LEARNING AND INTERPRETING POLICIES FOR SIMULTANEOUS ENTANGLEMENT REQUESTS IN QUANTUM NETWORKS

Anonymous authors
Paper under double-blind review

ABSTRACT

Future quantum networks will make use of entanglement to perform numerous tasks, such as sending quantum information over long distances, distributed quantum computing, and quantum sensing. In general, these tasks will need to be performed simultaneously in various regions of a network, while minimizing resources and latency. We will thus require policies for scheduling link-level entanglement resources, and using the link-level entanglement to create various forms of multipartite entanglement required for every task. In this work, we address this problem using reinforcement learning. We formulate a Markov Decision Process for the problem and use double deep Q-networks (DQN) with Message Passing Neural Networks (MPNNs), experience replay buffers, and curriculum training to obtain policies. The key physical parameter is the probability of link-level entanglement generation, i.e., the link activation probability. We show that our policies maintain 100% success for up to 71% lower link activation probability than the baseline heuristics for a set of physically relevant network topologies. We then examine an additional constraint where experiment (task) placements are restricted to specific hardware types and demonstrate a similar advantage in performance over heuristics, with our policy maintaining at least an 80% success rate for up to a 59% lower link activation probability. Finally, to understand how our policies obtain their advantage, we explore methods to interpret the learned policy by defining metrics enabling conclusions to be drawn about the model's behavior and by tasking a large language model (LLM) to derive a novel heuristic given example actions taken by the DQN-trained policy. We find that the LLM heuristic performs similarly to the DQN-trained policy in performance, indicating a promising method for interpretable policy extraction for large quantum networks, where direct training becomes computationally expensive.

1 INTRODUCTION

One of the grand visions of quantum science and technologies is the realization of a quantum internet (Wehner et al., 2018), a globally interconnected network of networked quantum computers that will, in conjunction with the existing (classical) internet, enable a host of applications, including those with known quantum advantages. Such applications include quantum computation, sensing, interferometry, and secure communication (Wei et al., 2022; Azuma et al., 2023).

While much work remains to realize this vision (Awschalom et al., 2021), networked quantum computation is an especially intriguing idea already, due to the fact that it could be used to scale up quantum computation itself. One of the current challenges hindering large-scale quantum computation on a single quantum computer is the qubit overhead of error correction. Currently, monolithic quantum computer architectures exist that can only support roughly hundreds of physical qubits, while qubit overhead for fault-tolerant error correction imposes a requirement of many more qubits (Zhao et al., 2026). Scaling beyond these values in monolithic architectures, while enabling fault tolerance via error correction, remains a challenge. Using entanglement to connect many such computers offers a path towards large-scale quantum computation. In fact, a successful demonstration of Grover's algorithm and the execution of other arbitrary two-qubit gates in a distributed setting has recently been shown (Main et al., 2025).

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

The global development of quantum network test beds is increasing (Sasaki et al., 2011; Hofmann et al., 2012; Knaut et al., 2024; Pompili et al., 2021; 2022; Chung et al., 2021; 2022; Craddock et al., 2024). In these networks, quantum nodes are physically connected using fiber optic cables that are later used to create entanglement pairs between neighboring nodes. However, not all quantum circuits can be directly executed on the network due to a mismatch between the quantum circuit pattern and the physical topology of the network. For example, if a network were constructed from four single-qubit quantum nodes connected in a ring-like fashion, it would not support a 4-qubit quantum circuit containing three two-qubit CNOT gates between q_0 and q_1, q_1 and q_2, and q_2 and q_0. In this case, a "virtual" link is needed to support one of the CNOT operations. To obtain a virtual link between distant nodes u and w, both independently sharing entanglement with v, one performs entanglement swapping Żukowski et al. (1993) between both pairs u-v and v-w, resulting in a shared entanglement between u and w. This process can be repeated to enable virtual links over greater distances.

Previous research has examined linear programming (Khatri, 2022; Iñesta et al., 2023; Fan et al., 2025), decision transformers, (Seok et al., 2026), and learning approaches Reiß & van Loock (2023); Le & Nguyen (2022); Haldar et al. (2024); Li et al. (2025) to address the problem of efficiently resolving a queue of virtual link generation requests on the network in minimal time, while maximizing the fidelity of the resulting virtual link. However, in our work, we examine the problem of fulfilling not the queue of virtual link requests but a given queue of "jobs" or "experiments" that end-users may request to execute. Each such experiment is a particular *subgraph* of physical and/or virtual links in the network, which could correspond to a multipartite state used for quantum sensing or for a distributed quantum computation. Given a set of such subgraphs, each requiring a given duration of time to execute, our goal is to determine an optimal policy that generates all subgraphs in the network, minimizing the time required. This formulation was examined in Ni et al. (2026), in which they propose heuristics to efficiently schedule the experiments. For this problem, virtual link generation is a *subroutine* necessary to *enable* the execution of an arbitrary distributed quantum circuit. In the quantum analogy of the classical network stack, fulfilling virtual link generation requests is contained in the so-called connectivity layer, whereas the problem we examine lies in the link layer as it involves requesting the required virtual links from the connectivity layer to schedule the experiments (Pirker & Dür, 2019). The problem has real-world implications in that broader usage of a mature quantum internet will result in a high number of end-users seeking to execute circuits of inherently variable topology demands on a network limited to a static physical topology.

We also examine an additional constraint in which a quantum circuit is confined to specific quantum nodes in the network for reasons such as hardware compatibility. The network operator will need to efficiently schedule the execution of the circuits on the network while maximizing fidelity over them and satisfying the constraint.

We find that our models outperform baselines in terms of robustness to changes in link activation probability, measured by the success rate and mean episode length. In particular, we find that our models succeed more frequently and more efficiently in more lossy environments than the baselines, indicating that lossy, near-term quantum network implementations can benefit from DQN for multipartite entanglement scheduling tasks. Additionally, this performance extends to a realistic constraint in which certain nodes in the network may be preferred by the network's users, which limits their experiment's possible locations on the network, ultimately increasing the role bottleneck nodes might have. Lastly, by prompting an LLM to infer a policy from example trajectories generated by the DQN model, we were able to extract a policy–expressible using natural language– that performs nearly as well as our DQN model, showing the significance of LLMs in extracting performant yet interpretable policies given a DQN training process.

2 Network Model and Problem Formulation

Let G be a network of nodes N and a set of physical links, E, representing optical connections between nodes. Let there be μ parallel links (sublinks) between neighboring nodes, enabled by multiple quantum memories operating on each node. Thus G is a multi-graph. Hereafter, a physical link between two nodes refers to any of the μ sublinks between them.

The network evolves in discrete time steps $t \in \mathbb{N}_0$. E is partitioned into active and inactive physical links. At each time step, an inactive physical link attempts to activate, succeeding with probability

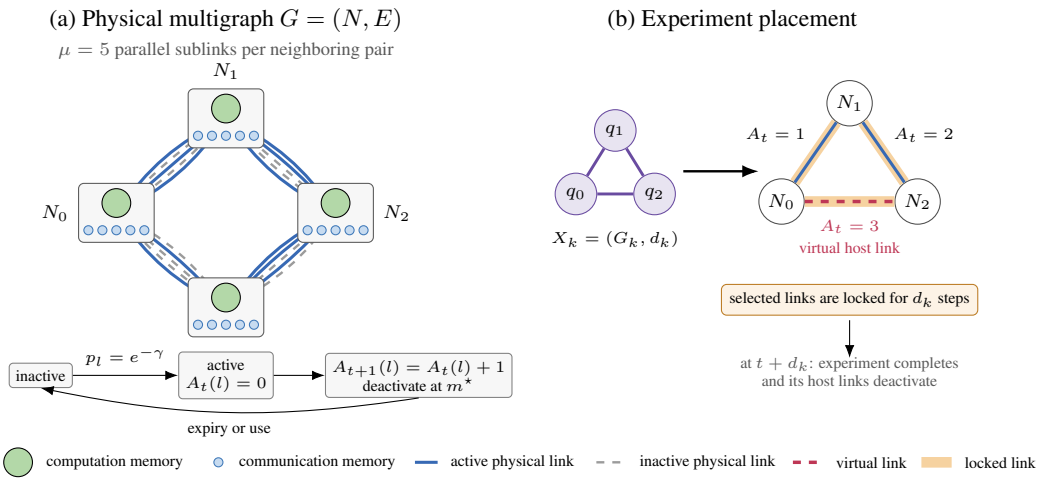

Figure 1: (a) A quantum network composed of $N = 4$ quantum memories organized as a ring, with $\mu = 5$ parallel sublinks between neighboring nodes. Links move from an inactive state to an active state with probability $p_l = e^{-\gamma}$ after which their age increments from zero to $m^\star$. (b) Entanglement swapping is performed across link N_0-N_1 and link N_1-N_2 with ages 1 and 2 respectively, establishing a virtual link between N_0-N_2 with age 3. This enables experiment X_k to be placed on host nodes N_0, N_1, N_2. The links are consumed for d_k steps before being released.

$p_l = e^{-\gamma}$, where γ represents the attenuation coefficient of the link. Upon successful activation, the link l initializes at age 0 and increments each step so that its age satisfies $A_t(l) \in \{0, 1, ..., m^\star - 1\}$, remaining active for at most $m^\star \in \mathbb{N}$ time steps before deactivating.

A "virtual" link may be generated between two non-neighboring nodes via entanglement swapping over their shortest connecting path of links (both physical and virtual), P. Given P, if $\sum_{l \in P} A_t(l) < m^\star$, a virtual link may be generated between the path's endpoints, whose age is $\sum_{l \in P} A_t(l)$. Once generated, all $l \in P$ become inactive. Hence, in addition to E, we have $V = (N \times N) \backslash E$ of generable virtual links in the network G. We assume that entanglement swapping succeeds deterministically.

Let $\mathcal{X}$ represent the set of experiments to be executed across distributed nodes. Each experiment defines a topology of shared entangled pairs between quantum nodes. Gate speeds, teleportation overhead, classical communications, and other factors constitute the duration of the experiment. These parameters define an individual experiment $X_k \in \mathcal{X}$, $X_k = (G_k, d_k)$, where $G_k = (N_k, E_k)$ is the required topology of experiment k and $d_k > 0$ is its duration. Figure 1 describes the network dynamics on a four node graph, including virtual link generation and experiment placement.

Let $\tilde{G}_t = (N, \tilde{E}_t \cup V_t)$ be named the "active" graph, whose set of edges is composed of active physical links $\tilde{E}_t$ and generated virtual links V_t at time t. An experiment can be "placed" onto $\tilde{E}_t \cup V_t$ if (i) G_k is subgraph isomorphic to $\tilde{G}_t$ (that is, there exists an injective mapping $f : N_k \to N$ such that $(u, v) \in E_k \Rightarrow (f(u), f(v)) \in \tilde{E}_t \cup V_t$), and (ii) for all active host links $l \in \tilde{E}_t \cup V_t$, $A_t(l) \leq m^\star - d_k$, i.e. no link will deactivate before d_k time steps pass. Once X_k is placed, all host links are considered locked for the duration d_k and cannot be re-assigned or used for virtual link generation. Upon completion of X_k at step $t + d_k$, the host links are deactivated, after which they repeatedly try to activate with success probability p_l.

We also consider the constraint in which an experiment X_k may limit its placement to a subset of host nodes. As the quantum network may be heterogeneously composed of various types of quantum memory, a user may desire to delay the placement of their experiment until the memory with desired characteristics can be utilized. Define $c(u)$ as the set of "colors" of node u. Each host node $u \in N$ maintains a non-empty set $c(u)$, while the color set of an experiment node $v \in N_k$, $c(v)$, may be empty, indicating hardware agnosticism. The placement of an experiment is only valid under the same conditions as previously described, with the addition that the mapped nodes $v \to f(v)$ satisfy

$c(v) \cap c(f(v)) \neq \emptyset \vee c(v) = \emptyset$, i.e. the colors of the host and experiment nodes agree so long as an experiment node has color preferences.

2.1 Markov Decision Process Formulation

We use reinforcement learning to train a model to minimize the time to place all experiments in $\mathcal{X}$. The underlying mathematical model, the Markov Decision Process (MDP), describes the state of the environment at time t, and the set of valid actions. The agent receives at each time step: (1) the number of memories available in each node normalized by μ; (2) the active sublinks in the topology and their age normalized by $m^\star$; (3) the time remaining for sublinks locked by an experiment; and (4) which experiments have been completed. The set of actions available to the agent is either to place an experiment, to generate a virtual link, or to wait (take no action). The validity of placements and virtual link generations is signaled using a binary action mask. After taking a specific action, the environment returns a reward equal to the sum of the relevant reward terms.

Subgraph edit distance-aware reward. Placing an experiment $X_k = (G_k, d_k)$ at time t requires G_k to be subgraph isomorphic to $\tilde{G}_t$. We define a custom notion of *subgraph edit distance* (SED) between G_k and $\tilde{G}_t$ as the minimum number of additional edges needed in G_k for it to be subgraph isomorphic to $\tilde{G}_t$. Assume that the model chooses to generate a virtual link at time t. Let $\Delta\,\mathrm{SED} = \mathrm{SED}_t - \mathrm{SED}_{t-1}$ be negative if the minimum SED between the topology of an unassigned experiment and any of its placements in $\tilde{G}_t$ decreased, where the minimum is calculated over all experiments and all placements. Then the reward received by the agent for virtual link generation, r_{vl}, is defined as

$$r_{\text{vl}} = \begin{cases} -r_{\text{pen}} & \text{if } \Delta\,\mathrm{SED} > 0 \\ r_{\text{base}} - \alpha\,\mathrm{SED}_t - \beta(\Delta\,\mathrm{SED} + 1) & \text{otherwise} \end{cases} \tag{1}$$

where $r_{\text{pen}}, r_{\text{base}} > 0$ and $\alpha, \beta > 1$ are tunable parameters. If the virtual link increases the minimum SED between any unassigned experiment topology and $\tilde{G}_t$, the agent is penalized by r_{pen}, representing an unhelpful generation. If the virtual link decreases the overall minimum SED, then the agent receives a reward composed of a base amount, a linear penalty proportional to the remaining distance (SED_t), and a shaping reward for the magnitude of step progress ($\Delta\,\mathrm{SED}$). We clarify this choice through the following example. Consider two virtual link generations, both of which cause $\Delta\,\mathrm{SED} = -1$. Let Action A reduce the overall SED from $4 \to 3$, while Action B reduces it from $1 \to 0$. Under Action A, placing any experiment requires at least 3 additional virtual links. In contrast, Action B enables the model to place the experiment in the next time step (assuming that the links did not expire). The weighting of r_{vl} by SED_t incentivizes the model to prioritize virtual links that allow a quicker placement of the experiment, since the goal is to minimize the needed time.

Betweenness centrality-aware reward. We introduce an additional reward depending on the maximum betweenness centrality of nodes along the path whose links were consumed to generate a virtual link. The betweenness centrality of a node u measures the number of shortest paths, taken between all node pairs in the graph, that include u along the path. A greater value indicates that a node acts more like a bottleneck. We compute its value based on the active edge set $\tilde{E}_t \cup V_t$ and normalize it against the maximum possible betweenness centrality out of all generable virtual links at that time step. If at time t, $V_t^+ \subseteq V$ is the set of generable virtual links, $l \in V_t^+$ is the virtual link chosen, and $P(l)$ is the shortest path among $\tilde{E}_t \cup V_t$ between the endpoints of l, then

$$r_{\text{bottleneck}} = r_{\text{base}}^{\text{bottleneck}} \times \frac{\max_{p \in P(l)} g(p)}{\max_{v \in V^+} \left\{ \max_{q \in P(v)} g(q) \right\}} \in [0, r_{\text{base}}^{\text{bottleneck}}] \tag{2}$$

is the reward given to the agent, where g computes the betweenness centrality of a node. Here, $r_{\text{base}}^{\text{bottleneck}} > 0$ is a hyperparameter.

Experiment complexity-aware reward for placement When the agent chooses to place an experiment X_k with edge set E_k, it receives a reward $r_{\text{exp}} = r_{\text{base}}^{\text{exp}} \times \frac{|E_k|}{\kappa}$, where $|E_k|$ is the number of edges in G_k, $\kappa > 0$ is a scaling parameter weighting the edge count, and $r_{\text{base}}^{\text{exp}}$ is an overall weighting parameter. Since more complex experiments require more intermediate virtual links, and thus more

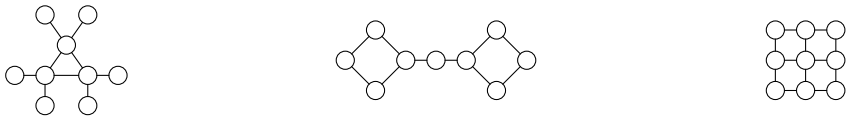

Figure 2: The starlink (left), dumbbell (center), and grid (right) topologies.

time to place, the greater penalty received in placing the experiment should be offset by a greater reward. A static reward would incentivize the agent to greedily place simpler experiments instead of normalizing against the required spatial resources.

Episode length penalty To incentivize the agent to minimize the overall time required to place all experiments in $\mathcal{X}$, we penalize the agent for each time step used with $r_{\text{step}} < 0$.

3 Reinforcement Learning Framework

We use double deep Q-learning to train our model. In this method, an online network (student) interacts with the environment through actions that a target network (teacher) evaluates, with the disagreement between the two models being the objective function to minimize. Higher values of γ increase the difficulty in placing experiments due to fewer simultaneous active links in a given interval of time. Therefore, training solely under high environment noise leads to difficulties in converging to an optimal policy, as any learning signal is subdued by the environment's noise. To improve policy performance at higher noise levels, we employ curriculum training with a replay buffer. Using curriculum training, the model learns a policy in low-noise conditions, which is then modified as it further trains in high-noise conditions. By exposing the model to low-noise environments, it is able to converge on a high-fidelity policy early on, while adjusting to simultaneously increasing stochasticity. We curriculum-train the model in 11 phases, each phase $\mathcal{P}$ corresponding to a specific γ of 11 linearly-interpolated values of the interval $[1.5, 5.8]$. (p_l decreases from ≈ 0.22 to ≈ 0.003 over the course of the curriculum.) If $\mathcal{P} > 1$, the replay buffer is seeded with episode roll-outs from the model from $\mathcal{P} - 1$. Then the online network is trained until a maximum step count is reached or phase mastery is achieved. Phase mastery is achieved after sufficient convergence and the demonstration of a mean success rate of 100% over a window of 100 episodes. Finally, the saved weights are used to both warm-start and seed the next phase $\mathcal{P} + 1$.

We maintain a state transition buffer (containing tuples of the state observed, the action taken, and the reward received) with capacity for 100,000 transitions. The transitions from 500 episodes rolled-out by the online network are pushed to the buffer. These constitute an "expert" section of the buffer and are kept unmodified for the phase. Transitions generated using an ϵ-greedy approach by the online network during its training are pushed to the buffer. During the update phase, a mini-batch of 256 samples is drawn from the buffer with the ratio of 25% expert transitions to 75% online network transitions, which is used to compute the gradient of the loss function and for updating the target network using Polyak updates as described in Appendix A. Since the model learns from graph-structured information, we employ a Message Passing Neural Network (MPNN) for topology-aware decision-making. Details of the MPNN are left for Appendix C.

4 Experimental Setup

4.1 Evaluation Topologies

We train three models to operate in three topologies as shown in Figure 2. The starlink topology shown represents a hub of three independent networks linked together. The dumbbell topology simulates two networks bridged by a bottleneck node. The grid topology is more dense, representing a network that may be implemented in urban settings or in a data center. In all evaluation settings, we set $|N| = 9$, $m^\star = 52$, $\mu = 5$, and $\mathcal{X} = \{(K_4, 1), (K_4, 1)\}$, where K_4 is the complete graph of 4 nodes. We truncate an episode if either experiment remains unplaced after 200 time steps. We consider the episode to be successful if it is not truncated.

4.2 BASELINE HEURISTIC ALGORITHMS

To benchmark the performance of our models, we compare their performance against baseline heuristics. The first baseline algorithm is a greedy approach we call AGECRITICALFIRST, which places a valid experiment whose corresponding host links expire the soonest, followed by generating the virtual link with the least resulting age. The second baseline algorithm is another greedy approach we call SHORTESTHOPFIRST, which places the first experiment in the action index, followed by generating the virtual link whose physical path distances is shortest. Whereas AGECRITICALFIRST operates greedily with respect to time, SHORTESTHOPFIRST operates greedily with respect to space. The third heuristic, DEGREECENTRICTHRESHOLDROUTING (DCTR), places the first experiment in the action index, followed by generating the virtual link whose centrality of its endpoints $u, v \in N$, $S = \deg(u) + \deg(v)$ is greatest.

4.3 COLOR CONSTRAINT

We demonstrate the ability to train a policy given the coloring constraint described in Section 2. Figure 3 shows the specific coloring of the starlink network in our evaluations. Results on the colored dumbbell and grid topologies are left to Appendix D. The experiment durations are left unchanged. With this coloring, the policy must learn to overcome resource contention in the high-degree region of the network. Placing the red experiment requires at least one virtual link that consumes links along a path that includes the green nodes, which themselves must host the green experiment. This setup mimics a real-world scenario where a specific kind of hardware that exists at topologically-distant locations in the network is requested by an experiment at the same time as more centrally-located hardware is requested by another experiment.

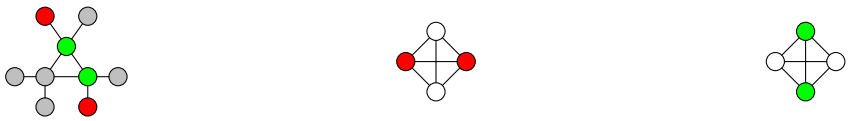

Figure 3: The coloring of the starlink network topology and experiment set. The red nodes of the red experiment must be mapped to the red nodes on the starlink, and similarly for the green nodes on the green experiment. In each experiment, the uncolored nodes can be mapped to any node on the topology, including the colored ones.

5 RESULTS

We are interested in the ability of a policy to operate in varying noise levels driven by γ, primarily because it describes the effect of link activation probability on the ability to place experiments efficiently. In all subsequent plots in this section, 25 values of γ were chosen between the interval $[1.5, 5.8]$ to evaluate 100 independent episodes, from which the success rate and the mean step count are calculated. Appendix E describes the selection of $\mathcal{P}$ for each topology by comparing the performance after each phase of curriculum training.

5.1 MODEL VS BASELINES ON UNCOLORED STARLINK TOPOLOGY

We compare $\mathcal{P} = 9$ of the model to the baseline heuristics using the noise robustness test on the starlink topology. Figure 4 shows that AGECRITICALFIRST, SHORTESTHOPFIRST, and DCTR begin to fail at $\gamma \approx 3.5$ ($p_l \approx 0.030$), whereas the model begins to fail at $\gamma \approx 4.5$ ($p_l \approx 0.011$). At $\gamma = 4.0$ ($p_l \approx 0.018$), the model places all experiments in approximately 95 steps, whereas all heuristics require approximately 170 steps. Additionally, the model first fails at a 59% lower probability than the best performing heuristic AGECRITICALFIRST, and at a 66% lower probability than the worst performing heuristic DCTR, indicating greater capability in high noise. For near-term quantum networks above $\gamma > 3.0$, the model succeeds faster, minimizing network resource consumption. The local strategy of AGECRITICALFIRST and SHORTESTHOPFIRST greedily consumes active sublinks for virtual link generation as soon as they become available, inherently producing a greater proportion of unhelpful virtual links. Across most of the higher γ range, representing near-term quantum

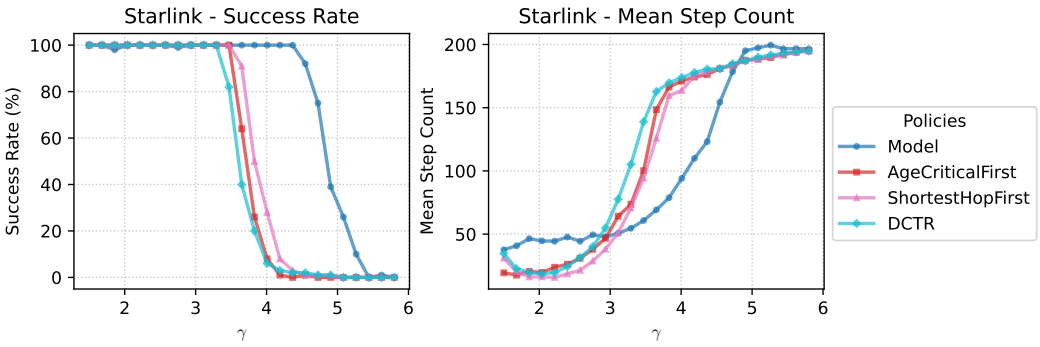

Figure 4: Comparison of success rate and mean step count between the starlink topology for $\mathcal{P} = 9$ of the model, AGECRITICALFIRST, SHORTESTHOPFIRST, and DCTR.

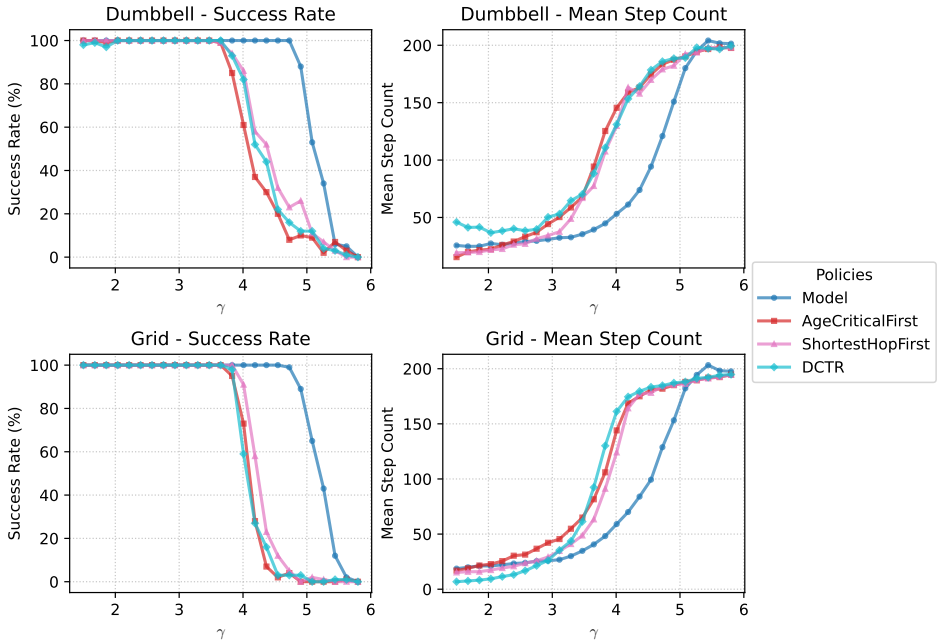

Figure 5: Success rate and mean step count as a function of γ, corresponding to link activation probability, on the dumbbell and grid topologies for $\mathcal{P} = 8$ for both models, AGECRITICALFIRST, SHORTESTHOPFIRST, and DCTR.

network capabilities, the learned policy completes each episode in fewer steps on average and maintains a higher success rate.

5.2 MODEL VS BASELINES ON UNCOLORED DUMBBELL AND GRID TOPOLOGIES

Figure 5 demonstrates the performance of the Deep Q-Network against baseline heuristics on the dumbbell and grid topologies. This highlights generalization across topologies featuring a bottleneck along the dumbbell's bridge node and path redundancy due to the high connectivity of the grid. On the dumbbell topology, the model first fails at 66% lower probability than the best performing heuristic DCTR, and at 71% lower probability than the worst performing heuristic AGECRITICALFIRST. The model also maintains a strictly lower mean step count across the operational regime. At $\gamma = 4.0$, the model places the experiments in approximately 50 steps, whereas the baseline heuristics require between 125 and 150 steps. In the grid topology, the model first fails at a 51% lower probability than the best performing heuristic SHORTESTHOPFIRST, and at a 59% lower probabil-

ity than the worst performing heuristic AGECRITICALFIRST. At $\gamma = 4.0$, the model takes around 55 steps to complete an episode, whereas the policies require the baselines require approximately between 125 and 160 steps.

5.3 MODEL VS BASELINES ON COLORED STARLINK TOPOLOGY

We compare the model to the baseline algorithms on the colored starlink topology in Figure 6. The figure shows the model succeeding frequently at low values of γ, where the heuristics cannot reliably succeed, likely because they exhaust the finite memories of the bottleneck node needed to form the required virtual link, whereas the model carefully generates its virtual links. In particular, the model begins to fail more than 20% of episodes at a 51% lower probability than the best performing heuristic AGECRITICALFIRST, and at a 59% lower probability than the worst performing heuristic DCTR. The model completes its episodes at least 60 steps quicker than the most efficient heuristic for all values of $\gamma < 4.1$.

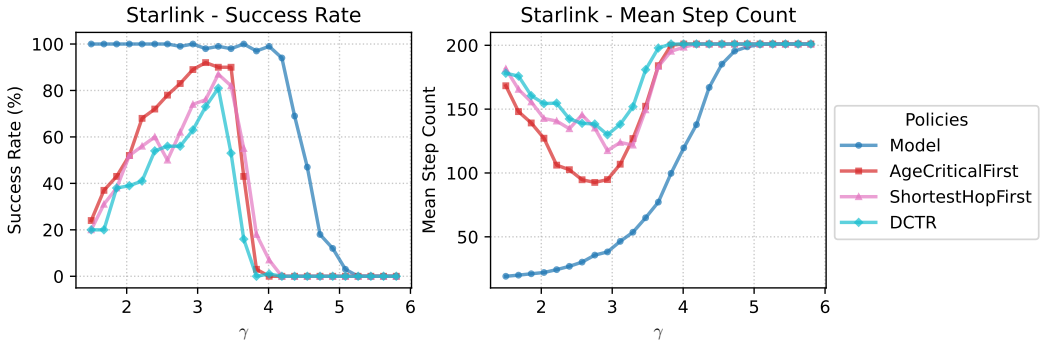

Figure 6: Comparison of success rate and mean step count between $\mathcal{P} = 7$ of the model and baseline heuristics on the colored variant of the problem on the starlink topology.

6 BEHAVIOR PROFILING

6.1 LLM POLICY DISTILLATION

We tasked Gemini 3.1 Pro with deriving a novel policy for the uncolored starlink topology ($N = 9$), given examples of actions taken by the DQN-based policy. Details about the prompt are given in Appendix F. The policy returned by the LLM partitions the nodes into hub and leaf nodes, processes each experiment in order, identifies their optimal placements (prioritizing hub nodes), and places them if possible. If no experiment can be placed, then the virtual links useful for the optimal placements are generated. This process continues until all experiments of $\mathcal{X}$ have been placed. Figure 7 shows the comparison of the LLM-based policy to the DQN-based policy. The DQN-based policy succeeds more frequently than the LLM-based one for γ values less than $\gamma \approx 5.0$. For $\gamma > 3.2$, the DQN-based model completes the episodes in fewer steps than the LLM's heuristic.

6.2 METRICS

We define three metrics to identify empirical differences between policies, enabling the identification of specific reasons why certain policies might succeed more frequently. HOLDINGTIME measures the average time that a sublink is active and unused across all active links during an episode. BRIDGESPAN measures the average length of the shortest physical path between the two endpoints of all generated virtual links. HUBANCHORBIAS measures the average maximum degree of the endpoints of a generated virtual link. Specifically, the average of $\max(\deg(u), \deg(v))$ is taken over the virtual links generated whose endpoints are $u, v \in N$, where the degree is taken with respect to the static physical edge set. Figure 8 compares these metrics on the starlink topology. The model learned a patient policy, as shown by its greater HOLDINGTIME relative to the heuristics. Even when physical links are activated more frequently (low γ), the model avoids greedily consuming

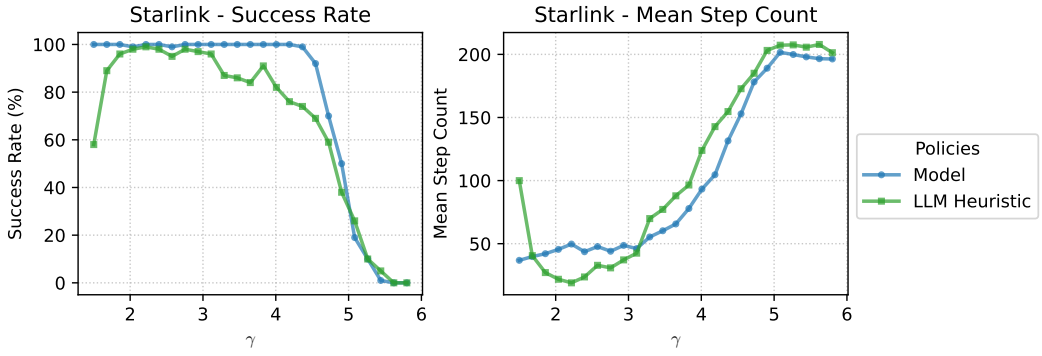

Figure 7: Robustness to changes in link activation probability on the uncolored starlink topology for $\mathcal{P} = 7$ of the model and the LLM-generated heuristic.

virtual links and instead waits for more activated physical links before consuming them. Interestingly, the LLM waits longer than the model after $\gamma \approx 3.5$ while maintaining relatively high success rates. Next, in the operational regime, the model prefers to consistently generate virtual links of lesser span than those generated by the baseline heuristics. This result shows the model has more reliably learned to optimally place a K_4-based experiment. Note that in the starlink topology, the K_4 experiment requires at least two 2-path virtual links to be placed, and any longer links are unnecessary. The LLM clearly identified this requirement in designing its heuristic, as shown by its constant BRIDGESPAN of two. Lastly, the model demonstrates a low HUBANCHORBIAS relative to all other policies, indicating that it prefers to schedule its experiments at peripheral nodes. On the other extreme, the LLM heuristic always generates virtual links with an endpoint in the hub, given its HUBANCHORBIAS of four for all values of γ.

7 CONCLUSION

The work demonstrates an application of reinforcement learning techniques in probabilistic environments to quantum networks. Whereas prior results optimize connectivity-layer scheduling policies, our model optimizes link-layer scheduling tasks under realistic constraints, obtaining a policy that operates reliably even if link activation probabilities are low. Our results show that the use of reinforcement learning to obtain policies can ultimately aid the design and expansion of near-term quantum network topologies and the efficient management of end-user requests. We also give initial demonstrations of providing trajectories performed by a classically-trained model, whose policy itself is difficult to interpret, to a frontier LLM enabling it to create a novel policy that performs well and can described using natural language. Whereas directly measuring certain aspects of a policy's behavior gives specific, targeted insights that can allude to a general policy, our work suggests that LLMs can be a useful tool for extracting a policy that is both easily described and useful in more general cases and in larger networks.

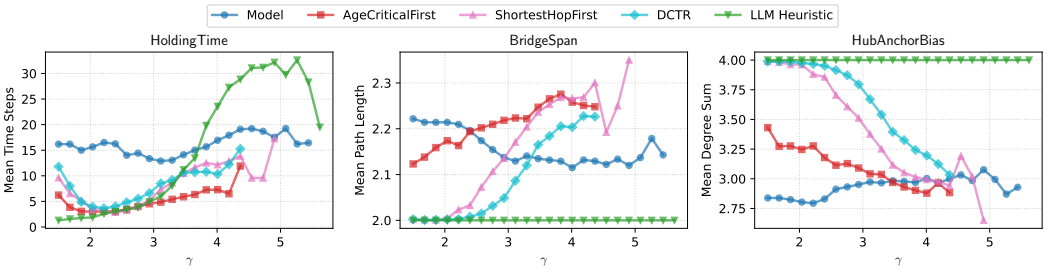

Figure 8: Three metrics HOLDINGTIME, BRIDGESPAN, and HUBANCHORBIAS measured on the uncolored starlink topology over successful episodes for $\mathcal{P} = 7$ of the model, the baseline heuristics, and the LLM heuristic. Absent points are a result of a policy's 0% success rate for that value of γ.

8 AI USE DISCLOSURE

In this work, we used generative AI tools, in particular Gemini 3.1 Pro, to aid in implementing an initial version of the MPNN and the high-level training loop. We additionally conducted experiments with Gemini 3.1 Pro to obtain the results in Section 6.1. We have not used generative AI tools to interpret results or to design or provide feedback on research methodology or experiments, or to support qualitative or thematic data analysis. The uses of AI to generate synthetic data sets, to develop theoretical models or conceptual frameworks, to formulate mathematical claims, to assist with writing proofs, to propose or refine hypotheses, to assist with translation, and to clean or reformat the dataset, were not applicable to this work. Additionally, we used generative AI tools to write software code to generate the plots, and to create an initial version of Figure 1. We manually modified and extended the AI-assisted implementation to suit our problem formulation, simultaneously reviewing and revising the implementation. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

To ensure the reproducibility of our training process, we have included the source code used to train and evaluate our models, and to produce plots, in the Supplemental Material section of our submission. The source code includes the environment and training configurations used to produce the plots shown here. A README.md file is present in the repository with instructions.

REFERENCES

David Awschalom, Karl K. Berggren, Hannes Bernien, Sunil Bhave, Lincoln D. Carr, Paul Davids, Sophia E. Economou, Dirk Englund, Andrei Faraon, Martin Fejer, Saikat Guha, Martin V. Gustafsson, Evelyn Hu, Liang Jiang, Jungsang Kim, Boris Korzh, Prem Kumar, Paul G. Kwiat, Marko Lončar, Mikhail D. Lukin, David A.B. Miller, Christopher Monroe, Sae Woo Nam, Prineha Narang, Jason S. Orcutt, Michael G. Raymer, Amir H. Safavi-Naeini, Maria Spiropulu, Kartik Srinivasan, Shuo Sun, Jelena Vučković, Edo Waks, Ronald Walsworth, Andrew M. Weiner, and Zheshen Zhang. Development of Quantum Interconnects (QuICs) for Next-Generation Information Technologies. *PRX Quantum*, 2:017002, February 2021. doi: 10.1103/PRXQuantum.2.017002. URL https://doi.org/10.1103/PRXQuantum.2.017002.

Koji Azuma, Sophia E. Economou, David Elkouss, Paul Hilaire, Liang Jiang, Hoi-Kwong Lo, and Ilan Tzitrin. Quantum repeaters: From quantum networks to the quantum internet. *Reviews of Modern Physics*, 95:045006, December 2023. doi: 10.1103/RevModPhys.95.045006. URL https://doi.org/10.1103/RevModPhys.95.045006.

Joaquin Chung, Gregory Kanter, Nikolai Lauk, Raju Valivarthi, Wenji Wu, Russell R. Ceballos, Cristián Peña, Neil Sinclair, Jordan Thomas, Si Xie, Rajkumar Kettimuthu, Prem Kumar, Panagiotis Spentzouris, and Maria Spiropulu. Illinois Express Quantum Network (IEQNET): Metropolitan-scale experimental quantum networking over deployed optical fiber. *arXiv:2104.04629*, 2021. URL https://arxiv.org/abs/2104.04629.

Joaquin Chung, Ely M. Eastman, Gregory S. Kanter, Keshav Kapoor, Nikolai Lauk, Cristián H. Peña, Robert K. Plunkett, Neil Sinclair, Jordan M. Thomas, Raju Valivarthi, Si Xie, Rajkumar Kettimuthu, Prem Kumar, Panagiotis Spentzouris, and Maria Spiropulu. Design and Implementation of the Illinois Express Quantum Metropolitan Area Network. *IEEE Transactions on Quantum Engineering*, 3:1–20, 2022. doi: 10.1109/TQE.2022.3221029. URL https://doi.org/10.1109/TQE.2022.3221029.

Alexander N. Craddock, Anne Lazenby, Gabriel Bello Portmann, Rourke Sekelsky, Mael Flament, and Mehdi Namazi. Automated Distribution of Polarization-Entangled Photons Using Deployed New York City Fibers. *PRX Quantum*, 5:030330, Aug 2024. doi: 10.1103/PRXQuantum.5.030330. URL https://link.aps.org/doi/10.1103/PRXQuantum.5.030330.

Xiaojie Fan, Caitao Zhan, Himanshu Gupta, and C. R. Ramakrishnan. Optimized Distribution of Entanglement Graph States in Quantum Networks. *IEEE Transactions on Quantum Engineering*, 6:1–17, 2025. doi: 10.1109/TQE.2025.3552006. URL https://doi.org/10.1109/TQE.2025.3552006.

Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. Neural message passing for quantum chemistry. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, pp. 1263–1272. JMLR.org, 2017.

Stav Haldar, Pratik J. Barge, Sumeet Khatri, and Hwang Lee. Fast and reliable entanglement distribution with quantum repeaters: Principles for improving protocols using reinforcement learning. *Physical Review Applied*, 21(2), February 2024. ISSN 2331-7019. doi: 10.1103/physrevapplied.21.024041. URL http://dx.doi.org/10.1103/PhysRevApplied.21.024041.

Julian Hofmann, Michael Krug, Norbert Ortegel, Lea Gérard, Markus Weber, Wenjamin Rosenfeld, and Harald Weinfurter. Heralded Entanglement Between Widely Separated Atoms. *Science*, 337(6090):72–75, 2012. ISSN 0036-8075. doi: 10.1126/science.1221856. URL https://science.sciencemag.org/content/337/6090/72.

Álvaro G. Iñesta, Gayane Vardoyan, Lara Scavuzzo, and Stephanie Wehner. Optimal entanglement distribution policies in homogeneous repeater chains with cutoffs. *npj Quantum Information*, 9:46, 2023. URL https://doi.org/10.1038/s41534-023-00713-9.

M. Żukowski, A. Zeilinger, M. A. Horne, and A. K. Ekert. 'Event-ready-detectors' Bell experiment via entanglement swapping. *Physical Review Letters*, 71:4287–4290, December 1993. doi: 10.1103/PhysRevLett.71.4287. URL https://link.aps.org/doi/10.1103/PhysRevLett.71.4287.

Sumeet Khatri. On the design and analysis of near-term quantum network protocols using Markov decision processes. *AVS Quantum Science*, 4(3):030501, 09 2022. ISSN 2639-0213. doi: 10.1116/5.0084653. URL https://doi.org/10.1116/5.0084653.

Can M. Knaut, Aziza Suleymanzade, Yan-Cheng Wei, Daniel R. Assumpcao, Pieter-Jan Stas, Yan Qi Huan, Bartholomeus Machielse, Erik N. Knall, Madison Sutula, Gefen Baranes, Neil Sinclair, Chawina De-Eknamkul, David S. Levonian, Mihir K. Bhaskar, Hongkun Park, Marko Lončar, and Mikhail D. Lukin. Entanglement of Nanophotonic Quantum Memory Nodes in a Telecom Network. *Nature*, 629:573–578, 2024. URL https://doi.org/10.1038/s41586-024-07252-z.

Linh Le and Tu N. Nguyen. DQRA: Deep Quantum Routing Agent for Entanglement Routing in Quantum Networks. *IEEE Transactions on Quantum Engineering*, 3:1–12, 2022. doi: 10.1109/TQE.2022.3148667.

Jan Li, Tim Coopmans, Patrick Emonts, Kenneth Goodenough, Jordi Tura, and Evert van Nieuwenburg. Optimising entanglement distribution policies under classical communication constraints assisted by reinforcement learning. *Machine Learning: Science and Technology*, 6(3):035024, aug 2025. doi: 10.1088/2632-2153/adeefa. URL https://doi.org/10.1088/2632-2153/adeefa.

D. Main, P. Drmota, D. P. Nadlinger, E. M. Ainley, A. Agrawal, B. C. Nichol, R. Srinivas, G. Araneda, and D. M. Lucas. Distributed quantum computing across an optical network link. *Nature*, 638(8050):383–388, 2025. doi: 10.1038/s41586-024-08404-x. URL https://doi.org/10.1038/s41586-024-08404-x.

Gongyu Ni, Davide Ferrari, Lester Ho, and Michele Amoretti. Advanced scheduling strategies for distributed quantum computing jobs. *Computer Networks*, 288:112622, 2026. ISSN 1389-1286. URL https://doi.org/10.1016/j.comnet.2026.112622.

A Pirker and W Dür. A quantum network stack and protocols for reliable entanglement-based networks. *New Journal of Physics*, 21(3):033003, mar 2019. doi: 10.1088/1367-2630/ab05f7. URL https://doi.org/10.1088/1367-2630/ab05f7.

M. Pompili, C. Delle Donne, I. te Raa, B. van der Vecht, M. Skrzypczyk, G. Ferreira, L. de Kluijver, A. J. Stolk, S. L. N. Hermans, P. Pawełczak, W. Kozlowski, R. Hanson, and S. Wehner. Experimental demonstration of entanglement delivery using a quantum network stack. *npj Quantum Information*, 8:121, 2022. URL https://doi.org/10.1038/s41534-022-00631-2.

Matteo Pompili, Sophie L. N. Hermans, Simon Baier, Hans K. C. Beukers, Peter C. Humphreys, Raymond N. Schouten, Raymond F. L. Vermeulen, Marijn J. Tiggelman, Laura dos Santos Martins, Bas Dirkse, Stephanie Wehner, and Ronald Hanson. Realization of a multi-node quantum network of remote solid-state qubits. *Science*, 372:259–264, 2021. URL https://doi.org/10.1126/science.abg1919.

Simon D. Reiß and Peter van Loock. Deep reinforcement learning for key distribution based on quantum repeaters. *Physical Review A*, 108:012406, Jul 2023. doi: 10.1103/PhysRevA.108.012406. URL https://doi.org/10.1103/PhysRevA.108.012406.

M. Sasaki, M. Fujiwara, H. Ishizuka, W. Klaus, K. Wakui, M. Takeoka, S. Miki, T. Yamashita, Z. Wang, A. Tanaka, K. Yoshino, Y. Nambu, S. Takahashi, A. Tajima, A. Tomita, T. Domeki, T. Hasegawa, Y. Sakai, H. Kobayashi, T. Asai, K. Shimizu, T. Tokura, T. Tsurumaru, M. Matsui, T. Honjo, K. Tamaki, H. Takesue, Y. Tokura, J. F. Dynes, A. R. Dixon, A. W. Sharpe, Z. L. Yuan, A. J. Shields, S. Uchikoga, M. Legré, S. Robyr, P. Trinkler, L. Monat, J.-B. Page, G. Ribordy, A. Poppe, A. Allacher, O. Maurhart, T. Länger, M. Peev, and A. Zeilinger. Field test of quantum key distribution in the Tokyo QKD Network. *Opt. Express*, 19(11):10387–10409, May 2011. doi: 10.1364/OE.19.010387. URL https://opg.optica.org/oe/abstract.cfm?URI=oe-19-11-10387.

YeongJun Seok, Ihsan Ullah, Youn-Hee Han, Chankyun Lee, and Wonhyuk Lee. End-to-end entanglement request scheduling in quantum networks via topology-aware decision transformer. *J. Opt. Commun. Netw.*, 18(1):1–15, Jan 2026. doi: 10.1364/JOCN.569435. URL https://doi.org/10.1364/JOCN.569435.

Stephanie Wehner, David Elkouss, and Ronald Hanson. Quantum internet: A vision for the road ahead. *Science*, 362(6412), 2018. ISSN 0036-8075. URL https://doi.org/10.1126/science.aam9288.

Shi-Hai Wei, Bo Jing, Xue-Ying Zhang, Jin-Yu Liao, Chen-Zhi Yuan, Bo-Yu Fan, Chen Lyu, Dian-Li Zhou, You Wang, Guang-Wei Deng, Hai-Zhi Song, Daniel Oblak, Guang-Can Guo, and Qiang Zhou. Towards real-world quantum networks: A review. *Laser & Photonics Reviews*, 16(3):2100219, 2022. doi: https://doi.org/10.1002/lpor.202100219. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/lpor.202100219.

Chen Zhao, Casey Duckering, Andi Gu, Nishad Maskara, and Hengyun Zhou. Towards ultra-high-rate quantum error correction with reconfigurable atom arrays, 2026. URL https://arxiv.org/abs/2604.16209.

9 APPENDIX

A DOUBLE DEEP Q-LEARNING

Each training time step, the model either chooses a random action with probability ϵ, or selects the action recommended by the online network. We use a dynamic epsilon-decay strategy, in which ϵ decays from ϵ_0 to ϵ_n over an n-length schedule given by

$$\epsilon_t = \epsilon_0 + \min\left(1, \frac{t}{n}\right)(\epsilon_n - \epsilon_0).$$

The environment processes the action and the action, along with the reward received, is pushed to the replay buffer previously described. At a certain frequency, the online and target network are updated. A minibatch of 256 transitions is sampled from the replay buffer. Given a state transition (s, a, r, s'), the online network parameterized by $\boldsymbol{\theta}$ selects the best action a^* for each next state s' by taking the $\arg\max$ over all Q-values computed over $\mathcal{A}$:

$$a^* = \arg\max_{a'} Q(s', a'; \boldsymbol{\theta})$$

Then, the target network parameterized by $\boldsymbol{\theta}^-$ computes its Q-value for a^* by $Q(s', a^*; \boldsymbol{\theta}^-)$ and the temporal difference is computed to give a ground-truth indicator of the value of selecting action a in state s, given the true reward, r received by taking action a.

$$Y_t^{\text{DoubleQ}} = r + \gamma_{\text{RL}} \cdot Q\left(s', a^*; \boldsymbol{\theta}^-\right)(1 - \text{done})$$

where done $\in \{0, 1\}$ is the terminal indicator, and γ_{RL} is the discount factor. Then the online network computes $Q(s, a; \boldsymbol{\theta})$, its *current* estimate for the value of that action, and $\boldsymbol{\theta}$ is updated by minimizing the Mean Squared Error loss against Y_t^{DoubleQ} via Adam. Simultaneously, the target network parameters $\boldsymbol{\theta}^-$ undergo Polyak soft-updates at every training step according to:

$$\boldsymbol{\theta}^- \leftarrow \tau\boldsymbol{\theta} + (1 - \tau)\boldsymbol{\theta}^-$$

where $\tau \ll 1$ ensures that target Q-values transition smoothly without introducing moving-target instabilities.

B OBSERVATION SPACE

The physical network state is encoded into a graph-structured observation dictionary to interface with the MPNN in Appendix C. All scalar features are normalized to $[0, 1]$ to stabilize gradient propagation. The observation tensor at time t consists of:

- Node Features ($\mathbf{h}_v^t \in \mathbb{R}^1$): The number of available memories of node v, normalized by the maximum number of memories in the network.
- Edge Features ($\mathbf{e}_{vw} \in \mathbb{R}^2$): The first dimension is the sublink's age scaled by the decoherence limit ($A_t(l)/m^\star$). The second dimension is the remaining execution time in the case that the link is locked by a placed experiment, normalized by $m^\star$.
- Global Status ($\mathbf{g} \in \{0, 1\}^{|\mathcal{X}|}$): A binary indicator vector where $g_k = 1$ if experiment X_k has not yet been placed, and 0 if placed.

C MESSAGE PASSING NEURAL NETWORK

Since the model learns from graph-structured information, we employ a Message Passing Neural Network (MPNN) defined in Gilmer et al. (2017) for topology-aware decision-making. We encourage the reader to reference the work for conceptual details of MPNNs. The work defines a message function M_t and a vertex update function U_t used during the message-passing phase of the forward pass. Following their notation, our message function $M_t(\mathbf{h}_v^t, \mathbf{h}_w^t, \mathbf{e}_{vw})$ is parameterized by a 2-layer MLP with ReLU activations:

$$M_t(\mathbf{h}_v^t, \mathbf{h}_w^t, \mathbf{e}_{vw}) = \text{MLP}_m(\mathbf{h}_v^t, \mathbf{h}_w^t, \mathbf{e}_{vw})$$

where

$$\text{MLP}_m(\mathbf{h}_v^t, \mathbf{h}_w^t, \mathbf{e}_{vw}) = \mathbf{W}_2^{(m)}\text{ReLU}\Big(\mathbf{W}_1^{(m)}[\mathbf{h}_w^t \| \mathbf{h}_v^t \| \mathbf{e}_{vw}] + \mathbf{b}_1^{(m)}\Big) + \mathbf{b}_2^{(m)}$$

Here, $\|$ denotes vector concatenation, $\mathbf{W}_1^{(m)} \in \mathbb{R}^{d\times(2d+d_e)}$ and $\mathbf{W}_2^{(m)} \in \mathbb{R}^{d\times d}$ are the weight matrices specific to the message-passing phase, and $\mathbf{b}_1^{(m)} \in \mathbb{R}^d$ and $\mathbf{b}_2^{(m)} \in \mathbb{R}^d$ are bias vectors. The term d denotes the hidden representation dimension, and d_e is the dimension of the edge features. In our setup, $d = 128$ and $d_e = 2$. The messages directed at a given node v are sum-pooled over its neighborhood $\mathcal{N}(v)$:

$$\mathbf{m}_v^{(t+1)} = \sum_{w\in\mathcal{N}(v)} M_t(\mathbf{h}_v^t, \mathbf{h}_w^t, \mathbf{e}_{vw})$$

Whereas M_t is responsible for defining the communication transferred between adjacent nodes, the update function U_t is responsible for updating each node's state as a consequence of the aggregated messages. U_t acts on both the node's original hidden state $\mathbf{h}_v^t$ and the accumulated message $\mathbf{m}_v^{t+1}$ via a secondary MLP:

$$\mathbf{h}_v^{t+1} = U_t(\mathbf{h}_v^t, \mathbf{m}_v^{(t+1)}) = \text{MLP}_u([\mathbf{h}_v^t \| \mathbf{m}_v^{(t+1)}])$$

In our configuration, message-passing and updating occurs for $T = 3$ layers.

The final phase of the forward pass involves a readout function R, which computes a global graph embedding to serve as the input to the discrete action head. The readout function produces the vector $\hat{\mathbf{y}}_t$ containing the Q-values for each action. We utilize a permutation-invariant mean-pooling operation over the final node states, which is then concatenated with a global status vector $\mathbf{g}$: $\hat{\mathbf{y}}_t = R(\{\mathbf{h}_v^{(T)} \mid v \in N\}, \mathbf{g}) = \text{MLP}_q(\mathbf{x}_{\text{readout}})$ where the fused readout representation is defined as $\mathbf{x}_{\text{readout}} = \Big[\frac{1}{|N|}\sum_{v\in N}\mathbf{h}_v^{(T)} \,\|\, \mathbf{g}\Big]$. The Q-value projection head evaluates this representation using

$$\text{MLP}_q(\mathbf{x}_{\text{readout}}) = \mathbf{W}_2^{(q)}\text{ReLU}\Big(\mathbf{W}_1^{(q)}\mathbf{x}_{\text{readout}} + \mathbf{b}_1^{(q)}\Big) + \mathbf{b}_2^{(q)}$$

where $\mathbf{W}_1^{(q)} \in \mathbb{R}^{d\times(d+|\mathcal{X}|)}$, $\mathbf{W}_2^{(q)} \in \mathbb{R}^{|\mathcal{A}|\times d}$, $\mathbf{b}_1^{(q)} \in \mathbb{R}^d$, and $\mathbf{b}_2^{(q)} \in \mathbb{R}^{|\mathcal{A}|}$. Here, $|\mathcal{A}|$ is the total size of the discrete action space, and $|\mathcal{X}|$ is the number of scheduled experiments tracked by $\mathbf{g}$. Before decision-making, $\hat{\mathbf{y}}_t$ is action-masked to heavily penalize illegal actions. Illegal actions have their Q-values modified to -10^9 in the action head before action selection.

D Performance on Colored Dumbbell and Grid Topologies

We test the training procedure on colored variants of the dumbbell and grid topologies, whose colorings are shown below:



The experiment colorings are the same as in the colored starlink case. Here, to place the red and green K_4 experiments, the agent must generate virtual links between the red and green nodes at opposite corners. Thus the required physical links along the intermediate paths are contested, and the agent needs to learn when to optimally generate virtual links accordingly. Figure 10 shows the performance on the colored variant of the dumbbell and grid topologies.

In the dumbbell case, there are limited routes to generate the virtual links needed to support a K_4 topology. The model performs similarly to the AgeCriticalFirst heuristic, which most directly relieves the $m^\star$ constraint. In the region where the model outperforms AgeCriticalFirst, we see the models' greater episode lengths. The model performs with at least a 20% success rate for 16% lower link activation probability than the best-performing heuristic AgeCriticalFirst, and

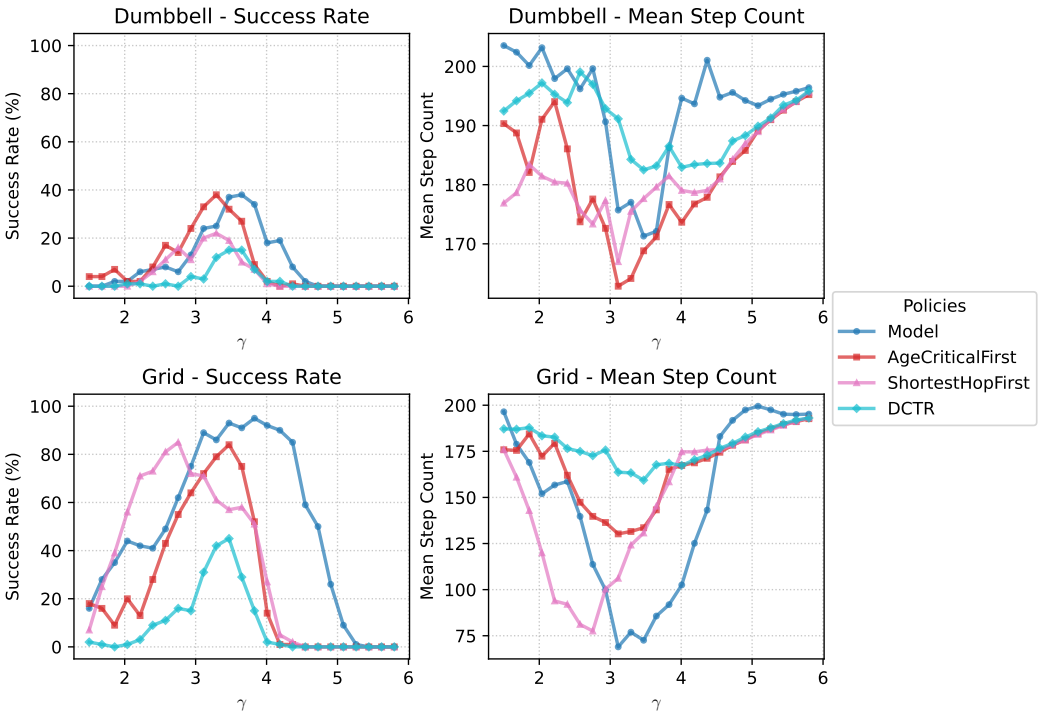

Figure 10: Comparison of the robustness to changes in link activation probability on colored dumbbell and grid topologies, with $\mathcal{P} = 6$ and $\mathcal{P} = 7$ of the model, respectively.

a 90% lower link activation probability than the worst-performing heuristic, DCTR, which does not reach above a 20% success rate for any γ.

In the grid case, the model maintains above an 80% success rate for 59% lower link activation probabilities than the best-performing heuristic AGECRITICALFIRST, and 94% lower link activation probabilities than the worst-performing heuristic DCTR, which does not reach above 80% success.

E CURRICULUM PHASE EVOLUTION

E.1 UNCOLORED VARIANT

Figure 11, shows a comparison of robustness to noise between all 11 phases of the curriculum training on the uncolored variant on the starlink, dumbbell, and grid topologies. Recall that phase $\mathcal{P} > 1$ is used to seed the experience replay buffer for phase $\mathcal{P} + 1$, i.e. phase $\mathcal{P}$ bootstraps phase $\mathcal{P} + 1$. The figures clarify whether the model retains its ability to succeed in low-noise settings even when subjected to high-noise in later phases. Whereas phases $\mathcal{P} \leq 9$ succeed perfectly in low γ settings in the starlink topology, later phases begin to deteriorate in performance, indicating that experiences with high noise overrides the policy learned in earlier phases.

E.2 COLORED VARIANT

Figures12 shows a comparison of the robustness to noise between each of 11 phases of curriculum training on the colored variant on each topology. Contrary to the uncolored case, early phases were more robust than later phases.

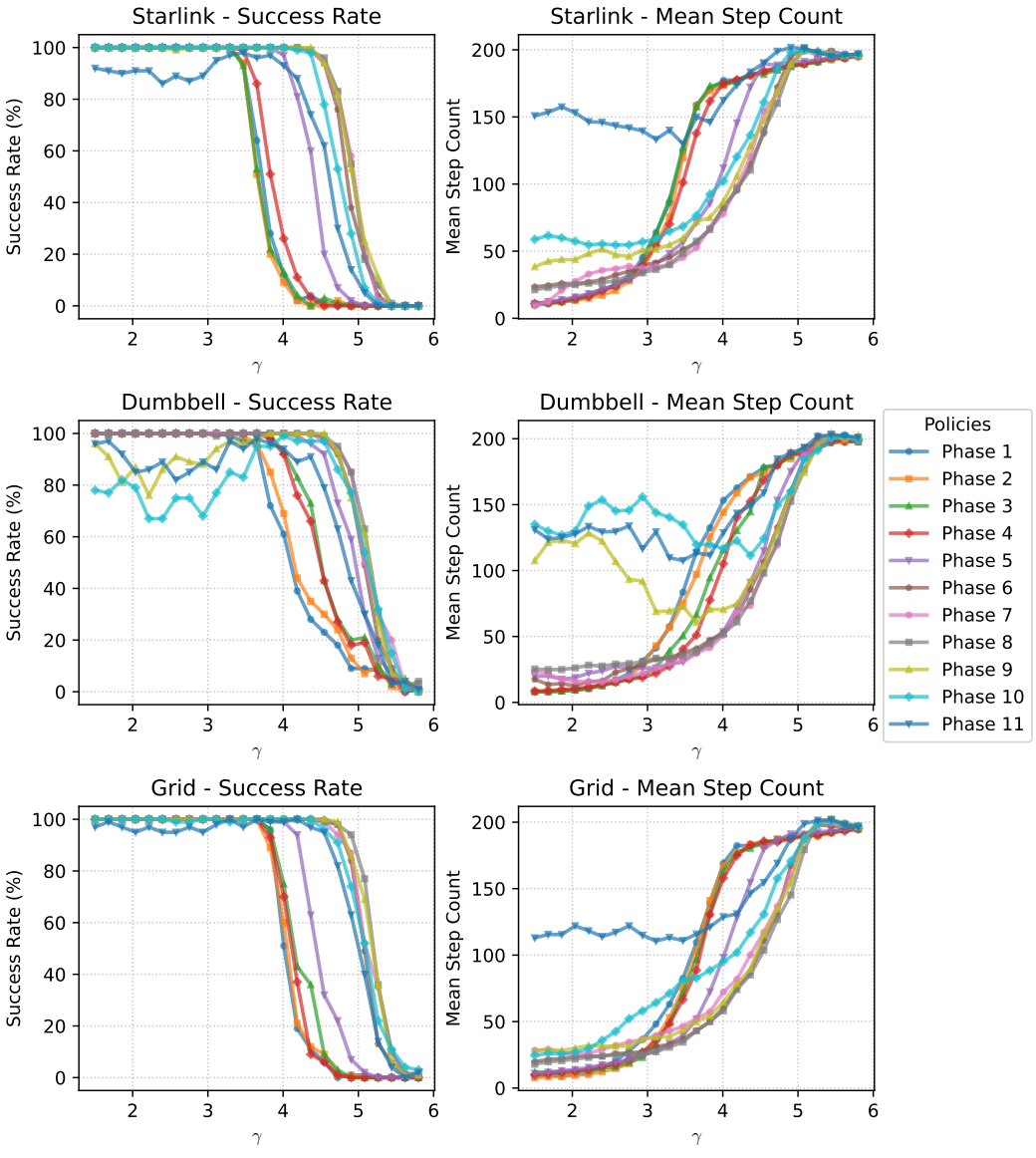

Figure 11: Comparison of the robustness to changes in link activation probability for each phase of the curriculum training process for each uncolored topology. We find that ending the curriculum training early at $\mathcal{P} = 9$, $\mathcal{P} = 8$, and $\mathcal{P} = 8$ for the starlink, dumbbell, and grid topologies respectively provides the most robustness to changes in noise. The training experience in higher noise environments appear to degrade the policy's ability to perform in low-noise settings.

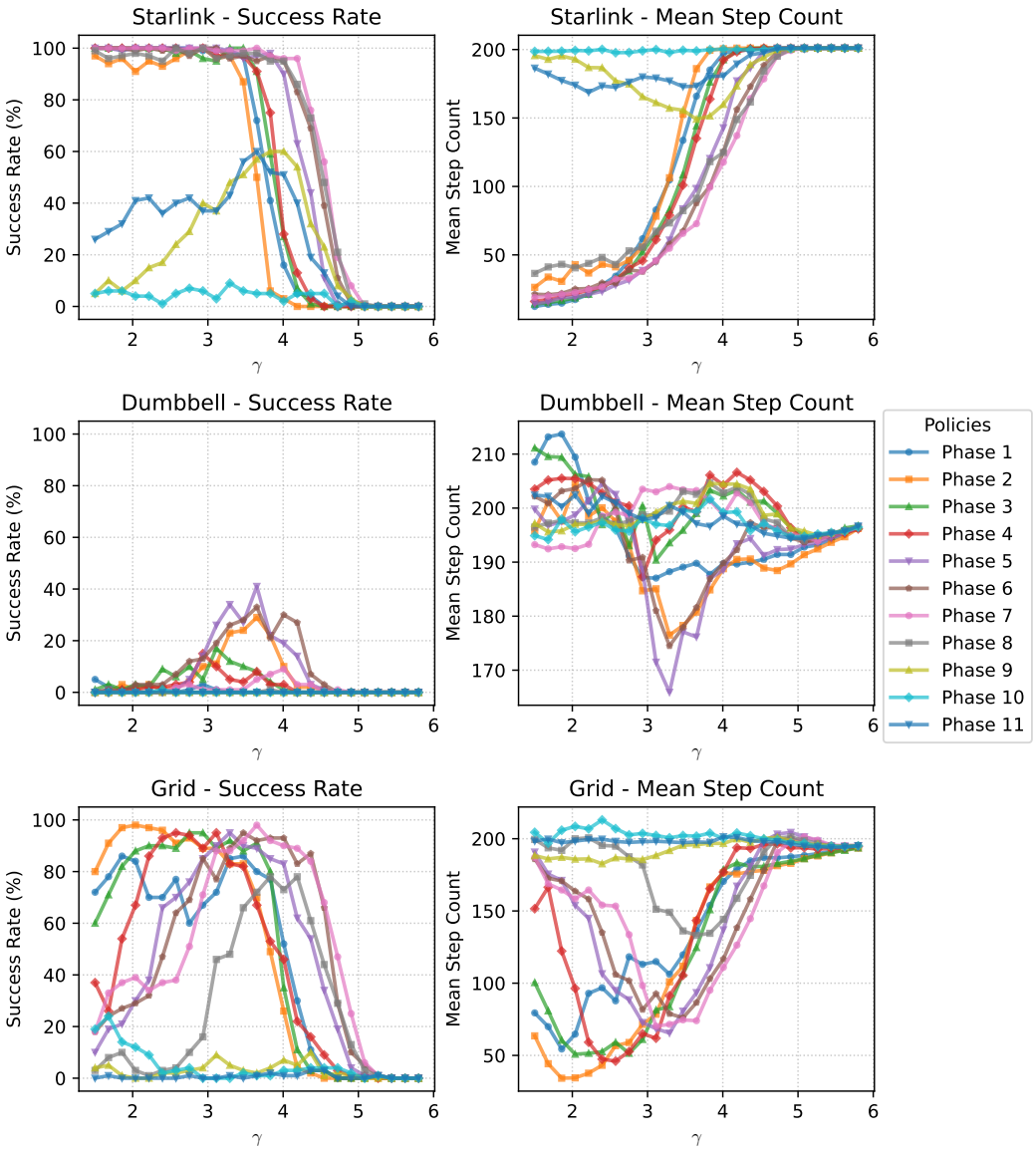

Figure 12: Comparison of the robustness to changes in link activation probability for each phase of the curriculum training process for the colored problem variant on each topology. We find that ending the curriculum training early at $\mathcal{P} = 7$, $\mathcal{P} = 6$, and $\mathcal{P} = 7$ respectively provided the greatest robustness to changes in noise on the starlink, dumbbell, and grid topologies.

F LLM PROMPT

To generate the LLM heuristic for the uncolored starlink environment, we supplied a version of Gemini 3.1 Pro available in September 2026 the following prompt:

```
You are an expert policy extractor whose task is to analyze the given
    execution traces of an agent acting on a starlink topology with N=9
    nodes attempting to minimize the steps needed to assign a set of
    experiments. Derive a general heuristic that will outperform the
    model on a starlink constructed using 3 hub nodes but with any number
     of leaf nodes evenly distributed among the hub nodes.

The adjacency matrix for the host network of the execution traces is:
[[0, 1, 1, 0, 0, 1, 0, 0, 1],
[1, 0, 1, 0, 1, 0, 0, 1, 0],
[1, 1, 0, 1, 0, 0, 1, 0, 0],
[0, 0, 1, 0, 0, 0, 0, 0, 0],
[0, 1, 0, 0, 0, 0, 0, 0, 0],
[1, 0, 0, 0, 0, 0, 0, 0, 0],
[0, 0, 1, 0, 0, 0, 0, 0, 0],
[0, 1, 0, 0, 0, 0, 0, 0, 0],
[1, 0, 0, 0, 0, 0, 0, 0, 0]]

The set of experiments to be assigned formatted as (adjacency matrix,
    duration) was:
{([[0, 1, 1, 1], [1, 0, 1, 1], [1, 1, 0, 1], [1, 1, 1, 0]], 1), ([[0, 1,
    1, 1], [1, 0, 1, 1], [1, 1, 0, 1], [1, 1, 1, 0]], 1)}

The maximum age of any link was m* = 52

The probability of physical link activation was p_link = 0.0111

5 parallel physical links exist between nodes, each of which is activated
     at each time step with probability p_link.

Virtual links are deterministically generated by consuming physical links
     along the shortest paths between nodes, and exist for m* before
    deactivating.

Output your heuristic as pseudocode.
```

The following is an excerpt from the execution traces provided to the LLM, specifically the first episode. The LLM can infer the time between virtual link generations, correspond their endpoint indices to the host network based on the adjacency matrix provided, and view the consequence of each action in terms of reward and episode result.

```
    === EPISODE 1/15 | GAMMA: 4.5 | SEED: 42 ===
  Step 000 | Action: WAIT                          | Reward: -15.000
  Step 001 | Action: WAIT                          | Reward: -15.000
  ...
  Step 008 | Action: WAIT                          | Reward: -15.000
  Step 009 | Action: WAIT                          | Reward: -15.000
  Step 010 | Action: WAIT                          | Reward: -15.000
  Step 011 | Action: GENERATE_VL(0, 6)             | Reward: -15.000
  Step 012 | Action: WAIT                          | Reward: -15.000
  ...
  Step 021 | Action: WAIT                          | Reward: -15.000
  Step 022 | Action: WAIT                          | Reward: -15.000
  Step 023 | Action: WAIT                          | Reward: -15.000
  Step 024 | Action: WAIT                          | Reward: -15.000
  Step 025 | Action: GENERATE_VL(1, 8)             | Reward: -20.000
  Step 026 | Action: WAIT                          | Reward: -15.000
  Step 027 | Action: WAIT                          | Reward: -15.000
  Step 028 | Action: WAIT                          | Reward: -15.000
  Step 029 | Action: WAIT                          | Reward: -15.000
```

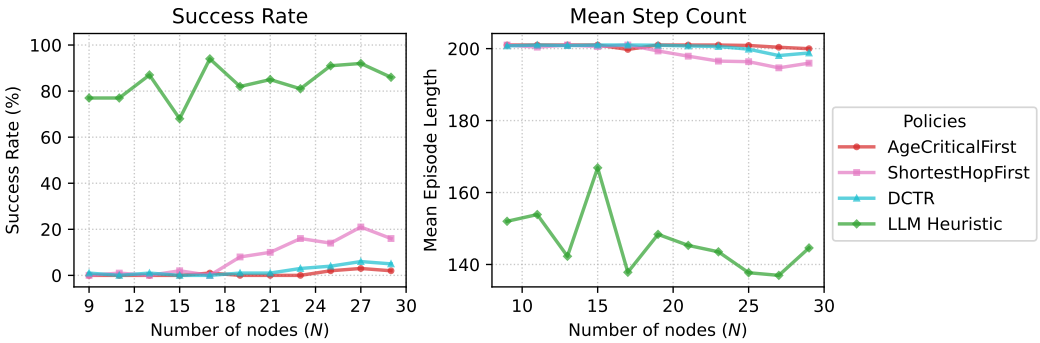

Figure 13: Comparison of the success rate and mean episode length for each heuristic as a function of topology size on the uncolored starlink topology. Each episode was run at constant $\gamma = 4.5$ ($p_l \approx 0.011$)

```
Step 030 | Action: WAIT                                | Reward: -15.000
...

Step 109 | Action: WAIT                                | Reward: -15.000
Step 110 | Action: WAIT                                | Reward: -15.000
Step 111 | Action: WAIT                                | Reward: -15.000
Step 112 | Action: PLACE_EXPERIMENT(idx=0, offset=11) | Reward:
    +385.000
Step 113 | Action: WAIT                                | Reward: -15.000
...
Step 193 | Action: WAIT                                | Reward: -15.000
Step 194 | Action: WAIT                                | Reward: -15.000
Step 195 | Action: PLACE_EXPERIMENT(idx=1, offset=32) | Reward:
    +385.000
Result: TERMINATED (Success) | Total Reward: -2274.000 | Total Steps:
    196
```

We also compared the success rates of the LLM heuristic against the baseline heuristic for increasing sizes of the uncolored starlink topology in Figure 13. In these tests, the number of central hub nodes remained fixed at three, so that the additional nodes were evenly distributed on the peripheral nodes. We note that the LLM generated heuristic performs well independently of network size, indicating that once a heuristic is obtained from the LLM, it can be reused as the network expands to include more nodes.