehn, G. Hellstern, M. Hüls, Y. Ji, I. Polian, A. S. Bhatia, T. Wellens, Benchmarking the performance of portfolio optimization with qaoa, Quantum Information Processing 22 (2022) 25. URL: https://doi.org/10.1007/s11128-022-03766-5. doi:10.1007/s11128-022-03766-5.

[27] K. Kurowski, T. Pecyna, M. Slysz, R. Różycki, G. Waligóra, J. Weglarz, Application of quantum approximate optimization algorithm to job shop scheduling problem, European Journal of Operational Research 310 (2023) 518–528. URL: https://www.sciencedirect.com/science/article/pii/S0377221723002072. doi:https://doi.org/10.1016/j.ejor.2023.03.013.

[28] R. Herrman, P. C. Lotshaw, J. Ostrowski, T. S. Humble, G. Siopsis, Multi-angle quantum approximate optimization algorithm, Scientific Reports 12 (2022) 6781. URL: https://doi.org/10.1038/s41598-022-10555-8. doi:10.1038/s41598-022-10555-8.

[29] M. Chalupnik, H. Melo, Y. Alexeev, A. Galda, Augmenting qaoa ansatz with multiparameter problem-independent layer, in: 2022 IEEE International Conference on Quantum Computing and Engineering (QCE), IEEE Computer Society, Los Alamitos, CA, USA, 2022, pp. 97–103. URL: https://doi.ieeecomputersociety.org/10.1109/QCE53715.2022.00028. doi:10.1109/QCE53715.2022.00028.

[30] S. Sim, P. D. Johnson, A. Aspuru-Guzik, Expressibility and entangling capability of parameterized quantum circuits for hybrid quantum-classical algorithms, Advanced Quantum Technologies 2 (2019) 1900070. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/qute.201900070. doi:https://doi.org/10.1002/qute.201900070. arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/qute.201900070.

[31] J. Qin, Review of ansatz designing techniques for variational quantum algorithms, Journal of Physics: Conference Series 2634 (2023) 012043. URL: https://dx.doi.org/10.1088/1742-6596/2634/1/012043. doi:10.1088/1742-6596/2634/1/012043.

[32] J. Wurtz, P. J. Love, Classically optimal variational quantum algorithms, IEEE Transactions on Quantum Engineering 2 (2021) 1–7. doi:10.1109/TQE.2021.3122568.

[33] Y. Du, M.-H. Hsieh, T. Liu, D. Tao, Expressive power of parametrized quantum circuits, Phys. Rev. Res. 2 (2020) 033125. URL: https://link.aps.org/doi/10.1103/PhysRevResearch.2.033125. doi:10.1103/PhysRevResearch.2.033125.

[34] F. Brozzi, G. Turati, M. F. Dacrema, Exploring the role of hamiltonian expressibility in ansatz selection for variational quantum algorithms, in: Proceedings of the International Workshop on AI for Quantum and Quantum for AI (AIQxQIA 2024) co-located with the 23rd International Conference of the Italian Association for Artificial Intelligence (AIxIA 2024), CEUR Workshop Proceedings, CEUR-WS.org, 2024.

[35] J. R. McClean, S. Boixo, V. N. Smelyanskiy, R. Babbush, H. Neven, Barren plateaus in quantum neural network training landscapes, Nature Communications 9 (2018). URL: http://dx.doi.org/10.1038/s41467-018-07090-4. doi:10.1038/s41467-018-07090-4.

[36] A. Arrasmith, M. Cerezo, P. Czarnik, L. Cincio, P. J. Coles, Effect of barren plateaus on gradient-free optimization, Quantum 5 (2021) 558. URL: https://doi.org/10.22331/q-2021-10-05-558. doi:10.22331/q-2021-10-05-558.

[37] A. Arrasmith, Z. Holmes, M. Cerezo, P. J. Coles, Equivalence of quantum barren plateaus to cost concentration and narrow gorges, Quantum Science and Technology 7 (2022) 045015. URL: https://dx.doi.org/10.1088/2058-9565/ac7d06. doi:10.1088/2058-9565/ac7d06.

[38] M. Cerezo, P. J. Coles, Higher order derivatives of quantum neural networks with barren plateaus, Quantum Science and Technology 6 (2021) 035006. URL: https://dx.doi.org/10.1088/2058-9565/abf51a. doi:10.1088/2058-9565/abf51a.

[39] Z. Holmes, K. Sharma, M. Cerezo, P. J. Coles, Connecting ansatz expressibility to gradient magnitudes and barren plateaus, PRX Quantum 3 (2022) 010313. URL: https://link.aps.org/doi/10.1103/PRXQuantum.3.010313. doi:10.1103/PRXQuantum.3.010313.

[40] M. Larocca, P. Czarnik, K. Sharma, G. Muraleedharan, P. J. Coles, M. Cerezo, Diagnosing Barren

Plateaus with Tools from Quantum Optimal Control, Quantum 6 (2022) 824. URL: https://doi.org/10.22331/q-2022-09-29-824. doi:10.22331/q-2022-09-29-824.

[41] T. Volkoff, P. J. Coles, Large gradients via correlation in random parameterized quantum circuits, Quantum Science and Technology 6 (2021) 025008. URL: https://dx.doi.org/10.1088/2058-9565/abd891. doi:10.1088/2058-9565/abd891.

[42] H. R. Grimsley, S. E. Economou, E. Barnes, N. J. Mayhall, An adaptive variational algorithm for exact molecular simulations on a quantum computer, Nature Communications 10 (2019) 3007. URL: https://doi.org/10.1038/s41467-019-10988-2. doi:10.1038/s41467-019-10988-2.

[43] H. L. Tang, V. Shkolnikov, G. S. Barron, H. R. Grimsley, N. J. Mayhall, E. Barnes, S. E. Economou, Qubit-ADAPT-VQE: An adaptive algorithm for constructing hardware-efficient ansätze on a quantum processor, PRX Quantum 2 (2021). URL: https://doi.org/10.1103%2Fprxquantum.2.020310. doi:10.1103/prxquantum.2.020310.

[44] Y. S. Yordanov, V. Armaos, C. H. W. Barnes, D. R. M. Arvidsson-Shukur, Qubit-excitation-based adaptive variational quantum eigensolver, Communications Physics 4 (2021). URL: https://doi.org/10.1038%2Fs42005-021-00730-0. doi:10.1038/s42005-021-00730-0.

[45] C. Feniou, M. Hassan, D. Traoré, E. Giner, Y. Maday, J.-P. Piquemal, Overlap-adapt-vqe: practical quantum chemistry on quantum computers via overlap-guided compact ansätze, Communications Physics 6 (2023) 192. URL: https://doi.org/10.1038/s42005-023-01312-y. doi:10.1038/s42005-023-01312-y.

[46] L. Zhu, H. L. Tang, G. S. Barron, F. A. Calderon-Vargas, N. J. Mayhall, E. Barnes, S. E. Economou, Adaptive quantum approximate optimization algorithm for solving combinatorial problems on a quantum computer, Phys. Rev. Res. 4 (2022) 033029. URL: https://link.aps.org/doi/10.1103/PhysRevResearch.4.033029. doi:10.1103/PhysRevResearch.4.033029.

[47] A. G. Rattew, S. Hu, M. Pistoia, R. Chen, S. Wood, A domain-agnostic, noise-resistant, hardware-efficient evolutionary variational quantum eigensolver, 2020. arXiv:1910.09694.

[48] D. Chivilikhin, A. Samarin, V. Ulyantsev, I. Iorsh, A. R. Oganov, O. Kyriienko, Mog-vqe: Multiobjective genetic variational quantum eigensolver, 2020. arXiv:2007.04424.

[49] U. Las Heras, U. Alvarez-Rodriguez, E. Solano, M. Sanz, Genetic algorithms for digital quantum simulations, Phys. Rev. Lett. 116 (2016) 230504. URL: https://link.aps.org/doi/10.1103/PhysRevLett.116.230504. doi:10.1103/PhysRevLett.116.230504.

[50] L. Cincio, Y. Subaşı, A. T. Sornborger, P. J. Coles, Learning the quantum algorithm for state overlap, New Journal of Physics 20 (2018) 113022. URL: https://dx.doi.org/10.1088/1367-2630/aae94a. doi:10.1088/1367-2630/aae94a.

[51] Y. Du, T. Huang, S. You, M.-H. Hsieh, D. Tao, Quantum circuit architecture search for variational quantum algorithms, npj Quantum Information 8 (2022) 62. URL: https://doi.org/10.1038/s41534-022-00570-y. doi:10.1038/s41534-022-00570-y.

[52] M. Bilkis, M. Cerezo, G. Verdon, P. J. Coles, L. Cincio, A semi-agnostic ansatz with variable structure for variational quantum algorithms, Quantum Machine Intelligence 5 (2023) 43. URL: https://doi.org/10.1007/s42484-023-00132-1. doi:10.1007/s42484-023-00132-1.

[53] R. S. Sutton, A. G. Barto, Reinforcement learning - an introduction, Adaptive computation and machine learning, MIT Press, 1998. URL: https://www.worldcat.org/oclc/37293240.

[54] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, O. Klimov, Proximal policy optimization algorithms, CoRR abs/1707.06347 (2017). URL: http://arxiv.org/abs/1707.06347. arXiv:1707.06347.

[55] J. Achiam, Spinning Up in Deep Reinforcement Learning, 2018.

[56] S. Giordano, M. A. Martin-Delgado, Reinforcement-learning generation of four-qubit entangled states, Phys. Rev. Res. 4 (2022) 043056. URL: https://link.aps.org/doi/10.1103/PhysRevResearch.4.043056. doi:10.1103/PhysRevResearch.4.043056.

[57] E.-J. Kuo, Y.-L. L. Fang, S. Y.-C. Chen, Quantum architecture search via deep reinforcement learning, 2021. arXiv:2104.07715.

[58] X. Zhu, X. Hou, Quantum architecture search via truly proximal policy optimization, Scientific Reports 13 (2023) 5157. URL: https://doi.org/10.1038/s41598-023-32349-2. doi:10.1038/s41598-023-32349-2.

[59] L. Moro, M. G. A. Paris, M. Restelli, E. Prati, Quantum compiling by deep reinforcement learning, Communications Physics 4 (2021) 178. URL: https://doi.org/10.1038/s42005-021-00684-3. doi:10.1038/s42005-021-00684-3.

[60] Y.-H. Zhang, P.-L. Zheng, Y. Zhang, D.-L. Deng, Topological quantum compiling with reinforcement learning, Phys. Rev. Lett. 125 (2020) 170501. URL: https://link.aps.org/doi/10.1103/PhysRevLett.125.170501. doi:10.1103/PhysRevLett.125.170501.

[61] T. Fösel, M. Y. Niu, F. Marquardt, L. Li, Quantum circuit optimization with deep reinforcement learning, 2021. URL: https://arxiv.org/abs/2103.07585. arXiv:2103.07585.

[62] E. Grant, L. Wossnig, M. Ostaszewski, M. Benedetti, An initialization strategy for addressing barren plateaus in parametrized quantum circuits, Quantum 3 (2019) 214. URL: http://dx.doi.org/10.22331/q-2019-12-09-214. doi:10.22331/q-2019-12-09-214.

[63] H. Singh, S. Majumder, S. Mishra, Benchmarking of different optimizers in the variational quantum algorithms for applications in quantum chemistry, The Journal of Chemical Physics 159 (2023) 044117. URL: https://doi.org/10.1063/5.0161057. doi:10.1063/5.0161057.

[64] M. Fernández-Pendás, E. F. Combarro, S. Vallecorsa, J. Ranilla, I. F. Rúa, A study of the performance of classical minimizers in the quantum approximate optimization algorithm, Journal of Computational and Applied Mathematics 404 (2022) 113388. URL: https://www.sciencedirect.com/science/article/pii/S0377042721000078. doi:https://doi.org/10.1016/j.cam.2021.113388.

[65] F. Glover, G. Kochenberger, R. Hennig, Y. Du, Quantum bridge analytics i: A tutorial on formulating and using qubo models, Annals of Operations Research 314 (2022) 141–183. URL: https://doi.org/10.1007/s10479-022-04634-2. doi:10.1007/s10479-022-04634-2.

[66] A. Lucas, Ising formulations of many np problems, Frontiers in Physics 2 (2014). URL: https://www.frontiersin.org/articles/10.3389/fphy.2014.00005. doi:10.3389/fphy.2014.00005.

[67] E. Pelofske, G. Hahn, H. Djidjev, Solving large maximum clique problems on a quantum annealer, in: S. Feld, C. Linnhoff-Popien (Eds.), Quantum Technology and Optimization Problems, Springer International Publishing, Cham, 2019, pp. 123–135.

[68] D. C. McKay, C. J. Wood, S. Sheldon, J. M. Chow, J. M. Gambetta, Efficient z-gates for quantum computing, Physical Review A 96 (2017). URL: http://dx.doi.org/10.1103/PhysRevA.96.022330. doi:10.1103/physreva.96.022330.