Q-DDCA: Decentralized Dynamic Congestion Avoid Routing in Large-Scale Quantum Networks

Lutong Chen, *Graduate Student Member, IEEE*, Kaiping Xue, *Senior Member, IEEE*, Jian Li, *Member, IEEE*, Ruidong Li, *Senior Member, IEEE*, Nenghai Yu, Qibin Sun, *Fellow, IEEE*, and Jun Lu

***Abstract*— The quantum network that allows users to communicate in a quantum way will be available in the foreseeable future. The network capable of distributing Bell state entangled pairs faces many challenges due to entanglement decoherence and limited network performance, especially when the network scale is enormous. Many entanglement distribution protocols have been proposed so far, and most of them are in a centralized and synchronized manner, which may be infeasible in large-scale networks. As such, in this paper, we propose a full spontaneous version of quantum networks in which the quantum nodes autonomously manage multiple entanglement distribution requests. However, one major issue is that quantum nodes have little knowledge about the network, especially the congestion (e.g., some nodes may have no usable quantum memories). We present a routing algorithm to adaptive evaluate the congestion on the neighbor nodes to avoid potential congestion. We use SimQN, the new network layer simulation platform built by our research team, to evaluate our proposed design. The result demonstrates that it can adapt to changes in network resources and reduce the drop rate that eventually leads to a higher entanglement distribution rate but remains fair for multiple requests to use the network resources fairly and achieve a more balanced throughput.**

***Index Terms*— Entanglement distribution, routing algorithm, fidelity, quantum network.**

I. Introduction

IN RECENT years, quantum internet [1], [2], [3] based on entanglement principles in physics makes it possible for remote devices to transmit information [4] in a secure way. Quantum network is capable of many neoteric functionalities out of the classic network, including quantum key distribution [3], [5], [6], [7], quantum computing [8], [9], [10], [11], [12], time synchronization [13], and etc. Among them, a vast majority of applications need to use long-distance Bell state entangled pairs, and quantum network distributes such entangled pairs between remote entities in the network [14], [15].

One challenge is how to build a feasible and large-scale quantum network considering quantum imperfection [16], [17] in this NISQ era [18]. On the one hand, the entangled state will be decohered. The critical metric, fidelity, drops dramatically during entanglements distribution because of the imperfect quantum operations and the elapsed time. Thus, an entanglement has an extremely short lifespan. On the other hand, nodes in quantum networks have limited concurrent capabilities because quantum memories [19] to store qubits are rare, and entanglement generation (i.e., one node produces an entangled pair and shares it with a neighbor node) [20] is currently stochastic and complex. Consequently, it is difficult to concurrently distribute long-distance entangled pairs in the network.

So far, several works have been proposed to address the above issues by introducing a centralized controller [21], [22], [23], [24]. The controller can obtain a global view, making it easier to schedule entanglement distributions efficiently. However, it may bring out an enormous overhead in communication and computation, which is critical in a quantum network, especially in a large-scale one. Another recurring requirement is substantial time synchronization by dividing time into a series of time slots so that the controller can periodically collect the information and give instructions, while such a task is challenging even in a classic network. These problems urge us to develop a decentralized and asynchronous entanglement distribution mechanism to overcome the shortcomings introduced by the centralized control model.

To solve such problems, in this paper, we propose a decentralized and asynchronous concurrent long-distance entanglement distribution network model without a controller or time division. Instead, like traditional network protocols, transmissions are managed by quantum nodes themselves. More specifically, the source node decides the sending rate (i.e., the entanglement distribution in processing concurrently). The entanglement distribution is hop-by-hop, in which all routers on a path will perform entanglement swapping in order. Moreover, all network resources, including quantum memories, are preemptive (i.e., not preserved for a specific request before transmission) to maintain fairness among multiple requests.

Manuscript received 24 March 2022; revised 31 March 2023; accepted 1 June 2023; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Editor O. Gurewitz. This work was supported in part by the National Scientific and Technological Innovation 2030 Major Project of Quantum Communications and Quantum Computers under Grant 2021ZD0301301, in part by the Anhui Initiative in Quantum Information Technologies under Grant AHY150300, in part by the Youth Innovation Promotion Association Chinese Academy of Sciences (CAS) under Grant Y202093, and in part by the Japan Society for the Promotion of Science (JSPS) KAKENHI under Grant 23H03380. *(Corresponding author: Kaiping Xue.)*

Lutong Chen, Kaiping Xue, Jian Li, Nenghai Yu, and Qibin Sun are with the School of Cyber Science and Technology, University of Science and Technology of China, Hefei, Anhui 230027, China (e-mail: kpxue@ustc.edu.cn).

Ruidong Li is with the Institute of Science and Engineering, Kanazawa University, Kakuma, Kanazawa 920-1192, Japan.

Jun Lu is with the School of Cyber Science and Technology and the Department of Electronic Engineering and Information Science, University of Science and Technology of China, Hefei, Anhui 230027, China.

Digital Object Identifier 10.1109/TNET.2023.3285093

Instead, the multiple requests spontaneously acquire resources with the first-in-first-out (FIFO) policy. In this situation, with the absence of a controller, network nodes are challenging to collect network status, especially for those that vary extremely, such as network congestion. Here, network congestion occurs when quantum nodes lack quantum memories or links (optical fiber) reach capacity. To address such dynamic variable network status, we design a quantum routing algorithm for this decentralized network to find the optimal next-hop adaptively, named **Q**uantum **D**ecentralized **D**ynamic **C**ongestion **A**void routing algorithm (Q-DDCA). To avoid congestion, Q-DDCA is designed to be an adaptive algorithm in which routers select the next-hop one based on the evaluated congestion level and then request it to require the necessary resources. Q-DDCA also guarantees the fidelity on the distributed entangled pairs by inventing the concept of short-term destination, where the entanglements are firstly distributed to short-term destinations so that entanglements distillation is performed on that node to upgrade the fidelity. In addition, we present the two bounds in the routing algorithm to adaptively produce routing results according to the network status and restrict decoherence.

The main goal of Q-DDCA is to maximize the network entanglement distribution rate (EDR), which is defined as the sum of entangled pairs distributed per second of multiple requests (a serial entanglement distribution between a source node and a destination node). Moreover, the proposed scheme should guarantee the fidelity of the distributed entangled pairs as many quantum applications (e.g., BB84 [25], or Teleportation [26], [27]) have an essential requirement on fidelity [28] to ensure security or availability. Unfortunately, achieving the fidelity goals means that more network resources are used even if they will be severely attenuated with EDR. Finally, the proposed algorithm should work for multiple requests and ensure each request uses the network fairly.

We develop a discrete-event driven simulator, SimQN [29], to evaluate the performance of the proposed algorithms. The results show that the performance can benefit from flexibly selecting the paths to avoid congestion and achieve 107.49% higher EDR than the shortest path algorithm based on Dijkstra's Algorithm. Meanwhile, the entanglements drop rate remains low and acceptable. Also, network resources are assigned to multiply requests fairly, which indicates that Q-DDCA can be more fair compared to both connection-oriented entanglement distribution models and the shortest path algorithm in a decentralized model. Our routing solution can improve the performance of large-scale quantum networks, enabling them to support various applications, such as enhancing the security of internet communication and facilitating distributed quantum computation.

This paper makes the following contributions:

- We propose a novel decentralized and asynchronous entanglement distribution model for the large-scale quantum Internet to avoid the communication and computation overhead introduced by centralized models. To address the fidelity downgrade issue and enlarge the network scale, we introduce short-term destinations to upgrade fidelity through entanglement distillation.
- We develop a spontaneous routing algorithm, Q-DDCA. In this algorithm, we formulate two bounds for path relaxing and fidelity guarantee, respectively. Also, we use the expected number of hops as the utility function that can adapt to the network states and take advantage of multiple resource allocation attempts.
- According to the discrete-event simulator, we construct a large-scale system-level quantum network simulation platform for experiments that demonstrates the superior performance of the proposed Q-DDCA algorithms in both efficiency and fairness.

The rest of the paper is organized as follows. First, we brief related works in Section II. In Section III, we state the entanglement distribution problem, the system model, and the designing goals as well. Then, we present our decentralized entanglement distribution model in Section IV. After that, we elaborate our routing algorithm in Section V and conduct simulations and evaluations in Section VI. Finally, we conclude our work in Section VII.

II. Related Work

The entanglement distribution is the primary task of a quantum network. Some challenges have been discussed to complete this task, including entanglement distribution scheduling [21], [22], [30], [31], routing algorithms [24], [32], [33], and resource allocation [28] for quantum networks. Several mechanisms have been proposed to meet these challenges, focusing on a particular quantum network topology. For the entanglement distribution problem, Wenhan Dai et al. proposed an effective entanglement distribution algorithm for the repeaters chain topology [21]. Also, fidelity and waiting time computation algorithms for the repeater chains have been proposed in [22]. As for the routing problem, a greedy routing algorithm for the lattice quantum network has been proposed [24]. It uses time slots to manage resource allocation, and entanglement generation and swapping are two phases in one slot.

In this paper, we mainly focus the study on a network with arbitrary topology and multiple requests. Pirandola [23] analyzed the end-to-end channel capacity in a quantum network. Franco et al. [26] proved that finding the optimal path in a quantum network is tough. Van Meter et al. [33] developed a routing algorithm based on Dijkstra's Algorithm but mainly focusing on a single request. Chakraborty et al. [34] proposed a routing algorithm based on multi-commodity flow optimization. Gyongyosi et al. also proposed a decentralized routing algorithm [35] mainly considering the success possibility and fidelity. We use a similar strategy for fidelity guarantee and improve it considering time. Also, it uses a centralized controller to find the optimal result. Also, Li et al. [28] developed an effective and detailed routing algorithm, including resource allocation, path determination, and entanglement distribution. These works are very effective if the global information is available or a centralized controller exists, such as a software-defined network (SDN) architecture, is introduced [36], [37]. Shi and Qian [38] proposed Q-Pass and Q-Cast routing algorithms for multiple requests, where

they assume a time synchronization and divide the time into four phases.

Compared to those works, we intend to propose a decentralized protocol of routing algorithm, similar to OSPF (Open Shortest Path First) in the classic Internet. Our scheme also does not require time synchronization, which can be difficult to achieve in large-scale networks. In such cases, it becomes challenging for the network to efficiently distribute entangled pairs as nodes must make decisions autonomously with limited information about the network's overall status. Despite this, we believe this task must be investigated. Both centralized control and time synchronization may bring an unavoidable computation overhead or communication delay in a large-scale quantum internet. As a result, we intend to develop a decentralized and asynchronous quantum Internet and propose Q-DDCA as a customized routing algorithm.

III. Problem Statement

A. System Model

Let graph $G = (V, E)$ denote the quantum network where $V = \{u_1, u_2, \ldots, u_N\}$ denotes the node set, u_i is the i-th node and N is the total number of nodes. The quantum nodes are capable of LOCC (local operations with classic communication) [39] and equipped with m_i quantum memories [40], [41], [42], i.e., nodes can store m_i qubits for T seconds. $E = \{(u, v)|u \in V, v \in V\}$ denotes the edge set, and each tuple $(u, v) \in E$ is a quantum channel that can transmit quantum information (usually an optical fiber) that directly connects u and v, and entangled pairs can be generated between these two nodes. We assume that the fidelity of entangled pairs generated between neighbor nodes is f_0. The fidelity is a $[0, 1]$ variable that describes the state difference between the entangled pair and the target state. Further, $C(u, v)$ is the limited capacity of the channel (u, v) [17], [43]. In this paper, we define capacity as the number of entangled pairs that can be generated on the channel in a period. Finally, let $N(u) = \{v|(u, v) \in E\}$ be the neighbor set of the node u.

Consider that the quantum network serves multiple entanglement distribution requests concurrently. Let $R = \{(u_k^s, u_k^d, f_k)|k = 1, 2, \ldots r, u_k^s \in V, u_k^d \in V, f_k \in [0, 1]\}$ be the set of all requests, where (u_k^s, u_k^d, f_k) is the k-th request and r is the number of concurrent requests. In each request $r_k = (u_k^s, u_k^d, f_k)$, u_k^s is the source node, while u_k^d is the destination node. f_k is the required fidelity, i.e., the distributed entanglements should have a fidelity higher than f_k.

In this paper, we adopt a layered Quantum Internet protocol stack [44], [45], where link-layer protocols [20], [46] are used to distribute link-layer entangled pairs, and this paper mainly focuses on network-layer functionality such as routing and distributing remote entangled pairs between non-neighbor nodes. Besides, we assume that a classic network exists so that all nodes can communicate and send control messages to each other.

We assume that the nodes have limited information about the network since the network is decentralized. We require a neighbor discovery protocol [47] to run on the nodes so that they can detect the network topology. It is achievable as the quantum networks are relatively static and connected by optical fibers [48]. Meanwhile, considering the heavy overhead of the neighbor discovery procedure, the nodes only obtain the correct topology within limited hops. Beyond that, the topology result is allowed to be incorrect and untimely. Furthermore, the nodes will have no information about other nodes (even their neighbors). They only know the status of entangled pairs that are connected to them.

TABLE I
Notations Table

Notation	Meaning
V	the node set
E	the quantum channel set
R	the request set
N	the number of nodes
r	the number of requests
u_i	the i-th node
m_i	quantum memory's size on u_i
r_k	the k-th request
u_k^s	the source node of r_k
u_k^d	the destination node of r_k
u_{curr}	the current node
u_{next}	the next-hop node
$C(u, v)$	the link capacity between node u and v
$N(u)$	node u's neighbor nodes
EDR	the network entanglement distribution rate
EDR_k	the entanglement distribution rate of r_k
f_k	the required fidelity of r_k
L_k	the maximized hops to guarantee fidelity
L'_k	a predetermined hops between short-term destinations
L_{total}	the actual hops of a success entanglement distribution
L_{prec}	the distance between u_k^s and u_{curr}
L_{remain}	the distance between u_{curr} and u_k^d
δ	the extended length per hop
wnd_k	the sending rate of r_k
t	the time of one resource request attempt
T	the memory cut-off time
T_{coh}	the coherence time
T_{total}	the actual time of one success entanglement distribution
$E(n)$	the expected number of swapping of a n-hop path
f_0	the initial fidelity on the link
$f(n)$	the fidelity after a n-hop distribution
ω	the state parameter of a Werner state
ω_n	the state parameter of a n-hop distribution
m	the m-th attempt of requiring resources to the neighbor node
M	the maximized number of attempts before cut-off time T
S_C	the candidate set
x_i	select the i-th node in S_C as the next-hop node
q_i	the acceptance rate of the i-th node in S_C

B. Design Goals

In this paper, we intend to propose a mechanism to distribute entangled pairs for multiple concurrent requests in a decentralized and asynchronous quantum network, as well as a corresponding routing algorithm. The design goals of the entanglement distribution schema include decentralization, high efficiency, fairness, and guaranteed fidelity.

1) Decentralization: The proposed quantum network and its routing algorithm should be a decentralized one to avoid the communication and computation overhead brought out by centralized control. The entanglement distribution request should be managed by the nodes autonomously.

2) High Efficiency: The primary goal for the network design of a quantum network is high throughput. The entanglement

distribution rate (*EDR*) indicates the total network throughput, and $EDR = \sum_{k=1}^{r} EDR_k$, where EDR_k is the number of entangled pairs distributed per second for the request r_k.

3) Fairness: Multiple requests should be handled fairly, and network resources should be assigned to requests equally when they pass via the same nodes. Our schema should guarantee that multiple requests use the network resources fairly so that no requests will be blocked or starved.

4) Fidelity Guaranteed: The distributed entanglements must have high fidelity to meet the request's requirement. For example, distributed entangled pairs for request r_k must be above f_k.

IV. A Decentralized Entanglement Distribution Scheme for Large-Scale Quantum Network

In this section, we develop a hop-by-hop decentralized entanglement distribution network through which the source node handles the entanglement distribution procedure to replace the controller. As mentioned in [44], two possible network models exist in quantum networks: circuit-switching and packet-switching. In this paper, we adopt the best-effort packet-switching model because it seeks to eliminate the need for a centralized controller and time synchronization in large-scale quantum networks. Consequently, all nodes must function autonomously and asynchronously. Specifically, the entangled pairs are distributed hop-by-hop, or routers on the path perform entanglement swapping to distribute entangled pairs iteratively. However, the entangled pairs decohere during the procedure. Thus, we design the conception of short-term destinations, where entanglement distillation [49], [50], [51] can be performed to provide the fidelity guarantee. Considering those factors, we will propose a routing algorithm for this network.

A. A Hop-by-Hop Entanglement Distribution Network

Since each request is handled in a decentralized manner, we mainly focus on one request $r_k = (u_k^s, u_k^d, f_k)$ in this section. The source node u_k^s manages r_k and controls how to distribute entangled pairs. To improve the efficiency, u_k^s handles wnd_k entangled pairs in distributing them concurrently. Here, we use wnd_k to denote the window size, *i.e.,* the number of concurrent entangled pair distribution for request r_k. For example, Fig. 1 shows an example of r_k, where u_k^s distributes entangled pairs to u_k^d in such a way that at most $wnd_k = 3$ entangled pairs can be managed at once. Currently, there are three entangled pairs in transmissions, including two entangled pairs that have been distributed to u_i and one to u_j. If entangled pairs are distributed to the destination or are dropped during the transmission, a new entangled pair is generated and begins to be distributed from the source node again.

Source node u_k^s performs the following operations recursively to distribute entangled pairs until they reach the destination, as shown in Fig. 2. Assume that the current entangled pair is distributed between u_k^s and u_i, as shown in Fig. 2(a). u_j and u_l are the neighbors of u_i, and u_l is in congestion. In Fig. 2(b), u_k^s runs a routing algorithm to decide the next hop (for example, u_j is selected). It communicates with u_j to acquire the necessary quantum memories. If u_j accepts that request, the EPR generator on edge (u_i, u_j) generates an entangled pair, as illustrated in Fig. 2(c). Finally, u_i performs a quantum swapping [52] to distribute a new entangled pair between u_k^s and u_j in Fig. 2(d). If successes, u_j can find the optimal next-hop node and distribute the entangled pairs to it. Thus, these operations are repeated until the entangled pair is distributed to u_k^d. Alternatively, if u_j refuses such a request, u_i has to drop the existing entangled pairs, notify the source node, or wait for another attempt (mentioned in Section V-B).

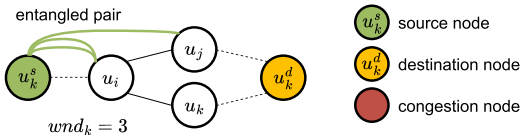
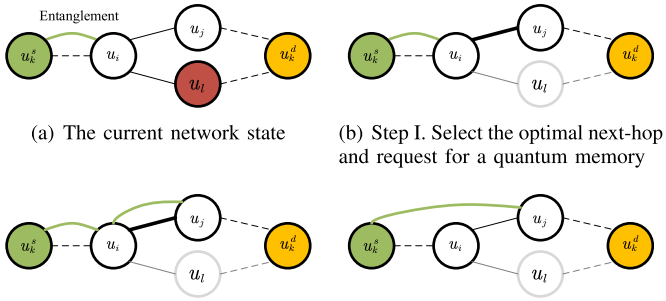


Fig. 1. An example of request r_k with $wnd_k = 3$. The green node is the source node, and the yellow node is the destination node. The dotted line represents the omission of multi-hop links.



(a) The current network state

(b) Step I. Select the optimal next-hop and request for a quantum memory

(c) Step II. Generate an adjacent entangled pair with an EPR generator

(d) Step III. Carry out entanglement swapping to distribute entangled pairs

Fig. 2. The process of hop-by-hop entangled pair distribution. The red node is in congestion.

B. The Short-Term Destination for Entanglement Distillation

During the procedure of entanglement swapping and qubit transmission, the entangled pairs decohere, and the fidelity decreases. A further mechanism must be proposed to meet the fidelity requirement to provide a fidelity guarantee for requests. In this part, we investigate how to meet the fidelity requirement. This paper assumes that the entangled pairs generated by EPR generators have a fidelity above f_0, and the required fidelity of r_k is f_k.

One difficulty here is that it is hard to measure fidelity without breaking entangled pairs due to the quantum No-Cloning Theorem. To this end, we observe that evaluating the low bound of fidelity is easier and more feasible. If we can evaluate the low bound of the fidelity, a strict distillation strategy can be performed to guarantee the final fidelity of the distributed entangled pairs. Such evaluation is based on the following rules: First, we insist that a qubit can be stored in a memory for no more than T seconds. Otherwise, the entangled pair is considered to be dropped. This requirement guarantees the low bound of fidelity after storing in a quantum memory. Further, it is possible to model the fidelity decrease after an entanglement swapping. Thus, it is possible to evaluate the low

bound of fidelity during the distribution process and further calculate the maximum number of hops before the fidelity is lower than the requested fidelity f_k. A similar mechanism is proposed [34], although it does not consider the time escape and decoherence while storing.

Assume that the entangled pairs are in Werner state [53]. The density matrix can be presented as

$$\rho(\omega) = \omega \left|\Psi^+\right\rangle \left\langle\Psi^+\right| + (1-\omega)\mathbb{I}_4/4,$$

where $\omega \in [0,1]$ is the ingredients of the Bell state (maximized entangled), and $1-\omega$ is the white noise (decoherence state), and $\mathbb{I}_4$ is the identity matrix of dimension 4. The corresponding fidelity is

$$F(\omega) = (1+3\omega)/4. \tag{1}$$

Obviously, the parameter ω is an equivalent of fidelity and measures the purity of an entangled pair in the Werner state.

We also consider decoherence in our model and mainly focus on two kinds of decoherence. The first comes from the noise during a qubit stored in a quantum memory, and we model this noise based on the following model. The fidelity of the Werner state entangled pair decays exponentially over time when stored in a quantum memory [53]. That is, ω drops to $\omega_{\text{delayed}} = \omega \cdot e^{-t/T_{\text{coh}}}$ after being stored for time t, where T_{coh} is a memory's attribution indicating the speed of decoherence.

The second source for fidelity to downgrade is from the quantum swapping [22]. Let's say there are two entangled pairs, with the fidelity f_1 and f_2. Equivalently, the corresponding parameter is ω_1 and ω_2. After the entanglement swapping, the parameter of the new entanglement becomes $\omega_{\text{swap}} = \omega_1 \cdot \omega_2$.

Consider a hop-by-hop entanglement distribution procedure, remember that f_0 denote the initial entanglement fidelity, and ω_0 be $\frac{4f_0-1}{3}$ respectively. For an entangled pair that has been distributed after n hops (i.e., after $n-1$ entanglement swapping), the fidelity is $f(n)$, and $\omega_n = \frac{4f(n)-1}{3}$. Following the hop-by-hop entanglement distribution strategy mentioned in the last section, a new entangled pair is generated between the current and next-hop nodes. Then the current node performs an entanglement swapping using the original n hop entangled pair and the newborn entangled pair. If successful, a new entangled pair is distributed between the source node and the next-hop node. Since the original n hop entangled pair can not be stored on the node for more than T during this period. Thus, the low bound of the original entangled pair before entanglement swapping is $\omega_n' \geq \omega_n \cdot e^{-T/T_{\text{coh}}}$. As the newly generated entangled pair's fidelity is ω_0, the fidelity after swapping is

$$\begin{aligned}
\omega_{n+1} &\geq \omega_0 \cdot \omega_n \cdot e^{-T/T_{\text{coh}}} \\
&\geq \omega_0^2 \cdot \omega_{n-1} \cdot e^{-2T/T_{\text{coh}}} \\
&\cdots \\
&\geq \omega_0^n \cdot \omega_1 \cdot e^{-\frac{nT}{T_{\text{coh}}}} \\
&= \omega_0^{n+1} \cdot e^{-\frac{nT}{T_{\text{coh}}}}.
\end{aligned}$$

If the final entangled pair is distributed through n hops, we require that the final fidelity is above f_k (the fidelity requirement in request r_k). Then, we have $f(n) \geq f_k$. In a more strict sense,

$$\begin{aligned}
f(n) &= \frac{1+3\omega_n}{4} \\
&\geq \frac{1+3\omega_0^n \cdot e^{-\frac{(n-1)T}{T_{\text{coh}}}}}{4} \geq f_k.
\end{aligned}$$

Let $\alpha = e^{-T/T_{\text{cho}}}$ denotes the fidelity downgrade due to the noise in quantum memory. We can now calculate the maximum number of hops that an entangled pair can be distributed before the fidelity is lower than the threshold f_k.

$$L_k \leq \lfloor \frac{\log\left(\alpha \cdot \frac{4f_k-1}{3}\right)}{\log\left(\alpha \cdot \frac{4f_0-1}{3}\right)} \rfloor, \tag{2}$$

where L_k is the maximum number of hops for request r_k.

This result hints that the network scale is strictly limited because if the network has a path longer than L_k, the final fidelity of the distributed entangled pairs can not be guaranteed. To address this issue, we propose the idea of short-term destinations to enlarge the scale by conducting an entanglement distillation to raise the fidelity [45], [54].

Accordingly, for each request, it is possible to select a series of short-term destinations so that the distance between adjacent selected nodes does not exceed L_k. All entangled pairs are first distributed to those short-term destinations with the hop-by-hop entanglement distribution scheme to perform entanglement distillation. After that, the entangled pairs are further distributed to the next hops and eventually to the destination.

Algorithm 1 shows the entire hop-by-hop entanglement distribution procedure with consideration of fidelity. It is repeatedly executed until the destination is arrived. In Algorithm 1, u_{curr} represents the current node that entangled pair is shared between the source and u_{curr}; u_{next} is the next-hop node selected from a routing algorithm (described in Section V); and d_k' is the short-term destination. First, the algorithm checks the distance to the destinations. If the destination is far away from the current node (i.e., $dist(u_{\text{curr}}, u_k^d) > L_k$), a short-term destination d_k' is determined. Here, $dist(u,v)$ denotes the minimized number of hops between node u and v. We use the variable L_k' to denote the distance between the short-term destination and the current node. L_k' can be smaller than L_k to relax the path and enable rerouting within multiple candidate paths to avoid congestion. Otherwise, the entanglement will distribute to the final destination directly. Then, a routing algorithm will be performed to select the optimal next-hop u_{next} to the destination or short-term destination. A request will be sent to the next-hop node in order to preserve essential quantum resources, *e.g.,* quantum memories, and generate entangled pairs between u_{curr} and u_{next}. If the next-hop node can handle the entanglement distribution, it accepts this request, and the entangled pair is distributed to that next-hop node. Alternatively, if the next-hop node is in congestion (due to the lack of quantum memories), it will decline the request, and the existing entangled pairs will be dropped.

Algorithm 1 The Hop-by-Hop Entanglement Distribution Algorithm With Fidelity Guarantee

Input: The current node that the entanglement stays u_{curr}; The quantum network $G = (V, E)$; The request $r_k = (u_k^s, u_k^d, r_k)$;

$d'_k \leftarrow$ None;
while $u_{curr} \neq u_k^d$ **do**
  if $u_k^d \neq d'_k$ *and* $dist(u_{curr}, u_k^d) > L'_k$ **then**
    select a d'_k that $dist(u_{\text{curr}}, d'_k) = L'_k, (L'_k \leq L_k)$;
  else
    $d'_k \leftarrow u_k^d$;
  end
  $r'_k = (u_k^s, d'_k, f_k)$;
  $u_{\text{next}} \leftarrow$ RoutingAlgorithm$(u_{\text{curr}}, G, r'_k)$;
  if *result is not None* **then**
    generate a new entangled pair between u_{curr} and u_{next};
    perfrom entanglement swapping on u_{curr};
  else
    drop the entanglement;
    break;
  end
  $u_{\text{curr}} \leftarrow u_{\text{next}}$;
end

V. Q-DDCA: An Adaptive Multiple-Path Routing Algorithm

So far, we have developed a decentralized hop-by-hop entanglement distribution scheme with a fidelity guarantee for each request. One question left is how to select one optimal next-hop node u_{next} from all the neighbors to bring the best performance. Here, we present our Quantum Decentralized Dynamic Congestion Avoid routing algorithm (Q-DDCA) to leverage the nodes' congestion information to avoid congestion and select the optimal node. Meanwhile, we require the nodes only to know the topology of at most L_k hops to select the proper next-hop node.

The Q-DDCA routing algorithm should also consider the fidelity guarantee. That is, the routing result should keep the constraint that the total number of hops to the destination should be smaller than L_k, nor the fidelity will go below f_k. In Section V-A, we propose a strategy to meet the fidelity requirement. This strategy will first filter out all optional next-hop nodes that meet the fidelity requirement and form a candidate set S_C. After that, we design a utility function for the proposed routing algorithm as it critically affects the entanglement distribution rate. We present the utility function based on our analysis of the hop-by-hop entanglement distribution process in Section V-B. Finally, we illustrate the Q-DDCA algorithm in Section V-C.

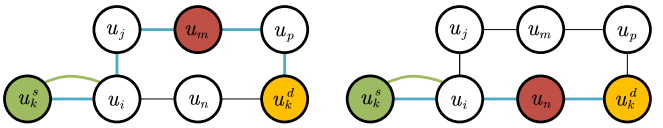
A. Selecting Feasible Next-Hops Based on Fidelity

Q-DDCA is a routing algorithm to decide the optimal next-hop node to achieve high throughput. In this section, we model the routing problem as follows. Consider the entangled pair of r_k is distributed between the source node u_k^s and a current node u_{curr}. The routing algorithm is performed on u_{curr} to decide the optimal next-hop node from all its neighbors. We assume that the minimum length (in the number of hops) from the source node to the short-term destination node is $L'_k (L'_k \leq L_k)$, and the length of the shortest path from the current node to the destination node is L_{remain}. Also, let L_{prec} denote the number of hops between the source and the current node. Again, the routing algorithm determines an optimal next-hop node to distribute entangled pairs. However, not all neighbor nodes are the proper next-hop nodes due to the fidelity consideration (the path will exceed L_k). Therefore, we design Q-DDCA to perform a pre-pruning on all neighbor nodes and choose a candidate node set S_C before further fine-grained routing determinations. We propose two pre-pruning constraints, hard and soft bound constraints, respectively, with different purposes.

The hard bound constraint is designed to meet the fidelity requirement. Q-DDCA selects the routing result of one of the most optimal next-hop nodes from all neighbor nodes. However, not all neighbors are feasible when considering the fidelity constraint. To guarantee fidelity, the path length between the source and the short-term destination must be smaller than L_k. For example, if L_k is 5 and the distance between the source node and the current node is 3, it suggests that the path selected from the current node to the (short-term) destination should be at most two hops left. In this case, some neighbor nodes are not feasible because the remaining path is longer than 2. Consequently, the hard bound is that the remaining path from the current node to the (short-term) destination should have at most $L_k - L_{\text{prec}}$ hops.

On the other hand, the soft bound constraint gives the routing algorithm some flexibility to trick off between a shorter path and a less congested path. As Q-DDCA is designed to avoid congestion, choosing a less congested node may benefit the entanglement distribution rate, even though it means a workaround in the topology. Since the distance between the source and short-term destination is L'_k, the hard bound requires that the path length is less than $L_k (L_k > L'_k)$. It allows us to choose longer but less congested paths. Considering that the minimum length between the current node u_{curr} and destination node $u_k^{d'}$ is L_{remain}, the soft bound is a relax assumption that all paths that the remaining length is shorter then $L_{\text{remain}} + \delta (\delta \leq L_k - L'_k)$ is accepted, where δ is the extension length per hop. On the contrary, if a path whose remaining length is larger than $L_{\text{remain}} + \delta$, it is not a candidate for the routing algorithm.

The relax variable, δ, affects the routing result and eventually influences the throughput. There are two kinds of inappropriate instances of δ that we want to avoid. Firstly, if δ is too large (almost near $L_k - L'_k$), there is an excellent chance that the previous node chooses a longer path too aggressively in order to avoid congestion. However, the chance for the subsequent nodes to avoid congestion will become smaller because the hard bound constraint must be met. In an extreme case shown in Fig. 3(a), if a node chooses to take a path of length L_k (and u_i choose u_j as the next-hop link since $\delta = 2$ allows to use this path), subsequent nodes will



(a) Aggressive Routing ($\delta \geq 2$). No room for later nodes to avoid congestion

(b) Conservative Routing ($\delta \leq 1$). Not able to make full use of multiple paths

Fig. 3. Two inappropriate instances with $L'_k = 3$ and $L_k = 5$. The red node is in congestion, and the bold blue line is the selected path. The entangled pair is between u_k^s and u_i currently.

ultimately lose the right to re-route to avoid congestion (on u_m). In the end, the route degenerates to the shortest path algorithm. On the contrary, if δ is too small ($\delta \approx 0$), all nodes always choose the shortest paths too conservatively. As a result, the advantages of adaptive routing will not be able to take effect, and the throughput will be likely to be limited by the bottleneck, as shown in Fig. 3(b).

To further evaluate δ, we generalize δ as the average extension length per hop. On the one hand, we have

$$L_{\text{total}} = L'_k + L_{\text{total}} \cdot \delta < L_k,$$

where L_{total} is the actual total number of hops from the source node to the short-term destination, and $L_{\text{total}} \cdot \delta$ is the total number of wake round hops or the extra length of rerouting path. This equation indicates that

$$\delta = \frac{L_{\text{total}} - L'_k}{L_{\text{total}}} \leq \frac{L_k - L'_k}{L'_k},$$

since $L'_k \leq L_{\text{total}} \leq L_k$. As L'_k is close to L_k, the δ is likely to be smaller than 1. In this case, we can assume that the extension length per hop is 1 with probability $p = \delta \approx \frac{L_k - L'_k}{L'_k}$. As a result, the current node accepts all paths where the remaining length from the current node to the destination node is L_{remain}. Besides, the current node has a chance to reroute to those paths with $L_{\text{remain}} + 1$ hops under the probability p.

Algorithm 2 shows selecting all candidate neighbor nodes and generating the candidate set. This algorithm is a sub-procedure of the entire routing algorithm of Q-DDCA. We introduce two bounds to calculate whether a certain next-hop satisfies the fidelity and relaxing requirements.

Algorithm 2 Calculate the Candidate Neighbor Node Set to Meed Fidelity Guarantee

Input: The current node u_{curr}; The destination u_k^d; the number of hops before the current node L_{prec}; the maximum hops that fit the fidelity requirement, L_k;
Output: The candidate set S_C;

```
S_C ← ∅;
for each neighbor u_i in N(u_curr) do
    soft_bound_result ← False;
    hard_bound_result ← False;
    L_i ← dict(u_i, u_k^d);
    // soft bound constraint
    if L_i = L_remain − 1 then
        // hops is not extended
        soft_bound_result ← True;
    end
    if L_i = L_remain then
        // hops is extended by 1
        soft_bound_result ← True, under probability
          p = L_k/L'_k − 1;
    end
    // hard bound constraint
    if L_i ≤ L_k − L_prec then
        hard_bound_result ← True;
    end
    if hard_bound_result ∧ soft_bound_result then
        S_C ← S_C ∪ {u_i};
    end
end
```

B. The Utility Function of the Routing Algorithm

After calculating the candidate set, all options in this set are supposed to meet the fidelity requirement. Now, Q-DDCA should select one optimal option. As we mentioned in the design goal, the primary goal of Q-DDCA is to achieve high throughput. Note that it is possible to extend our routing algorithm to use other metrics or utility functions. Such utility functions are also run on the nodes to determine the proper next-hop node. This paper focuses on the throughput, defined as the quantum network's entanglement distribution rate (EDR) or the number of entangled pairs distributed per second for all requests. We design the utility function based on a theoretical model and analyze the stochastic process of entanglement distribution on a path to achieve this goal.

We start by considering one entangled pair's distribution on a fixed n-hops path. Let $E(n)$ be the expected number of swapping operations before an entangled pair is distributed to the destination. $E(n)$ may be larger than L_k because the entangled pair may be dropped when the next-hop node has no quantum memories or the EPR generator fails to generate link-layer entangled pairs. For example, if a path has $n = 5$ nodes, an entangled pair is dropped after being distributed for 4 hops. After that, a new entangled pair begins to be distributed from the source node again. At this time, it succeeds in distributing an end-to-end entangled pair. In this case, the eventual number of entanglement swapping $E(n) = 9$.

Also, we assume the averaged time of a successful entanglement distribution T_{total} is no more than $(T + \tau) \cdot E(n)$, where τ is the time for entanglement swapping and the classic communication delay while T is the maximum time that a quantum memory can keep this entangled pair on a node before decoherence. Since there are at most wnd_k entangled pairs distributing at the same time for r_k, the distribution rate is

$$EDR_k = \frac{wnd_k}{T_{\text{total}}} \geq \frac{wnd_k}{(T + \tau) \cdot E(n)}. \tag{3}$$

Since both T and τ are the pre-determined attributions of the quantum network, given a fixed wnd_k, the routing algorithm should minimize $E(n)$. Let q be the probability that one node accepts the request of the coming entangled pairs, and $1 - q$

is the drop probability. We construct a discrete-time Markov chain to calculate $E(n)$. Assume the expected number of entanglement distribution for an l-length path is $E(l)$, and the entangled pair is now distributed to the next $(l+1)$-hop, we have the recursive equation:

$$E(l+1) = q \cdot [E(l)+1] + (1-q) \cdot [E(l)+1+E(l+1)], \quad and\ E(1) = \frac{1}{q},$$

where q is the success possibility of the qubit being distributed to the next-hop node. If this procedure succeeds, the final number of entanglement distribution is $E(l)+1$. Alternatively, if the entangled pair is dropped and a new entangled pair begins to distribute from the source node again, the expected number of entanglement swapping is $E(l)+1+E(l+1)$ with the possibility $1-q$. From the recursive formulas, we get

$$E(n) = \frac{E(n-1)+1}{q} = \sum_{i=1}^{n} (\frac{1}{q})^i. \tag{4}$$

From Eq. (4), we can observe the following two conclusions. On the one hand, the routing algorithm should prefer a shorter path. It eventually reduces the average number of entanglement swapping before a successful entanglement distribution, and EDR_k. It indicates that the shortest path algorithm based on Dijkstra's algorithm is preferred [55]. On the other hand, increasing the acceptance rate on each node is very effective because $E(n)$ is a polynomial of $1/q$. The network should reduce the drop rate and avoid retransmission. Therefore, we have proposed two approaches for Q-DDCA: multiple attempts and expected minimum hops, respectively.

1) Multiple Attempts of Distribution Request: The node congestion is the major reason for dropping a distributing entangled pair. When a quantum node selects an optimal next-hop node, it will ask the next-hop node to preserve quantum memories and generate a link-layer entangled pair. However, if the next-hop node has no free memory, it declines the distribution request, which causes a drop. To reduce the drop rate, Q-DDCA allows being performed for multiple rounds before the entangled pair decohered. In each round, a new next-hop node is selected, and it attempts to preserve resources on that node. The only constraint is that the total time for all those attempts must be lower than T to avoid decoherence. Specifically, when the current node, u_{curr}, tries to distribute an entangled pair to the next-hop, u_{next}, it will ask u_{next} to preserve one quantum memory for distribution. It could happen that u_{next} might be congested, and no memory is available for that request. In this case, u_{next} will deny the request. Otherwise, it will accept the request and wait for transmission. During this process, a classic communication happens, as shown in Fig. 2(b). We assume that such communication costs t time, which means there are at most $M = T/t$ attempts on each node. Multiple attempts help to reduce the drop rate. Formally, the acceptance probability of M attempts is $1-(1-q)^M$, where q is the acceptance rate in a multiple-attempts situation, i.e., the ratio of the number of times the node accepts to the total number of resource allocation application sent to it. In this case, the expected number of swapping before entanglement is distributed to the destination is $E(n)^M = \sum_{i=1}^{n} [1-(1-q)^M]^{-i}$. We find that more attempts will reduce the average number of hops before a successful entanglement distribution.

2) Use the Shortest Path: Since $E(n)$ shows that the best effort of improving the throughput is to use a path with fewer hops, we can adopt a proactive utility that prefers choosing a minimum hops path but still considers the network status. Q-DDCA requires that every node u_{curr} measure the acceptance rate of their neighbors, and this information is in the utility function. The u_{curr} should choose a path with a minimized expected number of hops to distribute entangled pairs. In the m-th attempt (in total M attempts), the expected number of hops is

$$(1-(1-q_i)^{M-m})\text{dist}(u_i, u_k^d) + (1-q_i)^{M-m}\text{dist}(u_k^s, u_k^d), \tag{5}$$

where q_i is the acceptance rate of neighbor u_i and $\text{dist}(u_i, u_k^d)$ is the distance between u_i and u_k^d, in terms of the number of hops, calculated by Dijkstra's algorithm. We only consider the acceptance rate of the next-hop node and assume that all subsequent transmissions are successful because we have no information on subsequent routing. Finally, the next-hop selection problem is formulated as

$$\begin{aligned} \min\ & f = \sum_{i=0}^{|S_C|} x_i \cdot [(1-(1-q_i)^{M-m})\text{dist}(u_i, u_k^d) \\ & (1-q_i)^{M-m}\text{dist}(u_k^s, u_k^d)], \\ \text{s.t.}\ & \sum_{i=0}^{|S_C|} x_i = 1, \\ & x_i = \{1,0\}, i = 1,2,\ldots,|S_C|, \\ & q_i \in [0,1], i = 1,2,\ldots,|S_C|, \\ & u_i \in S_C, \end{aligned} \tag{6}$$

where S_C is the candidate set that contains all possible options to meet the fidelity requirement, $|\cdot|$ is the cardinality of a set. Let x_i be a boolean value indicating whether to select $u_i \in S_C$ as the next-hop and q_i be the acceptance rate of u_i. The complexity of this problem is $\mathcal{O}(|C|)$, and one simple iteration can find the optimal option.

C. The Implementation of the Q-DDCA

The proposed algorithm should select the optimal next-hop node according to entangled pairs' fidelity, topology information, and node's congestion status. Q-DDCA can be divided into two steps. Firstly, it calculates a candidate node set S_C, where all nodes are feasible considering the fidelity requirement. In this step, we develop two bounds, including a soft and hard bound, to select all feasible nodes. Secondly, the optimal next-hop node should be further selected from S_C. We consider network congestion states and use the expected minimum number of hops as the utility function in this period.

Note that, the decisions made by Q-DDCA are mainly based on the following information: 1) Network topology. We assume that the topology is static for a period of time, and all network nodes share this information. Every node could

calculate the shortest path between any pair of nodes using Dijkstra's Algorithm. 2) Congestion information. As mentioned in Section V-B, the acceptance rate of one neighbor plays a vital role in determining the proper neighbor node to provide the optimal throughput, and all nodes must obtain this information. Remember that in our hop-by-hop distribution scheme, the node will require its next-hop node to accept the entanglement distribution (permit distributing entangled pairs via the next-hop node). The next-hop node may accept it or drop it. The acceptance rate is a ratio of the number of entanglement distributions a neighbor node accepts to the number of that total required. As a result, each node can statistically measure the acceptance rate of its neighbor nodes. Specifically, it counts the number of entanglement distributions sent to the neighbor node and the number the neighbor node accepts in a given time. Then, it can calculate the ratio as the acceptance rate. For example, if node u_i sends 5 requests to u_j and u_j accepts 3, the acceptance rate is 0.6. An extreme case to consider is that no request may be sent to some nodes in a period. In this case, the acceptance rate is unclear. In order to avoid this situation, we modify the original acceptance rate as $q_i = (acc_i + \epsilon)/(tol_i + \epsilon)$, where ϵ is a minimum value used to handle the situation that there is no request in this period. Because in this situation, q_i is now close to 1, indicating that this path is idle and is likely to accept the transmission. 3) The fidelity requirement f_k and the corresponding maximum number of hops L_k, which can be obtained and calculated by the source node.

Since the routing path is not pre-determined, some entangled pairs may be "lost" in a few worse cases, where it is better to be dropped and retransmitted than keep sending. Q-DDCA has a positive fast retransmission scheme by viewing "drops" as a virtual neighbor node. This drop node is virtually connected to both the current and destination nodes. The length between the "drop" node and the destination node is $2 \cdot \text{dist}(u_k^s, u_k^d)$, as we consider the expensive cost of retransmission and dropping. As a result, we add this penalty to minimize the occurrence of dropping by using twice the distance between the source and destination node as the length of the "drop" node. Only in this worse case will the node drop the entangled pairs and notify the source to redistribute a new one. Its q_{drop} equals 1 because it is always possible to discard an entangled pair.

Algorithm 3 shows how the Q-DDCA decides the optimal next-hop node at each node. The routing algorithm carries out at most M attempts for each entangled pair on a quantum node. Otherwise, the entangled pair will be dropped. In each attempt, Q-DDCA first selects a subset of neighbors S_C with two bounds we mentioned in Section V-A. Those bounds are used to guarantee that the fidelity will be at least the required f_k. Then, it selects an optimal next-hop node using the expected number of hops as the utility function. If the selected next-hop node is "drop", the entangled pair will be dropped positively. Otherwise, it will communicate with the selected next-hop to ensure that it has resources for entanglement distribution. If the next-hop accepts that request, the routing algorithm should return the next-hop node as an output. Alternatively, the next attempt begins.

Algorithm 3 Q-DDCA: The Decentralized Dynamic Congestion Avoid Routing Algorithm

Input: The current node that this entanglement is on, u_{curr}; the quantum network $G = (V, E)$; the request $r_k = (u_k^s, u_k^d, f_k)$); the maximum number of attempts $M = T/t$;

Output: The selected next-hop node u_{next}. If u_{next} is None, no next-hop is selected, and the entangled pair should be dropped.;

$m \leftarrow 1$, $u_{\text{next}} \leftarrow$ None;
for *Each neighbor* u_i *in* $N(u_{curr})$ **do**
  $L_i \leftarrow \text{dist}(u_i, u_k^d)$; // using Dijkstra algorithm.
end
$L_m \leftarrow \min_{1 \le i \le |N(u_{\text{curr}})|}\{L_i\}$, $L_{\text{drop}} \leftarrow 2 \cdot \text{dist}(u_k^s, u_k^d)$;
while $m \le M$ **do**
  $m = m + 1$;
  $S_C \leftarrow$ candidate set from Algorithm 2;
  find $u_i \in S_C$ that minimize Eq. (5);
  result $\leftarrow$ Request u_i to preserve a quantum memory;
  if *result is accept* **then**
    $u_{\text{next}} \leftarrow u_i$;
    break;
  end
end
if $dist(u_{next,u_k^d}) \ge L_{drop}$ **then**
  `// positively drop the entangled pair`
  $u_{\text{next}} \leftarrow$ None;
end

D. Discussion

This subsection discusses the correctness, efficiency, and fairness of Q-DDCA.

1) Correctness: Our routing algorithm should guarantee that every distributed entangled pair should meet the fidelity requirement. Our model designs the short-term destination to purify the entangled pairs on those nodes. We also use a relatively tight L_k to estimate the fidelity to prove robustness. We also set a hard bound in Q-DDCA so that the path length between two short-term destinations will not exceed L_k.

2) Efficiency: The network-flow theory allows us to evaluate the maximum transmission rate, though it does not guarantee fidelity. Like the network-flows theory, Q-DDCA utilizes multiple paths for entanglement distribution while satisfying the fidelity requirement. Besides, rerouting provides a higher transmission rate and robustness than the shortest path algorithm because it can predict congestion and detours to avoid it.

3) Fairness: Q-DDCA is fairer than the connection-oriented entanglement distribution schemes because it does not preserve any resource for predecessor requests so that the coming requests will not starve. Either, it does not allocate resources periodically like that in a centralized control scheme to avoid computation overhead. Q-DDCA allocates resources

preemptively, and all requests can use the network resource fairly and dynamically.

VI. Experiments and Evaluation

In this section, we conduct experiments to evaluate Q-DDCA's efficacy and fairness and compare it to other baseline algorithms, including the shortest path algorithm (SPA).

1) Evaluation Platform: The experiments are completed on SimQN [29], a system layer evaluation platform we built for simulating quantum networks. SimQN is a discrete-event driven simulator in Python 3, designed to assist quantum network investigation and evaluation easily. We have developed various physic models, including mixed-state qubit models, Bell state, Werner state, and isotropy state entangled pair models with varying granularity, functionality, and computational overhead. Also, unlike other platforms, we design the quantum nodes to be injectable with multiple applications to handle various complex behaviors. This paper is the first use case of SimQN, and in this evaluation, we build a quantum network of up to 50 nodes with random topology. We construct up to 10 arbitrary nodes to distribute entanglements spontaneously and asynchronously. We develop SimQN as an open-source project, and it is now released on GitHub (https://github.com/ertuil/simqn).

2) Competitors and Baselines: So far, we believe there is no other routing algorithm in a decentralized and asynchronous manner as we do in this paper. Q-Cast routing algorithm [38] is also a decentralized routing algorithm and assumes that nodes know local information about k-hop neighbors. Q-DDCA does not require nodes to have information about other nodes for free. Link status in at most 1-hop neighbors can be evaluated, but the classical communication overhead is also modeled. Besides, Q-DDCA guarantees the fidelity requirements and works fully asymmetric so that all nodes behave spontaneously without a time synchronization assumption.

As a result, we use an improved shortest path algorithm (SPA) as the baseline, where the shortest path is calculated based on Dijkstra's algorithm. To calculate the shortest path, SPA assumes that the network topology is static and available. In order to ensure the fidelity requirements, we also require that the path length does not exceed L_k. Like Q-DDCA, we assume it has at most M attempts before dropping an entanglement. In particular, when $M = 1$, SPA and Q-DDCA behave the same. It is because the utility function of Q-DDCA degenerates into the distance in this case.

3) Parameters Setting: We adopt a randomly generated topology to evaluate the routing algorithms in general scenarios. For most of our experiments, the network topology has 50 nodes. As for multiple requests, we randomly select the source and destination nodes. The quantum links that connect two neighbor nodes can be used to generate entangled pairs with a probability of 0.1, and the initial fidelity of the generated entangled pairs is $f_0 = 0.99$. Although, as far as we know, there are no mature quantum memory devices currently, recent work has demonstrated the feasibility of a quantum memory, and one qubit can be stored with high fidelity for minutes [56]. In these experiments, we set the decoherence time of a quantum memory to T_{coh} to be 5s and T to be 0.5s. As a result, the maximum fidelity decoherence is $\alpha = e^{-\frac{T}{T_{\text{coh}}}} \approx 0.9$. Also, we assume the final entangled pairs' fidelity requirement is above $f_k = 0.7$, and we can calculate $L_k = 6$. The time delay of the classic communication for resource allocation t is set to be $50\ ms$. Therefore, the maximum number of attempts is $M = T/t = 10$. In most experiments, we simulate the entanglement distribution for 10s, as it is found that EDR has become stable at this time.

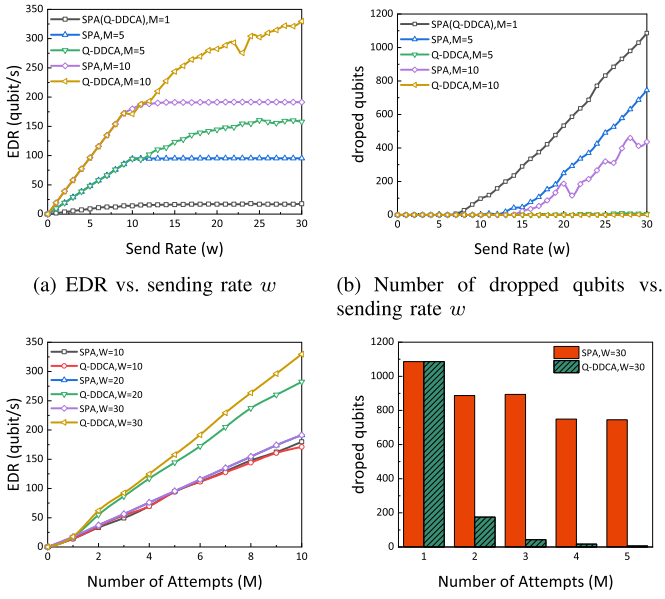
The wnd_k is the concurrency window that controls the number of entangled pairs distributed by a source node at the same time. In the experiments, we set wnd_k differently according to the path length. It is because the resources of the path depend on the path length, and a longer path requires a larger wnd_k to utilize the quantum memories. Specially, we use $wnd_k = w \cdot \text{dist}(u_k^s, u_k^d)$ for request r_k, where w is equivalent to resources used on one quantum node. For example, when the path has 3 hops with 10 memories on each interface (20 memories for the repeaters and 10 memories for the endpoint nodes), wnd_k is $3 \cdot 10 = 30$ to averagely use every memory on that path and reach the maximum sending rate before congestion. If the path has 6 hops, wnd_k can be increased to 60 to occupy all the quantum memories and therefore reach the highest throughput. This assumption is proved by the experiment result later, as it shows that if every request shares the same w, Q-DDCA can ensure that all requests use the quantum memories on a quantum node fairly.

A. Evaluation for One Request

We start from one single request scenario to evaluate whether the routing algorithm can fully use the path capacity. In our experiments, each node has 20 quantum memories, and the EPR generators produce entangled pairs at 1000 Hz. We examine the throughput of the two algorithms under different w (windows per hop) and M (number of attempts) settings.

Fig. 4 shows the EDR and dropped qubits of this request. We vary M to be 5 or 10. Then, we conduct simulations under different window size parameters w, as shown in Fig. 4(a). We can observe that the EDR increases simultaneously in SPA and Q-DDCA when w is relatively small. However, when w is over about 10, the EDR reaches the single path capacity, and SPA begins to drop entangled pairs, as we can see in Fig. 4(b), and EDR in SPA no longer increases in Fig. 4(a). On the contrary, EDR still increases in Q-DDCA. Meanwhile, the number of dropped entangled pairs remains low in Q-DDCA. It is because Q-DDCA detects the congestion on the original path and begins to utilize alternative paths. As a result, EDR in Q-DDCA grows smoothly. As a classic result, EDR is 329.5 qubits/s in Q-DDCA, 72.15% more than the EDR (191.4 qubits/s) of SPA when $M = 10$ and $w = 30$. Similarly, EDR in Q-DDCA (157.9 qubit/s) is 107.49% more than in SPA (76.1 qubit/s) when $M = 5$. Q-DDCA performs better than SPA, especially when the window per-hop w is relatively high, or the network is congested.

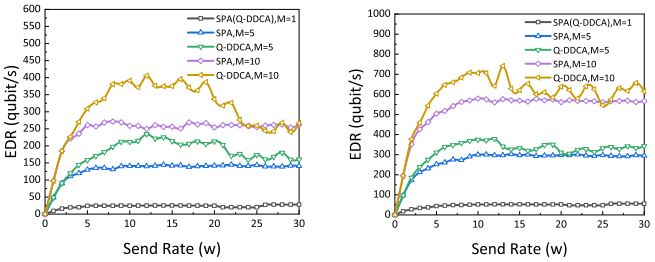
We also explored the influence of the number of attempts, as shown by Fig. 4(c) and Fig. 4(d). We find that with the



(a) EDR vs. sending rate w (b) Number of dropped qubits vs. sending rate w

(c) EDR vs. number of attempts M (d) Number of dropped qubits vs. M

Fig. 4. The EDR and the number of dropped qubits in single-request experiments.



(a) 5 requests (b) 10 requests

Fig. 5. The total EDR vs. sending rate w in multiple-requests experiments.

increase of M, EDR increases, as shown in Fig. 4(c). The EDR increases larger in Q-DDCA compared to the SPA algorithm. As for the drop rate, we find that Q-DDCA reduces the drop rate significantly when M goes large thanks to the congestion avoidance rerouting, as shown in Fig. 4(d). Overall, Q-DDCA can reach a higher EDR and drop fewer entangled pairs. For example, in the case of $w = 30$ and $M = 5$, SPA distributes 955 entangled pairs with 745 dropped entangled pairs. In comparison, Q-DDCA distributes 1579 entangled pairs in 10 seconds, with only 6 being dropped.

We specifically investigate how the congestion avoid function works in the Q-DDCA algorithm when $M = 5$ and $w = 30$. In this game, the request is to distribute entangled pairs from node u_{18} to node u_7. SPA uses the shortest path $\{u_{18}, u_{15}, u_3, u_7\}$, and distributes 955 entangled pairs in total. As for Q-DDCA distributes 1579 entangled pairs in 10 seconds, which is 165.34% compared to SPA. It also uses the same major path to distribute 393. However, the other 1186 entangled pairs are distributed from 16 different paths. This indicates that Q-DDCA's congestion avoidance schemes work fine. Moreover, the second contributed path is $\{u_{18}, u_{43}, u_{16}, u_9, u_7\}$ which distribute 211 entangled pairs. The least contributed path is $\{u_{18}, u_{31}, u_{13}, u_3, u_7\}$ and distributes only 1 entangled pairs. All those paths are shorter than L_k, and the distributed entangled pairs satisfy the fidelity requirement.

1) Discussion: The Q-DDCA algorithm has a higher EDR because it can adapt to the network congestion state to reroute or utilize multiple paths to significantly reduce congestion and drop rate. Further investigation finds that no path length is more than L_k. Consequently, Q-DDCA makes more use of resources in a single request scenario and breaks through the capacity of a single path.

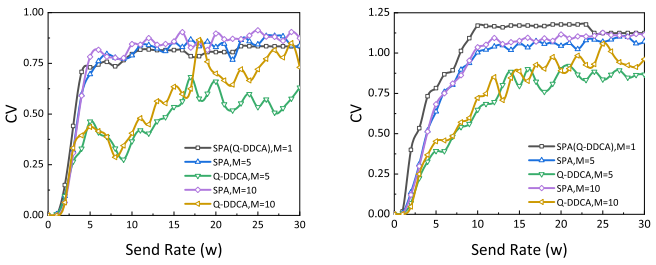
B. Evaluation for Concurrency Multiple Requests

We also investigate the multiple requests scenarios and explore the total EDR of multiple requests, the drop rate, and the fairness between requests.

In these experiments, 5 or 10 requests distribute entangled pairs concurrently and independently. The source node and the destination node are chosen randomly. Other parameters are the same with the single request scenario. The total EDR for 5 requests is shown in Fig. 5(b), and the total EDR for 10 requests is shown in Fig. 5(a) respectively. The result shows a new pattern that the total EDR decreases when w is relatively large. It is mainly because of network-wide congestion. Though Q-DDCA can avoid congestion on one path, it can not avoid enormous congestion when all potential paths are in congestion. For example, in the 5 requests experiment, the total EDR reaches the maximum when the window is $w = 12$. In this case, the EDR of Q-DDCA is 66.81% or 62.42% higher than SPA when $M = 5$ or $M = 10$, respectively. Similar to the single request scenario, we observe that as M increases, the total EDR also increases. Overall, Q-DDCA can achieve a higher EDR than SPA. The downgrade of the total EDR when w is relatively large indicates that a more efficient congestion avoidance algorithm should be studied for each request. Such a congestion avoidance algorithm controls the sending rate w based on the network congestion level like the classic TCP protocol.

We also count the dropped entangled pairs, showing that Q-DDCA has a lower drop rate than SPA. For example, when the sending rate ω is set to 12 and the number of attempts M is 10. If there are 5 requests distributing entanglements concurrently, Q-DDCA drops 394 qubits in total while SPA drops 1421. If the number of requests increases to 10, the corresponding result is 2109 with Q-DDCA and 3001 with SPA. Considering that the number of distributed entangled pairs differs between the two algorithms, the drop rate gap will be more significant.

We further examine the EDR for each request and the fairness. As we randomly pick the source and destination nodes, the final throughput of multiple requests should not be identically equal. Thus, instead of standard deviation, we use the coefficient of variation (CV) to measure the EDR difference, which is the ratio of the standard deviation and the average EDR. Fig. 6(a) and Fig. 6(b) show the CV in the 5 (or 10) requests experiment, respectively. The results show that Q-DDCA maintains a more average EDR for each



(a) 5 requests (b) 10 requests

Fig. 6. Fairness (CV) vs. sending rate w in multiple-requests experiments.

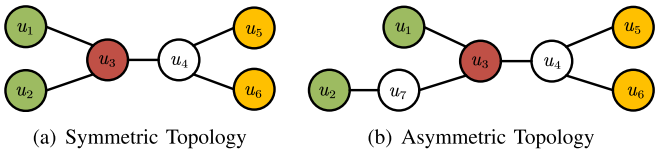
request. In the 5-requests experiment with $M = 10$ and $w = 30$, CV is 0.7300 with Q-DDCA while 0.8737 with SPA. In this experiment, the EDR for 5 requests is 875, 100, 977, 214, and 497 when using Q-DDCA. SPA results are 962, 80, 971, 26, and 584. Notice that SPA has two requests that distribute entangled pairs at a slow rate. Compared to these two algorithms, the EDR of the slowest request increases from 26 (SPA) to 214 (Q-DDCA), while the EDR of the faster flows does not increase significantly or even slightly decrease. In most cases, the result obtained by the Q-DDCA algorithm is more balanced and fairer for multiple requests.

1) Discussion: The multiple-requests experiment illustrates that Q-DDCA can still achieve a higher total EDR and lower drop rate than SPA, especially when a proper window per-hop w is set. Further investigation finds that multiple requests have a smaller EDR difference in Q-DDCA, which indicates much more fairness among multiple concurrency requests.

C. Microscopic Investigation on Resource Allocation

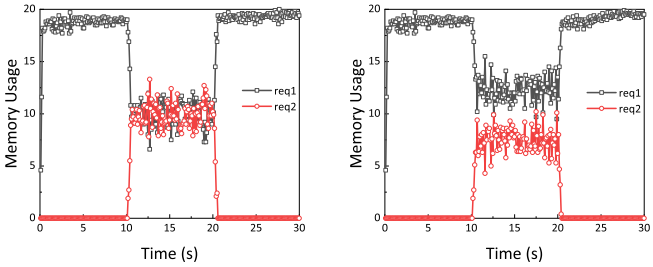
The experiments above study fairness from a global view, and we further investigate how multiple requests use the network resource, especially quantum memories. Ideally, multiple requests should use the network resources equally if they share the same link. Thus, this experiment shows how the resources are allocated to multiple requests, and we also investigate the factors that affect network fairness. However, using a random topology with many requests is not convenient. In this subsection, we reduce the network scale to a fixed dumbbell topology and mainly focus on two requests, as shown in Fig. 7. In the symmetric dumbbell topology, shown as Fig. 7(a), there are six nodes and two requests. u_1 will distribute entangled pairs to u_5 (*req1*) while u_2 will distribute entangled pairs to u_6 (*req2*). Nevertheless, it is not enough because it is common when two paths have different lengths. Still, in this case, both requests should achieve a similar EDR when other parameters are the same. Therefore, we build the asymmetric dumbbell topology by inserting an extra node between u_2 and u_3, as shown in Fig. 7(b).

To illustrate the detailed status of quantum memory used between two requests, we extend the simulation time to 30s. *req1* starts distributing entangled pairs at 0s and stops at 30s, while *req2* distributes entangled pairs from 10s to 20s. Also, we fix $M = 10$, $w = 10$, and 20 quantum memories on every quantum node. Leverage the monitor provided by SimQN, and we can collect the usage of quantum memories on u_3 every



(a) Symmetric Topology (b) Asymmetric Topology

Fig. 7. Dumbbell topologies for fairness evaluation between 2 requests.



(a) $w_1 = 10$, $w_2 = 10$ (b) $w_1 = 10$, $w_2 = 5$

Fig. 8. Memory usage of two requests at the node u_3.

0.01s. Fig. 7(c) shows the experimental result. At the first and the final 10 seconds, *req1* fully uses the 20 memories on u_3. When *req2* begins to send, *req1* occupies 9.268 memories and *req2* uses 10.676 on average. Eventually, *req1* distributed 4673 entanglements while *req2* distributes 969 entanglements in total. The result in the middle 10 seconds demonstrates that Q-DDCA can adjust quantum memory usage spontaneously and rapidly, enabling multiple requests to use network resources fairly. In the second round, we reduce the window of *req2* to 5 and run the same simulation. In this experiment, *req1* uses 12.490 memories, and *req2* uses 7.289 memories, respectively, when both requests are distributing entangled pairs at the same time as shown in Fig. 7(d). It illustrates the memory occupation status on u_3. The result indicates that the memory occupation is close to the window per-hop ratio $w_1/w_2 = 2$.

1) Discussion: This evaluation shows that the multiple requests' sending rate affects the allocation of resources on the shared links. If the requests set their sending rate proportional to the path length, the network will reach fairness, and the resources will be allocated equally to the multiple requests. Otherwise, a larger sending rate will lead the request to occupy more network sources. The results verify our assumption in the experiment's parameters setting.

VII. Conclusion

The critical factor affecting the performance of the quantum network is how to allocate quantum memories and other resources for multiple requests. Centralized control and global time synchronization models are often used to address this issue. However, such a model may bring massive communication and computing overhead and is unsuitable for a large-scale quantum internet. In this paper, we proposed a decentralized hop-by-hop entanglement distribution schema and a decentralized routing algorithm (Q-DDCA) that can probe and avoid network congestion. The main feature of Q-DDCA is the utility function of the expected shortest path according to the network congestion status and the fidelity

bounds that guarantee the fidelity requirement. Those features enable Q-DDCA to adjust the path according to the available network resources and meet the fidelity requirements. We evaluated Q-DDCA using the discrete-event simulator SimQN, built by our research team, and the results show that it could use multiple paths for entanglement distribution and reach a high entanglement distribution rate (107.49% higher than the shortest path algorithm as a classic result). Meanwhile, it can reduce the drop rate, adapt to new requests automatically, and maintain fairness in using network resources.

REFERENCES

[1] S. Pirandola and S. L. Braunstein, "Physics: Unite to build a quantum internet," *Nature*, vol. 532, no. 7598, pp. 169–171, Apr. 2016.

[2] S. Lloyd, J. H. Shapiro, F. N. C. Wong, P. Kumar, S. M. Shahriar, and H. P. Yuen, "Infrastructure for the quantum internet," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 34, no. 5, pp. 9–20, Oct. 2004.

[3] S. Wehner, D. Elkouss, and R. Hanson, "Quantum internet: A vision for the road ahead," *Science*, vol. 362, no. 6412, Oct. 2018, Art. no. eaam9288.

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

[5] C. H. Bennett and G. Brassard, "Quantum cryptography: Public key distribution and coin tossing," in *Proc. IEEE Int. Conf. Comput., Syst., Signal Process. (ICCSSP)*, Dec. 1984, pp. 175–179.

[6] A. K. Ekert, "Quantum cryptography based on Bell's theorem," *Phys. Rev. Lett.*, vol. 67, no. 6, p. 661, 1991.

[7] M. Wang et al., "A segment-based multipath distribution method in partially-trusted relay quantum networks," *IEEE Commun. Mag.*, early access, Mar. 7, 2023, doi: 10.1109/MCOM.010.2200672.

[8] M. Caleffi, A. S. Cacciapuoti, and G. Bianchi, "Quantum internet: From communication to distributed computing!" in *Proc. 5th ACM Int. Conf. Nanosc. Comput. Commun.*, Sep. 2018, pp. 1–4.

[9] L. Gyongyosi and S. Imre, "A survey on quantum computing technology," *Comput. Sci. Rev.*, vol. 31, pp. 51–71, Feb. 2019.

[10] Z. Li et al., "Building a large-scale and wide-area quantum internet based on an OSI-alike model," *China Commun.*, vol. 18, no. 10, pp. 1–14, Oct. 2021.

[11] A. W. Harrow and A. Montanaro, "Quantum computational supremacy," *Nature*, vol. 549, no. 7671, pp. 203–209, Sep. 2017.

[12] F. Arute et al., "Quantum supremacy using a programmable superconducting processor," *Nature*, vol. 574, no. 7779, pp. 505–510, 2019.

[13] P. Komar et al., "A quantum network of clocks," *Nature Phys.*, vol. 10, no. 8, pp. 582–587, 2014.

[14] H. J. Kimble, "The quantum internet," *Nature*, vol. 453, pp. 1023–1030, Jun. 2008.

[15] A. Pirker and W. Dür, "A quantum network stack and protocols for reliable entanglement-based networks," *New J. Phys.*, vol. 21, no. 3, Mar. 2019, Art. no. 033003.

[16] W. Kozlowski and S. Wehner, "Towards large-scale quantum networks," in *Proc. 6th Annu. ACM Int. Conf. Nanosc. Comput. Commun.*, Sep. 2019, pp. 1–7.

[17] L. Gyongyosi, S. Imre, and H. V. Nguyen, "A survey on quantum channel capacities," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 2, pp. 1149–1205, 2nd Quart., 2018.

[18] J. Preskill, "Quantum computing in the NISQ era and beyond," *Quantum*, vol. 2, p. 79, Aug. 2018.

[19] A. I. Lvovsky, B. C. Sanders, and W. Tittel, "Optical quantum memory," *Nature Photon.*, vol. 3, no. 12, pp. 706–714, Dec. 2009.

[20] A. Dahlberg et al., "A link layer protocol for quantum networks," in *Proc. Annu. Conf. ACM Special Interest Group Data Commun. Appl., Technol., Archit., Protocols Comput. Commun. (SIGCOMM)*. ACM, 2019, pp. 159–173.

[21] W. Dai, T. Peng, and M. Z. Win, "Optimal remote entanglement distribution," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 3, pp. 540–556, Mar. 2020.

[22] S. Brand, T. Coopmans, and D. Elkouss, "Efficient computation of the waiting time and fidelity in quantum repeater chains," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 3, pp. 619–639, Mar. 2020.

[23] S. Pirandola, "End-to-end capacities of a quantum communication network," *Commun. Phys.*, vol. 2, no. 1, pp. 1–10, May 2019.

[24] M. Pant et al., "Routing entanglement in the quantum internet," *NPJ Quantum Inf.*, vol. 5, no. 1, pp. 1–9, Mar. 2019.

[25] A. Sharma, V. Ojha, and S. K. Lenka, "Security of entanglement based version of BB84 protocol for quantum cryptography," in *Proc. 3rd Int. Conf. Comput. Sci. Inf. Technol.*, Jul. 2010, pp. 615–619.

[26] C. Di Franco and D. Ballester, "Optimal path for a quantum teleportation protocol in entangled networks," *Phys. Rev. A, Gen. Phys.*, vol. 85, no. 1, Jan. 2012, Art. no. 010303.

[27] A. S. Cacciapuoti, M. Caleffi, R. Van Meter, and L. Hanzo, "When entanglement meets classical communications: Quantum teleportation for the quantum internet," *IEEE Trans. Commun.*, vol. 68, no. 6, pp. 3808–3833, Jun. 2020.

[28] C. Li, T. Li, Y.-X. Liu, and P. Cappellaro, "Effective routing design for remote entanglement generation on quantum networks," *NPJ Quantum Inf.*, vol. 7, no. 1, pp. 1–12, Jan. 2021.

[29] L. Chen et al., "SimQN: A network-layer simulator for the quantum network investigation," *IEEE Netw.*, early access, Jan. 23, 2023, doi: 10.1109/MNET.130.2200481.

[30] L. Chen et al., "A heuristic remote entanglement distribution algorithm on memory-limited quantum paths," *IEEE Trans. Commun.*, vol. 70, no. 11, pp. 7491–7504, Nov. 2022.

[31] Z. Li, K. Xue, J. Li, N. Yu, D. S. L. Wei, and R. Li, "Connection-oriented and connectionless remote entanglement distribution strategies in quantum networks," *IEEE Netw.*, vol. 36, no. 6, pp. 150–156, Nov. 2022.

[32] J. Li et al., "Fidelity-guaranteed entanglement routing in quantum networks," *IEEE Trans. Commun.*, vol. 70, no. 10, pp. 6748–6763, Oct. 2022.

[33] R. Van Meter, T. Satoh, T. D. Ladd, W. J. Munro, and K. Nemoto, "Path selection for quantum repeater networks," *Netw. Sci.*, vol. 3, nos. 1–4, pp. 82–95, Dec. 2013.

[34] K. Chakraborty, D. Elkouss, B. Rijsman, and S. Wehner, "Entanglement distribution in a quantum network: A multicommodity flow-based approach," *IEEE Trans. Quantum Eng.*, vol. 1, pp. 1–21, 2020.

[35] L. Gyongyosi and S. Imre, "Decentralized base-graph routing for the quantum internet," *Phys. Rev. A, Gen. Phys.*, vol. 98, no. 2, Aug. 2018, Art. no. 022310.

[36] A. Aguado et al., "Secure NFV orchestration over an SDN-controlled optical network with time-shared quantum key distribution resources," *J. Lightw. Technol.*, vol. 35, no. 8, pp. 1357–1362, Apr. 15, 2017.

[37] W. Yu, B. Zhao, and Z. Yan, "Software defined quantum key distribution network," in *Proc. 3rd IEEE Int. Conf. Comput. Commun. (ICCC)*, Dec. 2017, pp. 1293–1297.

[38] S. Shi and C. Qian, "Concurrent entanglement routing for quantum networks: Model and designs," in *Proc. Annu. Conf. ACM Special Interest Group Data Commun. Appl., Technol., Archit., Protocols Comput. Commun. (SIGCOMM)*, 2020, pp. 62–75.

[39] E. Chitambar, D. Leung, L. Mančinska, M. Ozols, and A. Winter, "Everything you always wanted to know about LOCC (but were afraid to ask)," *Commun. Math. Phys.*, vol. 328, no. 1, pp. 303–326, May 2014.

[40] H.-J. Briegel, W. Dür, J. I. Cirac, and P. Zoller, "Quantum repeaters: The role of imperfect local operations in quantum communication," *Phys. Rev. Lett.*, vol. 81, no. 26, pp. 5932–5935, Dec. 1998.

[41] N. Sangouard, C. Simon, H. de Riedmatten, and N. Gisin, "Quantum repeaters based on atomic ensembles and linear optics," *Rev. Modern Phys.*, vol. 83, no. 1, pp. 33–80, Mar. 2011.

[42] W. J. Munro, K. Azuma, K. Tamaki, and K. Nemoto, "Inside quantum repeaters," *IEEE J. Sel. Topics Quantum Electron.*, vol. 21, no. 3, pp. 78–90, May 2015.

[43] S. Pirandola, R. Laurenza, C. Ottaviani, and L. Banchi, "Fundamental limits of repeaterless quantum communications," *Nature Commun.*, vol. 8, no. 1, Apr. 2017, Art. no. 15043.

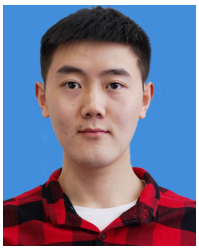
[44] J. Illiano, M. Caleffi, A. Manzalini, and A. S. Cacciapuoti, "Quantum internet protocol stack: A comprehensive survey," *Comput. Netw.*, vol. 213, Aug. 2022, Art. no. 109092.

[45] R. Van Meter, T. D. Ladd, W. J. Munro, and K. Nemoto, "System design for a long-line quantum repeater," *IEEE/ACM Trans. Netw.*, vol. 17, no. 3, pp. 1002–1013, Jun. 2009.

[46] M. Pompili et al., "Experimental demonstration of entanglement delivery using a quantum network stack," *NPJ Quantum Inf.*, vol. 8, no. 1, p. 121, Oct. 2022.

[47] S. Vasudevan, M. Adler, D. Goeckel, and D. Towsley, "Efficient algorithms for neighbor discovery in wireless networks," *IEEE/ACM Trans. Netw.*, vol. 21, no. 1, pp. 69–83, Feb. 2013.

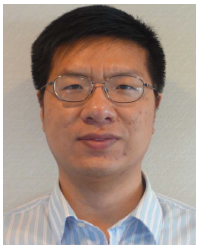
[48] L. J. Stephenson et al., "High-rate, high-fidelity entanglement of qubits across an elementary quantum network," *Phys. Rev. Lett.*, vol. 124, no. 11, Mar. 2020, Art. no. 110501.
[49] N. Kalb et al., "Entanglement distillation between solid-state quantum network nodes," *Science*, vol. 356, no. 6341, pp. 928–932, Jun. 2017.
[50] C. H. Bennett, D. P. DiVincenzo, J. A. Smolin, and W. K. Wootters, "Mixed-state entanglement and quantum error correction," *Phys. Rev. A, Gen. Phys.*, vol. 54, no. 5, pp. 3824–3851, Nov. 1996.
[51] R. Reichle et al., "Experimental purification of two-atom entanglement," *Nature*, vol. 443, no. 7113, pp. 838–841, Oct. 2006.
[52] T. E. Northup and R. Blatt, "Quantum information transfer using photons," *Nature Photon.*, vol. 8, no. 5, pp. 356–363, May 2014.
[53] R. F. Werner, "Quantum states with Einstein-Podolsky-Rosen correlations admitting a hidden-variable model," *Phys. Rev. A, Gen. Phys.*, vol. 40, no. 8, pp. 4277–4281, Oct. 1989.
[54] W. Dür and H. J. Briegel, "Entanglement purification and quantum error correction," *Rep. Prog. Phys.*, vol. 70, no. 8, pp. 1381–1424, Aug. 2007.
[55] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numerische Math.*, vol. 1, no. 1, pp. 269–271, Dec. 1959.
[56] P. Wang et al., "Single ion qubit with estimated coherence time exceeding one hour," *Nature Commun.*, vol. 12, no. 1, pp. 1–8, Jan. 2021.



Lutong Chen (Graduate Student Member, IEEE) received the bachelor's degree from the School of Cyber Science and Technology, University of Science and Technology of China, in 2020, where he is currently pursuing the Ph.D. degree. His research interests include quantum networking and network security.

Kaiping Xue (Senior Member, IEEE) received the bachelor's degree from the Department of Information Security, University of Science and Technology of China (USTC), in 2003, and the Ph.D. degree from the Department of Electronic Engineering and Information Science (EEIS), USTC, in 2007. From May 2012 to May 2013, he was a Post-Doctoral Researcher with the Department of Electrical and Computer Engineering, University of Florida. Currently, he is a Professor with the School of Cyber Science and Technology, USTC, where he is also the Director of the Network and Information Center. His research interests include next-generation internet architecture design, transmission optimization, and network security. He is an IET Fellow. He serves on the Editorial Board of several journals, including the IEEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING, the IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS, and the IEEE TRANSACTIONS ON NETWORK AND SERVICE MANAGEMENT. He has also served as a (Lead) Guest Editor for many reputed journals/magazines, including IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS, *IEEE Communications Magazine*, and IEEE NETWORK.

Jian Li (Member, IEEE) received the bachelor's degree from the Department of Electronics and Information Engineering, Anhui University, in 2015, and the Ph.D. degree from the Department of Electronic Engineering and Information Science (EEIS), University of Science and Technology of China (USTC), in 2020. From December 2020 to December 2022, he was a Post-Doctoral Researcher with the School of Cyber Science and Technology, USTC, where he is currently an Associate Researcher. His research interests include wireless networks, next-generation internet, and quantum networks.



Ruidong Li (Senior Member, IEEE) received the bachelor's degree in engineering from Zhejiang University, China, in 2001, and the Ph.D. degree in engineering from the University of Tsukuba in 2008. He is an Associate Professor with the College of Science and Engineering, Kanazawa University, Japan. Before joining Kanazawa University, he was a Senior Researcher with the Network System Research Institute, National Institute of Information and Communications Technology (NICT). His current research interests include future networks, big data networking, blockchain, information-centric networks, the Internet of Things, network security, wireless networks, and quantum internet. He is a member of IEICE.

Nenghai Yu received the B.S. degree from the Nanjing University of Posts and Telecommunications, Nanjing, China, in 1987, the M.E. degree from Tsinghua University, Beijing, China, in 1992, and the Ph.D. degree from the Department of Electronic Engineering and Information Science (EEIS), University of Science and Technology of China (USTC), Hefei, China, in 2004. Currently, he is a Professor with the School of Cyber Security and the School of Information Science and Technology, USTC. His research interests include multimedia security, multimedia information retrieval, video processing, and information hiding.

Qibin Sun (Fellow, IEEE) received the Ph.D. degree from the Department of Electronic Engineering and Information Science (EEIS), University of Science and Technology of China (USTC), in 1997. He is currently a Professor with the School of Cyber Science and Technology, USTC. He has published more than 120 papers in international journals and conferences. His research interests include multimedia security and network intelligence and security.

Jun Lu received the bachelor's degree from Southeast University in 1985 and the master's degree from the Department of Electronic Engineering and Information Science (EEIS), University of Science and Technology of China (USTC), in 1988. Currently, he is a Professor with the Department of EEIS, School of Cyber Science and Technology, USTC. His research interests include theoretical research and system development in the field of integrated electronic information systems and network and information security. He is an Academician of the Chinese Academy of Engineering (CAE).