

Adaptive Routing for Meshed QKD Networks of Flexible Size Using Deep Reinforcement Learning

Tim Johann *, Sebastian Kühl and Stephan Pachnicke 

Chair of Communications, Research Institute of Electrical Engineering and Information Technology,
Kiel University, 24143 Kiel, Germany; sebastian.kuehl@tf.uni-kiel.de (S.K.); stephan.pachnicke@tf.uni-kiel.de (S.P.)

* Correspondence: tim.johann@tf.uni-kiel.de

Abstract

Quantum Key Distribution (QKD) networks guarantee information-theoretical security of exchanged keys, but key rates are still limited. This makes efficient and adaptive routing a critical challenge, especially in meshed topologies without quantum repeaters. Conventional shortest path routing approaches struggle to cope with dynamic key store filling levels and changes in network topologies, which leads to load imbalance and blocked connections. In this work, we propose an adaptive routing framework based on Deep Reinforcement Learning (DRL) for hop-wise end-to-end routing in unknown meshed QKD networks. The agent leverages Graph Attention Networks (GATs) to process the network states of varying topologies, enabling generalization across previously unseen meshed networks without topology-specific retraining. The agent is trained on random graphs with 10 to 20 nodes and learns a routing policy that explicitly balances key consumption across the network by utilizing a reward function that is based on the entropy of key store filling levels. We evaluate the proposed approach on the 14-node NSFNET topology under time-varying traffic demands. Simulation results demonstrate that the DRL-based routing significantly outperforms hop-based and weighted shortest path benchmarks, achieving up to a 18.7% increase in mean key store filling levels while completely avoiding key store depletion. These results highlight the potential of graph-based DRL methods for scalable, adaptive, and resource-efficient routing in future QKD networks.

Keywords: Quantum Key Distribution; QKD networks; adaptive routing; Deep Reinforcement Learning; Graph Attention Networks; load balancing; Resource-Aware Optimization

1. Introduction

Quantum Key Distribution (QKD) is a promising technology to guarantee information-theoretical secure exchange of secret keys, which can be used to secure sensitive data even in the presence of future quantum computers [1]. QKD in general is of great interest for the research community and also first commercial deployments happen. Recent advances in QKD protocols and their implementation increase the diversity of practical QKD technologies. For example, experimental work has demonstrated experimental implementations of the B92 protocol using pulsed lasers, illustrating alternative prepare-and-measure schemes for key generation [2]. Additionally, multi-wavelength QKD emulations combined with physical unclonable functions have been proposed to increase protocol flexibility and security [3]. Nevertheless, the realization of a meshed QKD network is challenging as quantum repeaters are still in development [4]. Therefore, the key exchange has to be performed over so-called “trusted nodes” in a hop-wise manner [5]. This means that the scarce key



Received: 14 January 2026

Revised: 4 February 2026

Accepted: 14 February 2026

Published: 16 February 2026

Copyright: © 2026 by the authors.

Licensee MDPI, Basel, Switzerland.

This article is an open access article distributed under the terms and

conditions of the [Creative Commons Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

resources which are stored between two neighboring nodes have to be consumed along the complete path between source and destination nodes to establish a secret end-to-end key. Additionally, commercially available QKD devices are still limited with respect to reach and communication distance and secret key rate, especially due to fiber loss. As a result, it is mandatory that the key stores are always able to provide sufficient secret keys to enable a smooth network operation. This stands in stark contrast to traditional communication networks, where wavelengths or time slots are blocked for a certain time and released after the transmission finished. Secret keys as a resource have to be newly generated after usage, this constraint-based routing optimization problem has gained much interest in the past, as conventional shortest path (SP) algorithms often rely on specifically selected weighting functions, which struggle to adapt to the dynamics of QKD networks [6]. Currently, different options for the routing optimization of meshed QKD networks are under investigation. In ref. [7] the authors analyze a decentralized approach, where the secret keys are distributed in advance instead of an on-demand key relaying. Another investigation uses Deep Reinforcement Learning (DRL) for the selection of precomputed paths for solving the routing, wavelength and time slot allocation problem [8]. Overall, reinforcement learning has gained interest to solve routing problems in optical networks [9,10]. In ref. [11] the advantages of Graph Neural Networks (GNNs) are leveraged for adaptive routing in classical communication networks, where the agent is able to compute paths even in the presence of link failures in the training topology. In this paper, we present an adaptive routing agent that is able to route end-to-end key requests in a hop-wise manner while focusing on load-balancing in unknown meshed QKD networks of different sizes. The agent is trained on random topologies with 10 to 20 nodes, enabling it to operate without retraining or adaptations. This stands in contrast to our approach from [12], which was limited to topologies with 10 nodes, restricting the agent to this particular network size. To achieve such flexibility, we leverage the graph processing capabilities of Graph Attention Networks (GATs) [13]. Unlike traditional state-of-the-art shortest path routing algorithms, the agent is independent of well-designed static weighting functions. As our agent learns its routing policy directly from network states, it can adapt to topology changes and varying key resource distributions without the need for topology-specific retraining.

2. Deep Reinforcement Learning Agent Design

The DRL agent is trained using the Proximal Policy Optimization (PPO) algorithm with the policy and value networks using 2 hidden layers each of size 64 and tanh as its activation function to find a sequence of actions to connect a source and destination node for an end-to-end key request. PPO improves stability compared with unconstrained policy gradients by using a clipped surrogate objective that prevents excessively large changes to the policy in each update, constraining the policy ratio between consecutive updates and avoiding destabilizing updates [14,15]. The agent is trained on random topologies with 10 to 20 nodes, with randomly chosen node degrees from 2 to 5. Four exemplary topologies with different numbers of nodes, edges and varying connectivity are shown in Figure 1. As the focus of this paper is on load balancing a high connectivity is desirable to give several alternative routing options.

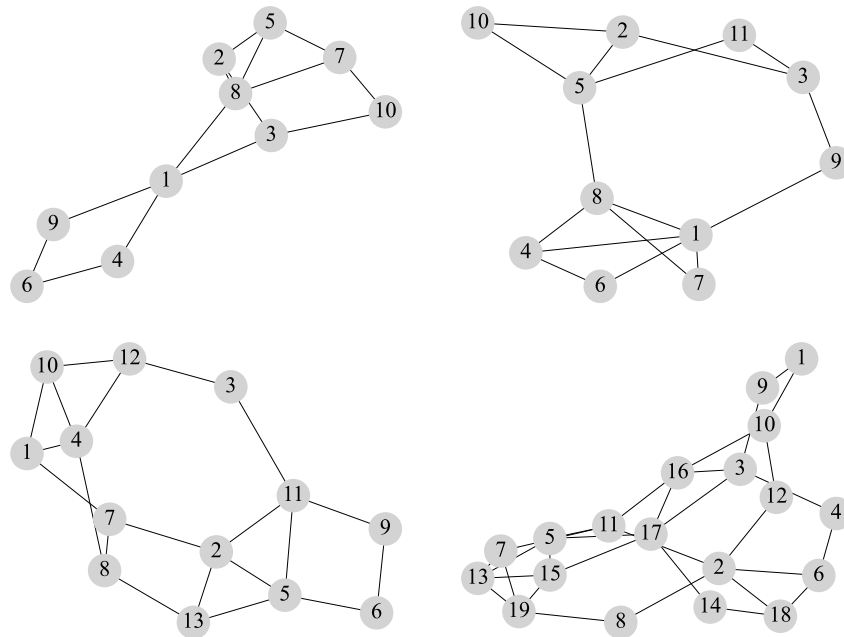


Figure 1. Exemplary training topologies with 10, 11, 13 and 19 nodes.

To enable direct observation of the different sized graph representations of the network topologies, a GAT based features extractor with 4 layers, each layer with 4 attention heads, is used. The attention heads enable the assignment of different weights to neighboring nodes, where each head learns a distinct weighting of node and edge features [13]. This allows the agent to emphasize different routing-relevant aspects of the local network state, such as key store filling levels or node connectivity, thereby improving decision-making in dynamically changing QKD network topologies. The GAT maps the node and edge features together with the edges themselves into a fixed sized intermediate representation with one feature per node in the topology. This is used as input to its policy and value networks. The state s contains the following information: the current position at node n of the agent in the network, the destination node $\hat{n}$, whether the current position has been visited before, the node degree, the mean key store filling levels of the direct neighbors of each node, the percentage of the key store filling levels of all edges and the required number of keys of the current end-to-end key request. A masking is applied to the state to keep the shape of the state constant as this is required by the python library stable-baselines3, which is used to implement the DRL agent [16]. Given this state s and action a selecting the next node $\bar{n}$, the reward function $R(s, a)$ is 0 when $n \neq \hat{n}$, provided that $\bar{n}$ is adjacent to n , sufficient keys are stored in the key store $(n, \bar{n})$ and $\bar{n}$ has not been visited before. Otherwise, the agent receives a reward of -1 . In the case of $n = \hat{n}$ the destination is reached and the agent is rewarded based on the change in entropy E of all initial key store filling levels f and current $\hat{f}$ to ensure load balancing, resulting in

$$r = 1 - \frac{\alpha \cdot (E(f) - E(\hat{f}))}{\beta}. \quad (1)$$

The entropy is calculated using the general form of Shannon entropy

$$E = - \sum_e (f_e \cdot \log(f_e)). \quad (2)$$

Hereby, f_e are absolute key store filling levels and not probabilities. The reward function is therefore normalized by the factor β . A higher entropy corresponds to a more

uniform distribution of key store filling levels across the network. The scaling factor α and normalization factor β in Equation 1 ensure the consistency of the reward for different sized topologies. The value for α has been empirically tuned to a value of 100. Figure 2 shows the influence of α on the reward. If α is too high, the reward may fluctuate significantly depending on key store filling levels, particularly over longer paths. This can negatively affect the agent's routing decisions and lead to suboptimal behavior. The normalization factor β is the maximum possible entropy of the edge-feature distribution, corresponding to a uniform utilization of all edges. This makes the reward more robust and less variant to changes in network topology and the number of edges and key stores. The agent is trained for 1 million actions with a learning rate of 2×10^{-4} . A training episode ends if the destination is reached or the agent performs an invalid action. At the end of an episode a new random topology is generated. It takes the trained agent around 1 to 10 ms to choose the next hop of a routing request based on the current network state.

In Figure 3 the evolution of the mean reward during training over the number of training steps is shown. After a short starting phase with mostly negative rewards, the agent optimizes its policy and performs better actions and therefore increases the mean reward. After around 800,000 steps the mean reward converges and the training stops after around 1 million steps.

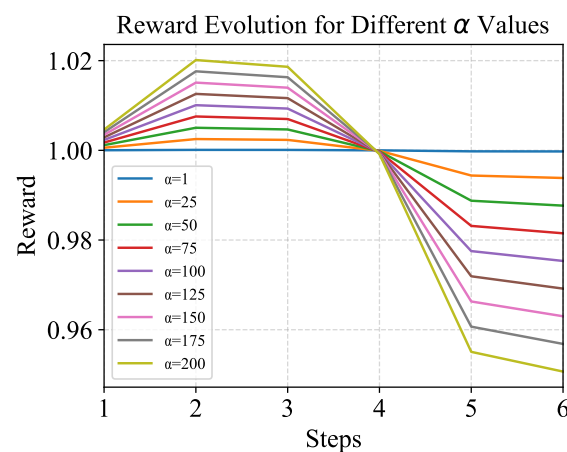


Figure 2. Evolution of the reward for different α values over the number of steps.

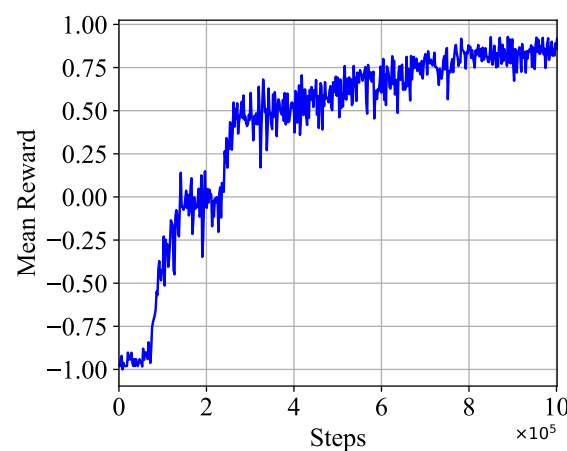


Figure 3. Evolution of the reward during training.

3. Simulation Setup

After the training phase the agent is placed in a network, which has not been used during training, to evaluate the generalization abilities of the agent. Here, we used the

14-node NSFNET topology depicted in Figure 4 and assumed a central SDN controller with full access to the network parameters. Due to range limitations of current QKD technology, trusted nodes are placed as relay nodes every 80 km to bridge longer distances. These relay nodes are ignored in the routing process as no routing decision is made in these nodes. The routing of an end-to-end secret key between two non-neighboring nodes takes place according to the XOR-approach, which was introduced in ref. [17].

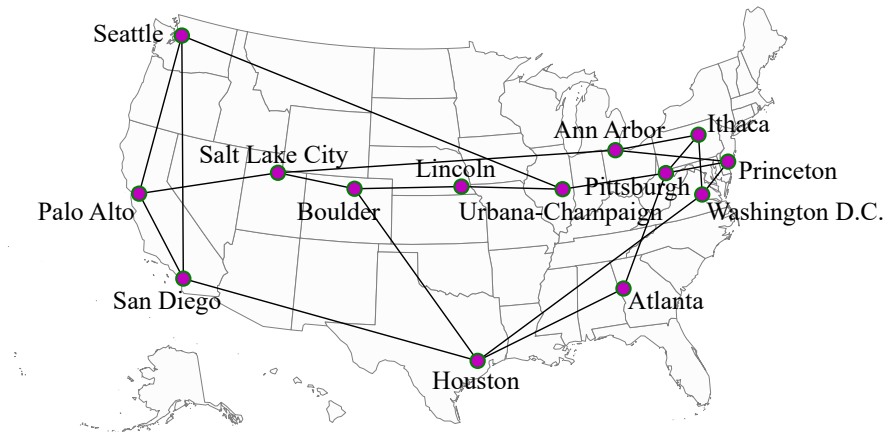


Figure 4. 14-node NSFNET topology used for evaluation of the adaptation abilities of the agent to an unknown topology.

The virtual key buffers are therefore consequently assumed only between two neighboring access nodes, which are depicted in Figure 4. The key stores are assumed to be of same size and completely filled at the beginning of the evaluation. The end-to-end key requests are generated based on a sinusoidal process over the course of the day with a higher number of requests drawn from a Poisson distribution during working hours (6 a.m. to 6 p.m.) and with a lower number of requests at lunch or at night time. We additionally account for differences in time zones. The requests are processed in time intervals of 5 min, where the DRL agent routes each request hop-by-hop consuming the secret keys from the used edges until it reaches the destination. The amount of required secret key material per request is drawn from a log-normal distribution and the values lie between 1% and 2% of the maximum key store filling level. This way the requests of a single timestep do not drain the key stores immediately. After processing all key requests of an interval, new keys are generated between the nodes and the key stores are refilled. For the sake of simplicity, we assumed random regeneration rates between 10% and 15% of the maximum key store filling levels. This way the key stores have the possibility to regenerate over time. As a benchmark for the routing performance of the DRL agent, we used two different shortest path algorithms which route the same end-to-end key requests. First, we used a simple hop-based shortest path (SP) algorithm, which minimizes the number of hops between source and destination nodes. Second, we implemented a weighted shortest path algorithm where the edge weights take the filling levels of the key stores into account. For both algorithms the cost of an edge that has an insufficient number of keys is set to infinity.

4. Results and Discussion

Figure 5 shows the heatmap of the mean key store filling levels for the different routing approaches. The simulation is repeated for 100 realizations for different random traffic patterns. The hop-based shortest path algorithm drains many key stores after a short time. This is expected, as the routing algorithm does not take the filling levels into account. Some stores work at their capacity limit most of the time, which leads also to the blocking of

some requests as some nodes are not reachable until enough keys are reproduced or only accessible via a long detour. The weighted SP considers the key store filling levels for path computation and performs better, but some key stores are still heavily utilized. This is leading to a blocking of certain edges for some time. The DRL agent distributes the traffic more evenly and shortens the duration where the edges have a low key store filling level. In contrast to the baseline algorithms, none of the key stores are drained completely.

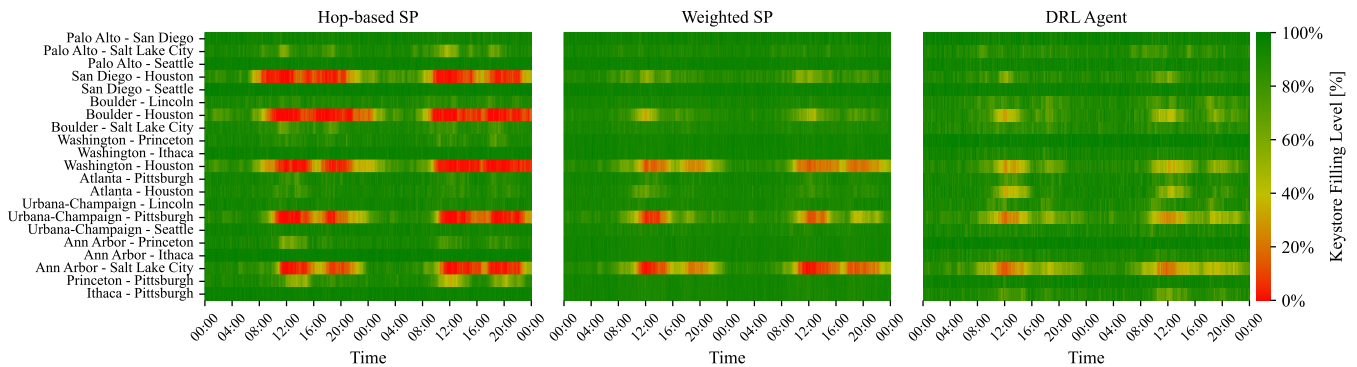


Figure 5. Heatmap of the mean key store filling levels of all edges in the topology over time. Low key store values are depicted in red and high key store levels are depicted in green.

Figure 6a exemplarily depicts the mean key store filling level and its standard deviation for the edge between Ann Arbor and Salt Lake City in more detail to underline the agent's routing performance. The DRL agent, following its learned policy, routes using slightly longer paths to distribute the load more evenly. This avoids phases of full utilization completely, shortens phases of high utilization and speeds up recovery time. As the network load increases in the morning, the DRL agent is able to keep the edges' key store filling level at around 15%, while both shortest path algorithms drain the store completely leading to blocking of connections. This is also reflected by the mean key store filling level over time. The hop-based SP shows a mean filling level of only 46.3% and the weighted SP shows a mean of around 48.6%. The DRL agent outperforms this by keeping the mean store level at a value of 57.7%, which means an increase of 18.7% compared to the weighted SP. In contrast to that, Figure 6b shows the edge between Ithaca and Pittsburgh, which shows a high key store filling level in general. During the time-span where other edges tend to overload, uses less heavily utilized used edges to relax the other key stores. At the same time, although the edge is used more frequently by the DRL agent, it is never over-utilized.

In Table 1 the percentage and absolute values for the different path length of the three algorithms is listed. As the focus the hop-based SP algorithm is the minimization of hops most of the path lengths are between 1 and 3 hops. Additionally, plenty of outliers are in the range between 4 and up to 9 hops. This is caused by the over-utilization of certain edges, which results into large detours to reach the node. The weighted SP algorithm shows a much lower amount of paths with more than 3 hops. The DRL agent shows the highest number of paths with 4 hops, but the lowest number of outliers compared to the other algorithms. This underlines the assumption that the agent prefers slightly longer paths to balance the load more evenly, without an over-utilization which would cause large detours as it is observable by looking at the hop-based algorithm.

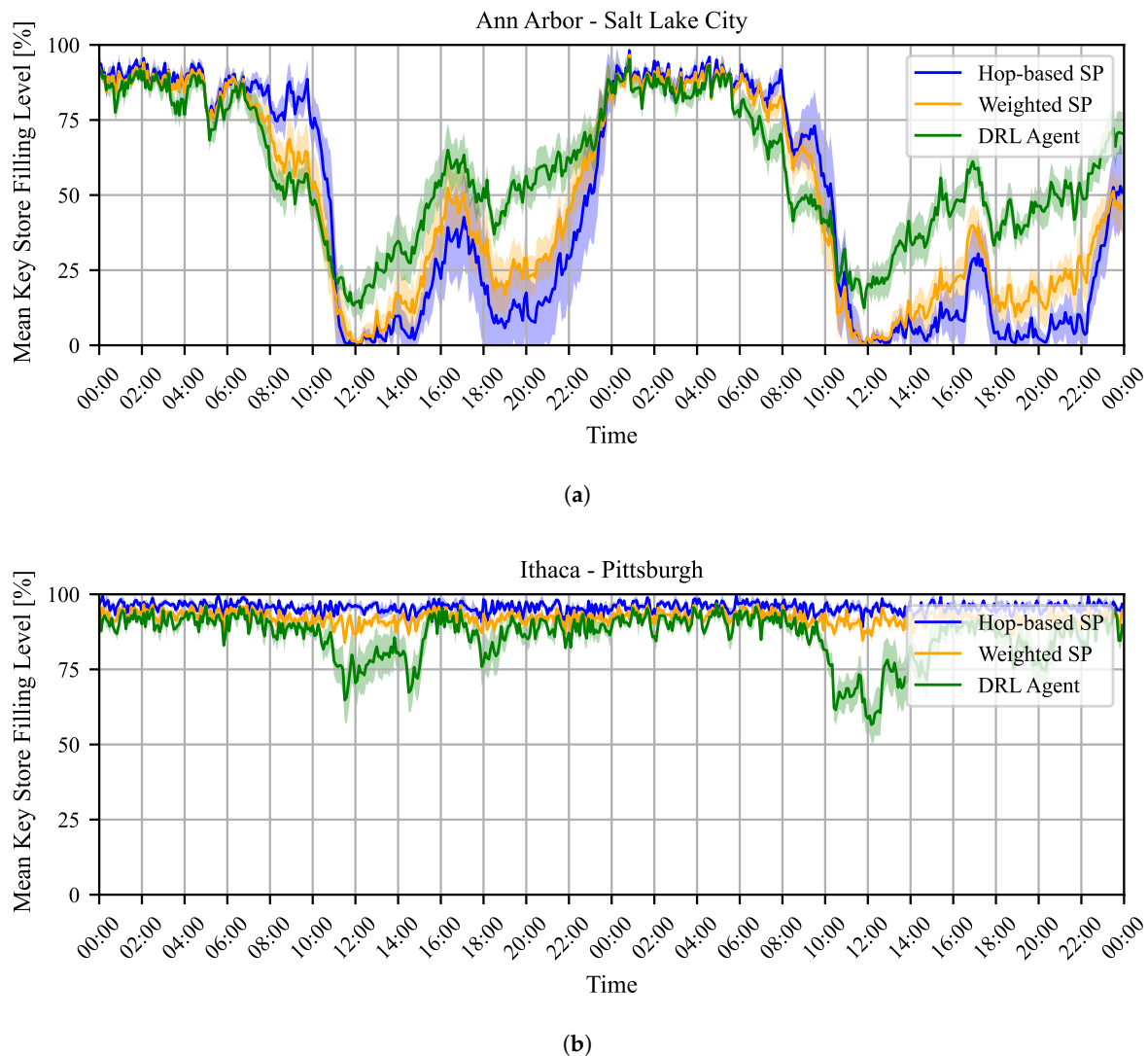


Figure 6. (a) Filling level of the key store between Ann Arbor and Salt Lake City over time. (b) Filling level of the key store between Ithaca and Pittsburgh over time.

Table 1. Path lengths of the different routing algorithms for 100 different simulation results with a total amount of 4,118,176 requests.

Path Length [Hops]	Hop-Based SP	Weighted SP	DRL Agent
1	22.69% (934,225)	22.98% (946,196)	23.00% (947,037)
2	38.82% (1,598,453)	39.69% (1,634,358)	38.38% (1,580,472)
3	35.83% (1,475,625)	36.84% (1,517,287)	33.18% (1,366,214)
4	1.46% (60,191)	0.43% (17,775)	5.38% (221,495)
5	0.43% (17,861)	0.05% (2099)	0.07% (2737)
6	0.12% (4889)	0.009% (387)	0.005% (204)
7	0.05% (1859)	0.002% (74)	0.0004% (16)
8	0.02% (706)	0.00% (0)	0.00002% (1)
9	0.00002% (1)	0.00% (0)	0.00% (0)
Blocked	0.43% (24,366)	0.00% (0)	0.00% (0)

Related work has also explored alternative routing strategies in QKD networks. For example, ref. [18] focuses on routing based on a well-designed weighting function. In ref. [19] the authors used DRL for route selection in fixed topologies for routing optimization. In contrast to these approaches and other state-of-the-art shortest path algorithms, where

e.g., the weighting function needs to be adapted to changes in the topology, the DRL agent is able to learn a robust policy without the need for retraining even if the topology is changing.

The proposed DRL agent is trained on high-level features such as edge key store filling levels. This design allows the agent to generalize to heterogeneous nodes with different hardware capabilities or protocol implementations, such as measurement-device-independent (MDI) QKD nodes or nodes supporting different degrees of freedom, without retraining. In such scenarios, the agent can adapt its routing policy based on the observed network state, effectively balancing load and avoiding bottlenecks, similar to the orchestration and protocol adaptation mechanisms described in ref. [20,21].

5. Conclusions

We presented, to the best of our knowledge, the first adaptive DRL routing agent, which is able to route end-to-end key requests in unseen meshed QKD networks of varying size without relying on conventional shortest path algorithms. This adaptability is reached by leveraging the graph processing abilities of GATs. The agent outperforms the well-known shortest path algorithms in terms of load-balancing. This allows a constant operation of the network by efficiently using the secret keys as a resource without over-utilization of certain edges. The investigations show that the DRL agent is capable to reach a higher mean key store filling level for heavily utilized edges. This has been shown by looking at the connection between Ann Arbor and Salt Lake City, where the DRL agent is able to increase the mean key store filling level by 18.7% compared to a weighted shortest path algorithm. To achieve this, the agent takes slightly longer paths into account to avoid bottlenecks.

Author Contributions: T.J. conceived the overall study and developed the methodology, with contributions from S.K. and S.P., T.J. conducted the analysis. T.J. and S.K. drafted the manuscript. S.P. secured funding and supervised the project. All authors have read and agreed to the published version of the manuscript.

Funding: This work has been performed in the framework of the QuNET+ML project and it is partly funded by the German Federal Ministry for Research, Technology and Space (16KISQ069).

Data Availability Statement: Inquiries considering further information about the simulation setup can be directed to the corresponding author.

Conflicts of Interest: The authors declare no conflicts of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

References

1. Scarani, V.; Bechmann-Pasquinucci, H.; Cerf, N.J.; Dušek, M.; Lütkenhaus, N.; Peev, M. The Security of Practical Quantum Key Distribution. *Rev. Mod. Phys.* **2009**, *81*, 1301–1350.
2. Gandelman, S.P.; Maslennikov, A.; Rozenman, G.G. Hands-On Quantum Cryptography: Experimentation with the B92 Protocol Using Pulsed Lasers. *Photonics* **2025**, *12*, 220.
3. Riggs, B.; Partridge, M.; Cambou, B.; Rios, M.A.; Heynssens, J.; Ghanaimiandoab, D. Multi-wavelength Quantum Key Distribution Emulation with Physical Unclonable Function. *Cryptography* **2022**, *6*, 36.
4. Azuma, K.; Economou, S.E.; Elkouss, D.; Hilaire, P.; Jiang, L.; Lo, H.-K.; Tzitrin, I. Quantum repeaters: From quantum networks to the quantum internet. *Rev. Mod. Phys.* **2023**, *95*, 045006.
5. Salvail, L.; Peev, M.; Diamanti, E.; Alléaume, R.; Lütkenhaus, N.; Länger, T. Security of trusted repeater quantum key distribution networks. *J. Comput. Secur.* **2010**, *18*, 61–87.
6. Nannicini, G.; Liberti, L. Shortest paths on dynamic graphs. *Int. Trans. Oper. Res.* **2008**, *15*, 551–563.
7. Álvarez Roa, M.; Stan, C.; Verschoor, S.; Monroy, I.T.; Rommel, S. Decentralized key distribution versus on-demand relaying for QKD networks. *J. Opt. Commun. Netw.* **2025**, *17*, 732–742.

8. Sharma, P.; Bhatia, S.; Prakash, S. Routing Based on Deep Reinforcement Learning in Quantum Key Distribution-secured Optical Networks. In *2023 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS), Jaipur, India, 2023*; IEEE: New York, NY, USA, 2023.
9. Mammeri, Z. Reinforcement Learning Based Routing in Networks: Review and Classification of Approaches. *IEEE Access* **2019**, *7*, 55916–55950.
10. Doherty, M.; Matzner, R.; Yamchi, R.S.; Bayvel, P.; Beghelli, A. Reinforcement learning for dynamic resource allocation in optical networks: Hype or hope? *J. Opt. Commun. Netw.* **2025**, *17*, D1–D17.
11. Rao, Z.; Xu, Y.; Yao, Y.; Meng, W. DAR-DRL: A dynamic adaptive routing method based on deep reinforcement learning. *Comput. Commun.* **2024**, *228*, 107983.
12. Johann, T.; Kühl, S.; Pachnicke, S. Deep Reinforcement Learning Based Topology Independent Routing and Load-Balancing in QKD Networks using Graph Neural Networks. In *25th Anniversary International Conference on Transparent Optical Networks (ICTON), Barcelona, Spain, 2025*; IEEE: New York, NY, USA, 2025.
13. Brody, S.; Alon, U.; Yahav, E. How Attentive are Graph Attention Networks? *arXiv* **2021**, arXiv:2105.14491.
14. Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; Klimov, O. Proximal Policy Optimization Algorithms. *arXiv* **2017**, arXiv:1707.06347.
15. Holzeitner, M.; Gruber, L.; Arjona-Medina, J.; Brandstatter, J.; Hochreiter, S. Convergence Proof for Actor-Critic Methods Applied to PPO and RUDDER. *arXiv* **2020**, arXiv:2012.01399.
16. Raffin, A.; Hill, A.; Gleave, A.; Kanervisto, A.; Ernestus, M.; Dormann, N. Stable-Baselines3: Reliable Reinforcement Learning Implementations. *J. Mach. Learn. Res.* **2021**, *22*, 1–8.
17. Johann, T.; Wenning, M.; Giemsa, D.; Dochhan, A.; Gunkel, M.; Fehenberger, T.; Pachnicke, S. Comparison and optimization of different routing methods for meshed QKD networks using trusted nodes. *J. Opt. Commun. Netw.* **2024**, *16*, 382–391.
18. van Duijn, T.; Verschoor, S.R.; Rommel, S.; Monroy, I.T. Routing strategies for quantum key distribution networks based on trusted relay nodes. In *Proceedings of the 2024 International Conference on Optical Network Design and Modeling (ONDM), Madrid, Spain, 6–9 May*; IFIP: Stockholm, Sweden, 2024.
19. Sharma, P.; Gupta, S.; Bhatia, V.; Prakash, S. Deep reinforcement learning-based routing and resource assignment in quantum key distribution-secured optical. *IET Quantum Commun.* **2023**, *4*, 136–145.
20. Lu, F.-Y.; Wang, Z.-H.; Zhou, Y.; Fan, Y.-X.; Wang, S.; Yin, Z.-Q.; Li, J.; He, D.-Y.; Wang, F.-X.; Chen, W.; et al. Fully heterogeneous prepare-and-measure quantum network for the next stage of quantum internet. *Nat. Commun.* **2025**, *16*, 11487.
21. Fan-Yuan, G.-J.; Lu, F.-Y.; Wang, S.; Yin, Z.-Q.; He, D.-Y.; Zhou, Z.; Teng, J.; Chen, W.; Guo, G.-C.; Han, Z.-F. Measurement-device-independent quantum key distribution for nonstandalone networks. *Photonics Res.* **2021**, *9*, 1881–1891.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.