Entangled Photon Source Pooling and Entanglement Distribution for Quantum Networks

Junyuan Shi*†, Yangming Zhao*†‡, Dong Zhang§, Bingheng Yan§, Chen Tian* and Chunming Qiao¶
*State Key Laboratory of Novel Software Technology, Nanjing University
† School of Intelligent Software and Engineering, Nanjing University Suzhou Campus
‡ Hefei National Laboratory § Inspur Data Co.,Ltd. ¶ University at Buffalo

2026 IEEE/ACM International Symposium on Quality of Service (IWQoS) | 979-8-3195-1703-6/26/$31.00 ©2026 IEEE | DOI: 10.1109/IWQoS70441.2026.11661033

***Abstract*—Entanglement distribution is a key problem in quantum networks. Conventional works on entanglement distribution either ignored that Entangled Photon Sources (EPSes) are precious resources or organized EPSes in an unscalable way. In this work, we propose an EPS Pooling and Entanglement Distribution (EPED) approach to efficiently utilize EPSes. The salient features of EPED include (i) EPSes are organized as several EPS pools; (ii) each EPS pool distributes entanglements only in its coverage area; and (iii) EPSes are dynamically assigned to create different entanglement links.**

There are two challenges in EPED: (i) when deploying EPS pools, we do not have any demand information; and (ii) we have to calculate the Entanglement Path (EP) for each demand as quickly as possible. To tackle these challenges, we design an algorithm with a constant approximation ratio to minimize the number of EPS pools required, and determine the number of EPSes provisioned to each pool using a primal-dual technique. By preparing candidate EPs for each demand, we solve the online EP selection problem for tens of demands in a network with hundreds of nodes in 6.5 ms. Extensive simulations show that EPED can save up to 8% EPS pools and enhance network throughput by up to 48%.

I. Introduction

Entanglement distribution plays a key role in both quantum computing and quantum communications. For distributed quantum computing, when two quantum bits (called qubits, which are usually implemented as photons) that have to go through the same binary gate are not on the same Quantum Computer (QC), we could perform such a remote quantum gate using a technique called cat-entangler [1, 2]. For quantum communications, a qubit is easy to get lost if we directly send it from one QC to another via a transmission medium due to its low energy. To address this issue, we leverage teleportation [3] to deliver quantum states instead of sending physical qubits. Both cat-entangler and teleportation need to consume entanglements, which are also referred to as Entanglement Connections (EC), established between the two corresponding QCs. An EC will be destroyed after it is used to perform a remote gate or teleport a qubit. Thus, we would try to concurrently establish as many ECs as possible. The number of ECs that can be concurrently established is also referred to as *network throughput* [4]–[6].

It is the simplest way to establish an EC between two QCs that we first generate a pair of entangled qubits with an Entangled Photon Source (EPS) and send each of them to one of these two QCs [7, 8]. However, it is difficult to deliver a qubit via a long fiber and free-space channels cannot be used in the daytime. Accordingly, this method is not suitable for a large-scale network that operates day and night. To address this scalability issue, many existing works assumed that there are enough EPSes placed at the middle of each physical link [9, 10] or at each quantum node [11]. By doing so, we can create Entanglement Links (ELs) over each physical link, and then stitch properly selected ELs via quantum swapping to establish end-to-end ECs. These works inherently assumed that quantum memory is the most precious quantum resource. However, in practice, EPSes, each of which costs tens of hundreds of dollars [12], are also very limited, as it is not affordable to statically deploy EPSes for every link to ensure that there are always enough EPSes to generate entangled qubits concurrently.

In this work, we propose an approach called EPED (EPS Pooling and Entanglement Distribution) to manage EPSes and distribute entanglements. In EPED, we organize all available EPSes into several pools as Micius [8]. By leveraging all-optical switching capability as in [13], each EPS pool can dynamically provision its EPSes to create ELs between different node pairs inside its coverage area. When establishing an EC between nodes that are far away from each other, *e.g.*, not covered by the same EPS pool, we first find an Entanglement Path (EP) consisting of a sequence of nodes, such that every two adjacent nodes are covered by the same EPS pool. Then, we create ELs between every two adjacent nodes and stitch these ELs together via quantum swapping to establish the end-to-end EC. Since EPED dynamically provisions EPSes in every pool to create ELs according to existing demands and the related EPs, it can significantly increase the EPS utilization compared with statically deploying each EPS for creating ELs over a specific physical link.

There are two challenging stages in EPED, *i.e.*, the offline network deployment stage and the online entanglement distribution stage. In the network deployment stage, given the locations of all quantum nodes, the number of available EPSes,

The work of Yangming Zhao was supported in part by the National Natural Science Foundation of China under Grants 62572233 and 62272428, in part by Quantum Science and Technology-National Science and Technology Major Project under grant 2021ZD0300705, in part by Fundamental and Interdisciplinary Disciplines Breakthrough Plan of the Ministry of Education of China under grant JYB2025XDXM118, and in part by the "111 Center" under grant B26023.

Yangming Zhao is the corresponding author.

and the coverage radius of each EPS pool, we determine where to deploy EPS pools and how many EPSes to deploy in each pool. In this stage, the key challenge lies in the uncertainty of demand information. In the online entanglement distribution stage, based on the demand information, we compute the EP for each demand and the number of ELs to be created between different node pairs for each EPS pool. The algorithms used in this stage should have low time complexity since a long algorithm running time will prolong the duration of time slots and reduce the effective network throughput.

To deploy an EPED-based quantum network, we propose a $17+\epsilon$ approximation algorithm to compute the positions of EPS pools in order to minimize the number of EPS pools built to ensure that ECs can be established between any node pairs, which in turn minimizes the network infrastructure cost. Then, by considering all potential demands, we compute the number of EPSes deployed in each EPS pool in order to maximize network throughput. In the entanglement distribution stage, we prepare some candidate EPs for every potential demand and calculate the probability of using each candidate EP. When demands arrive, we randomly select an EP for each of them according to this probability. By doing so, we can determine the EPs for 36 demands in a network with 100 nodes in 6.5 ms. Extensive simulations demonstrate that EPED can save 8% of the EPS pools required to build compared with a greedy method, and it can improve the network throughput by up to 31%, 48%, and 16%, respectively, compared with baselines such as shortest-path based entanglement routing, even EPS provisioning, and least workload first EL creation approaches.

The main contributions of this paper are:

- Propose a new network architecture to improve quantum network scalability and fully utilize EPSes (Section III)
- Design efficient algorithms to minimize infrastructure cost and improve network throughput (Sections IV & V)
- Conduct extensive simulations to demonstrate the superior performance of EPED (Section VI)

II. Background

In this section, we introduce preliminary knowledge about quantum networks, present the system model we study, and justify the assumptions we make in this work.

A. Preliminary

Vanilla entanglement distribution. To distribute an entanglement between Alice and Bob, we generate a pair of entangled qubits (called ebits) using an EPS and send each of these two ebits to Alice and Bob, respectively. Once both Alice and Bob receive one ebit, an entanglement is distributed. In other words, an Entanglement Connection (EC) is established between Alice and Bob.

Entanglement distribution between remote ends. When Alice and Bob are far away from each other, it would be difficult to establish an EC with the vanilla entanglement distribution since Alice or Bob may be far away from the EPS and an ebit would get lost when it is flying in the transmission medium. To solve this problem, we will stitch multiple Entanglement Links (ELs), which are physically entanglements created between two nodes close by each other, together, to establish an EC between Alice and Bob.

Specifically, we first find a physical path (called an Entanglement Path, EP) from Alice to Bob. Then, we create an EL between every two adjacent nodes along this EP. As a result, every intermediate node (so-called repeater) will hold two ebits, which belong to different ebit pairs. By performing quantum swapping, a repeater can stitch its two adjacent ELs. When all involved repeaters complete the quantum swapping, an EC between Alice and Bob is established.

Quantum resource requirement. To create an EL (u, v), we need an Entangled Photon Source (EPS) to generate a pair of ebits. To deliver each of the generated ebits to node u and node v, respectively, a quantum channel from the EPS to node u and a channel from the EPS to node v are also required. In addition, we need one unit of quantum memory at node u and node v, respectively, to keep an ebit. As a result, to establish an EC between Alice and Bob through an H-hop EP, we need H EPSes to generate H ebit pairs. From each of these EPSes, two quantum channels, each of which is connected to one end of the EL to be created by its ebit pair, are required. Two units of quantum memory are required at every intermediate repeater to keep ebits, while only one unit is needed on Alice's and Bob's sides, respectively.

Limited quantum resources. According to current techniques, we have only limited quantum resources available in a quantum network. For example, each quantum node, such as a repeater, has only tens of quantum memory units. In addition, an EPS costs tens of hundreds of dollars [12], and it may not be affordable to widely deploy them in a quantum network. Consider that a quantum channel would be a wavelength, and there would be more than one hundred available wavelengths in a fiber, in this work, we will focus on fully utilizing the available EPSes and quantum memory units in order to concurrently establish as many required ECs as possible.

B. System model and assumptions

Network model. In this work, we study photonic quantum networks with some quantum nodes (which can function as both end nodes and intermediate repeaters) whose locations have already been determined (in order to provide services to customers). Each quantum node carries a given number of quantum memory units. EPSes will be organized into several EPS pools. Due to engineering issues, we assume that there are some candidate positions to deploy EPS pools. In order to save infrastructure costs incurred by deploying EPS pools, we will minimize the number of EPS pools that must be built as long as all potential demands can be served (though some of them may be delayed).

It should be noted that in practice, we may have to deploy a given number of EPS pools in a network with the objective of not only serving all potential demands, but also maximizing network throughput. However, this tells a parallel story to this work, and we will leave it to our future work.

As in most previous works [5, 6, 14, 15], we assume a quantum network is operated in a time-slotted manner with a central controller. In each time slot, we collect all existing demands, determine the EP for each of them, create ELs, and perform quantum swapping to establish ECs.

EPS model. In order to fully utilize available EPSes, we leverage an all-optical switching architecture as in [13] to dynamically send ebits to different quantum nodes. However, due to signal attenuation, an ebit can be delivered for only a certain distance, r. Hereafter, if the distance between node u and the EPS pool w is less than r, we say node u is in EPS pool w's *coverage area* or node u is covered by EPS pool w. We deploy out-of-band channels between an EPS pool and every quantum node in its coverage area. Consequently, an EPS pool (*i.e.*, its EPSes) can create ELs between every node pair in its coverage area. For brevity, if both node u and node v are covered by the same EPS pool, we say there is a *quantum link* between them even though there is nodes u and v are not physically connected. To ensure there is always an EPS pooling scheme such that we can establish ECs for all potential demands in the network, we make the following assumption:

Assumption 1. *For every pair of nodes with potential demands between them, there is a path consisting of ordered quantum nodes such that every two adjacent nodes lie in the coverage area of at least one identical EPS pool.*

Entanglement distribution model. Due to signal attenuation, the success probability of delivering an ebit from an EPS pool w to a node u depends on the distance between them. In general, this success probability can be formulated as [4, 5]

$$q(w,u) = e^{-\alpha l_{wu}} \tag{1}$$

where α is a fixed parameter associated with the channel quality and l_{wu} is the length of the transmission medium between EPS pool w and node u. When EPS pool w needs to create an EL between node u and node v, both ebits must be successfully delivered. Thus, the success probability of creating an EL between node u and node v with an ebit pair generated by EPS pool w is

$$p_{uv}^{w} = q(w,u)q(w,v) = e^{-\alpha(l_{wu}+l_{wv})} \tag{2}$$

Demand model. A demand is specified by the two ends of the EC to be established. For a potential demand, we first prepare some candidate EPs for it based on its two ends. In every time slot, the central controller collects existing demands and determines (i) for which demands we will try to establish ECs; (ii) the EP used by each selected demand; and (iii) how many ELs an EPS pool we will try to create over each quantum link. For each demand, we will establish at most one EC in a time slot. For the node pairs requiring multiple ECs, they have to submit multiple demands concurrently.

C. Previous Works and Motivation

This work focuses on concurrently distributing as many end-to-end entanglements, *i.e.*, ECs, as possible in quantum networks. Most existing works in this area try to fully utilize quantum resources by optimizing the EP selection (also referred to as entanglement routing). In the early stage, researchers started by solving the entanglement routing problem on specific topologies, such as diamond topologies [16], ring or sphere topologies [17], star topologies [18], chain topologies [19], *etc.*. After that, Q-CAST [4], REPS [5], and Multi-R [6] considered entanglement routing on general topologies. In addition, EFiRAP [20] and Q-LEAP [21] leveraged purification to guaranty the fidelity of established ECs. In order to further increase the number of ECs that can be concurrently established, SEE [22] and [23] proposed segment-based entanglement routing methods. [24] built Quantum Overlay Networks (QONs) to efficiently utilize created ELs. All the above works assumed that quantum memory was the most precious resource and ignored that EPSes were also very limited in a quantum network.

ERSA [13] is perhaps the first work that considers limited EPSes. It organized all EPSes into a pool and dynamically assigned EPSes to create ELs over different physical links. Specifically, to create an EL over physical link (u,v) with ERSA, an EPS in the pool must generate a pair of ebits and send them to one end (say node u) of link (u,v). Then, node u will send one of these two ebits to node v through the quantum channel over link l. It is problematic to organize EPSes following the way described above since we can directly send two ebits to the two ends of an EC even if the two ends are not physically connected.

To address the ebit distribution issue in ERSA, [7] organized all EPSes into a single pool and directly distributed ebits generated by EPSes to the ends of ECs required to establish. To maximize the number of ECs that can be established concurrently, [7] optimized the position of the unique EPS pool and the number of EPSes assigned to establish different ECs. In a large scale quantum network, an ebit cannot be delivered to a node that is far away from the EPS pool.

LightER [11] allocated EPSes with some quantum nodes and assumed every ebit generated by a specific EPS can be delivered to one of its adjacent nodes. In this way, the success probability of creating ELs can be improved. However, when existing demands vary, some EPSes cannot be fully utilized since they would not be used to establish any ECs, which significantly reduces network throughput.

In this work, we organize EPSes into multiple pools, and the EPSes in each pool are dynamically assigned to create ELs in a limited area. By doing so, we can not only ensure the success probability of creating ELs, but also fully utilize EPSes.

III. EPED OVERVIEW

The proposed network architecture of EPED is shown in Fig. 1. Such a network consists of several EPS pools and many quantum nodes. Each EPS pool distributes entanglements (*i.e.*, creates ELs) between node pairs inside its coverage area. One EL can be created by different EPS pools as long as these EPS pools cover both ends. To establish an EC between two

TABLE I
NOTATION LIST

Parameters	Description
$\mathcal{C}$	The set of candidate EPS pools.
$\mathcal{W}$	The set of selected EPS pools.
$\mathcal{W}(e)$	The set of selected EPS pools that cover link e.
$\mathcal{E}$	The set of quantum links.
$\mathcal{N}$	The set of quantum nodes.
m_u	The number of quantum memory units carried by $u \in \mathcal{N}$.
$\mathcal{D}$	The set of all potential demands.
$E(d)$	The set of demand d's two ends.
$\mathcal{C}(w)$	The set of links over which ELs are created by EPS pool w.
$\mathcal{I}_d$	A binary parameter to indicate if demand d exists.
$\mathcal{T}$	The set of demand combinations that may exist.
$\mathcal{P}_d$	The set of candidate EPs for demand d.
$\mathcal{L}_w$	The set of quantum links over which ELs should be created by EPS pool $w \in \mathcal{W}$.
G	The number of available EPSes.
r	The coverage radius of an EPS pool (or EPS).
$d(u,v)$	The distance between u and v, $u, v \in \mathcal{N} \cup \mathcal{W}$.
q_e^w	The success probability of creating an EL over $e \in \mathcal{E}$ by an EPS in pool w.

Variables	Description
g_w	The number of EPSes allocated to EPS pool w.
y_d^p	The fraction of demand d will be fulfilled through its candidate EP p.
x_e^w	The number of ELs that EPS pool w will try to create over quantum link e.
θ	A factor to scale demand $d \in \mathcal{D}$ such that there are enough quantum resources to accommodate all demands.

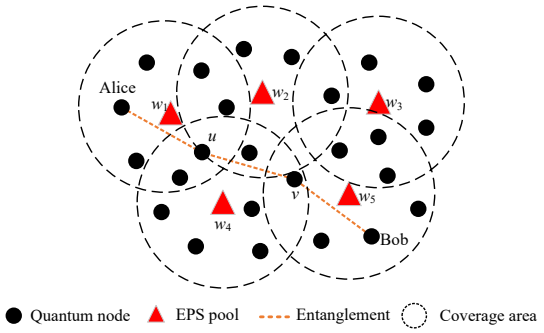


Fig. 1. The network structure of EPED.

nodes, *e.g.*, Alice and Bob, that the same EPS pool may not cover them, we first find an EP between them such that every two adjacent nodes are covered by the same EPS pool (*e.g.*, Alice$\rightarrow$ u $\rightarrow$ v $\rightarrow$Bob). Then, we can create ELs between every two adjacent nodes along this EP and establish the end-to-end EC by stitching ELs together via quantum swapping.

To build the proposed quantum networks, EPED takes care of both network deployment and online entanglement distribution. For network deployment, EPED minimizes the number of EPS pools to build as long as we can distribute entanglements between every pair of quantum nodes. This is to save the infrastructure cost of building a quantum network. In addition, EPED also assigns all available EPSes to the selected EPS pools to concurrently serve as many demands as possible. Given the locations of EPS pools and the number of EPSes in each pool, EPED prepares some candidate EPs for every potential demands based on their two ends.

When the quantum network starts running, EPED will enter the online entanglement distribution stage. Specifically, given a batch of demands, EPED determines which demands we will try to fulfill and through which candidate EP we will establish an EC for each selected demand. In this stage, EPED pursues two goals: (i) calculate the EP for each selected demand as soon as possible to reduce the duration of time slots; and (ii) concurrently serve as many demands as possible, *i.e.*, maximize network throughput.

In the following two sections, we will discuss the details of all algorithms we used for offline network deployment and online entanglement distribution, respectively.

IV. OFFLINE NETWORK DEPLOYMENT

In this section, we present the algorithms used to deploy networks in EPED. In Section IV-A, we first discuss how to select the positions to deploy EPS pools. Then, we present how to deploy available EPSes into EPS pools in Section IV-B. At last, we describe the method of preparing candidate EPs for each potential demand in Section IV-C.

A. Selection Positions for EPS Pool Deployment

Considering the policy and engineering issues, we assume that there are several candidate positions to deploy EPS pools (these positions are in the set $\mathcal{C}$). Then, we have to select the minimum number of positions to deploy EPS pools in order to save infrastructure costs. To ensure that we can establish ECs between every node pair in the network, the selected EPS pools (in $\mathcal{W}$) should satisfy the following two conditions: (i) between every pair of nodes, there is a feasible EP consisting of quantum links, and (ii) every selected EPS pool covers at least two nodes. The first condition ensures that we can establish ECs between every node pair, and the second condition is due to the fact that an EPS pool covering only one node is useless.

In EPED, we propose an EPS Pool Selection (EPSPS) algorithm to determine the locations for deploying EPS pools. This algorithm is summarized in Algorithm 1. In the EPSPS algorithm, we first scale the distance between nodes such that the coverage radius of each EPS pool becomes 1 (Line 1), and then leverage a Discrete Unit Disk Covering (DUDC) algorithm [25] to select some positions for deploying EPS pools such that every node is covered by at least one EPS pool (Line 2). So far, we cannot ensure that ECs can be established between every node pair. Accordingly, from Line 3 to 11, we sequentially select more EPS pools to address this issue. More specifically, in Line 3, we construct an auxiliary graph to check the network connectivity. If a quantum node is covered by two different EPS pools, this node can be used as a repeater to connect a pair of nodes, each of which is covered by one of these two EPS pools, respectively. Accordingly, as long as the auxiliary graph constructed in Line 3 is connected, we can find an EP to establish ECs for every node pair in the network. If this is not the case, $\boldsymbol{G}$ must have multiple partitions, and there must be a candidate EPS pool that can create ELs between two nodes, each of which is covered by selected EPS pools in different partitions. Otherwise, the Assumption 1 cannot hold. In Line 5, we find such two partitions, say $\boldsymbol{C}$ and $\boldsymbol{C}'$, and involve one more EPS pool to connect them (Lines 7–10). Given $\boldsymbol{C}$ and $\boldsymbol{C}'$, there may be multiple EPS pools that can

Algorithm 1: EPS Pool Selection (EPSPS) Algorithm

Input: The set of candidate EPS pools $\mathcal{C}$, the set of quantum nodes $\mathcal{N}$, the coverage radius of an EPS pool r

1 Scale the distance measurement such that the coverage radius of an EPS pool becomes 1.
2 Solve the DUDC using the algorithm in [25] to ensure every node is covered by one EPS pool in $\mathcal{W}$.
3 Construct an auxiliary graph $\boldsymbol{G}$ such that a selected EPS pool $w \in \mathcal{W}$ in represented by a vertex v in $\boldsymbol{G}$. There is an arc a between v_1 and v_2 in $\boldsymbol{G}$ if w_1 and w_2 cover at least one identical node.
4 **while** $\boldsymbol{G}$ *is not a connected graph* **do**
5 Find out two seperated partitions, *e.g.*, $\boldsymbol{C}$ and $\boldsymbol{C}'$, in $\boldsymbol{G}$ such that there exist v (covered by a $w \in \mathcal{C}$) and v' (covered by a $w' \in \mathcal{C}'$) that are covered by the same EPS pool.
6 $D_{min} \leftarrow 2r$, $w^* \leftarrow \emptyset$.
7 **foreach** *Node pair* v *and* v' *satisfying the above condition* **do**
8 $w \leftarrow \text{argmin}_{w \text{ covers } v \text{ and } v'}\{d(v,w) + d(v',w)\}$
9 **if** $d(v,w) + d(v',w) < D_{min}$ **then**
10 $D_{min} \leftarrow d(v,w) + d(v',w)$, $w^* \leftarrow w$
11 $\mathcal{W} \leftarrow \mathcal{W} + w^*$ and update $\boldsymbol{G}$ accordingly.
12 Remove all EPS pools that do not contribute to ensure solution feasibility.
13 **Return** $\mathcal{W}$

connect them. Among all these EPS pools, we choose the one that can achieve the largest success probability of creating an EL between two nodes that are covered by $\boldsymbol{C}$ and $\boldsymbol{C}'$, respectively.

Till now, the EPSPS algorithm has derived a feasible solution for EPS pool deployment. However, after involving more EPS pools to ensure solution feasibility (Lines 4–11), some selected EPS pools may be redundant. Accordingly, in Line 12, we remove all redundant EPS pools that do not contribute to the solution feasibility to further reduce infrastructure costs.

The performance of the EPSPS algorithm can be guarantied by the following theorem

Theorem 1. *The approximation ratio of the EPSPS algorithm is 17+ϵ.*

Proof. Suppose O^*_{DUDC} is the minimum number of EPS pools required in order to ensure that every node is covered by at least one EPS pool and O^*_{EPSPS} is the minimum number of EPS pools we need. Apparently, $O^*_{DUDC} \leq O^*_{EPSPS}$ since every feasible solution of the latter problem is also a feasible solution of the former one.

With the algorithm in [25], we can ensure every node is covered by at least one EPS pool with at most $9O^*_{DUDC} + \epsilon$ EPS pools. These EPS pools divide a network into at most $9O^*_{DUDC} + \epsilon$ partitions. Accordingly, we need at most another $8O^*_{DUDC} + \epsilon$ EPS pools to connect them. As a result, the EPSPS algorithm will derive a solution using at most $17O^*_{DUDC} + \epsilon \leq 17O^*_{EPSPS} + \epsilon$ EPS pools. This indicates that the approximation ratio of the EPSPS algorithm is 17+ϵ. □

It should be noted that the approximation ratio stated in Theorem 1 is derived based on the worst case. In practice, the EPSPS algorithm may derive a solution that is close to the optimal one. For example, $9O^*_{DUDC} + \epsilon$ EPS pools cannot divide the network into $9O^*_{DUDC} + \epsilon$ partitions since it is very likely that some of the quantum nodes are covered by the same EPS pool. In addition, some EPS pools are redundant and will be removed in Line 12 of Algorithm 2, which is not accounted for in the above proof.

B. EPS Assignment

Given the positions of all EPS pools, we can easily obtain all quantum links, *i.e.*, $\mathcal{E}$, and the success probability of creating an EL over each quantum link by EPS pool w, *i.e.*, q_e^w for all $e \in \mathcal{E}$ and $w \in \mathcal{W}$. To determine how many EPSes we have to deploy in each pool, we first prepare some candidate EPs for every *potential* demand (rather than the exact demands in the network). More specifically, we set the cost of each quantum link e as $-\sum_{w \in \mathcal{W}(e)} \log(q_e^w/|\mathcal{W}(e)|)$ and then leverage Yen's algorithm [26] to calculate the k-shortest paths for every potential demand. In EPED, the candidate EPs include all paths that are at most two hops longer than the shortest one. For demand d, all its candidate EPs are in the set $\mathcal{P}_d$. Then, we solve the following problem which takes into consideration all possible combinations of potential demands:

$$\text{Lexmax} \qquad \theta_d \tag{3}$$

subject to:

$$\theta_d \leq \sum_{p \in \mathcal{P}_d} y_d^p \leq 1 \quad \forall d \in \mathcal{D} \tag{3a}$$

$$\sum_{d \in \mathcal{D}} \sum_{p \in \mathcal{P}_d : e \in p} y_d^p I_d \leq \sum_{w \in \mathcal{W}} q_e^w x_e^w \quad \forall \{I_d\} \in \mathcal{T}, e \in \mathcal{E} \tag{3b}$$

$$\sum_{e \in \mathcal{C}(w)} x_e^w \leq g_w \quad \forall w \in \mathcal{W} \tag{3c}$$

$$\sum_{w \in \mathcal{W}} g_w \leq G \tag{3d}$$

$$\sum_{w \in \mathcal{W}} \sum_{e \in \mathcal{E} : (u,v) = e} x_e^w \leq m_u \quad \forall u \in \mathcal{N} \tag{3e}$$

$$x_e^w, g_w \in N \quad \forall d \in \mathcal{D}, w \in \mathcal{W}, e \in \mathcal{E} \tag{3f}$$

The objective is to maximize the scaling factor such that we have enough quantum resources to accommodate all scaled demands. Constraint (3a) limits the fraction of each demand that can be served. Constraint (3b) indicates that we have to create enough ELs over each link $e \in \mathcal{E}$ to establish ECs. In this constraint, $q_e^w x_e^w$ is the expected number of ELs that can be created over link e by EPS pool w. It is worth noting that in this constraint, we take *ALL* possible demand combinations into consideration, rather than considering a specific set of demands. This is because in the offline network deployment stage, we do not know which demands will be in the network

simultaneously. Constraints (3c) and (3d) limit the number of ELs that can be created by each EPS pool and the total number of EPSes that can be provisioned to EPS pools, respectively. Constraint (3e) is enforced by the number of quantum memory units carried by each quantum node. Constraint (3f) states that the number of ELs we attempt to create and the number of EPSes we deploy in an EPS pool are non-negative integers.

y_d^p, which indicates if an EC will be established through the EP p for demand d, should be a binary variable, however, we do not enforce this limitation. On the one hand, in the above formulation, demands may be scaled. When y_d^p is limited to binary, the scaled demands (with the volume of $\theta_d y_d^p$) should satisfy the resource capacity limitations. It is equivalent to relaxing the binary constraint enforced on y_d^p. On the other hand, without enforcing the binary constraint, y_d^p shows the "fraction" of demand d that will be fulfilled through EP p, and it hints at the probability that we will establish an EC for demand d through EP p in EPED.

Problem (3) is difficult to solve due to two issues. Firstly, constraint (3b) includes an infinite number (or at least a huge number) of constraints since it should hold for all possible demand combinations. In addition, some of the variables are integers, which makes Problem (3) be non-convex.

To address the first issue, a primal-dual based method can help reduce the number of constraints. At first, we consider the maximum number of ELs we have to create over quantum link e when demands in the network change. It can be derived by solving the following problem:

$$\text{maximize} \quad \sum_{d\in\mathcal{D}} \sum_{p\in\mathcal{P}_d:e\in p} y_d^p I_d \tag{4}$$

subject to:

$$\sum_{d:u\in E(d)} I_d \le m_u \quad \forall u \in \mathcal{N} \tag{4a}$$

$$I_d \le 1 \quad \forall d \in \mathcal{D} \tag{4b}$$

In the above formulation, the EP selection (*i.e.*, y_d^p) is a parameter. Given EPs, we calculate the maximum number of ELs that we have to create over each link when demand combination changes under the limitation of quantum memory. Based on the above fact, we have the following theorem:

Theorem 2. *It will not change the optimal objective value of* (3) *by substituting* (3b) *with the following inequality group*

$$\begin{cases} \sum_{u\in\mathcal{N}} \alpha(u) m_u + \sum_{d\in\mathcal{D}} \beta(d) \le \sum_{w\in\mathcal{W}} q_e^w x_e^w \\ \sum_{u\in E(d)} \alpha(u) + \beta(d) \ge \sum_{p\in\mathcal{P}_d:e\in p} y_d^p \quad \forall d \in \mathcal{D} \\ \alpha(u) \ge 0, \beta(d) \ge 0 \end{cases} \tag{5}$$

Proof. Let $\alpha(u)$ and $\beta(d)$ be the dual variable associated with (4a) and (4b), respectively. Then, the duality of (4) is

$$\text{minimize} \quad \sum_{u\in\mathcal{N}} \alpha(u) m_u + \sum_{d\in\mathcal{D}} \beta(d) \tag{6}$$

subject to:

$$\sum_{u\in E(d)} \alpha(u) + \beta(d) \ge \sum_{p\in\mathcal{P}_d:e\in p} y_d^p \quad \forall d \in \mathcal{D} \tag{6a}$$

$$\alpha(u) \ge 0, \beta(d) \ge 0 \tag{6b}$$

Algorithm 2: EPS Assignment (EPSA) Algorithm

Input: Formulation of (3)

1 Substitute (3b) with (5) and relax all integer constraints.
2 Solve the relaxed (3) and derive the solution $\hat{g}_w$ and $\hat{x}_e^w$.
3 $g_w \leftarrow \lceil \hat{g}_w \rceil$ for all $w \in \mathcal{W}$.
4 **while** $\sum_{w\in\mathcal{W}} g_w > G$ **do**
5 $\quad w \leftarrow \text{argmax}_{w\in\mathcal{W}:g_w\neq 1}\{g_w - \sum_{e\in\mathcal{C}(w)} \hat{x}_e^w\}$.
6 $\quad g_w \leftarrow g_w - 1$.
7 **Return** $\{g_w\}$

Due to the strong duality theorem [27], the objective values of (6) and (4) are identical. Accordingly, $\sum_{d\in\mathcal{D}} \sum_{p\in\mathcal{P}_d:e\in p} y_d^p I_d \le \sum_{w\in\mathcal{W}} q_e^w x_e^w$ indicates $\sum_{u\in\mathcal{N}} \alpha(u) + \sum_{d\in\mathcal{D}} \beta(d) \le \sum_{w\in\mathcal{W}} q_e^w x_e^w$. Combining with (6$a$) and (6$b$), we know (5) is a necessary condition to ensure constraint (3b).

On the other hand, when (5) holds, we have

$$\sum_{d\in\mathcal{D}} \sum_{p\in\mathcal{P}_d:e\in p} y_d^p I_d \le \sum_{d\in\mathcal{D}} (\sum_{u\in E(d)} \alpha(u) + \beta(d)) I_d$$
$$\le \sum_{u\in\mathcal{N}} \alpha(u) m_u + \sum_{d\in\mathcal{D}} \beta(d) \le \sum_{w\in\mathcal{W}} q_e^w x_e^w$$

The first inequality is due to the second inequality in (5), the second inequality is owned to (4a) and (4b), and the last inequality is the first inequality in (5). This shows that (5) is a sufficient condition to ensure constraint (3b). □

With Theorem 2, we can substitute constraint (3b) with a finite number of linear constraints. Now, we will discuss how to address the second issue, *i.e.*, the integer variables in (3). In general, we leverage a relaxation and rounding based method to obtain a near-optimal solution to an integer linear programming model. Following this idea, we propose a rounding based EPS Assignment (EPSA) Algorithm shown in Algorithm 2. In this algorithm, we relax the integer constraint in problem (3) and solve it (Lines 1–2). Based on the solution, $\hat{g}_w$, we assign $\lceil \hat{g}_w \rceil$ EPSes to each selected EPS pool $w \in \mathcal{W}$ (Line 3). According to (3d), we have $\sum_{w\in\mathcal{W}} \hat{g}_w = G$. Therefore, $\sum_{w\in\mathcal{W}} \lceil \hat{g}_w \rceil \ge G$. It means that we do not have enough EPSes if we always assign $\lceil \hat{g}_w \rceil$ EPSes to EPS pool w. To address this issue, we sequentially remove one EPS from the pool that has the most redundant EPS resources, *i.e.*, the EPS pool w associated with the largest $g_w - \sum_{e\in\mathcal{C}(w)} \hat{x}_w^e$ value (Lines 4–6). The only exception is that if an EPS pool has only one EPS, we cannot remove it in order to maintain network connectivity.

C. Candidate EPs Preparation

Once the EPS provisioning is determined, we can solve the following problem, which optimizes the probability of using different EPs, to maximize the scaling factor to serve an arbitrary potential set of demands:

$$\text{Lexmax} \quad \theta_d \tag{7}$$

Algorithm 3: Online Entanglement Routing (OER) Algorithm

Input: Solutions to problem (7) $\hat{y}_d^p$ and $\hat{x}_e^w$, the set of existing demand Ω

1 Initialize $y_d^p \leftarrow 0$, $x_e^w \leftarrow 0$, $\Phi \leftarrow \Omega$
2 **foreach** *Demand* $d \in \Omega$ **do**
3 Randomly select path $p \in \mathcal{P}_d$ with a probability of $\hat{y}_p^d$.
4 **if** *A path* p *is selected* **then**
5 $y_d^p \leftarrow 1$, $\Omega \leftarrow \Omega - d$
6 **foreach** e *along* p **do**
7 $w_e \leftarrow \text{argmax}_w\{\hat{x}_e^w - x_e^w \mid \sum_{e' \in C(w), e' \neq e} \lceil \frac{x_{e'}^w}{q_{e'}^w} \rceil + \frac{x_e^w + 1}{q_e^w} \leq g_w\}$
8 **if** $w_e = \emptyset$ **then**
9 $y_d^p \leftarrow 0$, $\Omega \leftarrow \Omega + d$, **break**
10 **if** $y_d^p = 1$ **then**
11 **foreach** e *along* p **do**
12 $x_e^{w_e} \leftarrow x_e^{w_e} + 1$
13 **foreach** *Demand* $d \in \Omega$ **do**
14 **foreach** $p \in \mathcal{P}_d$ **do**
15 **if** *Enough resources for another EC along* p **then**
16 $y_d^p \leftarrow 1$, **break**
17 **Return** $\{y_d^p\}$

subject to: $(3a)$–$(3c)$, $(3e)$

$$y_d^p \leq 1 \quad \forall d \in \mathcal{D}, w \in \mathcal{W} \tag{7f}$$

Different from (3), g_w is a parameter in (7) instead of a variable since it has been determined based on the EPSA algorithm. y_d^p can be treated as the probability of fulfilling demand d through EP p. Then, we have enough resources to create the required ELs over each quantum link in expectation.

V. Online Entanglement Distribution

In the online entanglement distribution stage, we have to establish ECs for some of existing demands (if not all, depending on resource limitations). In this section, we will present how to online select an EP for each existing demand.

If demand d establishes its EC going through the path p with the probability of $\hat{y}_d^p$, *i.e.*, the solution of (7), there will be enough resources to create ELs over every quantum link in expectation. Based on this observation, we propose the Online Entanglement Routing (OER) algorithm as shown in Algorithm 3 to select the EP for every demand.

In the OER algorithm, Φ is used to record the demands that have not received an EP, and w_e is the EPS pool selected to create an EL over link e when attempting to establish the EC for a demand. In Lines 2–12, we select the EP for each demand based on the solution of (7), *i.e.*, $\hat{y}_d^p$. Before selecting an EP for a demand, we should check if there are enough quantum resources to establish one more EP along this EP (Lines 6–9). After selecting EPs for demands based on $\hat{y}_d^p$, there would be some remaining quantum resources and some demands may not receive any EPs. From Line 13 to 16, we traverse all candidate EPs for every remaining demand to check if we can establish more ECs. This is to fully utilize quantum resources and maximize network throughput.

Theorem 3. *Suppose the solutions of* (7) *are* $\hat{\theta}_d$ *and* $\hat{y}_d^p$*, for a set of demands* $\{I_d\}$*, the number of ECs that can be established is at least* $\sum_{d \in \mathcal{D}} \hat{\theta}_d I_d$ *in expectation.*

Proof. In the optimal solution of (7), there must be $\sum_{p \in \mathcal{P}_d} \hat{y}_d^p = \hat{\theta}_d \quad \forall d \in \mathcal{D}$. Suppose t_d is the number of ECs we can establish for demand $d \in \mathcal{D}$. Based on the OER algorithm, an EC can be established for demand d with the probability $\sum_{p \in \mathcal{P}_d} \hat{y}_d^p = \hat{\theta}_d$. That is

$$t_d = \begin{cases} 1, \text{with the probability } \hat{\theta}_d \\ 0, \text{with the probability } \hat{\theta}_d \end{cases} \tag{8}$$

Then, the expected number of ECs that can be established for demand d is $E[t_d] = 1 \times \hat{\theta}_d + 0 \times (1 - \hat{\theta}_d) = \hat{\theta}_d$. As a result, the expected number of ECs that can be established in Lines 2–12 should be $E[\sum_{d \in \mathcal{D}} t_d I_d] = \sum_{d \in \mathcal{D}} E[t_d] I_d = \sum_{d \in \mathcal{D}} \hat{\theta}_d I_d$. In Lines 13–16, we will establish more ECs. It only increases the scaling factor of each demand. Accordingly, we can conclude the above theorem. □

Theorem 4. *Without checking the available quantum resources in Lines 6–9, we need at most* $F\lceil \hat{g}_w + C \rceil$ *EPSes in pool* w *with a high probability, where* $F = (\frac{4 \log |\mathcal{N}|}{\alpha} + 3)$*, and* C *is the maximum number of links covered by an EPS pool.*

Proof. Following the definition of t_d in (8), we have $E[t_d I_d] = \sum_{p \in \mathcal{P}_d} \hat{y}_d^p I_d$ and $\sum_{d \in \mathcal{D}} \sum_{p \in \mathcal{P}_d : e \in p} \hat{y}_d^p I_d = \sum_{w \in \mathcal{W}} q_e^w \hat{x}_e^w$. The expected number of ELs that have to be created over e satisfies $E[\sum_{e \in C(w)} \sum_{d \in \mathcal{D}} \sum_{p \in \mathcal{P}_d : e \in p} t_d I_d] = \sum_{w \in \mathcal{W}} \hat{x}_e^w q_e^w$. Assume $\dot{x}_e^w$ is the number of ELs that will be created over link e by EPS pool w, according to the EPS selection method in Line 7 where we always use the EPS pool that has to create the most ELs over e according to $\hat{x}_e^w$, we have $\sum_{e \in \mathcal{C}(w)} E[\dot{x}_e^w] \leq g_w + C$ since we always rounds up the number of ELs that to be created by EPS pool w.

Now, by defining $\lambda = \frac{1}{\sum_{d \in \mathcal{D}} I_d}$ and letting $\dot{x}_e$ be the number of ELs we must attempt to create over link e, we know

$$\begin{cases} \frac{\dot{x}_e^w \lambda}{g_w + C} \in [0, 1] \\ E[\frac{\sum_{e \in C(w)} \dot{x}_e^w \lambda}{g_w + C}] \leq \lambda \end{cases}$$

Suppose ρ is a positive value, by applying the Chernoff Bound [28], we have

$$\begin{aligned} &\mathbf{Pr}[\frac{\sum_{e \in C(w)} \dot{x}_e^w \lambda}{g_w + C} \geq (1 + \rho)\lambda] \\ =&\mathbf{Pr}[\frac{\sum_{e \in C(w)} \dot{x}_e^w}{g_w + C} \geq (1 + \rho)] \leq e^{-\frac{\rho^2 \lambda}{2 + \rho}} \end{aligned}$$

Now, we introduce a parameter F and assume

$$\mathbf{Pr}[\frac{\sum_{e \in C(w)} \dot{x}_e^w}{g_w + C} \geq (1 + \rho)] \leq e^{-\frac{\rho^2 \lambda}{2 + \rho}} \leq \frac{F}{|\mathcal{N}|^2} \tag{9}$$

(9) holds if $\rho \geq \frac{\log \frac{|\mathcal{N}|^2}{F} + \sqrt{\log^2 \frac{|\mathcal{N}|^2}{F} + 8\lambda \log \frac{|\mathcal{N}|^2}{F}}}{2\lambda}$, $|\mathcal{N}| \geq 2$.

By setting $F = \frac{1}{|\mathcal{N}|^2}$, (9) becomes

$$\mathbf{Pr}[\frac{\sum_{e \in C(w)} \dot{x}_e^w}{g_w + C} \geq (1+\rho)] \leq \frac{1}{|\mathcal{N}|^4} \tag{10}$$

where $\rho \geq \frac{2\log|\mathcal{N}| + 2\sqrt{\log^2|\mathcal{N}| + 2\lambda \log|\mathcal{N}|}}{\lambda}$. Since $\frac{4\log|\mathcal{N}|}{\alpha} + 2 > \frac{2\log|\mathcal{N}| + 2\sqrt{\log^2|\mathcal{N}| + 2\lambda\log|\mathcal{N}|}}{\lambda}$, (9) also holds when $\rho \geq \frac{4\log|\mathcal{N}|}{\alpha} + 2$. This demonstrates that EPS pool w needs fewer than $(1+\rho)(g_w + C) = (\frac{4\log|\mathcal{N}|}{\alpha} + 3)(g_w + C)$ EPSes. □

VI. Performance Evaluation

In this section, we will evaluate the performance of EPED using a custom in-house simulator. Simulations involve randomly generated networks with a certain number of quantum nodes (with their locations), EPSes, demands, and candidate positions of EPS pools. In every simulation trial, we ensure that there are more available EPSes than the candidate positions of EPS pools, and it is a feasible solution that assigns one EPS to every candidate EPS pool. For a given set of parameters, we run the simulation for 1×10^5 times and the averaged results are shown. The main findings include:

- Compared with a conventional greedy algorithm, EPED can save up to 8% EPS pools to serve all potential demands.
- The multi-path routing, EPS assignment, and EL creation in EPED improve network throughput by up to 31%, 49%, and 16%, respectively.

A. Simulation Methodology

Test case generation. To generate a case for evaluating the performance of EPED, we place a given number of quantum nodes into an 5×5 (km^2) square area and select a given number of candidate positions to deploy EPS pools. Given the coverage radius of an EPS pool (suppose it is r), we will check whether there is a path between every node pair and that the distance between every two adjacent nodes along this path is shorter than $1.8r$. If not, we scale down the entire area until such a path exists for every node pair. If it is not a feasible solution to deploy an EPS pool at every candidate position, we generate more candidate positions until there is a feasible solution.

Default parameters. There are 150 quantum nodes and 60 candidate positions to deploy EPS pools when we investigate the performance of minimizing the number of EPS pools required to ensure a feasible solution can be derived, while there are 100 quantum nodes and 60 candidate EPS pools when we investigate the performance of maximizing network throughput. The coverage radius of each EPS pool is 1. The parameter determining the success probability of creating an EL, *i.e.*, γ in (2), is 1×10^{-4}. Without clarification, there are 150 EPSes available and 36 demands in a network.

Comparison schemes. There are two goals to pursue in EPED. One is saving infrastructure costs, which is measured by the number of EPS pools to deploy. The other is maximizing network throughput, which is measured by the average number of ECs established per time slot (*i.e.*, ECs per time slot, epts). In terms of minimizing infrastructure costs, we compare EPED with the following two schemes:

- Greedy-based EPS pool selection (GD). We select the positions for deploying EPS pools one after another. In each iteration, we choose the position that can cover the most nodes that have not been covered by any selected EPS pools. When all nodes are covered, we added more EPS pools to ensure solution feasibility.
- Path-based EPS pool selection (PB). We check every node pair. If there is no feasible EP to establish ECs between two nodes, we find a path along which every two adjacent nodes are covered by the same EPS pool, and deploy more EPS pools to ensure there is a feasible EP between them.

To investigate the performance of EPED in maximizing throughput, we compare it with the following three schemes:

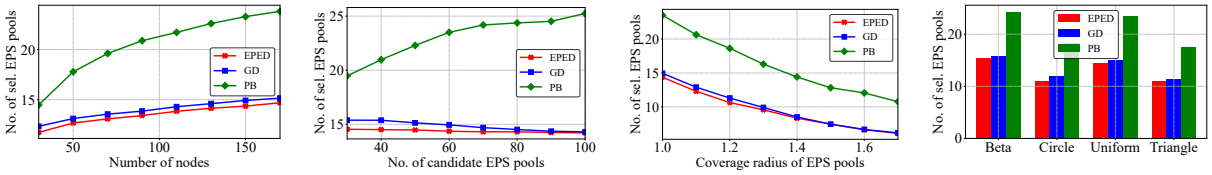
- Shortest-Path based routing (SP). By leveraging the EPSA algorithm to determine the EPS provisioning scheme, we establish the EC for every selected demand through the shortest path. This is to demonstrate the benefit of using the multi-path entanglement routing in EPED.
- Even EPS Assignment (EEA). We evenly assign all EPSes to selected EPS pools and leverage the OER algorithm to determine the EP of each demand. This demonstrates the superiority of the EPS assignment in EPED.
- Least Workload First (LWF). To fully utilize EPSes, we sequentially find the EP for each demand. If an EL can be created by EPSes in multiple pools, we will use an EPS in the pool with the least workload. This shows the superior performance of the EL creation method in EPED.

B. Performance of Saving Infrastructure Cost.

Impact of network scale. By varying the number of quantum nodes from 30 to 170, the required number of EPS pools is shown in Fig. 2(a). No matter which EPS pool selection scheme is adopted, more EPS pools are needed in larger networks since there are more nodes to serve. With PB, the number of required EPS pools increases faster than in the other two comparison schemes since it only considers a specific EP when introducing more EPS pools, which does not efficiently utilize the coverage areas of EPS pools.

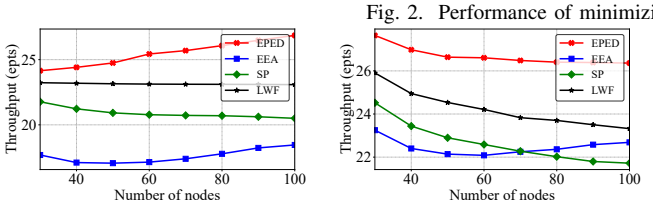
Impact of candidate EPS pools. When the number of candidate EPS pools varies from 30 to 100, EPED and GD need fewer EPS pools as shown in Fig. 2(b), since we have more choices to save the infrastructure cost. More candidate EPS pools introduce more solutions with the same infrastructure cost, and hence the performances of GD and EPED are very close when there are many candidate EPS pools. However, the required number of EPS pools increases with the number of candidate EPS pools when PB is adopted. This is because more EPs are available and the EPs for different demands are spread over the network, which prevents reusing the EPS pools deployed for a specific EP.

Impact of coverage radius. When the coverage radius of each EPS pool increases from 1 to 1.7 with a step size of 0.1, the number of required EPS pools is shown in Fig. 2(c).



(a) Impact of network scale. (b) Impact of candidate pools. (c) Impact of coverage radius. (d) Impact of node position.

Fig. 2. Performance of minimizing the required number of EPS pools.



(a) All long demands. (b) Mixed demands.

Fig. 3. Impact of traffic patterns when network scale changes.

Intuitively, the larger the coverage radius is, the fewer EPS pools we need. The EPS pool number decreases at a slower pace with the increase in EPS pool coverage radius. This is because when each EPS pool can cover a larger area, other EPS pools need a larger coverage radius increase to take over its function, *i.e.*, cover a set of quantum nodes.

Impact of node position distribution. How quantum nodes are distributed in the given area significantly impacts the number of required EPS pools. As in [7], we investigate the performances of all schemes by sampling the position (*i.e.*, its horizontal and vertical coordinates) of each node following four different distributions: (i) Beta, *i.e.*, the horizontal and vertical coordinates of each node are 5 times of a number following the distribution of Beta$(0.5, 0.5)$; (ii) Circle, *i.e.*, all nodes spread on a circle whose radius is 2.5 km; (iii) Uniform, *i.e.*, all nodes evenly spread in the given area; and (iv) Triangular, *i.e.*, the horizontal and vertical coordinates of each node follow the distribution of $\mathcal{T}(0, 2.5, 5)$. The simulation results are shown in Fig. 2(d). When all quantum nodes are distributed on a circle or the node coordinates follow the triangular distribution, we need fewer EPS pools than the other two test cases. This is because, in these two cases, all nodes are allocated closer to each other than in the other two cases, and an EPS pool can cover more nodes.

C. Performance of Maximizing Network Throughput

We investigate the performance of EPED to maximize network throughput based on two traffic patterns:

- Long demands. All ECs have to be established over EPs using multiple quantum links.
- Mixed Demands. We randomly select a set of nodes and assume there is a demand between every pair of nodes.

We do not investigate the case that all ECs can be established by one EPS pool since it has been well studied in [7].

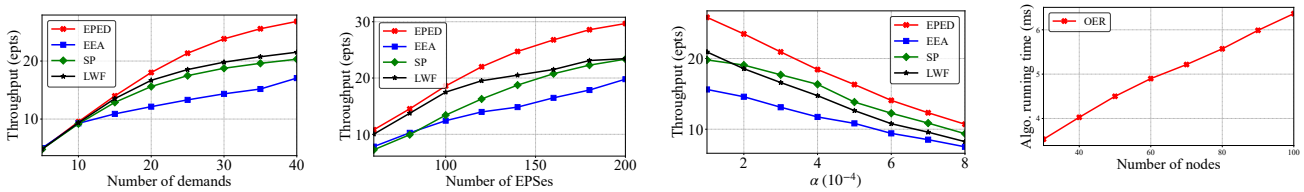
Impact of traffic patterns in networks with different scales. Fig. 3 shows the network throughput achieved by different comparison schemes under different traffic patterns when the number of quantum nodes changes. From this figure, we can make the following observations. Firstly, with more nodes in a network, EPED will achieve a larger network throughput when only long demands exist. This is because there are more available EPs to serve each demand. This analysis is validated by the performance achieved by the SP scheme. When only the shortest path can be used, network throughput decreases with the increase of network scale. However, when short demands also exist, the throughput achieved by EPED decreases since some short demands become long demands in a large network. In a network with 100 nodes, EPED achieves a similar performance regardless of which pattern demands follow, since almost all demands become long demands.

Secondly, when EEA is adopted, network throughput first decreases with the network scale and then increases. The decrease is because more EPS pools are needed and it leads to a lower utilization by evenly assigning EPSes to more EPS pools. When network scale continues increasing, evenly assigning EPS to each pool becomes a near-optimal solution since demands are spread sparsely over the network.

Last but not least, EPED outperforms the EEA, SP, and LWF by up to 49%, 31%, and 16%, respectively, when only long demands exist, while such performance improvements become 21%, 22%, and 13%, respectively, when serving mixed demands. The reason is that there is little optimization space to improve network throughput by changing the routing of demands with short EPs. Due to the space limitation, in the following, we only present the simulation results when only long demands exist since EPED is mainly designed for large quantum networks with many long demands.

Impact of workload. When the number of demands in a network changes from 5 to 40, the network throughput derived by all comparison schemes is shown in Fig. 4(a). When there are more demands in a network, we have more options to establish ECs and hence network throughput increases. The throughput achieved by LWF increases the slowest since the EPS pool with the least load may not be able to create an EL with a large success probability. In addition, throughput increases at a slower pace when the network experiences a heavy workload. This is because more demands will incur resource contention, and we cannot serve all additional demands.

Impact of available EPSes. When the number of available EPSes increases from 60 to 200, network throughput increases regardless of which comparison scheme is adopted as shown in Fig. 4(b). The network throughput achieved by EPED increases faster than that achieved by EEA since EPED intentionally provisions EPSes to pools that need to serve more demands. In addition, the throughput increase ratio achieved by all comparison schemes decreases when there are more than 100 available EPSes since other resources, *e.g.*, quantum memory, gradually become another bottleneck that limits network throughput.



(a) Impact of workload. (b) Impact of available EPSes. (c) Impact of success probability of creating ELs.

Fig. 4. Performance of maximizing network throughput.

Fig. 5. Algorithm running time.

Impact of success probability of creating ELs. When the parameter α in (2) increases from 1×10^{-4} to 8×10^{-4}, the network throughput derived by different comparison schemes is shown in Fig. 4(c). When α increases, the success probability of creating each EL decreases. Accordingly, network throughput also decreases. When α achieves 8×10^{-4}, it is unlikely to establish an EC going through a multi-hop EP. As a result, all comparison schemes achieve a closer network throughput, though EPED still outperforms others.

D. Algorithm Running Time

Fig. 5 shows how much time EPED spends running the OER algorithm. The results are collected from a desktop carrying an Intel Core i9-13900KF CPU and 64 GB of memory. From this figure, we can see that even when there are 100 nodes in the network, only 6.5 ms is needed to select EPs for all demands. Accordingly, the proposed EPED approach is feasible for large scale and heavily loaded quantum networks.

VII. Conclusions

In this work, we have proposed a new approach named EPED that organizes Entangled Photon Sources (EPSes) as multiple pools and dynamically assigns EPSes to create entanglement links (ELs) between nodes in their coverage areas. By stitching these ELs together with quantum swapping, we can establish Entanglement Connections (ECs) between every node pair. In EPED, we have proposed a series of efficient algorithms to determine the positions of EPS pools we have to deploy, the number of EPSes we should deploy into each pool, and how to dynamically provision EPSes to create ELs. We have conducted extensive simulations to demonstrate the superior performance of EPED.

References

[1] J. Eisert, K. Jacobs, P. Papadopoulos, and M. Plenio, "Optimal local implementation of nonlocal quantum gates," *Phys. Rev. A*, vol. 62, Oct. 2000.

[2] A. Yimsiriwattana and J. Lomonaco, "Generalized ghz states and distributed quantum computing," *AMS Contemporary Mathematics*, vol. 381, no. 131, 2005.

[3] M. Riebe, H. Häffner, C. F. Roos *et al.*, "Deterministic quantum teleportation with atoms," *Foundations of Physics*, vol. 429, p. 734–737, 2004.

[4] S. Shi and C. Qian, "Concurrent entanglement routing for quantum networks: Model and designs," in *Proceedings of the ACM SIGCOMM*, 2020.

[5] Y. Zhao and C. Qiao, "Redundant entanglement provisioning and selection for throughput maximization in quantum networks," in *Proceedings of the IEEE INFOCOM*, 2021.

[6] Y. Zeng, J. Zhang, J. Liu, Z. Liu, and Y. Yang, "Multi-entanglement routing design over quantum networks," in *Proceedings of the IEEE INFOCOM*, 2022.

[7] G. Iacovelli, F. Vista, N. Cordeschi, and L. A. Grieco, "A probability-based optimization approach for entanglement distribution and source position in quantum networks," *IEEE Journal on Selected Areas in Communications*, vol. 42, no. 7, pp. 1738–1748, 2024.

[8] J. Yin, Y. Cao, Y.-H. Li *et al.*, "Satellite-based entanglement distribution over 1200 kilometers," *Science*, vol. 356, no. 6343, pp. 1140–1144, 2017.

[9] 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.

[10] L. Chen, K. Xue, J. Li *et al.*, "A heuristic remote entanglement distribution algorithm on memory-limited quantum paths," *IEEE Transactions on Communications*, vol. 70, no. 11, pp. 7491–7504, 2022.

[11] Q. Zhu, Y. Zhao, H. Xu *et al.*, "Eps placement and lightweight entanglement routing for quantum data networks," in *IEEE ICDCS*, 2024, pp. 1190–1201.

[12] TOSHIBA, "QKD System Specifications," 2022. [Online]. Available: https://www.global.toshiba/ww/products-solutions/security-ict/qkd/products.html

[13] Q. Zhu, Y. Zhao, H. Xu, L. Huang, and C. Qiao, "Beyond entanglement routing: Source assignment and all-optical switching-based distribution," *IEEE Transactions on Networking*, vol. 33, no. 2, pp. 713–728, 2025.

[14] L. Yang, Y. Zhao, H. Xu, and C. Qiao, "Online entanglement routing in quantum networks," in *IEEE/ACM IWQoS*, 2022.

[15] L. Yang, Y. Zhao, L. Huang, and C. Qiao, "Asynchronous entanglement provisioning and routing for distributed quantum computing," in *Proceedings of the IEEE INFOCOM*, 2023.

[16] S. Pirandola, "End-to-end capacities of a quantum communication network," *Communications Physics*, vol. 3, pp. 1—-10, 2019.

[17] E. Schoute, L. Mancinska, T. Islam, I. Kerenidis, and S. Wehner, "Shortcuts to quantum network routing," *CoRR*, vol. abs/1610.05238, 2016. [Online]. Available: http://arxiv.org/abs/1610.05238

[18] G. Vardoyan, S. Guha, P. Nain, and D. Towsley, "On the stochastic analysis of a quantum entanglement switch," *SIGMETRICS Perform. Eval. Rev.*, vol. 47, no. 2, pp. 27—-29, dec 2019.

[19] M. Caleffi, "Optimal routing for quantum networks," *IEEE Access*, vol. 5, pp. 22 299–22 312, 2017.

[20] Y. Zhao, G. Zhao, and C. Qiao, "E2e fidelity aware routing and purification for throughput maximization in quantum networks," in *Proceedings of the IEEE INFOCOM*, 2022.

[21] J. Li, M. Wang, K. Xue, R. Li, N. Yu, Q. Sun, and J. Lu, "Fidelity-guaranteed entanglement routing in quantum networks," *IEEE Transactions on Communications*, vol. 70, no. 10, pp. 6748–6763, 2022.

[22] G. Zhao, J. Wang, Y. Zhao *et al.*, "Segmented entanglement establishment for throughput maximization in quantum networks," in *IEEE ICDCS*, 2022.

[23] Z. Li, J. Li, K. Xue, D. S. L. Wei, N. Yu, Q. Sun, and J. Lu, "Efficient remote entanglement distribution in quantum networks: A segment-based method," *IEEE Transactions on Network and Service Management*, vol. 21, no. 1, pp. 249–265, 2024.

[24] S. Pouryousef, N. K. Panigrahy, and D. Towsley, "A quantum overlay network for efficient entanglement distribution," in *IEEE INFOCOM*, 2023.

[25] M. Basappa, R. Acharyya, and G. K. Das, "Unit disk cover problem in 2d," *Journal of Discrete Algorithms*, vol. 33, pp. 193–201, 2015.

[26] J. Y. Yen, "An algorithm for finding shortest routes from all source nodes to a given destination in general networks," *Quarterly of Applied Mathematics*, vol. 27, pp. 526–530, 1970.

[27] S. Boyd and L. Vandenberghe, *Convex Optimization*. New York, NY, USA: Cambridge University Press, 2004.

[28] M. Mitzenmacher and E. Upfal, *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*. Cambridge University Press, 2005.