Entanglement Routing For Quantum Networks: A Deep Reinforcement Learning Approach

Linh Le*, Tu N. Nguyen*, Ahyoung Lee*, and Braulio Dumba†
*College of Computing and Software Engineering, Kennesaw State University, Marietta, GA 30060, USA.
†IBM T. J. Watson Research Center, Yorktown Heights, NY 10598 USA.
Email: *{linh.le, tu.nguyen, and ahyoung.lee}@kennesaw.edu, and †Braulio.Dumba@ibm.com.

***Abstract*—Quantum communications are gaining momentum in finding applications in a wide range of domains, especially those require high-security data transmissions. On the other hand, machine learning has achieved numerous breakthrough successes in various application domains including networking. However, currently, machine learning is not as much utilized in quantum networking as in other areas. With such motivation, we propose a machine-learning-powered entanglement routing scheme for quantum networks that aims to accommodate maximum numbers of demands (source-destination pairs) within a time window. More specifically, we present a deep reinforcement routing scheme that is called Deep Quantum Routing Agent (DQRA). In short, DQRA utilizes an empirically designed deep neural network that observes the current network states to schedule the network's demands which are then routed by a qubit-preserved shortest path algorithm. DQRA is trained towards the goal of maximizing the number of resolved requests in each routing window by using an explicitly designed reward function. Our experiment study shows that, on average, DQRA is able to maintain a rate of successfully routed requests above 80% in a qubit-limited grid network, and about 60% in extreme conditions i.e. each node can act as a repeater exactly once within a window. Furthermore, we show that the complexity and the computational time of DQRA are polynomial in terms of the sizes of the quantum networks.**

***Index Terms*—quantum network routing, deep reinforcement learning, quantum networks, deep learning.**

I. Introduction

There are high demands of network resources and security in today's network and the next-generation network systems since more devices connected to the Internet and new services created. Quantum network appears as a promising technology to enhance exchanged information security via the Internet [1], [2], [3]. A quantum network is built on the top of the conventional networks (e.g., network slicing) that is composed by various nodes (computers) equipped with quantum processors to process and deliver information in the form of quantum bits, called *qubits* [4], [5], [6].

Quantum networks are not designed to replaced the conventional network communication. In fact, they supplement the operation of the next-generation network system where quantum entanglement and swapping play the key role of quantum network technology. In particular, quantum entanglement is designed with the no-cloning theorem, in which it is impossible to produce independent and identical copies of any unknown quantum state. This addresses the fundamental problem of network security: *key distribution* [7], [8]. Specifically, quantum entanglement is set up based on a strong correlation between two particles (i.e., photons). Hence, quantum entanglement can enable the secured data transmission, called *teleportation*, as shown in Fig. 1. A routing path in the quantum network is therefore built based on quantum entanglement as well with the support of quantum repeater using swapping protocol to enable entanglement to be distributed over long distances [9]. We will present details of quantum repeater and swapping in §II-A. Since quantum entanglement is the nonlocal property of two qubits that are inextricably linked to each other, the question of how to construct routing paths for a (or multiple) given pair(s) of source and destination becomes how to assign qubit entanglements to appropriate routing paths. In order to design scalable quantum networks, existing works aim at proposing new optimization models to efficiently assign qubits and repeaters to different quantum entanglements. Authors in [10] discuss how to use a limited number of repeater to enable quantum communication [11] and propose multi-path routing in a diamond topology.

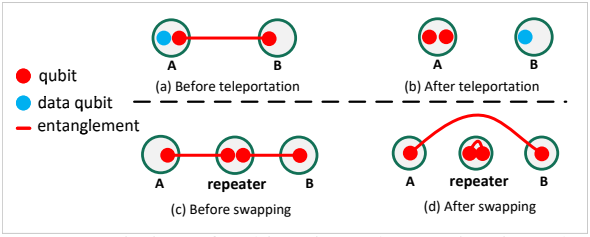


Fig. 1: Transmission of qubit using teleportation in (a-b) and quantum entanglement swapping in (c-d).

While machine learning has been a highly active research area recently, works on machine-learning-based routing in quantum networks are still relatively limited at this moment. We are able to find one of such research [12] which utilizes reinforcement learning to schedule entanglements. Specifically, the work in [12], focuses on optimizing entanglement times on quantum channels across a path to ensure an entanglement state between the two end nodes can be established before any channels decay. In this paper, we focus on using machine learning on a different aspect of routing, which is to allocate quantum channels to accommodate multiple communication requests in a quantum network. To our knowledge, there are currently no such works in the literature. With such motivation, in this paper, we present a deep reinforcement routing scheme that is called *Deep Quantum Routing Agent (DQRA)*. We start with modeling the problem of entanglement routing as a reinforcement learning problem with the following settings:

- **Network environment** includes information on the current network graph (i.e. which nodes can establish entanglement with which others), qubit capacity at each node, and a set of pending routing requests (demands).
- **Episodes** are the accommodation of all requests in a given window. An episode ends by solving all requests in a set, or by entering an environment state in which no pending requests can be solved. Each step in an episode refers to the accommodation of one request.
- **Actions** that the agent makes at each step include selecting then routing a pending request.
- **Rewards** that the agent receives after an action are designed to guide the agent to generate schedule that maximize the number of accommodated requests in a given time window.

DQRA then solves the problem of using a combination of a deep neural network and a shortest path algorithm. The neural network first observes the current environment state to select a pending request to accommodate. The selected request is then routed using a shortest path algorithm that uses a metric representing the qubit capacities among nodes across a path. We use two algorithms to train the deep network of DQRA, as a deep reward network, and as a deep Q network (DQN). In the first case, the neural network predicts the true reward obtained if a request is selected, and in the second case, the neural network predicts the Q-values of selecting the requests. Both training algorithms are guided by the previously mentioned reward function. Our experiment study shows that, on average, both DQRA models are able to maintain a rate of successfully routed requests above 80% in a qubit-limited quantum network, and approximately 60% in extreme conditions of network (i.e. all nodes can only be end nodes or repeater exactly once). We also empirically show that the routing time of DQRA increase as a polynomial function of network sizes (i.e. number of nodes). Specifically, the paper has the following **contributions** and **intellectual merits**:

1) We tackle the problem of entanglement routing in quantum networks from a machine learning perspective. Specifically, we propose a reinforcement learning model with specific designs of environments, actions, and rewards, that guide the agents towards a schedule that fulfills the most traffic requests within a time window.
2) We present DQRA, a deep reinforcement routing scheme that consists of an empirically designed deep neural network that schedules requests and a qubit-preserved shortest path algorithm that routes selected ones. The deep network of DQRA can be trained either as a deep reward network or a deep Q network. DQRA is shown to obtain good request-resolving rates even in qubit-scarce networks, and is scalable in terms of routing times.

Organization: The rest of the paper is organized as follows. Section §II discusses the concepts related to our work, and mathematically formalizes our research problem. We discuss DQRA in Section §III then present the experiment study in Section §IV. Finally, we conclude our paper in Section §V.

II. Network Model and Research Problem

In the following sections, an overview of the quantum network and basic mathematical notations are initially introduced. Formal research problem is also defined to demonstrate the key ideas behind the proposed model and methodology.

A. Quantum Network: An Overview

We first briefly introduce basic definitions in the context of a quantum network. The main components of quantum networks and their key roles are also briefly discussed as follows.

Quantum nodes are computers equipped with quantum processor(s) and are capable of manipulations on qubits. Specifically, the nodes can establish quantum entanglements between their qubits and qubits in other nodes, or perform entanglement swapping in the case they work as repeaters.

Quantum entanglement and teleportation are the process of establishing a quantum link between two qubits on two nodes so that their states are interdependent on each other. In short, this process involves performing a Bell State Measurement on qubits at two end-nodes then sending the co-relation through a classical transmission channel using two classical bits (e.g., via network slicing). Quantum entanglement is the mean to start a process of sending data via a quantum network, called *teleportation* as shown in Fig. 1. Henceforth, we use the term "neighbors" to refer to nodes that are capable of forming *direct* entanglement with each other, i.e., they are connected through a classical channel with a distance is set below 143km[1].

Entanglement swapping is used to extend quantum entanglement to a long-distant pair of nodes. In this case, each end node can have a qubit entangled with a qubit in a same intermediate node (i.e., repeater). The repeater then performs entanglement swapping on its own qubits which results in the entangled state between qubits of the two end nodes. We further refer to the intermediate node as a *repeater*. Since a repeater eventually needs to perform entanglement swapping, it must be able to gather at least two available qubits.

Quantum channels refer to established entanglement pairs between two neighbors. A chain of quantum channels to which are continuously connected is referred as a *quantum route*.

B. Network Model: Mathematical Notations and Illustration

Mathematically, we model a quantum network as a graph $G = (V, E)$ in which $V = \{v_i\}_{i=1}^{N}$ represents the set of nodes in the network of size N, and $E = \{e_{i,j}; v_i, v_j \in V\}$ represents the set of *pairwise neighborhood relationships* among the nodes. More specifically, the neighborhood relationship between v_i and v_j exists when there is a possibility to form a quantum channel between them, in other words, qubits in v_i and v_j can establish quantum entanglements. To elaborate, there exists $e_{i,j} \in E$ only when v_i and v_j are connected through a classical network channel, and the physical distance between them is within the threshold to establish qubit entanglements (i.e. 143 km).

[1]A long-distance entanglement decays exponentially with the physical distance between the two entangled nodes [1], [2], [3]. Quantum teleportation over a distance of 143 km has been deployed but still in the early stage.

We define $C = \{c_i; c_i \in \mathbb{N}\}_{i=1}^{N}$ as the set of qubit capacities in which c_i is the *maximum* number of qubits that node v_i can generate to form entanglements with its neighbors. If there exists $e_{i,j} \in E$, v_i and v_j can establish at most $\min(c_i, c_j)$ entangled qubit pairs between them. We set the lower bound $\min(\{c_i;\ \forall\ c_i \in C\}) = 2$ so that any node v_i can connect to at least two neighbors and perform entanglement swapping. We then define a quantum route in the network between the source v_s and the destination v_d as a path $p = \{e_{p_1,p_2}, e_{p_2,p_3}, ..., e_{p_{k-1},p_k}\}$ with $e_{p_1,p_2}, e_{p_2,p_3}, ..., e_{p_{k-1},p_k} \in E$; $v_{p_1} = v_s$; $v_{p_k} = v_d$ and $p_i \neq p_j$ for any pair of (i, j) in the path; any nodes beside v_s and v_d are repeaters in the path. p is further constrained on the number of qubits available in the repeaters. Mathematically, let q_i be the number of qubits in a repeater v_i that has already been utilized in other quantum channels, the remaining number of available qubits must be at least two, i.e., $c_i - q_i \geq 2$. A demand (v_s, v_d) in the set of network demands $\mathcal{D}$ is successfully resolved by determining a quantum routing path between them that satisfies both mentioned constrains.

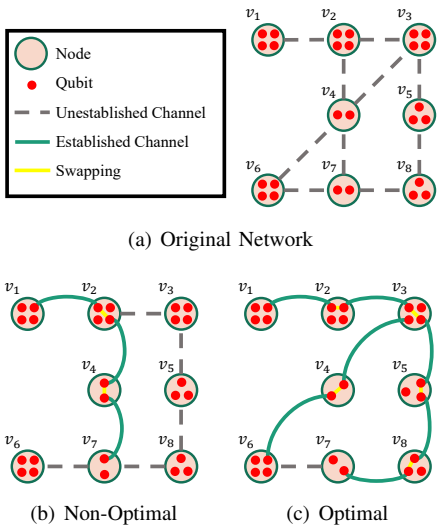
C. Problem Formulation

In this work, we advocate a novel routing scheme – dubbed *Quantum Routing Scheme ($\mathcal{Q}_{\mathcal{RS}}$)* – that implements a deep reinforcement learning model to circumvent the computability limitations of *heuristic* conventional qubit assignment-based schemes, while concurrently meeting the quality-of-service requirements (e.g., connectivity) of networks. Given a quantum network $G = (V, E)$ with a set of qubit capacities and a set of network demands $\mathcal{D}$, our goal is to maximize the entanglement routing rate achieved by a routing scheme $\mathcal{Q}_{\mathcal{RS}}$ as follows:

$$\mathcal{Q}_{\mathcal{RS}} = \max_{(v_s,v_d)\in\mathcal{D}} \sum \mathcal{P}_{(v_s,v_d)} 1_{\{\mathcal{E}_{(p,q)}=1\ \forall\ (p,q)\in\mathcal{P}_{(v_s,v_d)}\}} \quad (1)$$

Where $\mathcal{P}_{(v_s,v_d)}$ denotes the entanglement path connecting the source-destination pair (v_s, v_d). Here, $1_{\{.\}}$ is the indicator function; it is equal to one if the condition in the subscript is true, otherwise zero. $\mathcal{E}_{(p,q)}$ denotes the entanglement status between the two quantum nodes p and q. We further constraint the size of $\mathcal{D}$ to be a fixed integer k. More specifically, $\mathcal{Q}_{\mathcal{RS}}$ performs routing on a window of k source-destination pair at a time. It should be noted that this constraint does not reduce the generality of the research problem, since a request set can be split or padded to meet the size requirement.

As the number of paths for a set $\mathcal{D}$ grows exponentially with $|V|$ and $|\mathcal{D}|$, determining the optimal solution for all requests simultaneously could become highly challenging. Furthermore, the network capacity may not be able to accommodate all requests in $\mathcal{D}$. Consequently, we break the problem into two tasks which are 1) to schedule demands to accommodate and 2) to determine the path for each one. These two components make up the output of our $\mathcal{Q}_{\mathcal{RS}}$ scheme. We shows an illustrative example in Fig. 2. Fig. 2(a) shows the original network topology. At the same time, the network needs to accommodate two demands, (v_1, v_7) and (v_3, v_6). Fig. 2(b) demonstrates the case in which (v_1, v_7) is routed first using a least-hop strategy to select paths. This solution is not optimal as v_4 and v_7 do not have enough qubits left to accommodate the demand (v_3, v_6). Fig. 2(c) shows the optimal solution in which (v_3, v_6) is resolved first.



(a) Original Network

(b) Non-Optimal (c) Optimal

Fig. 2: An example on quantum network and routing solutions for two demands (v_1, v_7) and (v_3, v_6)

III. Deep Quantum Routing Agent

In this section, we describe our deep reinforcement routing scheme, henceforth referred to as Deep Quantum Routing Agent (DQRA). DQRA utilizes a novel deep neural network to schedule requests and a shortest-path algorithm for routing them. Depending on the training algorithm, the deep neural networks is referred to as a deep reward network (DRN) or a deep Q network (DQN). Since the network architectures are identical in both cases, for simplicity, we refer to the neural network as a Scheduling Network (SN) throughout this section. Given a quantum network $G(V, E)$ with a qubit capacity set C and a set of routing requests $\mathcal{D}$, we refer to the accommodation of all requests in $\mathcal{D}$ an *episode*, denoted as $\mathcal{E}$. $\mathcal{E}$ consists of a chain of consecutive action $a^{(t)}$ that DQRA makes at each step t to resolve *one* request in $\mathcal{D}$. An episode ends when all requests in $\mathcal{D}$ are resolved, or when no current requests can be accommodated (e.g., when all repeaters run out of qubits). The flow of DQRA is illustrated in Fig. 3. In the following sections, we discuss DQRA in terms of input states, output actions, reward function, architecture, and training.

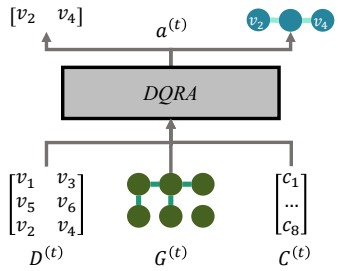


Fig. 3: An example of input and output of DQRA at step t.

A. Input State

At step t in an episode $\mathcal{E}$, an input state to DQRA consists of 1) the connectivity in the network $G^{(t)}(V, E^{(t)})$, 2) the qubit capacity of each node $C^{(t)}$, and 3) the request set $\mathcal{D}^{(t)}$. $G^{(t)}$ is generated as a subset of G in which edges that connect to nodes with current qubit capacity of 0 are removed. Then we extract the adjacency matrix $A^{(t)}$ of $G^{(t)}(V, E^{(t)})$ which is transformed to a row vector where only dimensions corresponding to edges that originally exist in G are kept. Let the input vector that represents $G^{(t)}$ be $\mathcal{A}^{(t)}$, then $\mathcal{A}^{(t)} = [\{A^{(t)}_{i,j}; e_{i,j} \in V\}]$. $\mathcal{D}^{(t)}$ is the second input to DQRA. In terms of representation, each node in a request is modeled as binary vector $v = [\{v_i; i = 1 \dots |V|\}]$ in which v_i is 1 if the node is i, and 0 otherwise. If a request is already resolved prior to step t, its source and destination are replaced with 0-vectors. Overall, each source-destination pair makes up one row in $\mathcal{D}^{(t)}$ which in turns become a binary matrix of size $k \times 2|V|$. Finally, the qubit capacity at step t is modeled as a vector $\mathcal{C}^{(t)} = [\{c^{(t)}_i; c_i \in C\}]$. All three components make up the input state $X^{(t)}$that DRN observes from the environment at step t, $X^{(t)} = \{\mathcal{A}^{(t)} \quad \mathcal{D}^{(t)} \quad \mathcal{C}^{(t)}\}$.

B. Output Action

At each step t in an episode, DQRA selects one among the pending requests $\mathcal{D}^{(t)}$ then assigns a path connecting the source and destination nodes. We utilize a hybrid approach, that is to use the deep network SN for scheduling the traffic requests, and a short-test path algorithm with customized metric to determine the path for each request.

In terms of scheduling, SN outputs a reward vector (or a Q vector the the case of DQN) $r^{(t)}$ of size $|\mathcal{D}|$: $r^{(t)} = \{r^{(t)}_0, r^{(t)}_1, \dots, r^{(t)}_{|\mathcal{D}|}\}$ in which $r^{(t)}_i$ represents the return at the end of step t if DQNA routes request i. Then, the *pending* request with the highest return value is selected in each step, $a^{(t)} = \text{argmax}_{i \in \mathcal{D}^{(t)}_P}(r^{(t)}_i)$, where $\mathcal{D}^{(t)}_P$ is the set of unresolved requests at t.

With $a^{(t)}$ selected, to determine the path between the source and destination, we utilize a shortest path approach in which the metric expresses the qubit capacity of the nodes. More specifically, a shortest path algorithm is applied. The weight of edge $e_{i,j}$ at step t is as $w^{(t)}_{i,j} = {}^1\!/_{\min(c^{(t)}_i, c^{(t)}_j)}$ where $c^{(t)}_i$ is the current capacity of node v_i. Using this metric, a shortest path of which total metric is low is more likely to traverse nodes with high qubit capacities and less likely to exhaust qubits in any nodes. To prevent a node with less than two qubits being selected as a repeater, we apply the shortest path algorithm on a node-induced sub-graph $\mathcal{G}^{(t)}$ of $G^{(t)}$ in which a node v_i is retained only if it has $c^{(t)}_i \geq 2$ *or* v_i is either the source or destination in the current request.

At the end of each action, the input states are updated for the next step $t + 1$. First, the row that associates to the request resolved by $a^{(t)}$ in $\mathcal{D}^{(t)}$ is replaced with 0's to obtain $\mathcal{D}^{(t+1)}$. Then, $\mathcal{C}^{(t)}$ is updated with new qubit capacities. Finally, $G^{(t+1)}$ is acquired by removing all edges that associate with any node v_i that has $c^{(t+1)}_i = 0$ and updating the weights of the edges using $\mathcal{C}^{(t+1)}$.

C. Reward Function

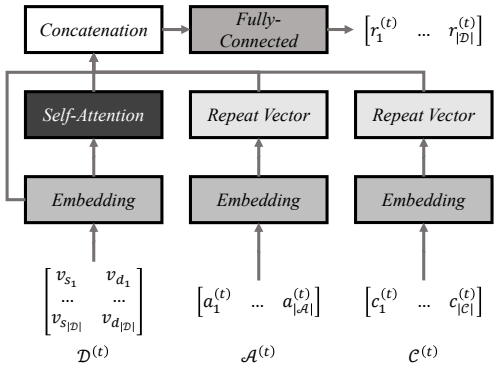


Fig. 4: DQRA deep neural network architecture

The reward function is specifically designed to increase the number of accommodated routing requests in a window. Mathematically, let the reward at step t be $R^{(t)}$, then

$$R^{(t)} = n^{(t)}_r \alpha + (|\mathcal{D}| - n^{(t)}_r)\beta f + \lambda R^{(t+1)}(1 - f) \quad (2)$$

where $n^{(t)}_r$ is the number of requests that are resolved from the beginning of the episode through step t, f is a binary indicator that is 1 when t is the ending step of and episode and 0 otherwise, $\alpha > 0$ is the reward term, $\beta < 0$ is the penalty term, and $\lambda \in [0, 1]$ is a discount factor. Both α, β, and λ are hyper-parameters to be selected during the training phase. Overall, the reward function design encourages the agent to find schedules that accommodate more requests, and penalizes schedules that end too early - the less requests solved, the heavier the penalties.

D. Scheduling Network Architecture

In this subsection, we describe the architecture of DRN. First, each input component among $\mathcal{D}^{(t)}$, $\mathcal{A}^{(t)}$, and $\mathcal{C}^{(t)}$ are fed into a different embedding network. In short, an embedding network consists of multiple fully-connected layers which transform an input vector into embedding vectors, usually of lower dimensionality. Let the mappings represented by the embedding networks for $\mathcal{D}^{(t)}$, $\mathcal{A}^{(t)}$, and $\mathcal{C}^{(t)}$ be $Emb_{\mathcal{D}}(\cdot)$, $Emb_{\mathcal{A}}(\cdot)$, and $Emb_{\mathcal{C}}(\cdot)$, respectively, then

$$\begin{aligned} Emb_{\mathcal{D}}(\mathcal{D}^{(t)}) = U_{\mathcal{D}} = [u_{s_1,d_1} \quad \dots \quad u_{s_{|\mathcal{D}|},d_{|\mathcal{D}|}}]^T \\ Emb_{\mathcal{A}}(\mathcal{A}^{(t)}) = u_{\mathcal{A}^{(t)}} \\ Emb_{\mathcal{C}}(\mathcal{C}^{(t)}) = u_{\mathcal{C}^{(t)}} \end{aligned} \quad (3)$$

For DRN to further learn the interrelationships among all the pending requests, $U_{\mathcal{D}}$ is input into a *Self-Attention* layer. In short, a self-attention layer outputs a "context" matrix in which each row represents the "context score" of a request given the rests in $\mathcal{D}^{(t)}$. Let the mapping by the self-attention layer be $\mathcal{S}_{\mathcal{D}}(\cdot)$, then $\mathcal{S}_{\mathcal{D}} = [\sigma_{s_1,d_1} \quad \dots \quad \sigma_{s_{|\mathcal{D}|},d_{|\mathcal{D}|}}]^T$ where σ_{s_i,d_i} is the context score of request i. We then repeat $u_{\mathcal{A}^{(t)}}$ and $u_{\mathcal{C}^{(t)}}$ $|\mathcal{D}|$ times, then concatenate them with the rows in $U_{\mathcal{D}}$ and $S_{\mathcal{D}}$ to form the complete embedding for each request:

$$U = \begin{bmatrix} u_{s_1,d_1} & \sigma_{s_1,d_1} & u_{\mathcal{A}^{(t)}} & u_{\mathcal{C}^{(t)}} \\ \dots & & & \\ u_{s_{|\mathcal{D}|},d_{|\mathcal{D}|}} & \sigma_{s_{|\mathcal{D}|},d_{|\mathcal{D}|}} & u_{\mathcal{A}^{(t)}} & u_{\mathcal{C}^{(t)}} \end{bmatrix} \quad (4)$$

Finally, U is input into a block of fully-connected layers that output a vector of $|\mathcal{D}|$ values $r_1^{(t)}, r_2^{(t)}, \ldots, r_{|\mathcal{D}|}^{(t)}$ that represent the reward if DQRA select each request. The architecture of DQRA's deep neural network is shown in Fig. 4.

E. Training Algorithm

We utilize two algorithms to train DQRA deep neural network, specifically, as a supervised deep reward network (DRN), and as a deep Q network (DQN). In the supervised case, DRN is trained as a supervised model to predict the true reward of taking an action. In an episode, we first randomize the request set $\mathcal{D}$. Then, for each step in an episode, the input states are fed to the model to generate actions then updated for the next step as described in Section §III-B. At the end of an episode, the reward values for each step are computed backward using equation (2).

In terms of training objective, we utilize *Mean Squared Error* function. Because the model only knows the reward of actions that it has taken, we utilize a *mask* vector $m^{(t)}$ of size $|\mathcal{D}|$ in which $m_i^{(t)} = 1$ if request i is selected in $a^{(t)}$, and 0 otherwise. $m^{(t)}$ prevents non-selected requests from contributing to the loss value. The loss for one episode $\mathcal{E}_i$ is computed as

$$\mathcal{L}_i = \sum_{t \in \text{episode}} (r^{(t)} * m^{(t)} - R^{(t)} m^{(t)})^2 \tag{5}$$

where $*$ represents an element-wise multiplication. Finally, the overall loss is averaged across episodes in a training batch of size n_b

$$\mathcal{L} = \frac{1}{n_b} \sum_{i \in \text{batch}} \mathcal{L}_i \tag{6}$$

In the DQN training scheme, two versions of QN are maintained, a predict network and a target network. the output of the predict network is considered the Q values of each action, and the loss of one episode $\mathcal{E}_i$ is as

$$\mathcal{L}_i = \sum_{t \in \text{episode}} (Q_P^{(t)} * m^{(t)} - Q_P'^{(t)} * m^{(t)})^2 \tag{7}$$

where $Q_P^{(t)}$ is the output of the predict network at step t, and $Q_P'^{(t)}$ is the target Q value at step t which is computed as:

$$Q_P'^{(t)} = (1 - l_r) * Q_P^{(t)} + l_r * (Q_T^{(t+1)} + \lambda R^{(t)}) \tag{8}$$

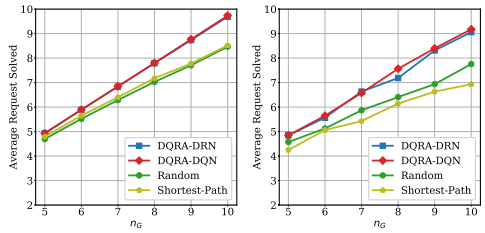
where l_r is a learning rate, $Q_T^{(t+1)}$ is the Q values output by the target network for $t+1$, λ and $R^{(t)}$ are the discount factor and reward function at t as described in Eq. (2). The overall loss for one training batch is also calculated by Eq. (6). The predict network can be trained using regular algorithms like Stochastic Gradient Descent; the target network is updated with the predict network's weights once every few hundred of epochs. During decision making phases, only the predict network is utilized.

IV. Experiment and Evaluation

We focus on grid networks of size $n_G \times n_G$ nodes, and use a routing window size of n_G requests, with $n_G \in \{5, 6, \ldots 10\}$. We test four cases with different nodes' qubit capacities: 1) $c_i = 4 \ \forall \ i$, 2) $c_i \in [3, 4] \ \forall \ i$, 3) $c_i \in [2, 4] \ \forall \ i$, and 4) $c_i = 2 \ \forall \ i$. The test cases represent networks with increasingly more limited qubit capacity. The last simulation represents the extreme case in which each node can act as a repeater for exactly one path, and, if a node is the source or destination of a request, then it cannot be a repeater anymore. In all test cases, each node is selected as the source or destination of requests within a window no more than once. For a fair comparison, we utilize the same architectures (in terms of number of layers and neurons) of deep networks for both DQRA using supervised reward network (denoted as DQRA-DRN) and DQRA using the deep Q network (DQRA-DQN). After fine-tuning, the selected deep network architectures are as 1) all embedding networks have two fully-connected layers, each of which has n_{E_i} neurons with $n_{E_i} = |X_i^{(t)}|, i \in \{\mathcal{A}^{(t)} \quad \mathcal{D}^{(t)} \quad \mathcal{C}^{(t)}\}$, and 2) the fully-connected block that outputs return values has three layers, each has $(|\mathcal{A}^{(t)}| + |\mathcal{D}^{(t)}| + |\mathcal{C}^{(t)}|)$ neurons. The hyper-parameters α, β, and λ, for the reward function in Eq. (2) are set at 0.2, -1, and 0.9, respectively. The learning rate l_r for training DQRA-DQN as in Eq. (2) is 0.1. Finally, the mini-batch size in all experiments, n_b in Eq. (6), is 512.

We implement two baseline routing strategies to compare with DQRA. The first (Random) performs random selection on requests, then assigning path to the selected request using the shortest path algorithm. The second (Shortest Path) strategy selects requests based on a shortest-first strategy. Specifically, the shortest paths for each pending requests are first determined, then the one with minimum value is chosen to accommodate. We use the average number of solved requests across 1000 windows as the evaluation metric. The five-run averaged experiment results are shown in Fig. 5. In all cases, the deep networks are trained for $10,000$ epochs. As shown in Fig. 5, the two versions of DQRA yield similar performances in all tests, both of which are significantly higher than the two naive baselines in all test cases. In Fig. 5(b) and (c) where each node has c_i randomized, DQRA models maintain a successful routing rates of over 80% in all network sizes, with DQRA-DQN being slightly better than DQRA-DRN. In the two extreme cases where each node has exactly four and two qubits (Fig. 5(a) and (d)), the performances of DQRA models are almost identical. In networks with $c_i = 4 \ \forall \ i$, the two achieve almost 100% resolving rates, whereas in networks with $c_i = 2 \ \forall \ i$, they achieve the rates from about 70% at $n_G = 5$ to approximately 59% at $n_G = 10$.

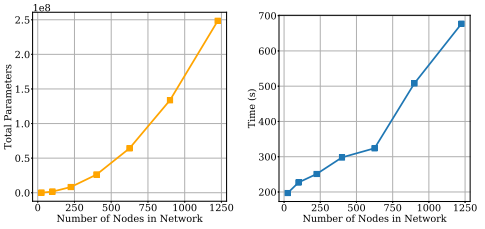
Next, we further examine the network size and time performance of DQRA models. In production phase, the two DQRA models should have equal time performance as they use identical prediction architectures. In this experiment, the grid networks has size $n_G \times n_G$ with $n_G \in \{5, 10, \ldots, 35\}$ and the window sizes are fixed to 10. Furthermore, we set the qubit capacity of all nodes to a very high number so all requests



(a) $c_i = 4$ (b) $c_i \in \{3, 4\}$

(c) $c_i \in \{2, 4\}$ (d) $c_i = 2$

Fig. 5: Models' performance on routing n_G requests in grid networks of size $n_G \times n_G$ nodes of qubit capacity c_i



(a) Total Parameters (b) Routing Time

Fig. 6: Total parameters and routing time of DQRA by number of nodes in the network

of the windows are always accommodated. Fig. 6 shows the DQRA's total number of parameters in the deep networks (a) and routing time (b) by the number of nodes in a network. As shown in Fig. 6(a), by the design of the deep network, the number of parameters increases as a polynomial function (specifically, quadratic) of the number of nodes. This is an advantage of DRN over DQN which must maintain two neural networks of the same size during training. As shown in Fig. 6(b), the routing times for 1,000 windows (10 requests each) increases with a linear pattern with respect to the numbers of nodes. This is to be expected, as being deterministic, the mapping of inputs to outputs in any neural networks has polynomial time complexities; and the shortest path algorithm is utilized exactly once for any demands.

V. Conclusion

Routing in quantum networks, one of the key problems of the next-generation network system, and machine learning, one of today leading research areas, have not seen much integration currently. With such motivation, this paper aims to bridge that gap with a new machine-learning-powered quantum routing model for quantum networks. In particular, we have proposed a deep reinforcement routing scheme (DQRA) to construct routing paths for all demands in quantum networks. We have modeled the problem of entanglement routing in quantum networks as a reinforcement learning problem that consists of input states, actions, and rewards. At each step in a routing window, we construct input states of the model by considering the quantum network's states, the qubit capacities of nodes, and the set of pending demands; the actions were defined as the demand selection to accommodate and the path connecting the two ends of the demand; and the rewards were designed to guide the model towards schedules that fulfill the maximum number of demands in a time window. In terms of architectures, DQRA consists of two components, an empirically designed deep neural network that was used to observe the current input states to decide the routing schedule, and routing paths of the selected demands were then determined by a qubit-preserved shortest path algorithm. We further utilized two algorithms to train the deep network of DQRA, as a deep reward network, and as a deep Q network (DQN). Experiment results show that, on average, DQRA is able to maintain a fulfilling rate at above 80% in a qubit-limited grid network, and about 60% in extreme conditions i.e. each node can act as repeater exactly once within a time window.

Acknowledgement

This research was supported in part by the US NSF grant CNS-2103405.

References

[1] C. Cicconetti, M. Conti, and A. Passarella, "Request scheduling in quantum networks," *IEEE Transactions on Quantum Engineering*, vol. 2, pp. 2–17, 2021.

[2] W. Dai, T. Peng, and M. Z. Win, "Optimal remote entanglement distribution," *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 3, pp. 540–556, 2020.

[3] C. Li, T. Li, Y.-X. Liu, and P. Cappellaro, "Effective routing design for remote entanglement generation on quantum networks," *npj Quantum Information*, vol. 7, pp. 1–12, 2020.

[4] S. Shi and C. Qian, "Concurrent entanglement routing for quantum networks: Model and designs," ser. SIGCOMM '20. New York, NY, USA: Association for Computing Machinery, 2020, p. 62–75.

[5] J. Chung, G. S. Kanter, N. Lauk, R. Valivarthi, W. Wu, R. R. Ceballos, C. Pena, N. Sinclair, J. Thomas, S. Xie, R. Kettimuthu, P. Kumar, P. Spentzouris, and M. Spiropulu, "Illinois express quantum network (ieqnet): metropolitan-scale experimental quantum networking over deployed optical fiber," in *Defense + Commercial Sensing*, 2021.

[6] A. Dahlberg, M. Skrzypczyk, T. Coopmans, L. Wubben, F. Rozpundefineddek, M. Pompili, A. Stolk, P. Pawełczak, R. Knegjens, J. de Oliveira Filho, R. Hanson, and S. Wehner, "A link layer protocol for quantum networks," in *Proceedings of the ACM Special Interest Group on Data Communication*, ser. SIGCOMM '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 159–173.

[7] M. Mehic, M. Niemiec, S. Rass, J. Ma, M. Peev, A. Aguado, V. Martin, S. Schauer, A. Poppe, C. Pacher, and M. Voznak, "Quantum key distribution: A networking perspective," *ACM Comput. Surv.*, vol. 53, no. 5, Sep. 2020.

[8] A. S. Cacciapuoti, M. Caleffi, F. Tafuri, F. S. Cataliotti, S. Gherardini, and G. Bianchi, "Quantum internet: Networking challenges in distributed quantum computing," *IEEE Network*, vol. 34, no. 1, pp. 137–143, 2020.

[9] L. Gyongyosi and S. Imre, "Entanglement availability differentiation service for the quantum internet," *Scientific Reports*, vol. 8, 2018.

[10] S. Pirandola, R. Laurenza, C. Ottaviani, and L. Banchi, "End-to-end capacities of a quantum communication network," *Communications Physics*, vol. 2, 2019.

[11] ——, "Fundamental limits of repeaterless quantum communications," *Nature Communications*, vol. 8, 2017.

[12] S. Khatri, "Policies for elementary links in a quantum network," *Quantum*, vol. 5, p. 537, 2021.