Reward Design for RL-Based Initial Qubit Mapping: From Reward–Metric Alignment to Compiled-Circuit Quality

Federico Lentini

[1]Delft University of Technology (EEMCS) • Supervisor(s): Sebastian Feld, Akash Kundu • Responsible Professor: Matthijs Spaan

1. Background

- Quantum algorithms are usually defined over **logical qubits** and assume full qubit connectivity. Real quantum processors have **limited hardware connectivity**: two-qubit gates can only be executed directly between **physically connected qubits.**
- **Initial mapping** assigns logical qubits to physical qubits before routing. If interacting qubits are mapped to non-adjacent physical qubits, routing must insert additional SWAP gates.
- A good mapping leads to **fewer SWAPs** → lower number of gates → lower accumulated error. Since hardware fidelities are non-uniform, **reliable connections** should also be preferred.

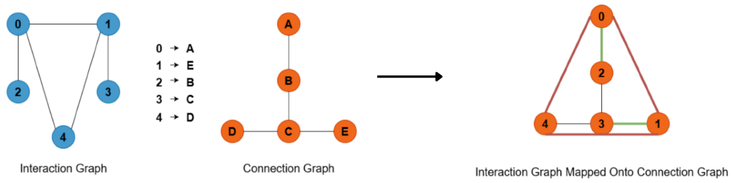

Figure 1. Process of mapping an **interaction graph** (logical qubits and two-qubit interactions) onto a **connection graph** (physical qubits of the hardware with available connections). Figure adapted from [2].

2. Research Objective

In RL-based initial mapping, the agent constructs the logical-to-physical qubit assignment sequentially. The **reward function** evaluates the quality of this mapping and therefore determines which mapping strategies the agent learns.

However, graph-level rewards are computed before routing, using only the interaction and hardware graphs. They therefore act as **proxies** for downstream compiled-circuit quality. This work formulates reward design as a **reward–metric alignment** problem.

Main question: *How well are graph-level rewards aligned with downstream compiled-circuit metrics, and how do agents trained with these rewards perform against established baselines?*

- **Q1:** How well do graph-level rewards align with their intended compiled-circuit metrics?
- **Q2:** How do reward formulation and reward timing affect PPO performance?
- **Q3:** How does RL-based initial mapping compare with Random, SABRE, and metric-specific oracle baselines?

3. Methodology

- **RL environment:** QGYM [5] initial-mapping environment with binary interaction graphs, fidelity-weighted hardware graphs, Maskable PPO, and action masking.
- **Experimental setup:** Random 5-qubit circuits, depth 8, two-qubit-gate probability 0.6, five hardware topologies, five PPO seeds, and $250\,000$ training timesteps.
- **Compilation and baselines:** PPO mappings are passed to Qiskit as `initial_layout`, followed by SABRE routing and ALAP scheduling; compared with Random, SABRE [1], and exhaustive metric-specific oracles.
- **Evaluation:** 100 unseen circuits and MQT Bench [3] circuits; metrics: compiled SWAP count and Log ESP; reward alignment measured with Spearman correlation over all $5! = 120$ mappings.

4. Reward Formulations

We evaluate graph-level rewards that score an initial mapping before routing, using hardware distance, edge fidelity, adjacency, and graph centrality:

- **Direct adjacency:** rewards interactions mapped to directly connected physical qubits.
- **Distance:** penalizes the hardware distance between interacting logical qubits and targets lower routing overhead.
- **Fidelity-path:** favours mappings that place interacting qubits on or near reliable hardware paths.
- **Hybrid distance–fidelity:** combines routing distance and hardware reliability into a balanced objective.
- **Centrality-based:** matches highly connected or central logical qubits with central physical qubits using degree or closeness rankings.

To test how feedback frequency affects PPO learnability, each main reward is evaluated using three timing variants:

- **Terminal:** feedback only after the complete mapping.
- **Shaped:** feedback after every qubit assignment.
- **n-step shaped:** feedback after every two assignments.

Evaluation Metrics

Compiled SWAP Count:

$$N_{\mathrm{SWAP}}(C_{\mathrm{comp}}) = \sum_{g \in C_{\mathrm{comp}}} \mathbb{I}[g = \mathrm{SWAP}].$$

Counts routing SWAPs, since input circuits contain none. Lower is better.

Log ESP:

$$\log \mathrm{ESP}(C_{\mathrm{comp}}) = \sum_{g \in G_{2q}(C_{\mathrm{comp}})} \log F_{e(g)}.$$

Log estimated success probability over compiled two-qubit gates. Higher is better.

Notation: C_{comp}: compiled circuit; g: compiled gate; $\mathbb{I}$: indicator function; G_{2q}: compiled two-qubit gates; $e(g)$: hardware edge used by g; $F_{e(g)}$: its fidelity.

5. Reward–Compiled Metric Alignment

For each circuit–topology pair, all $5! = 120$ initial mappings are scored using each graph-level reward and then evaluated after Qiskit compilation. Spearman correlation measures how similarly the reward and the compiled metric rank the mappings.

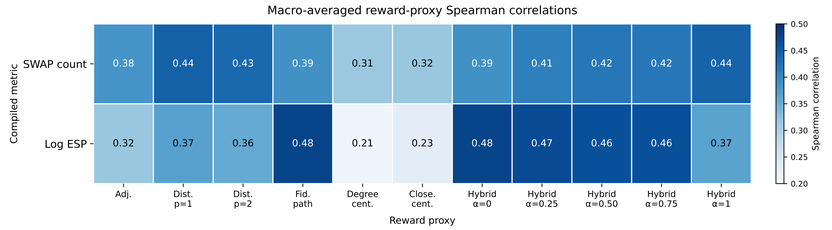

Figure 2. **Macro-averaged Spearman correlations between reward scores and compiled metrics.**

Takeaway. The correlations follow the intended reward objectives: Distance aligns best with compiled SWAP count ($\rho = 0.44$); fidelity-path aligns best with Log ESP ($\rho = 0.48$). Hybrid balances both ($\rho = 0.42$ SWAP, $\rho = 0.46$ Log ESP). Correlations remain moderate: graph-level rewards are useful but incomplete proxies for downstream compilation performance.

6. Downstream Compiled Performance

Terminal PPO results. PPO outperforms Random but remains behind SABRE. Hybrid improves over Random by $\mathbf{20.3}\%$ in SWAP count and $\mathbf{14.0}\%$ in log-error cost. Fidelity-path improves log-error cost by only $\mathbf{7.1}\%$, despite its strongest alignment with Log ESP.

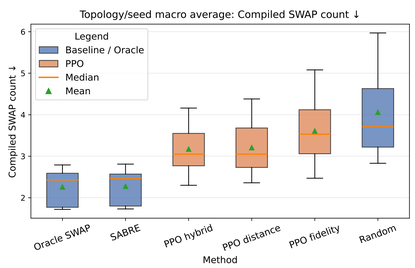

Figure 3. **Terminal PPO: SWAP count.** Lower is better.

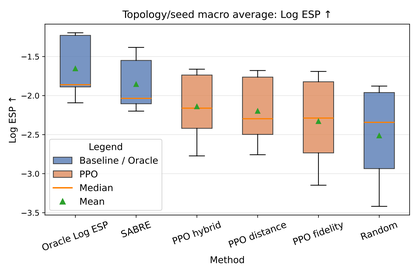

Figure 4. **Terminal PPO: Log ESP.** Higher is better.

Takeaway. Reward quality alone is not enough to guarantee performance: the reward must also provide a signal that PPO can learn effectively.

Reward Shaping results. Shaping has little effect on distance and hybrid. For fidelity-path, improvement over Random increases from 10.4% to 19.9% in SWAP count and from 7.1% to 14.8% in log-error cost. The n-step shaped variant achieves similar improvements.

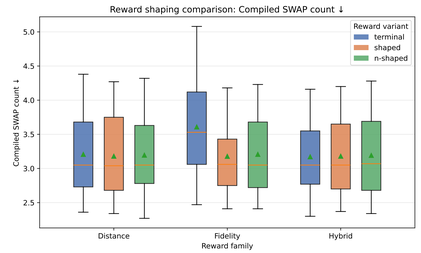

Figure 5. **Reward timing: SWAP count.**

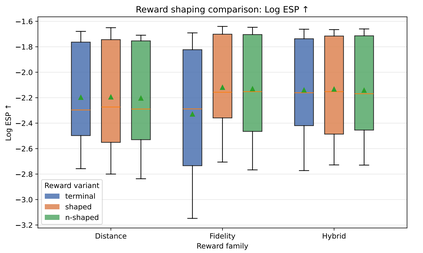

Figure 6. **Reward timing: Log ESP.** Higher is better.

Takeaway. Reward alignment and learnability are distinct: fidelity-path is strongly aligned with Log ESP, but requires shaped feedback to become an effective PPO training signal. Reward timing is therefore shown as an important design choice in reward engineering.

MQT Bench results. SABRE achieves the best mean performance across benchmark families, Random performs worst, and PPO variants show similar intermediate performance.

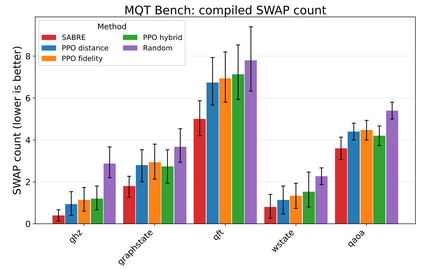

Figure 7. **MQT Bench: compiled SWAP count.** Lower is better.

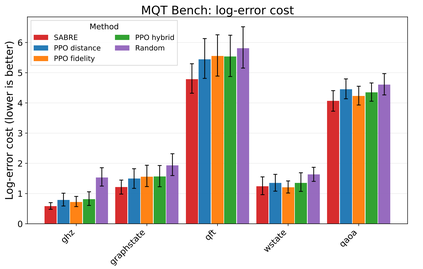

Figure 8. **MQT Bench: log-error cost.** Lower is better.

Takeaway. No reward provides a consistent advantage, indicating limited generalization beyond the random-circuit training distribution.

7. Conclusions & Next Steps

- **Graph-level rewards are useful but incomplete proxies:** correlations reach only $\rho \leq 0.48$, partly because binary interaction graphs omit interaction frequency, gate order, and dependencies that affect routing.
- **Reward alignment and learnability are distinct:** rewards must provide a signal that PPO can learn effectively; reward timing can improve this signal through denser feedback.
- **The rewards guide PPO towards better-than-random mappings,** but performance remains behind SABRE and generalization to MQT Bench is limited.
- **Next steps:** develop a broader framework combining alignment, learnability, and routing awareness, using richer circuit representations and evaluations on larger devices and more diverse circuits.

References

[1] G. Li, Y. Ding, and Y. Xie. Tackling the qubit mapping problem for nisq-era quantum devices. In *Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems*, ASPLOS '19, page 1001–1014, New York, NY, USA, 2019. Association for Computing Machinery.

[2] R. A. Oancea, S. van der Linde, W. de Kok, M. Sabatelli, and S. Feld. Optimizing initial qubit mappings under fixed gate error rates using deep reinforcement learning. In *International Conference on Innovations for Community Services*, pages 189–208. Springer, 2025.

[3] N. Quetschlich, L. Burgholzer, and R. Wille. Mqt bench: Benchmarking software and design automation tools for quantum computing. *Quantum*, 7:1062, 2023.

[4] R. S. Sutton, A. G. Barto, et al. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998.

[5] S. van der Linde, W. de Kok, T. Bontekoe, and S. Feld. qgym: A gym for training and benchmarking rl-based quantum compilation. In *2023 IEEE International Conference on Quantum Computing and Engineering (QCE)*, volume 02, pages 26–30, 2023.