Optimizing Initial Qubit Mappings Under Fixed Gate Error Rates Using Deep Reinforcement Learning

Rares Adrian Oancea[1], Stan van der Linde[1], Willem de Kok[1], Matthia Sabatelli[2], and Sebastian Feld[3(✉)]

[1] The Netherlands Organisation for Applied Scientific Research (TNO), Oude Waalsdorperweg 63, 2597 AK The Hague, The Netherlands
[2] University of Groningen, Nijenborgh 7, 9747 AG Groningen, The Netherlands
[3] QuTech, Delft University of Technology, Mekelweg 5, 2628 CD Delft, The Netherlands
s.feld@tudelft.nl

Abstract. Quantum computing promises to execute some tasks exponentially faster than classical computers. Quantum compilation, which transforms algorithms into executable quantum circuits, involves solving the initial mapping problem, crucial for optimizing qubit assignment and minimizing gate error rates. This study explores Deep Reinforcement Learning (DRL) for initial mapping across various qubit topologies, considering fixed gate error rates. Previous DRL approaches have succeeded but didn't account for fixed error rates, used only one algorithm (PPO), and focused on a single topology with 20 qubits. The trial-and-error nature of Reinforcement Learning makes it ideal for initial mapping. DRL agents, using multiple policy gradient algorithms (A2C, PPO with and without action masking, and TRPO), compute high-quality mappings for small- and medium-scale quantum architectures. While effective, their efficiency decreases with larger systems, necessitating further optimization. Fine-tuning hyperparameters and action masking prevent illegal actions and enhance accuracy. Although currently not surpassing tools like Qiskit or achieving scalability for larger systems, this study highlights DRL's potential for initial mapping in quantum computing, encouraging further innovation and refinement.

Keywords: Quantum compilation · Deep reinforcement learning

1 Introduction

In quantum computing, gate-based quantum computers execute a series of gates on qubits to perform computations, which are believed to efficiently handle [10,12] tasks that are otherwise intractable for classical computers [17]. Algorithms for these systems are often designed using a quantum circuit representation, specifying which gates are applied to which qubits and in what sequence. Although this representation is hardware-agnostic, executing arbitrary quantum circuits on actual hardware is complex. Furthermore, the high-level design of

S. Zielinski et al. (Eds.): I4CS 2025, CCIS 2513, pp. 189–208, 2025.
https://doi.org/10.1007/978-3-031-94263-1_12

a quantum circuit often does not take into account the restrictions imposed by the physical quantum hardware [30]. This process, known as quantum compilation [2], involves converting the circuit into a hardware-specific executable algorithm [20]. Among numerous compilation steps and possible optimization passes [19], quantum compilation [15] generally includes steps like qubit routing [7] and initial mapping [30], both recognized as NP-hard optimization problems [27]. The main roadblock which restricts the execution of quantum algorithms on quantum computers is given by the manner in which qubits are connected on the hardware. Depending on how the qubits are connected on the target architecture, some of the gates on the circuit may not be executable on the quantum computer, due to the connectivity constraints of the hardware.

This concept is illustrated in Fig. 1, which shows the interaction graph representation of a quantum circuit, with virtual qubits, on the left, and the connectivity scheme of some quantum hardware on the right, denoted as the "connection graph", with physical qubits. The connection graph represents the physical qubits and their connections on the hardware. In contrast, the interaction graph represents the virtual qubits and their interactions within the circuit. The challenge is to map each virtual qubit to a physical qubit in a way that minimizes the difference between the logical interactions and the available physical connections. More formally, we define a connection graph $G_C = (V, E_C)$, where the vertices $v \in V$ represent the physical qubits of the hardware and the edges $(v, u) \in E_C$ represent the qubit connections. Next, we define an interaction graph $G_I = (V, E_I)$, with vertices $v \in V$ the virtual qubits of the circuit and the edges $(v, u) \in E_i$ the interactions in the circuit. In the initial mapping problem, the objective is to find a bijection $f : V \rightarrow V$, which maps every virtual qubit to a physical qubit such that some cost function on the mapped edges E_M is minimized, where $E_M = \{(f(v), f(u)) : (v, u) \in E_I\}$. In some cases, a perfect mapping might exist where $E_M \subset E_C$. However, this is not always the case, and the objective should be some cost function of $E_M \backslash E_C$, i.e., a cost function of connections that exist in the mapped interaction graph, but not in the connection graph.

The graph at the bottom of the figure shows the manner in which the virtual qubits on the circuit were mapped to the quantum computer. For example, virtual qubits 0 and 2 from the circuit were mapped to physical qubits A and B from the hardware. This connection can be executed, as there is a connection between A and B in the connectivity scheme. On the other hand, the mapping of qubits $0 - A$ and $1 - E$ leads to a connection which can not be executed, since there is no link between qubits physical A and E in the connection graph. In this example, it is easy to see which connections can and can not be executed, and why the mapping proposed in Fig. 1 does not enable the compilation of the algorithm on the example hardware. However, this setting can get much more complex, with almost endless possibilities to customize the algorithms, both in terms of qubit and gate count. Furthermore, quantum computers have evolved significantly, with current architectures featuring 53 [11], 97 [1] or even 127 qubits [4].

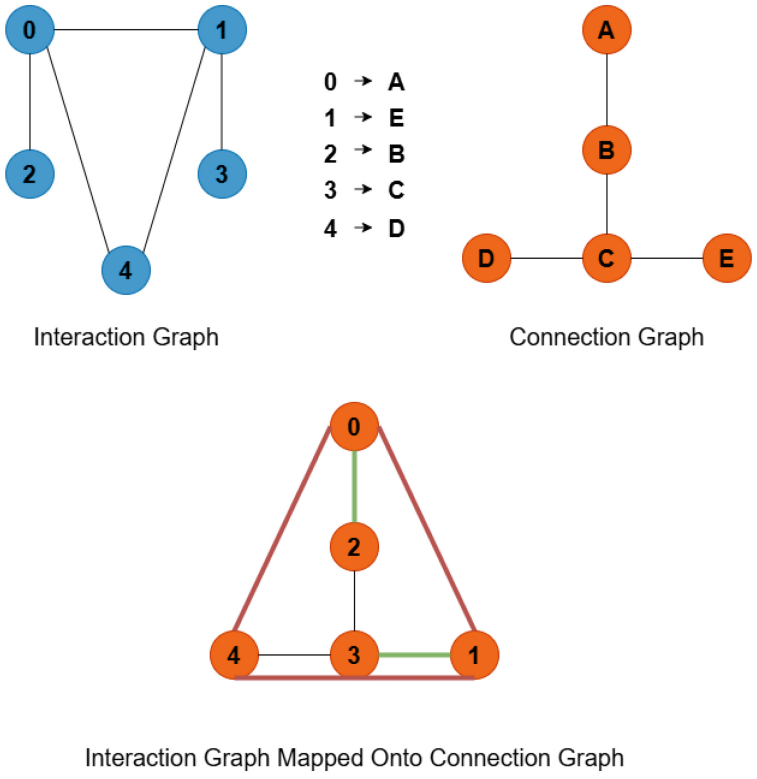

Fig. 1. Process of mapping an interaction graph (a compressed representation of a quantum circuit, on the left) to a connection graph (a representation of the quantum hardware, on the right). The proposed qubit mapping leads to two executable connections ($(0, 2)$ and $(3, 1)$) and three non-executable connections ($(0, 4)$, $(0, 1)$ and $(4, 1)$).

High-quality compilation strategies are essential components to allow for this computational advantage over classical computers [22], because in the current Noisy Intermediate-Scale Quantum (NISQ) era, hardware is prone to noise [18]. The compilation influences the resulting fidelity of running the circuit on the actual hardware in many ways, such as by increasing the circuit depth and by determining which qubit connections are used for which multi-qubit gates [13]. If there are bad qubits or bad connections, it is crucial to avoid using them, generally achieved by routing a mapped circuit, to ensure optimal performance.

One task in the compilation step is to relate every virtual qubit in the circuit to a physical qubit in the hardware in such a way that all gates can be operated. Ideally, this should be done to minimize the total noise of running the whole circuit [13]. Similarly to the situation portrayed in Fig. 1, a high quality mapping would result in as few non-executable interactions as possible. This task is not obvious, as quantum hardware topologies are not all-to-all connected. In prac-

tice, the task is split into two steps that both are NP-hard [6]. Firstly, as already mentioned, initial mapping relates every virtual qubit to a physical qubit. Secondly, the routing stage involves changing the assignment of virtual qubits to physical qubits to enable the execution of gates that could not be executed under the initial mapping. This process effectively moves quantum information and can be technically realized in different ways, such as using SWAP, Hadamard [29] or BRIDGE gates. The initial mapping is a crucial step as it affects the total noise of the final executable circuit in two ways. Namely, it can alter the amount of SWAP and BRIDGE gate insertions in the routing stage, and it influences which qubits and qubit connections are used for which gates. Finding a good initial mapping is challenging because the routing process changes the qubit assignments dynamically. The initial mapping is not static; it serves as a starting point, and the mapping changes at each time step as the quantum information is moved.

This research focuses on improving the initial mapping step, a crucial task in quantum computing. Initial mapping involves assigning virtual qubits to physical qubits, and this step directly influences all subsequent activities, including the routing of qubits and execution of quantum gates. Making optimal decisions at this stage is essential, as it can reduce the number of adjustments during routing, reduce error rates, and ultimately achieve a higher-quality execution of quantum algorithms [18]. By optimizing this process, we aim to enhance the overall performance and reliability of quantum computations. The task at hand will tackle a modified version of the initial mapping problem, which will incorporate the problem description described above, as well as various hardware constraints. The previously described problem is a generalization of the subgraph isomorphism problem, which is a NP-complete problem [6]. The version of the problem considered for this project will consider various hardware constraints imposed by the target hardware. One such constraint is that of gate error rates, which in this context translates to the quality of qubits or of the connections between qubits. This added layer of difficulty increases the overall complexity of the problem. Notably, since the graph isomorphism problem is a special case of this problem, it can be classified as NP-hard.

With this objective in mind, this paper proposes a Deep Reinforcement Learning (DRL) based approach to compute initial mappings for given quantum circuits and topologies. This branch of Artificial Intelligence has gained significant interest within the quantum community, and previous research has shown promising results within the compilation process as well [9,14]. However, they do not take into account gate error rates, and they do not show the scalability of proposed RL approaches. To this end, several policy gradient DRL methods (Advantage Actor-Critic (A2C) [16], Proximal Policy Optimization (PPO) [24], Trust Region Policy Optimization (TRPO) [23], and PPO with action masking) will be employed to investigate whether high quality initial mappings can be computed for various qubit connectivity schemes which feature varying degrees of error rates on their connections. Furthermore, a database of quantum com-

puter connectivity schemes is introduced to facilitate benchmarking and analysis of DRL agents on various quantum compilation steps.

The remainder of the paper is structured as follows. Section 2 provides an overview of previous attempts to solve the initial mapping problem using DRL. Section 3 details the workings of the environment and the specific DRL algorithms employed in this research. Section 4 presents findings from experiments conducted on three simulated quantum devices. Finally, Sect. 5 summarizes the study's key findings and their implications.

2 Related Work

The goal of the initial mapping step is to obtain a mapping which (heuristically) minimizes the need for later adjustments imposed by hardware connectivity limitations and gate error rate variations. Previous research has investigated whether RL agents are able to successfully compute high quality initial mappings. In this section, a comprehensive overview and discussion of this previous work is provided.

Huang et al. [9] separate qubit mapping into two separate subproblems, namely discovering high quality initial mappings and minimizing SWAP insertion, respectively. The authors start by proposing a RL-based approach which uses a self-attention model to extract features from a circuit and formulate initial mapping as a sequence to sequence learning task. The neural architecture includes three encoders: two that process distinct aspects of the input data, and one dedicated to feature extraction. Each encoder incorporates multi-head attention mechanisms [28]. Afterwards, a RL algorithm makes use of these mechanisms to compute high quality initial mappings. The method employed by [9] is the REINFORCE algorithm [31] with baseline algorithm, which is a policy gradient algorithm. This policy gradient algorithm takes the extracted features as input and outputs the probabilities of mapping virtual to physical qubits. The baseline, which usually comes in the form of a function or even a constant, as long as it is independent of the actions, is used to stabilize the learning procedure. The second step, namely the qubit routing stage, involves a novel Dynamically Extract and Route (DEAR) framework, which extracts circuits iteratively and uses the A* algorithm to determine when and where it is necessary to insert SWAP gates. Their results show that method REINFORCE with baseline architecture obtains initial mappings that require 12% fewer gate insertions when used in combination with DEAR for qubit routing. This comparison was made against a nearest neighbor transformation approach [5] and a dynamic look-ahead heuristic [32]. The RL architecture was trained using 140 circuits from the IBM QX set, with 19 circuits used for testing. The target hardware that the circuits were mapped to is the IBM QX20 architecture.

Another recent attempt to tackle the qubit mapping problem with a RL approach was made by Li et al. in [14]. The authors divided the task similarly to Huang et al. in [9], however, the key difference is that they employed an RL-based architecture for the routing stage, using an isomorphic graph search

algorithm for initial mappings. Li et al. [14] adopted a value-based approach, specifically the Dueling Deep-Q Network, to minimize the number of SWAP gate insertions. The B131 circuit set was used for training and testing the RL model, targeting the IBM QX20 architecture. Li et al. [14] reported 63% fewer SWAP gate insertions compared to simulated annealing and heuristic search.

2.1 Limitations

Previous research in tackling either initial mapping or qubit routing using various RL techniques has shown that RL agents are indeed capable of both computing high quality initial mappings and reducing the number of SWAP gate insertions, for a 20 qubit topology. The studies conducted by Huang et al. [9] and Li et al. [14] have investigated the application of RL agents in mapping circuits to a hardware topology with 20 qubits, displaying promising results in this field.

Meanwhile, larger and more complex architectures have been developed, with quantum computers featuring up to 127 [4] qubits. The increase in scale goes hand in hand with an increase in the complexity of the initial mapping problem. Therefore, the demand for efficient and high quality initial mapping techniques grows. However, aforementioned research only focused on quantum architectures featuring up to 20 qubits.

Furthermore, the two-qubit gate error rates were not taken into account in the previously discussed research, while this is a crucial aspect of the compilation process. Some connections may have a poorer quality than others, which can severely impact the quality of the mapping and also lead to a higher error rate of the final circuit.

Thus, our proposal applies DRL models across various quantum architectures with different qubit connectivity schemes. A distinguishing feature is the inclusion of fixed gate error rates, allowing to test model scalability under diverse conditions.

3 Methodology

We use the QGYM library [27] to investigate whether the application of DRL agents can result in high quality initial mappings for various quantum computer topologies with incorporated gate error rates. The package functions in a manner similar to the well known GYM package [3], in the sense that it provides a number of environments on which RL agents can be applied. The main purpose of QGYM is to provide environments which represent various stages of the quantum compilation process, including those for the initial mapping, qubit routing, and scheduling stages.

In the following, Subsect. 3.1 provides a detailed explanation of the environment's functionality. Next, Subsect. 3.2 discusses the characteristics of the DRL algorithms. Subsection 3.3 then outlines the topologies employed and their integration into the experiments. Finally, Subsect. 3.4 offers insights into the development of the experiments.

3.1 Environment Functionality

We focus on experiments within the "Initial Mapping" environment from QGYM, which involves mapping virtual qubits of a quantum circuit to physical qubits on quantum hardware. In this environment, both the quantum circuit and hardware topology are represented as graphs. The quantum circuit is represented by a graph after reduction to an interaction graph, where virtual qubits are vertices, and two-qubit gates are represented by edges (i.e., Fig. 1). This representation omits single-qubit gates, gate types, gate order, and the amount of two-qubit gates between qubits. This reduction improves the training and evaluation times of the RL agent (i.e., the agent efficiency), while neglecting some aspects that do impact the overall performance (e.g. the gate error rates) of the compiled circuit. This trade-off between agent efficiency and compiled circuit performance can be tweaked by including edge weights to account for aspects like the number of gates between qubits and/or the order of gate execution in the original circuit. The quantum hardware is represented by a connection graph, with vertices representing physical qubits and edges indicating hardware connections. The edges can be weighted between 0 and 1 to reflect the gate error rates of the hardware connections, with 1 indicating perfect qubit connection reliability and 0 no reliability.

The logic for transposing the initial mapping problem to a reinforcement learning setting goes as follows. The setup begins with a fixed connection graph, static across all episodes. Each episode introduces a novel interaction graph for the agent to observe, alongside an initially empty mapping. At every step, the agent can map a virtual qubit to a physical qubit until the mapping is fully established. In theory, this process enables the training of agents that are capable of managing various interaction graphs on a predetermined hardware layout. The process of mapping an interaction graph to a connection graph is depicted in Fig. 2. At each step, a virtual qubit is mapped to a physical qubit, until a complete mapping is obtained (i.e., every virtual qubit is mapped to a physical qubit).

In this work, we want to optimize the initial mapping on its own, whereas more commonly the initial mapping is optimized together with the routing compilation step. In the combined setting the optimization target often is minimization of the total CNOT-, T- or gate count or the overall gate error rate of the mapped and routed circuit. These targets make sense as they all contribute in a clear manner to the optimization of the full quantum compilation pipeline. However, what this means for the optimization target of the initial mapping step is not self-evident. Nonetheless, we propose an optimization metric for the initial mapping which aims to anticipate minimizing the routing while also taking into account the gate error rates of the hardware connections. Roughly speaking, the metric tries to minimize the "mismatch" between the interaction graph and the connection graph and utilize the connections with highest reliability (i.e., lowest error rate). This optimization is done in the training phase, via the reward function r. The reward function emits reward signals $R(a)$ to the DRL agent, guiding its training process and policy.

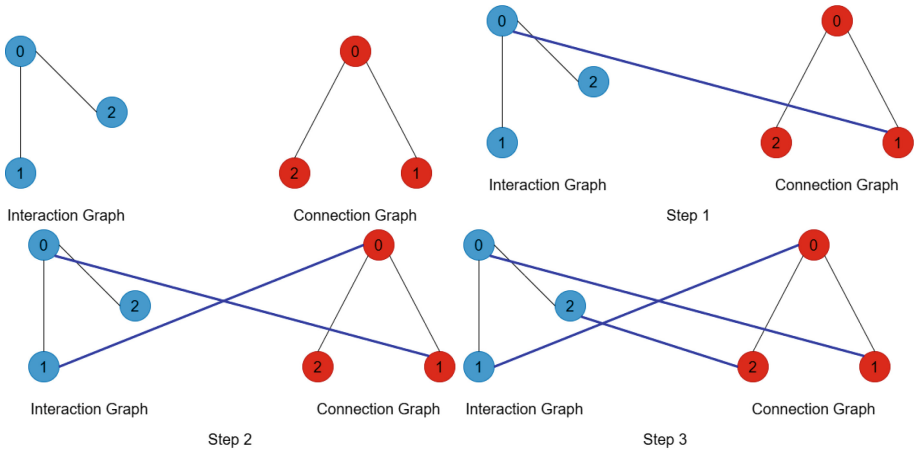

Fig. 2. Process of mapping an interaction graph to a connection graph. At each step, a virtual qubit is mapped to a physical qubit.

The reward is computed based on the previous state, the action chosen by the agent, and the new state after the action has been executed. Besides these three preset rewarder types, the design of QGYM is focused towards customization, and as such new rewarders can be implemented. A visual representation of what it means to have a "good" or "bad" mapped edge can be seen in Fig. 1. The figure displays an interaction graph on the left, and a connection graph on the right, each with five nodes. In the middle, the mappings between logical and physical qubits can be observed, and the mapped connection graph is found at the bottom of the image. In this picture, virtual qubits 0 to 4 are mapped to physical qubits A, E, B, C and D, respectively. This qubit mapping results in two good edges, namely edge $(0, 2)$ and edge $(3, 1)$ in the mapped connection graph at the bottom of the image. Furthermore, the obtained mapping leads to three bad edges, $(0, 4)$, $(0, 1)$ and $(4, 1)$.

The agents that will be trained on the environment can also perform legal or illegal actions. A visual representation of how an agent can select illegal actions in the environment is shown in Fig. 3. In this example, if the agent first maps virtual qubit 0 in the interaction graph to physical qubit 0 in the connection graph, it can not map the same virtual qubit to another physical qubit. In a similar manner, if any virtual qubit is mapped to physical qubit 0, it will not be allowed to map another virtual qubit to the same physical qubit.

The reward signal is computed as follows. First, a partial reward function is precomputed. This reward function $r(q_{i,p}, q_{j,p})$ defines a reward for an edge in the interaction graph that is mapped to the physical qubits $q_{i,p}$ and $q_{j,p}$ based on the error rates of the connection graph. Specifically, the reward function is defined as follows:

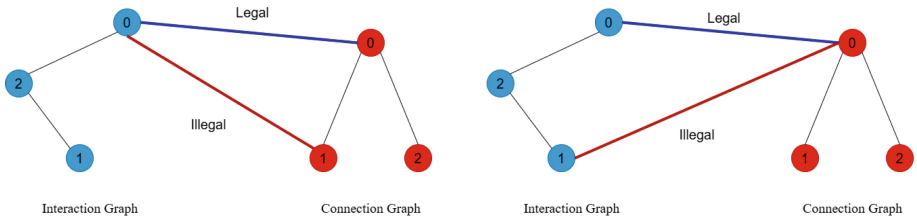

Fig. 3. Figure depicting the difference between legal and illegal actions. An action is illegal if a virtual qubit is mapped to two physical qubits or if two virtual qubits are mapped to the same physical qubit.

$$r(q_{i,p}, q_{j,p}) = \min_{P \in \mathcal{P}(q_{i,p}, q_{j,p})} \left(\prod_{e \in P} (1 - \mathrm{err}(e)) \right)^{-1}, \tag{1}$$

where $\mathcal{P}(q_{i,p}, q_{j,p})$ is the set of all simple paths from $q_{i,p}$ to $q_{j,p}$ in the connection graph. Note that r uses the path with the lowest combined error rates. This concept simulates the introduction of SWAP and/or BRIDGE gates in the routing step of the compilation process. This partial reward can then be used to compute a score for a completed mapping M

$$S(M, E_I) = \sum_{\{q_{i,v}, q_{j,v}\} \in E_I} r(M(q_{i,v}), M(q_{j,v})). \tag{2}$$

The reward signal for the agent is then constructed as follows:

$$R(a) = \begin{cases} -0.2, & \text{if } a \text{ is illegal} \\ 0, & \text{if } a \text{ does not complete the mapping} \\ \exp\left(S(M, E_I))^{-1}\right), & \text{if } a \text{ completes the mapping} \end{cases} \tag{3}$$

where M is the mapping from virtual qubits to physical qubits build by the agent. Note that only the last action in an episode gives a reward and that all other actions give no reward or a negative reward in the case of an illegal action.

3.2 DRL Algorithms

To address the initial mapping problem, three policy gradient DRL algorithms were implemented: Proximal Policy Optimization (PPO) [24], Trust Region Policy Optimization (TRPO) [23], and Advantage Actor-Critic (A2C) [16]. The main reason for choosing these algorithms comes from their availability, provided through the STABLE BASELINES 3 library [21]. This package facilitates their implementation, as well as thorough customization of hyperarameters and architecture depth. PPO stabilizes training by clipping the probability ratio during policy updates, allowing for controlled changes and balancing exploration and exploitation, which is essential for complex tasks like initial mapping, in the context of quantum compilation. TRPO maximizes a surrogate objective while

constraining the Kullback-Leibler divergence between policies, ensuring stable updates and reducing performance collapse, making it reliable for precise mapping tasks. A2C combines policy learning (actor) and value estimation (critic) with an advantage function to improve the accuracy of the value estimate, contributing to more stable learning and making it well-suited for the complexities of initial mapping.

Additionally, a variant of PPO with illegal action masking will be applied to handle complex topologies with many qubits, where the probability of selecting illegal actions increases. This masking aims to prevent such actions, improving learning efficiency [26].

These DRL models are implemented using the STABLE BASELINES 3 library [21]. PPO and A2C are directly supported by the library, while TRPO and Maskable PPO are implemented using the STABLE BASELINES 3 CONTRIB package.

3.3 Target Topologies

The four DRL models were trained to compute initial mappings for various target topologies from the leading superconducting devices, provided by vendors such as IBM, Rigetti and Google. An example showing some topologies that were used in the experiments can be seen in Fig. 4, showing the qubit connectivity schemes for the Rigetti Aspen, IBMQ Tokyo and Google Sycamore devices. All topologies are represented as NETWORKX graphs, with random edge weights in the range of $[0.85, 1]$, different hardware qubit connectivity schemes, and qubit counts. The edge weights were set in the range $[0.85, 1]$ because values lower than 0.85 would denote poor qubit connection qualities. Having edge weights close to, or lower than 0.5 for instance, would indicate two-qubit gates that are only corrected half the time. For this, we developed a dataset of quantum computer qubit connectivity schemes with connections encoded as lists of tuples, to be retrieved for the connection graph before training. Each tuple denotes a bidirectional connection between two qubits.

The dataset includes connectivity schemes with 5 to 8 qubits, such as IBMQ Athens or Rigetti Agave; architectures with 15 to 20 qubits from IBMQ Melbourne, QuTech Surface17, IBMQ Tokyo, and Almaden; and 27 to 28 qubits from IBMQ Kolkata, Falcon, and Cambridge devices. To explore mappings for larger topologies, the dataset includes IBMQ Rochester and Google Sycamore (both 53 qubits), IBMQ Manhattan (65 qubits), QuTech Surface97 (97 qubits), and IBMQ Washington (127 qubits). These qubit connectivity schemes can be loaded into the connection graph component and used in the QGYM Initial Mapping environment to train DRL agents.

3.4 Experimental Setup

The learning rate α and discount factor γ are key hyperparameters in Deep Reinforcement Learning, influencing the speed of learning and the balance between immediate and future rewards. The learning rate determines the magnitude of

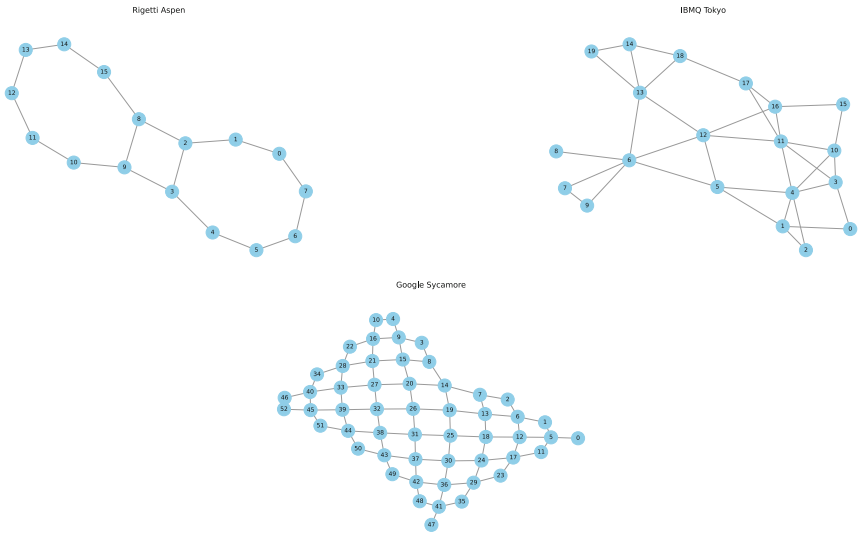

Fig. 4. Examples of quantum computer qubit connectivity schemes, for the Rigetti Aspen (16 qubits), IBMQ Tokyo(20 qubits) and Google Sycamore (53 qubits) devices

adjustment to the weights of our network with each step during the learning process. The discount factor regulates the importance assigned to future rewards compared to obtaining more immediate rewards. A grid-search procedure will optimize these parameters using PPO (with action mask), TRPO, and A2C on two small-scale architectures: IBMQ Casablanca and CC Light. The best performing values, determined by the average rewards achieved, will guide agent training across all topologies in the database. Fine-tuning was conducted for all four DRL models on these 7-qubit devices, and parallelization was employed to enhance computation efficiency. The procedure took approximately one week using 8 CPUs. Each training procedure will consider $400,000$ training steps, with $\gamma \in \{0.8, 0.85, 0.9, 0.95, 0.99\}$ and $\alpha \in \{0.0003, 0.0001, 0.003, 0.001, 0.1\}$.

The best performing hyperparameter values determined during fine-tuning will be used in the training of the DRL algorithms on each of the 25 topologies in the database. The number of timesteps for each training session will depend on the number of qubits in the topology, with larger qubit counts requiring more training time to ensure that the neural network in the DRL algorithm learns an effective policy for the given environment. The relationship between the number of timesteps (i.e., interactions between the DRL agent and the environment) and number of qubits in the target topology is assumed to be quadratic, beginning at $200,000$ timesteps for topologies with 4 qubits. The reason for this design choice is that, for training and evaluating the DRL agent on a simple, four qubit topology, $200,000$ training steps were sufficient to obtain high quality mappings, hence it was considered to be a good starting point. The exact number of training timesteps allocated, given the number of qubits, can be seen in Fig. 5. Given

that the number of environment interactions varies according to the number of qubits in the target topology, so does the number of interactions graphs used during training. As the agents are tasked with mapping interaction graphs to connection have that contain a greater number of qubits, the agents will require more training time, and consequently more interaction graphs in order to converge to an optimal solution. When dealing with larger connection graphs, the agent needs to be exposed to more interaction graphs during training because the number of possible interaction graphs increases significantly with the number of qubits. This ensures the agent can effectively learn and adapt to the expanded range of interactions. Table 1 shows the number of interaction graphs that were used throughout the training procedure, for each number of qubits facilitated by the hardware.

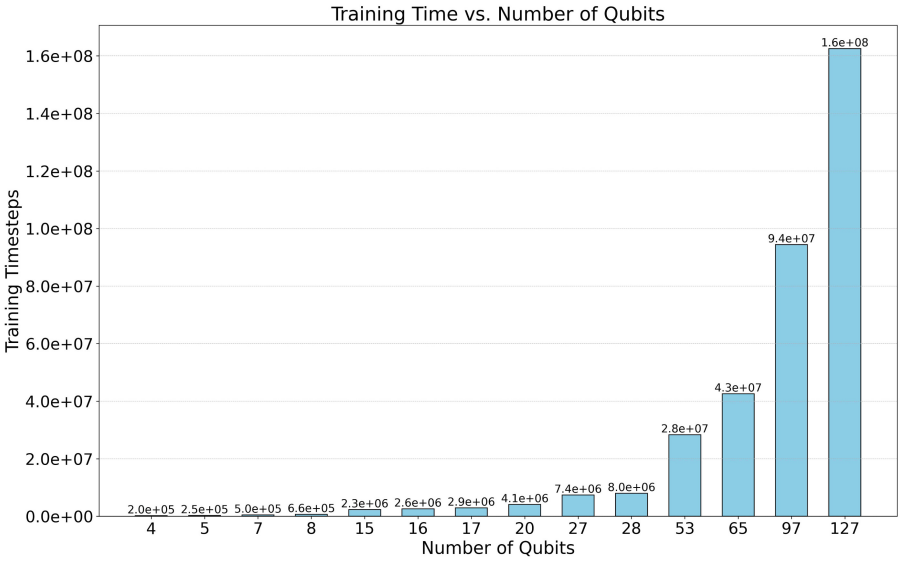

Fig. 5. Training timesteps required for different number of qubits

Similarly, as the complexity of the quantum computer architecture increases, the number of hidden layers in the neural networks will be expanded to better capture the intricate patterns and dependencies within the data. Deeper networks with more hidden layers can model more complex relationships, enabling the agent to make more informed decisions in challenging environments. This increase in network depth helps to ensure that the model has sufficient capacity to learn and generalize effectively from the higher-dimensional feature space associated with more complex topologies.

Table 1. Number of interaction graphs used in training

# Qubits in Target Hardware	# Interaction Graphs
4	50,000
5	50,781
7	71,540
8	81,920
15	152,473
16	162,500
17	172,977
20	203,125
27	273,509
28	284,388
53	534,618
65	655,060
97	973,038
127	1,279,527

During evaluation, trained DRL agents map, in a greedily manner by following the policies learned during training, interaction graphs generated using the Erdős-Rényi model [8], where each pair of nodes is connected with a probability $p = 0.5$. Null graphs (no edges) are excluded. The RL paradigm is generally a data hungry approach, and agents do not receive information from a fixed dataset. Instead, they receive information by continuously interacting with the environment [25]. The information that reinforcement learning agents receive from the environment keeps changing over the course of the learning process because of this continuous cycle of interaction of obtaining feedback. This eventually results in a substantial number of interactions for long simulations, resulting in often sample-inefficient solutions. For this reason, both training and evaluation data comes in the form of randomly generated graphs, not from an actual dataset of quantum circuit qubit connectivity schemes.

Performance is assessed over ten runs, each with 100 unique graphs, with the mean reward and a gate error rate-aware distance metric:

$$D = \frac{S}{|E_I|} \geq 1, \tag{4}$$

where S the sum mentioned in (2), and $|E_I|$ is the number of edges in the interaction graph. This metric is such that a value of 1 indicates an optimal mapping and higher values indicate worse mappings. A distance valued to 1 indicates that all mappings are executable on the hardware. Values larger than 1 denote that not all mappings and executable, and modifications need to be done to the obtained initial mapping in the routing stage. This approach helps

to evaluate the agents' ability to learn mappings with an uniform metric that is scalable between different topology sizes as well. Finally, the models will be compared against a random policy and a random mapper in terms of rewards obtained during the training procedure, and will also be compared to the random mapper distance-wise. These two metrics were chosen because the reward provides insight into how the agent develops during the training procedure, while the distance portrays the performance during evaluation, during which no learning takes place. The random policy will perform actions completely at random, both legal and illegal, while the random mapper will only perform random, legal actions.

4 Results

In this section we provide the performance comparison of the DRL models when applied to three quantum architectures, the IBMQ Casablanca, Tokyo and Kolkata devices (7, 20, 27 qubits). These architectures were chosen, as their results show a trend, displaying the evolution of the performance of the DRL agents as the complexity (i.e., the number of qubits) of the topologies increases. We also provide the hyperparameters obtained during the finetuning procedure, visible in Table 2.

Table 2. Learning rates and discount factors for different models

Model/Parameters	α	γ
PPO	0.0003	0.8
A2C	0.0003	0.85
TRPO	0.001	0.8
PPO with invalid action mask	0.0003	0.99

The choice of parameters for the PPO algorithm balances learning stability with moderate emphasis on future rewards. The configuration of the A2C model uses the same learning rate as PPO but has a slightly higher discount factor, indicating a greater focus on future rewards. The optimal learning rate indicated by the finetuning procedure for TRPO reflects the method's capability for stable policy updates despite faster learning. Its discount factor is similar to PPO's. When it comes to the variant of PPO with an invalid action mask, the results from finetuning indicated that a lower learning rate with a significantly higher discount factor would be optimal. This configuration is particularly useful in environments with numerous invalid actions, ensuring that long-term rewards are heavily prioritized.

The evaluation results for the IBMQ Casablanca device (7 qubits) can be seen in Fig. 6(a) shows the four DRL models having similar median rewards around 0.15 and similar IQR spreads. It is TRPO that shows the presence of some

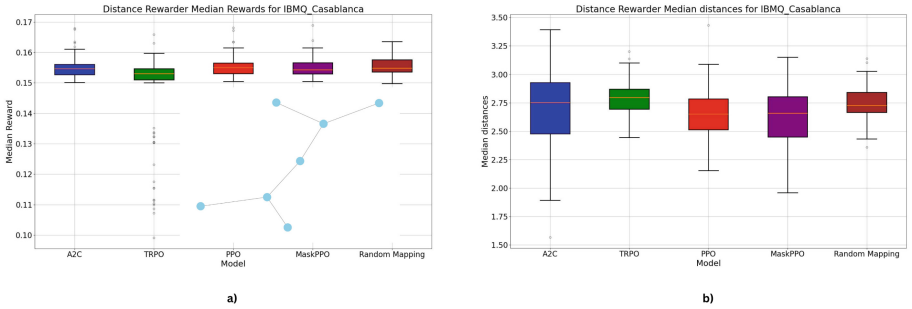

Fig. 6. Median rewards and distances for the IBMQ Casablanca device

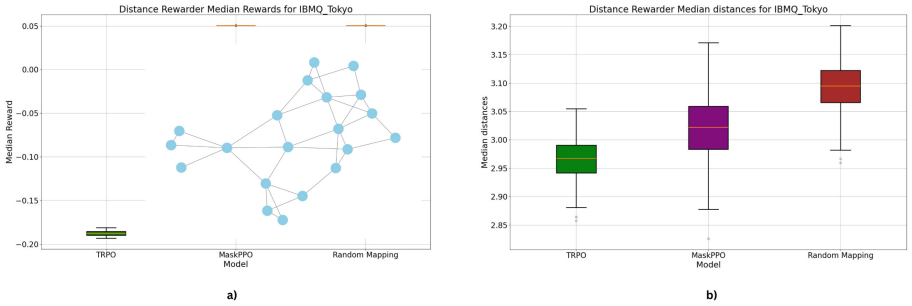

Fig. 7. Median rewards and distances for the IBMQ Tokyo device

outliers reaching towards the median reward of 0.1. In Fig. 6(b), it is shown that that PPO, along with its illegal action masking variant, have a smaller median distance compared to the other models, around 2.65. TRPO has the highest median distance value above 2.75, while A2C has an exact median distance of 2.75. MaskPPO was an IQR spread in the range $[2.5, 2.75]$, A2C an IQR spread in the range $[2.5, 2.9]$, whereas random mapping and TRPO have the smallest IQR spreads out of all four. A2C has the highest spread, with the whiskers of the boxplot having their minimum value towards 1.75.

For the IBMQ Tokyo device (20 qubits, Fig. 7), only TRPO and Maskable PPO have finished the evaluation procedure. In Fig. 7(a), PPO with invalid action masking and random mapping have a median reward of 0.05, while TRPO displays a median reward just above -0.2. Maskable PPO and TRPO outperform the random mapping technique in terms of mapping quality Fig. 7(b). Random mapping has a median distance of 3.1, PPO with invalid action masking displays a median distance between 3.0 and 3.05, whereas TRPO has a slightly lower median distance, between 2.95 and 3.0.

Moving on to a more complex architecture, Fig. 8 displays the results for the IBMQ Kolkata device, which features 27 qubits. For this device, the A2C and PPO with invalid action masking models are evaluated. In Fig. 8(a), it can

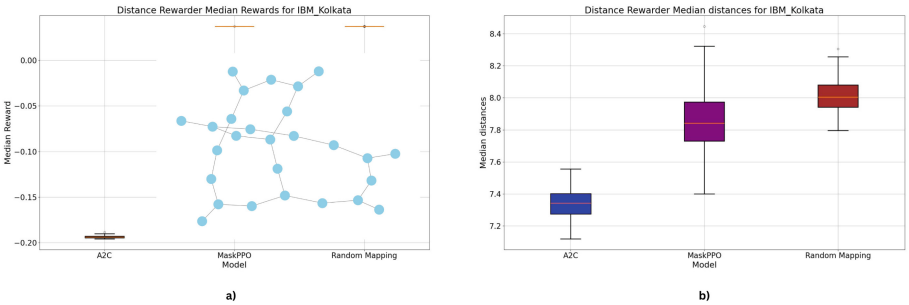

Fig. 8. Median rewards and distances for the IBMQ Kolkata device

be seen that Maskable PPO and random mapping achieved a median reward around 0.05, with A2C around −0.20. For this 27 qubit quantum device, Fig. 8(b) portrays the fact that the two DRL models outperform the random mapping approach. In terms of the quality of the obtained mappings, the random mapping has a median distance of 8.0, Maskable PPO a median distance just above 7.8, and A2C a median distance just below 7.4.

4.1 Discussion

Across the various quantum architectures tested, the performance of the Deep Reinforcement Learning (DRL) models, A2C, TRPO, PPO, and MaskPPO was consistent for small qubit topologies but varied significantly as the complexity of the topologies increased. For topologies featuring 5 qubits, such as the IBMQ Casablanca, all four DRL models displayed relatively similar median rewards and median distances, indicating comparable performance. However, the MaskPPO variant, which incorporates invalid action masking, slightly outperformed other models, particularly in avoiding illegal actions and achieving more stable median rewards. As we moved to more complex topologies however, such as the IBMQ Tokyo or Kolkata devices, with 20 and 27 qubits, the differences between models became more pronounced. While MaskPPO continued to perform well, the other algorithms started to display inconsistencies.

One key observation across all topologies is the relationship between the median distance achieved by the models and the number of qubits in the device. For smaller topologies, the DRL models typically achieved distances close to the optimal value 1. For these devices, all DRL models had median distances around 1.5 to 1.6, which aligns with what is expected for small circuits. This proves that Deep Reinforcement Learning agents are capable of computing high quality optimal initial mappings for small scale quantum computer architectures. For devices with 20 qubits, such as the IBMQ Tokyo, the DRL models continued to show an upward trend in median distances, reaching values around 4.9 and 5.5. The random mapping approach, in these scenarios, often produced results comparable to the DRL models, indicating that the complexity of larger devices

limited the ability of DRL models to uncover high quality mappings. For topologies with more than 20 qubits, such as the IBMQ Kolkata, the upward trend in median distance continued. For example, on the IBMQ Kolkata device, A2C and MaskPPO models displayed median distances between 7.4 and 7.8, which are somewhat higher distances than those observed in the smaller, previous topologies. The random mapping approach, for this quantum computer architecture, yielded a median distance of around 8.0, which shows that DRL models, though slightly better on this occasion, are having a harder time achieving distances that are close to the optimal value.

4.2 Limitations

One limitation of this project is the design of the fine-tuning procedure for the DRL models, which was focused on small-scale topologies. This approach may lead to suboptimal values for the learning rate (α) and discount factor (γ) when applied to more complex topologies. Expanding the fine-tuning process to include a wider range of qubits could improve generalization and performance but would require significantly more computational resources.

Furthermore, the failure of PPO to complete the evaluation procedure, due to convergence to a suboptimal policy, suggests a need for tuning the policy's entropy coefficient to promote more diverse action sampling and avoid premature convergence. This issue indicates that the agent may have become trapped in a local optimum or not explored the action space sufficiently, leading to suboptimal performance. Another limitation involved TRPO's application to large-scale quantum devices, which revealed challenges related to the exploding gradient phenomenon. The complexity of the task-mapping virtual qubits onto physical qubits-further compounds this issue, as it involves satisfying various constraints such as connectivity and gate error rates, leading to steep gradients in the loss landscape.

These challenges highlight the need for network architectures with deeper layers or specialized structures to capture intricate dependencies, along with optimization techniques to ensure stability and effective learning for complex quantum topologies. Addressing these limitations would likely involve more sophisticated fine-tuning procedures, tailored reward mechanisms, and advanced optimization strategies to enhance the robustness and scalability of DRL models for quantum computing applications.

5 Conclusion

In this study, we evaluated the performance of four policy gradient DRL models (A2C, PPO with and without action masking, and TRPO) on the task of computing high-quality initial mappings between virtual qubits of quantum circuits and physical qubits of various quantum computers. The analysis focused on two metrics, namely the median rewards and median distances. The findings reveal that policy gradient DRL models can generate high-quality initial mappings across small to medium hardware topologies.

For smaller topologies with up to 20 qubits, such as IBMQ Casablanca and Tokyo, the algorithms achieved near-optimal mappings in terms of distance, with consistent performance. However, as the scale increased to larger devices like the IBMQ Kolkata (27 qubits), the DRL agents had a harder time learning optimal policies. PPO with action masking demonstrated the highest adaptability and consistency, effectively balancing scalability and gate error rate considerations, and minimizing distances in complex scenarios. This demonstrates that DRL approaches, particularly PPO with action masking, are capable of effectively solving the initial mapping problem across a wide variety of quantum computer architectures.

The results indicate that, while DRL models can generate high-quality initial mappings for small to moderately sized architectures, they require careful calibration to perform effectively on larger systems. For complex topologies, some DRL models, mostly PPO, but occasionally TRPO and A2C, failed to train properly, preventing successful evaluation. This highlights the effectiveness of action masking (as used with PPO), which improves performance, especially in larger topologies. Although numerical improvements in median distances were modest, they could lead to reduced error rates and enhanced reliability for qubit connections in quantum computations.

The main bottlenecks for scaling DRL models to larger qubit systems include increased complexity in control systems and computational bottlenecks requiring more complex calibration procedures. Implementing multi-core architectures can help distribute the computational load and manage larger qubit counts more effectively. Leveraging advanced accelerated computing technologies, such as GPUs and AI-driven techniques, can significantly reduce computational time and cost for optimization tasks, such as the one tackled in this study. Fine-tuning DRL models on more complex architectures before training and evaluation can also lead to better performance.

Future research could look into comparing the performance of these trained RL agents to some well-known mappers, such as those from Qiskit. A proper benchmark against these other mappers would require some extra translation methods, as our agents work with interaction graphs instead of quantum circuits. Moreover, generating useful random circuits is a non-trivial task. Fine-tuning the models first on more complex architectures (i.e., more than 20 qubits), which is a resource-intensive task itself, could lead to improved performance during evaluation and result in higher quality mappings.

Acknowledgments. We thank QuTech and TNO for their support in realizing this research.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Bandic, M., Almudever, C.G., Feld, S.: Interaction graph-based characterization of quantum benchmarks for improving quantum circuit mapping techniques. Quantum Mach. Intell. **5**(2), 40 (2023)
2. Bandic, M., Feld, S., Almudever, C.G.: Full-stack quantum computing systems in the nisq era: algorithm-driven and hardware-aware compilation techniques. In: 2022 Design, Automation & Test in Europe Conference & Exhibition (DATE), pp. 1–6. IEEE (2022)
3. Brockman, G., et al.: OpenAI gym. arXiv preprint arXiv:1606.01540 (2016)
4. Cheng, C.Y., Yang, C.Y., Wang, R.C., Kuo, Y.H., Cheng, H.C., Huang, C.Y.: Qubit mapping toward quantum advantage. arXiv preprint arXiv:2210.01306 (2022)
5. Cheng, X., Guan, Z., Zhu, P.: Nearest neighbor transformation of quantum circuits in 2D architecture. IEEE Access **8**, 222466–222475 (2020)
6. Cook, S.A.: The complexity of theorem-proving procedures. In: Logic, Automata, and Computational Complexity: The Works of Stephen A. Cook, pp. 143–152 (2023)
7. Cowtan, A., Dilkes, S., Duncan, R., Krajenbrink, A., Simmons, W., Sivarajah, S.: On the qubit routing problem. arXiv preprint arXiv:1902.08091 (2019)
8. Erdos, P., Rényi, A., et al.: On the evolution of random graphs. Publ. Math. Inst. Hung. Acad. Sci **5**(1), 17–60 (1960)
9. Huang, C.Y., Lien, C.H., Mak, W.K.: Reinforcement learning and dear framework for solving the qubit mapping problem. In: Proceedings of the 41st IEEE/ACM International Conference on Computer-Aided Design, pp. 1–9 (2022)
10. Karuppasamy, K., Puram, V., Johnson, S., Thomas, J.P.: A comprehensive review of quantum circuit optimization: current trends and future directions. Quantum Rep. **7**(1), 2 (2025)
11. Kharkov, Y., Ivanova, A., Mikhantiev, E., Kotelnikov, A.: Arline benchmarks: automated benchmarking platform for quantum compilers. arXiv preprint arXiv:2202.14025 (2022)
12. Kwon, S., Tomonaga, A., Lakshmi Bhai, G., Devitt, S.J., Tsai, J.S.: Gate-based superconducting quantum computing. J. Appl. Phys. **129**(4) (2021)
13. Lao, L., Van Someren, H., Ashraf, I., Almudever, C.G.: Timing and resource-aware mapping of quantum circuits to superconducting processors. IEEE Trans. Comput. Aided Des. Integr. Circuits Syst. **41**(2), 359–371 (2021)
14. Li, Y., Liu, W., Li, M.: Deep reinforcement learning for mapping quantum circuits to 2D nearest-neighbor architectures. Adv. Quantum Technol. **7**(2), 2300289 (2024)
15. Maronese, M., Moro, L., Rocutto, L., Prati, E.: Quantum compiling. In: Quantum Computing Environments, pp. 39–74. Springer, Cham (2022)
16. Mnih, V., et al.: Asynchronous methods for deep reinforcement learning. In: International Conference on Machine Learning, pp. 1928–1937. PMLR (2016)
17. Nielsen, M.A., Chuang, I.L.: Quantum Computation and Quantum Information. Cambridge University Press, Cambridge (2010)
18. Preskill, J.: Quantum computing in the NISQ era and beyond. Quantum **2**, 79 (2018)
19. Quetschlich, N., Burgholzer, L., Wille, R.: Compiler optimization for quantum computing using reinforcement learning. In: 2023 60th ACM/IEEE Design Automation Conference (DAC), pp. 1–6. IEEE (2023)
20. Quetschlich, N., Burgholzer, L., Wille, R.: MQT predictor: automatic device selection with device-specific circuit compilation for quantum computing. ACM Trans. Quantum Comput. **6**(1), 1–26 (2025)

21. Raffin, A., Hill, A., Gleave, A., Kanervisto, A., Ernestus, M., Dormann, N.: Stable-baselines3: reliable reinforcement learning implementations. J. Mach. Learn. Res. **22**(1), 12348–12355 (2021)
22. Rattacaso, D., Jaschke, D., Ballarin, M., Siloi, I., Montangero, S.: Quantum circuit compilation with quantum computers. arXiv preprint arXiv:2408.00077 (2024)
23. Schulman, J., Levine, S., Abbeel, P., Jordan, M., Moritz, P.: Trust region policy optimization. In: International Conference on Machine Learning, pp. 1889–1897. PMLR (2015)
24. Schulman, J., Wolski, F., Dhariwal, P., Radford, A., Klimov, O.: Proximal policy optimization algorithms. arXiv preprint arXiv:1707.06347 (2017)
25. Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction. MIT Press, Cambridge (2018)
26. Tang, C.Y., Liu, C.H., Chen, W.K., You, S.D.: Implementing action mask in proximal policy optimization (PPO) algorithm. ICT Express **6**(3), 200–203 (2020)
27. Van Der Linde, S., De Kok, W., Bontekoe, T., Feld, S.: qgym: a gym for training and benchmarking RL-based quantum compilation. In: 2023 IEEE International Conference on Quantum Computing and Engineering (QCE), vol. 2, pp. 26–30. IEEE (2023)
28. Vaswani, A., et al.: Attention is all you need. In: Advances in Neural Information Processing Systems, vol. 30 (2017)
29. Wille, R., Burgholzer, L., Zulehner, A.: Mapping quantum circuits to IBM QX architectures using the minimal number of swap and h operations. In: Proceedings of the 56th Annual Design Automation Conference, pp. 1–6 (2019)
30. Wille, R., Hillmich, S., Burgholzer, L.: Efficient and correct compilation of quantum circuits. In: 2020 IEEE International Symposium on Circuits and Systems (ISCAS), pp. 1–5. IEEE (2020)
31. Williams, R.J.: Simple statistical gradient-following algorithms for connectionist reinforcement learning. Mach. Learn. **8**, 229–256 (1992)
32. Zhu, P., Guan, Z., Cheng, X.: A dynamic look-ahead heuristic for the qubit mapping problem of NISQ computers. IEEE Trans. Comput. Aided Des. Integr. Circuits Syst. **39**(12), 4721–4735 (2020)