A Fast Heuristic Entanglement Distribution Algorithm for Quantum Repeater Chains

Wenkang Cen, Jinbei Zhang, *Member, IEEE*, Kechao Cai, *Member, IEEE*, Shihai Sun, and John C. S. Lui, *Fellow, IEEE, ACM*

***Abstract*— Entanglement distribution via probabilistic entanglement swapping across a quantum repeater chain connecting two quantum nodes is a challenging problem. The difficulty lies in the exponential number of possible swapping structures within the repeater chain, necessitating efficient search algorithms, especially as the chain length increases. In this paper, we first explore the algorithmic design to facilitate the search for the optimal swapping structure along a repeater chain, aiming to maximize the entanglement distribution rate. Second, we examine the computational complexities of various algorithms and find that prior approaches exhibit excessively high complexities. Thus, we propose an efficient dynamic programming-based algorithm, FastHED, that leverages heuristics to expedite the search for the optimal swapping structure. Our theoretical analysis reveals that the upper bound of the proposed algorithm's computational complexity is $O(n(\log n)^3)$ (more precisely, $O(n(\log n)^2 \log\log n)$ when $n \leq 2^{29}$), a significant improvement over the existing algorithm with a complexity of $O(n^2 \log n)$, where n denotes the repeater chain's length. Additionally, we design a best-first framework to evaluate the performance of different algorithms. Numerical results show that our algorithm achieves a higher average entanglement distribution rate than existing algorithms.**

***Index Terms*— Quantum repeater chain, entanglement distribution, entanglement swapping, dynamic programming.**

I. Introduction

QUANTUM Internet, with its unique quantum advantages, has gained significant attention in recent years, and is regarded as an important technical direction for future communication technologies. Quantum entanglements or EPR pairs [1], [2] (also referred to as ebits) are the vital resources for transmitting quantum information in the Quantum Internet. They are also critical in other upper-layer applications, such as quantum conference key agreement [3], [4], clock synchronization [5], [6] and multiparty quantum computation [7]. A central challenge in these applications is efficiently establishing quantum entanglement for quantum communication.

There are two methods for establishing quantum entanglement between two nodes. The first method, known as *entanglement generation*, directly creates entanglement between any two end nodes that are directly linked by a quantum channel, such as an optical fiber or a free-space link. However, it suffers from exponential decay [8], [9], [10], limiting the entanglement distance of the two nodes. The second method, called *entanglement swapping* [11], can generate quantum entanglement between two end nodes that are far apart. Quantum repeaters play a crucial role in entanglement swapping. Acting as intermediate nodes that connect the two end nodes through quantum channels, quantum repeaters can establish long-distance entanglement by leveraging two short-distance entanglements across two short quantum channels via Bell State Measurements (BSM). When multiple quantum repeaters exist between two end nodes, entanglement swapping can be repeatedly performed at the repeaters to create a long distance end-to-end entanglement, mitigating the exponential decay in long-distance entanglement.

Using the aforementioned methods, Quantum entanglement Routing (QR) [12] has been introduced to determine the optimal scheme for distributing quantum entanglements between any arbitrary pair of nodes, with the objective of maximizing the entanglement distribution rate (EDR).[1] Solving the QR problem involves two steps in general. The initial step entails selecting the most suitable quantum repeaters to form a quantum path [12], [13], [14], [15], also known as a quantum repeater chain within the quantum network. In this repeater chain, adjacent repeaters are interconnected by quantum channels, referred to as single-hop elementary links. The collection of all these elementary links constitutes a multi-hop repeater chain, establishing connectivity between the end nodes. The second step is to find the optimal entanglement swapping protocol on the selected repeater chain [11], [16], [17]. Entanglement swapping is typically probabilistic [12], [18], [19], [20], [21] because entanglement is swapped conditional on the occurrence of preset photon coincidence events for photonic entanglement swapping [22], [23]. To realize entanglement distribution using current quantum technologies, there are many aspects that need to be optimized for a given quantum path, such as designing cutoff policies [24] for

Received 2 December 2023; revised 15 August 2024; accepted 2 October 2024; approved by IEEE TRANSACTIONS ON NETWORKING Editor D. Elkouss. Date of publication 17 October 2024; date of current version 14 February 2025. This work was supported in part by the National Key Research and Development Program of China under Grant 2022YFB2902700; in part by NSF, China, under Grant 62471505, Grant 62071501, and Grant 62202508; and in part by Shenzhen Science and Technology Program under Grant JCYJ20220818102011023, Grant 20220817094427001, and Grant ZDSYS20210623091807023. The work of John C. S. Lui was supported in part by the Research Grants Council of Hong Kong (RGC) under Grant SRFS2122-4202. *(Corresponding author: Jinbei Zhang.)*

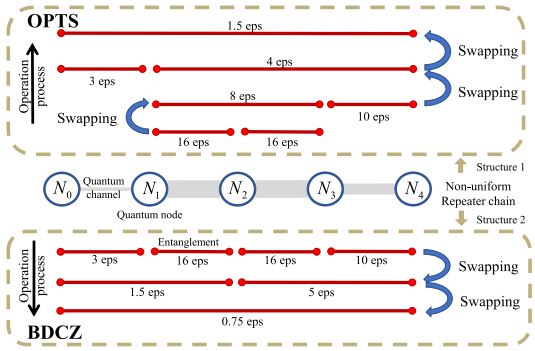
Wenkang Cen, Jinbei Zhang, Kechao Cai, and Shihai Sun are with the School of Electronics and Communication Engineering, Sun Yat-sen University, Shenzhen Campus, Shenzhen 518107, China (e-mail: cenwk@mail2.sysu.edu.cn; zhjinbei@mail.sysu.edu.cn; caikch3@mail.sysu.edu.cn; sunshh8@mail.sysu.edu.cn).

John C. S. Lui is with the Department of Computer Science and Engineering, The Chinese University of Hong Kong, Hong Kong (e-mail: cslui@cse.cuhk.edu.hk).

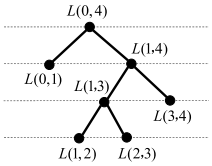
This article has supplementary downloadable material available at https://doi.org/10.1109/TNET.2024.3475378, provided by the authors.

Digital Object Identifier 10.1109/TNET.2024.3475378

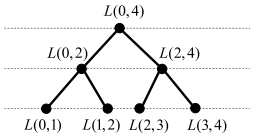
[1]Entanglement distribution rate is the number of distributed entanglements per unit of time (in ebit per second or eps).



(a) Two different swapping structures conducted in a 4-hop chain.



(b) The OPTS binary tree.



(c) The BDCZ binary tree.

Fig. 1. This 4-hop non-uniform repeater chain contains quantum channels with distinct entanglement generation capacities, such as 3 eps on $N_0 - N_1$, 16 eps on $N_1 - N_2$ and $N_2 - N_3$ and 10 eps on $N_3 - N_4$. Each swapping operation would reduce the capacity down to half of the minimum input capacities (i.e., take $p_{swap}^{(k)} = 0.5$ in (3)). The **opt**imal **s**wapping structure (OPTS) is better than BDCZ [11] structure. Where $L(i, j)$ denotes the chain from N_i to N_j. Two corresponding binary trees are illustrated at the bottom of the figure.

avoiding decoherence and distillation protocols [25] to improve the fidelity.[2] With the development of cutting-edge quantum technologies, we assume negligible effects from decoherence and other factors, thus reducing the swapping protocol optimization problem to a swapping structure optimization problem. Essentially, the swapping structure for a repeater chain is the order of entanglement swapping operations. The swapping structure can be viewed as a binary tree where entanglement swapping operations can be executed in parallel in the left and right subtrees by iteratively dividing the repeater chain into two shorter chains. For a uniform chain, where all the elementary links have the same EDR, Briegel et al. [11] introduced a strict parallel swapping structure called BDCZ (named after the authors) to maximize the EDR. However, in a non-uniform chain where the EDRs on the elementary links are heterogeneous, a strict symmetric parallel scheme like BDCZ may not be optimal. For example, as illustrated in Fig. 1, a better swapping structure exists in the non-uniform chain, yielding a higher EDR than the strict parallel swapping structure. In the 4-hop non-uniform repeater chain as shown in Fig. 1(a), BDCZ (Structure 2) achieves an end-to-end EDR of 0.75 eps via symmetric entanglement swapping operations. In contrast, the optimal swapping structure (OPTS, Structure 1) attains an end-to-end EDR of 1.5 eps using asymmetric entanglement swapping operations.

There have been several salient studies on designing algorithms to find the optimal swapping structure in the non-uniform chain to obtain a higher EDR. Jiang et al. [16] and Goodenough et al. [17] proposed the regular dynamic programming (DP) technique for searching for the optimal swapping structure on a multi-hop chain. They employed a bottom-up approach, constructing the global optimal structure by recursively assembling optimal swapping structures on shorter sub-chains. However, these studies mainly focused on maximizing the entanglement distribution rate while neglecting the algorithmic computation complexity. Note that an algorithm with high computational complexity has an inherent drawback: it may not quickly adapt to changing quantum network parameters, resulting in an outdated optimal swapping structure. This leads to a dilemma: whether to adhere to the previous swapping structure, which may no longer be optimal, or to rerun the algorithm with the latest parameters. Therefore, an effective algorithm should not only aim to achieve a higher EDR, but can also efficiently find the optimal swapping structure with low computation complexity to accommodate the time varying nature of quantum networks.

In this paper, we introduce a novel DP-based algorithm, *Fast Heuristic Entanglement Distribution* algorithm (FastHED), designed to reduce the computational complexity of finding the near-optimal swapping structures. Instead of merely using a bottom-up approach, FastHED leverages novel pruning techniques to identify the least essential sub-chains by top-down pruning before starting the DP search. Furthermore, we establish a recursive expression for swapping-structure dependent EDR over a multi-hop chain. Additionally, we derive an upper bound for the computational complexity of FastHED. To evaluate the average EDR of various algorithms in entanglement distribution, we also develop a best-first entanglement distribution framework and integrate it to tackle the entanglement distribution problem in chains with time varying parameters. Numerical results show the superiority of FastHED over existing algorithms.

The remainder of the paper is as follows. In Sec. II, we provide an overview of related work. In Sec. III, we present the modeling of quantum network and entanglement distribution on repeater chains. In Sec. IV, we introduce the problem structure of heuristic dynamic programming and present the details of FastHED. In addition, we derive an upper bound for the computational complexity for FastHED. In Sec. V, we develop a best-first entanglement distribution framework. In Sec. VI, we illustrate the superiority of FastHED using numerical simulations. We conclude and address some potential problems for future research in Sec. VII.

II. Related Work

Entanglement distribution, a sub-task of Quantum Routing, is crucial for quantum networking. Existing research mainly addresses two challenges in the QR problem. The first challenge, known as path selection [12], [13], [14], [26], [27], [28], [29], [30], [31], [32], is to design an algorithm to identify the most suitable repeaters to form a path from the source node to the destination node. The second challenge, termed swapping protocol optimization, is to design a swapping protocol on the selected path to attain a high EDR between the two end nodes. A core challenge in swapping protocol optimization is known as swapping structure optimization problem.

Chang and Xue [33] pointed out that the order (swapping structure) of distributing entanglement significantly impacts

[2]Fidelity is an important metric for assessing the disparity between actual entanglement and perfect entanglement, ranging from 0 to 1. A higher fidelity value indicates a closer approximation to perfect entanglement.

the EDR. Briegel et al. [11] proposed a parallel structure called BDCZ to minimize the entanglement distribution waiting time (maximize the EDR) for a uniform chain. However, when the elementary links between the repeaters are non-uniform, the strict symmetric parallel scheme of BDCZ is not optimal as illustrated in Fig. 1.

Goodenough et al. [17] proved that the number of all possible swapping structures in a repeater chain is super-exponential. Shchukin and van Loock [34] designed a Markov decision process solution for finding the optimal swapping structure, but the computational complexity is high due to the super-exponential size of the state space. There are also several attempts on reducing the computation complexity of the entanglement distribution problem. Dai et al. [21] transformed the problem into a linear programming (LP) problem with polynomial computational complexity. But the number of decision variables for an n-hop chain is $O(n^3)$, thus the complexity of solving this LP optimization exceeds $O(n^3)$. In [16], Jiang et al. first provided a dynamic programming [35] based solution with polynomial complexity. Other works reduced the complexity by leveraging heuristics in quantum networks [17], [20], [36] (e.g., the optimal EDR can be achieved by conducting quantum operations on entanglements with similar EDRs). Goodenough et al. [17] leveraged heuristics in the DP-based algorithm with a computational complexity of $O(n^2 \log n)$ to find a near-optimal swapping structure, which is the state-of-the-art algorithm for a non-uniform chain. This algorithm outperforms the ones presented in [11], [37], [38], and [39]. Some researchers [19], [40], [41], [42] derived the closed-form expression of the waiting time in entanglement distribution and conducted theoretical analyses of various swapping structures' performance. Nevertheless, it is important to note that finding the exact expression is computationally intensive. Recently, a more practical entanglement swapping protocol with imperfect quantum memories has been studied; for instance, Iñesta et al. [24] addressed the short-live quantum memory problem and Haldar et al. [25] designed a fast and reliable entanglement distribution protocol considering imperfect memories.

In [43] and [44], the authors designed a more general swapping method (i.e., n-fusion) through Greenberger-Horne-Zeilinger (GHZ) Measurements for the swapping. Using GHZ measurements, Zeng et al. [45] significantly enhanced the EDR.

Our work differs from the existing works primarily in two key aspects. First, we propose a DP-based solution for swapping structure optimization that is significantly faster than that in [17]. Our top-down pruning techniques can eliminate redundant computations in the bottom-up DP searching. Second, we design a best-first framework that can provide insights into: i) the tradeoff between reducing the computation complexity of an entanglement distribution algorithm and maximizing the EDR, and ii) how to leverage a fast algorithm to achieve a high EDR on a repeater chain with time varying parameters.

III. System Model

In this section, we first introduce the network model of a quantum repeater chain and describe how to compute the EDR of any entanglement swapping structure on a repeater chain.

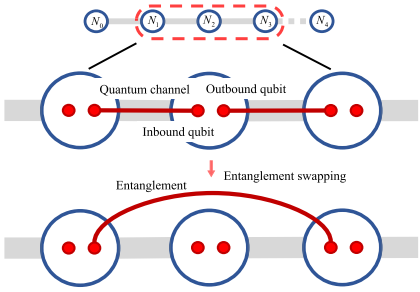


Fig. 2. An n-hop repeater chain. Every repeater can perform entanglement swapping to glue two short entanglements to create a long entanglement.

A. Network Model

Consider an n-hop repeater chain $N_0 - \cdots N_i - N_{i+1} - \cdots N_n, 0 \leq i \leq n$, connecting the two end nodes, i.e., the source node N_0 and the destination node N_n, as shown in Fig. 2. Let $L(i, j)$ denote the sub-chain from the left node N_i to the right node N_j in the n-hop repeater chain, $0 \leq i, j \leq n$ and $i \neq j$. This repeater chain consists of the following components:

- *Quantum nodes*: These nodes can perform a series of quantum operations including quantum bit (qubit) measurements and quantum information transmission. Each quantum node has a quantum memory for storing unused entangled qubits.
- *Quantum repeaters*: Quantum repeaters function as intermediate nodes within the repeater chain. Similar to the relay nodes in classical networks, they can perform entanglement swapping operations to create long-distance entanglements. Similar to [46], we assume[3] each repeater is equipped with a sufficient quantum buffer, i.e., a sufficiently-size and sufficiently long-lived memory for entanglement swapping. As shown in the red dashed box in Fig. 2, the repeater node N_2 can perform entanglement swapping on two entangled qubits of two short entanglements, and generate a new long-distance entanglement connecting N_1 and N_3.
- *Quantum channels*: Quantum channels, also known as elementary links, can connect adjacent nodes directly. For those non-adjacent nodes with no direct quantum channel links, the generation of entanglement is only feasible with the assistance of quantum repeaters.
- *Classical communication network*: This network interconnects all quantum nodes and transmits the measurement results of quantum operations.

B. Entanglement Generation Over an Elementary Link

For two adjacent nodes N_i and N_{i+1} connected by a quantum channel, one can establish an eflow[4] connecting the two nodes by implementing Entanglement Generation [8], [46], [51]. For example, an entangled qubit pair can be locally prepared at N_i, then N_i sends one of the entangled qubits to N_{i+1} via the quantum channel and an entanglement

[3]We acknowledge that considering limited quantum memory (e.g., limited size in [41], short lifespan in [24], [25]) is crucial. However, with advancements of quantum technologies, future quantum repeaters may have sufficient buffers [47], [48], [49], [50].

[4]An eflow is a series of quantum entanglements that can connect the two end nodes of a chain [21].

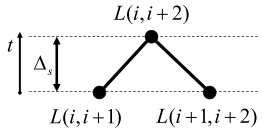


Fig. 3. By performing entanglement swapping on N_{i+1}, one can construct an eflow over $L(i,i+2)$ from two eflows over $L(i,i+1)$ and $L(i+1,i+2)$.

connecting N_i to N_{i+1} is generated [51]. The entanglement generation attempt may not always succeed. An important parameter to evaluate the generation capability of $N_i - N_{i+1}$ is called entanglement generation success probability $p_{gen}^{(i,i+1)}$. An entanglement connecting N_i to N_{i+1} is successfully generated with a probability $p_{gen}^{(i,i+1)}$; otherwise, no entanglement is obtained. We assume the entangled qubit can be stored in a quantum memory for a sufficiently long time. By continuously conducting entanglement generation, an eflow connecting N_i to N_{i+1} can be established with the EDR:

$$R(i,i+1) = p_{gen}^{(i,i+1)} \cdot f, \tag{1}$$

where f denotes the operation frequency, i.e., the number of attempts per second.

C. Entanglement Distribution Over a Multi-Hop Repeater Chain

For a two-hop sub-chain $L(i,i+2)$, an eflow over $L(i,i+2)$ can be established from the two eflows over the adjacent elementary links $L(i,i+1)$ and $L(i+1,i+2)$ with entanglement swapping operations on N_{i+1} as shown in Fig. 3. Let $p_{swap}^{(i+1)}$ be the success probability of entanglement swapping on N_{i+1}. We assume that the time required for an entanglement swapping operation, referred to as a swapping duration and denoted as Δ_s, is close to 0 and even negligible [34]. Then according to [46], the EDR of the eflow over $L(i,i+2)$ can be expressed as:

$$R(i,i+2) = \min\{R(i,i+1), R(i+1,i+2)\} \cdot p_{swap}^{(i+1)}. \tag{2}$$

In other words, the EDR $R(i,i+2)$ of the long eflow between N_i and N_{i+2} relies on the smaller EDRs (the larger waiting times) of the two short eflows, $R(i,i+1)$ and $R(i+1,i+2)$, and the success probability of entanglement swapping $p_{swap}^{(k)}$. More generally, any eflow over a sub-chain $L(i,j)$ can be constructed from two eflows over its two sub-chains, $L(i,k)$ and $L(k,j)$, by performing entanglement swapping operations on the repeater N_k that is between N_i and N_j. Given that two short eflows with EDRs $R(i,k)$ and $R(k,j)$ have been established and safely stored in quantum memories and entanglement swapping is performed on N_k, the EDR of the eflow over the sub-chain $L(i,j)$ can be expressed as:

$$R(i,j) = \min\{R(i,k), R(k,j)\} \cdot p_{swap}^{(k)}. \tag{3}$$

Establishing an eflow over an n-hop chain from the eflows over n elementary links is to distribute entanglements and perform a sequence of entanglement swapping operations on the intermediate repeaters in the chain. The order of these entanglement swapping operations can be represented via a binary tree, which we refer to as the *swapping structure*. As shown in Fig. 4, each node of the binary tree represents a sub-chain, with the root node corresponding to $L(0,n)$ and the leaf nodes representing the elementary links. In the binary tree, an eflow over a sub-chain $L(i,j)$ (a parent node) can be constructed from the eflows over its two sub-chains (its two children nodes). As shown in Fig. 4, to construct an eflow over an n-hop chain at the root node (or $L(0,n)$) of the binary tree, we first create eflows over the elementary links at the leaf nodes. Using these eflows over the elementary links, we can then conduct entanglement swapping operations simultaneously on the intermediate repeaters to create eflows over longer sub-chains, and form the corresponding parent nodes in the first swapping duration. This process is repeated until the eflow over $L(0,n)$ is constructed. Let the time for entanglement generation over elementary links be Δ_{gen} (a unit of time) and the depth of the binary tree be h, i,e., the number of all swapping durations required to establish the desired eflow. After $\Delta_{gen} + h\Delta_s = \Delta_{gen}$, the desired eflow is obtained since Δ_s can be negligible [34].

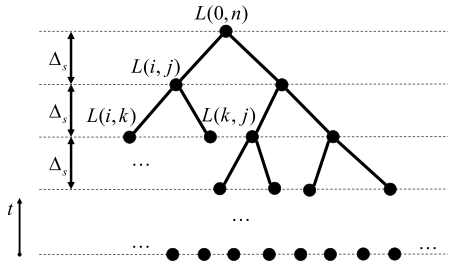


Fig. 4. Each parent (sub-chain $L(i,j)$) has two children (sub-chains $L(i,k)$ and $L(k,j)$), which are separated by a repeater in $L(i,j)$. The optimal repeater N_{k^*} that maximizes the EDR of $L(i,j)$ can be identified using (4).

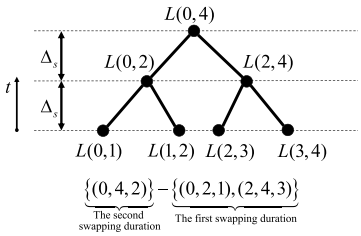


Fig. 5. Following BDCZ, one can construct an eflow over $L(0,4)$ in two swapping durations. In the first swapping duration, we perform entanglement swapping on N_1 and N_3 to construct eflows over $L(0,2)$ and $L(2,4)$ simultaneously. In the second swapping duration, we perform entanglement swapping on N_2 and construct an eflow over $L(0,4)$.

Let us clarify how to derive the swapping structure from a binary tree. Fig. 5 shows an example of the entanglement distribution process on a 4-hop chain. We assume the eflows over all elementary links are established at the beginning of the entanglement distribution procedure. In the first swapping duration, we select two eflows over $L(0,1)$ (a leaf node) and $L(1,2)$ (another leaf node), and perform entanglement swapping on N_1 to construct an eflow over $L(0,2)$ (a parent node). At the same time, we can perform entanglement swapping on N_3 to construct an eflow over $L(2,4)$ using eflows over $L(2,3)$ and $L(3,4)$. In the next swapping duration, we select the two eflows over $L(0,2)$ and $L(2,4)$, and perform entanglement swapping on N_2 to construct an eflow over $L(0,4)$. The swapping structure in Fig. 5 can be represented

TABLE I
NOTATIONS

Notation	Description
n	Hop-length of chain $L(0,n)$
N_i	The i-th repeater node
$p_{gen}^{(i,i+1)}$	Success probability of entanglement generation
$p_{swap}^{(k)}$	Success probability of entanglement swapping on N_k
$\tau_{(i,i+1)}$	Generation time of entanglement from N_i to N_{i+1}
$L(i,j)$	The sub-chain from node N_i to N_j
k^*	The index of the optimal repeater for $L(i,j)$
$\mathcal{L}^{(s)}$	The all sub-chains needed to be computed at level s
$\mathbf{R}$	A 2D array saving the EDRs of all sub-chains
$\mathbf{S}$	A 2D array saving the optimal repeater indices
$\mathbf{C}$	The C-loop pruning array
$\mathbf{K}$	The K-loop pruning array
$\mathfrak{Q}$	The swapping structure
$[\alpha_k^{(s)};\beta_k^{(s)}]$	The k-th segment at level s
$\gamma^{(s)}$	The upper bound of the length of segments at level s
$l^{(s)}$	The upper bound of the length of sub-chains at level s
$\phi(n,s)$	The upper bound of the complexity of FastHED
$\psi(n)$	Function of computational complexity of NOPTS-DP
λ_i	The EDR of the i-th run in best-first framework
$\overline{\lambda}$	The average EDR in best-first framework

as $\{(0,4,2)\}-\{(0,2,1),(2,4,3)\}$, where each tuple (i,j,k) indicates the operation of entanglement swapping on N_k to construct an eflow over $L(i,j)$ from eflows over $L(i,k)$ and $L(k,j)$. The notation $\{\cdot\}$ represents a group of entanglement swappings indicated by the tuples that can be performed simultaneously. Another example of a swapping structure $\{(0,4,1)\}-\{(1,4,3)\}-\{(1,3,2)\}$ is shown in Fig. 1(b).

The goal of entanglement distribution is to find the optimal swapping structure for constructing the eflow over an n-hop chain with the maximum EDR. In other words, finding the optimal repeaters for conducting entanglement swapping. To find the optimal repeater that maximizes the EDR of the eflow over $L(i,j)$, one can search through all the intermediate repeaters and calculate the EDRs of the corresponding eflow using (3). Hence, we establish the following recurrence structure for the *maximum EDR* of the entanglement between N_i and N_j for $i \le j-2$,

$$R(i,j) = \max_{i+1\le k\le j-1} \left\{ \min\left\{R(i,k), R(k,j)\right\} \cdot p_{swap}^{(k)} \right\}. \quad (4)$$

Note that if the EDRs of all the nodes (sub-chains) in a binary tree satisfy (4), the associate swapping structure is optimal and yields the maximum EDR for $L(0,n)$.

IV. Heuristic Dynamic Programming for EDR Maximization

In this section, we introduce a dynamic programming algorithm designed to identify the optimal swapping structure using the recurrence structure in (4). Following that, we present two pruning techniques for this algorithm using heuristics.

A. Dynamic Programming Algorithm

A brute-force search for the optimal swapping structure in (4) requires splitting the n-hop chain into halves recursively. However, this approach is impractical due to the exponential growth in computation time for searching all possible swapping structures as the repeater chain length increases. Therefore, the dynamic programming-based algorithm was introduced in [16] and [17] to leverage the swapping structures of sub-chains to reduce the redundant computation and improve the computational efficiency.

Algorithm 1 Pure DP Search (Pure-DP)

Input: Chain length n, Entanglement generation probability $p_{gen}^{(i,i+1)}, \forall i \in [0,n-1]$, Swapping probability $p_{swap}^{(k)}, \forall k \in [1,n-1]$
Output: The indices of optimal repeaters $\mathbf{S}$, the entanglement distribution rates $\mathbf{R}$, and the optimal swapping structure $\mathfrak{Q}$

1 **for** $i \in [0,n-1]$ *// Compute the EDRs of the elementary links* **do**
2 $R(i,i+1) \leftarrow 1/p_{gen}^{(i,i+1)}$;
3 **end**
4 **for** $d \in [2,n]$ *// Compute bottom-up* **do**
5 **for** $i \in [0,n-d]$ *// C-loop* **do**
6 $j = i+d$;
7 **for** $k \in [i+1, j-1]$ *// K-loop* **do**
8 Find the optimal repeater N_{k^*} with the highest EDR among all N_k using (4);
9 **end**
10 *// Save the results*
11 $S(i,j) \leftarrow k^*$;
12 $R(i,j) \leftarrow \min\{R(i,k^*), R(k^*,j)\} \cdot p_{swap}^{(k)}$;
13 **end**
14 **end**
15 $\mathfrak{Q} \leftarrow$ FindOrder($\mathbf{S}$, 0, n); *// The details are shown in Algorithm 2*
16 **return** $\mathbf{S}$, $\mathbf{R}$ and $\mathfrak{Q}$

Specifically, with the recurrence structure in (4), one can establish long eflows from short eflows over sub-chains using a bottom-up approach. There are $O(n^2)$ sub-chains as $0 \le i \le n$ and $0 \le j \le n$. For each sub-chain $L(i,j)$, a naive algorithm needs to search $O(n)$ $(j-i-2)$ times for the optimal repeater N_{k^*} from N_{i+1} to N_{j-1}. To keep track of the swapping structures in the sub-chains, the indices of the optimal repeater for the sub-chains are saved in a two-dimensional (2D) array $\mathbf{S} \in \mathbb{Z}^{(n+1)\times(n+1)}$, and the entry $S(i,j)$ of $\mathbf{S}$ records the index of the optimal repeater for $L(i,j)$. The EDRs of the sub-chains are saved in a 2D array $\mathbf{R} \in \mathbb{R}^{(n+1)\times(n+1)}$, and the entry $R(i,j)$ of $\mathbf{R}$ records the maximum EDR of the eflow over $L(i,j)$.

Such an algorithm is called Pure-DP and is presented in Algorithm 1. The optimal swapping structure $\mathfrak{Q}$ is initialized as an empty queue, and the EDRs of the elementary links are computed in Line 2.[5] The first two for-loops (from Lines 4 and 5 to Lines 13 and 14) are called *sub-chain search loops* (C-loop) as they traverse all the sub-chains. The for-loop from Line 7 to Line 8, termed as *near-optimal repeater search loop* (K-loop), is to find the optimal repeater for every sub-chain. In Line 4, d represents the position of the upper right diagonal line of $\mathbf{S}$ and $\mathbf{R}$. For instance, as shown in Fig. 6,

[5]Hereafter, we use $[a,b]$ to denote the set of integers between a and b.

Algorithm 2 Findorder

Input: $\mathbf{S}$, i, j
Output: the swapping structure $\mathfrak{Q}$
1 Initialize U as a set $\{(i, j, S(i, j))\}$; *// The last group in the swapping structure*
2 Initialize $\mathfrak{Q}$ as an empty queue and push U to $\mathfrak{Q}$; *// $\mathfrak{Q}$ is a queue of sets*
3 **while** *U is not empty* **do**
4 Initialize a new set U'; *// The next group*
5 **for** $(\tilde{i}, \tilde{j}, S(\tilde{i}, \tilde{j})) \in U$ *and* $\tilde{i} < \tilde{j} - 1$ **do**
6 Insert $(\tilde{i}, S(\tilde{i}, \tilde{j}), S(\tilde{i}, S(\tilde{i}, \tilde{j})))$ and $(S(\tilde{i}, \tilde{j}), \tilde{j}, S(S(\tilde{i}, \tilde{j}), \tilde{j}))$ to U'; *// Except the elementary links*
7 **end**
8 Push U' to $\mathfrak{Q}$;
9 $U \leftarrow U'$;
10 **end**
11 **return** $\mathfrak{Q}$;

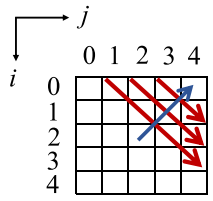


Fig. 6. The search process of Algorithm 1.

for a 4-hop repeater chain, when $d = 1$, the C-loop would search for all the elementary links $L(i, i + 1)$ and compute the upper-right diagonal (the longest red arrow) entries of $\mathbf{R}$ and $\mathbf{S}$. When $d = 2$, the C-loop searches all the sub-chains of length 2 and compute the upper-right diagonal (the second longest red arrow) of $\mathbf{R}$ and $\mathbf{S}$. Thus, iterating d from 2 to n follows a *bottom-up* search manner as shown by the blue arrow in Fig. 6. In Line 15, Algorithm 1 recovers the optimal swapping structure $\mathfrak{O}$ for an n-hop repeater chain from $\mathbf{S}$ using the function FINDORDER presented in Algorithm 2.

In Algorithm 2, the output $\mathfrak{Q}$ is the optimal swapping structure defined in Sec III-C. We use a set U (consisting of tuples) to store a group $\{\cdot\}$ of a swapping structure $\mathfrak{Q}$.

Specifically, the eflow over chain $L(i, j)$ is established by performing entanglement swapping on the repeater $N_{S(i,j)}$ in the end of a swapping duration. Thus, in Line 1, U is initialized with $U = \{(i, j, S(i, j))\}$ to represent the end group. We push U to the queue $\mathfrak{Q}$ in Line 2. We then use U to construct the remaining groups (the previous swapping duration) for establishing an eflow over $L(i, j)$. We traverse all tuples in U to identify the tuples in the next group (the previous swapping duration). For ease of description, we use U' to represent the next group of U. For each tuple $(\tilde{i}, \tilde{j}, S(\tilde{i}, \tilde{j}))$ in U, to establish the eflow over $L(\tilde{i}, \tilde{j})$, the eflows over $L(\tilde{i}, S(\tilde{i}, \tilde{j}))$ and $L(S(\tilde{i}, \tilde{j}), \tilde{j})$ should have been established. By performing entanglement swapping operations on $N_{S(\tilde{i},S(\tilde{i},\tilde{j}))}$ and $N_{S(S(\tilde{i},\tilde{j}),\tilde{j})}$, one can establish the eflows over $L(\tilde{i}, S(\tilde{i}, \tilde{j}))$ and $L(S(\tilde{i}, \tilde{j}), \tilde{j})$, respectively. We insert the two tuples $(\tilde{i}, S(\tilde{i}, \tilde{j}), S(\tilde{i}, S(\tilde{i}, \tilde{j})))$ and $(S(\tilde{i}, \tilde{j}), \tilde{j}, S(S(\tilde{i}, \tilde{j}), \tilde{j}))$ to U' in Line 6. Until all the tuples in U have been traversed, we obtain U' and push it to $\mathfrak{Q}$ in Line 8. Then, we update U as U' in Line 9 to construct the next group of U'. The while-loop from Line 3 to Line 10 stops when U' becomes empty, indicating that U is the first swapping duration (group). The computational complexity of Algorithm 2 is $O(j - i)$ with the input of i and j. This is because we traverse all tuples in $\mathfrak{Q}$ and the number of tuples is equal to the number of intermediate nodes in $L(i, j)$, i.e., $j - i - 1$. Therefore, the computational complexity for finding the optimal swapping structure of $L(0, n)$ from $\mathbf{S}$ is $O(n)$.

Now let us present the computational complexity of Algorithm 1. For the C-loop, its computational complexity is $O(n^2)$ as there are total $O(n^2)$ sub-chains. For the K-loop, the computational complexity is $O(n)$ as K-loop searches all intermediate repeaters in the sub-chain $L(i, j)$. Hence, the total computational complexity of Pure-DP shown in Algorithm 1 for searching for the optimal repeaters is $O(n^3)$.

B. K-Loop Pruning Technique

In this subsection, we introduce a heuristic pruning technique, K-loop pruning, proposed in [17] to improve Pure-DP.

As shown in Algorithm 1, Pure-DP searches for the optimal repeaters among all nodes for every sub-chain in the K-loop in Line 7. However, there is no need to search all the nodes. [17] introduced the following heuristic: the optimal eflow over a long repeater chain with i_1+i_2 hops can be established using the eflow on its two short disjoint sub-chains with i_1 and i_2 hops with high probability as long as,

$$|i_1 - i_2| \leq 2\log(i_1 + i_2). \tag{5}$$

By letting $n = i_1 + i_2$ in (5), it follows that $i_1, i_2 \in [n/2-\log n, n/2+\log n]$, which indicates that the index of the optimal repeater for an n-hop chain is in $[n/2 - \log n, n/2 + \log n]$ with high probability. The base of the log function is 2 in this paper. Goodenough et al. [17] proved that such a heuristic is sufficient and proposed a pruning technique to reduce the complexity of Pure-DP. Specifically, for a sub-chain $L(i, j)$, the search interval for the optimal repeater can be narrowed by using (5), and the recurrence structure in (4) can be rewritten as:

$$R(i,j) = \max_k \{\min\{R(i,k), R(k,j)\} \cdot p_{swap}^{(k)}\},$$
$$\tfrac{(i+j)}{2} - \log(j-i) \leq k \leq \tfrac{(i+j)}{2} + \log(j-i). \tag{6}$$

By incorporating the pruning technique in (6), [17] proposed an algorithm known as NOPTS-DP (Near-OPTimal Structure Dynamic Programming). NOPTS-DP suggested a minor modification of Pure-DP, achieved by replacing the search interval in Line 7 in Algorithm 1 with the search interval from (6). The algorithm is near-optimal, although there would be some exceptional cases where the indices of the optimal repeaters are not in the search interval. We would like to point out that NOPTS-DP performs well even in these exceptional cases, as the repeaters identified by NOPTS-DP are close to the optimal ones.

Compared with Pure-DP, NOPTS-DP has a lower computational complexity. Specifically, with the pruning technique in (6), the length of the search interval in the K-loop is reduced from $O(n)$ to $O(\log n)$ and the C-loop is the same as that in

Pure-DP. Hence, the computational complexity of NOPTS-DP is $O(n^2 \log n)$.

In the following, for ease of presentation, we use a 2D array, $\mathbf{K} \in \mathbb{Z}^{(n+1)\times(n+1)}$, to store the lengths of the search intervals in the K-loop in Algorithm 1. Specifically, for the NOPTS-DP algorithm, $K(i,j)$ equals $\log(j-i)$ as the length of the search interval for the sub-chain $L(i,j)$ is $2\log(j-i)$ as indicated in (6). As such, the search interval for the sub-chain $L(i,j)$ in the K-loop can be expressed as follows,

$$I(i,j) = [\tfrac{(i+j)}{2} - K(i,j), \tfrac{(i+j)}{2} + K(i,j)]. \quad (7)$$

Using $\mathbf{K}$, we narrow the search interval for the K-loop to speed up the search for the near-optimal swapping structure, and we call such pruning technique as the K-loop pruning technique.

Algorithm 3 Findchildren

Input: a set of tuples representing the parent sub-chains $\mathcal{L}$, K-loop pruning array $\mathbf{K}$.
Output: a set of tuples representing the children sub-chains $\tilde{\mathcal{L}}$.

```
Initialize L̃ as an empty set;
for L(i,j) ∈ L do
    Determine I(i,j) from K(i,j) as per (7);
    for k ∈ I(i,j) do
        Insert L(i,k) and L(k,j) to L̃;
    end
end
return L̃;
```

C. Further Complexity Reduction in NOPTS-DP

It is important to note that the K-loop pruning technique in NOPTS-DP is not the only method for reducing computational complexity. There are other efficient techniques that can also reduce complexity by leveraging the same heuristic in (5). In this subsection, our objective is to design an algorithm that maximizes the reduction of computational complexity in Pure-DP using heuristic in (5). We introduce our C-loop pruning technique, present an efficient algorithm for computing the C-loop pruning array, describe our fast heuristic entanglement distribution algorithm FastHED, and conclude with a discussion on the two pruning techniques.

1) C-Loop Pruning Technique: Our C-loop pruning technique also uses the heuristic in (5). First, we introduce two key concepts.

The first concept is "*division*", denoting the derivation of children sub-chains from their parent sub-chains using (5). Fig. 7(a) shows the relation between a parent sub-chain (the square marked with striped shading) and its children sub-chains (the blue squares). For any sub-chain $L(i,j)$, its children are collected in $\{L(i,k), L(k,j) | k \in [(\frac{i+j}{2}) - \log(j-i), (\frac{i+j}{2}) + \log(j-i)]\}$. The details of a division are shown in Algorithm 3. Algorithm 3 takes inputs of a set $\mathcal{L}$ of sub-chains (parent set) and K-loop array $\mathbf{K}$, and returns the set $\tilde{\mathcal{L}}$ of children sub-chains. In Line 2 to 7, for each $L(i,j) \in \mathcal{L}$, we obtain its children as shown in Fig. 7(a) and insert the children to the children set $\mathcal{L}'$. The output is the union of children of every sub-chains in $\mathcal{L}$.

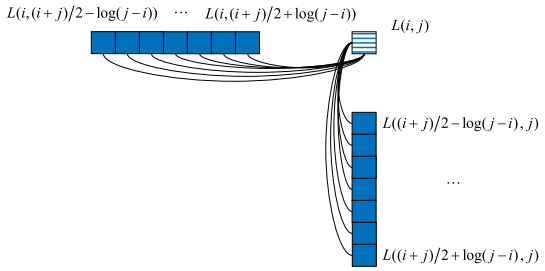
Now let us introduce the second key concept.

Definition 1 (Level): We call the set of new sub-chains created by division from $\mathcal{L} = \{L(0,n)\}$ for s times ($s \geq 0$) as the s-th level *children sub-chains and denote the children set as $\mathcal{L}^{(s)}$.*

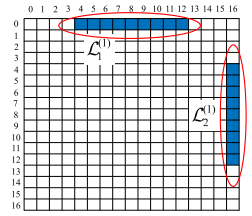
By Definition 1, $\mathcal{L}^{(0)}$ is the set of the 0^{th} level sub-chains, $\{L(0,n)\}$, with no division. With divisions, we can find all the descendant sub-chains of the ancestor chain $L(0,n)$. Division on $\mathcal{L}^{(0)}$ generates the set of 1st level sub-chains $\mathcal{L}^{(1)} = \mathcal{L}_1^{(1)} \bigcup \mathcal{L}_2^{(1)}$ as its children, where $\mathcal{L}_1^{(1)} = \{L(0,i_1) | i_1 \in [n/2 - \log n, n/2 + \log n]\}$ and $\mathcal{L}_2^{(1)} = \{L(i_1,n) | i_1 \in [n/2 - \log n, n/2 + \log n]\}$. Here, we use a notation $\mathcal{L}_k^{(s)}$ (called a child set) to denote a component of a set $\mathcal{L}^{(s)}$ (called a children set). Upon the s-th division, we have 2^s child sets $\mathcal{L}_k^{(s)}, k \in [1, 2^s]$ at level s, and the children set $\mathcal{L}^{(s)}$ is a union of these child sets, i.e., $\mathcal{L}^{(s)} = \bigcup_{k=1}^{2^s} \mathcal{L}_k^{(s)}$. Now let us explain how to obtain child sets at level s from the children set $\mathcal{L}^{(s-1)}$. Note that $\mathcal{L}^{(s-1)}$ also can be expressed as a union of child sets at level $s-1$: $\bigcup_{k=1}^{2^{s-1}} \mathcal{L}_k^{(s-1)}$. For all children of $\mathcal{L}_k^{(s-1)}$, they are categorized into two groups, namely a left child set and a right child set. The left child set is $\mathcal{L}_{2k-1}^{(s)} = \bigcup_{k \in I(i,j)} \{L(i,k) | \forall L(i,j) \in \mathcal{L}_k^{(s-1)}\}$ and the right child set is $\mathcal{L}_{2k}^{(s)} = \bigcup_{k \in I(i,j)} \{L(k,j) | \forall L(i,j) \in \mathcal{L}_k^{(s-1)}\}$. For each child set $\mathcal{L}_k^{(s-1)}$ at level $s-1$, it is associated with two child sets, indexed as $2k-1$ and $2k$ at level s. The left child set $\mathcal{L}_{2k-1}^{(s)}$ contains all the sub-chains with the left end nodes N_i, while the right child set $\mathcal{L}_{2k}^{(s)}$ contains the sub-chains with the right end nods N_j for each $L(i,j) \in \mathcal{L}_k^{(s-1)}$. Thus, when k is even, $\mathcal{L}_k^{(s)}$ is the right child set of $\mathcal{L}_{k/2}^{(s-1)}$. Conversely, $\mathcal{L}_k^{(s)}$ is the left child set of $\mathcal{L}_{(k-1)/2}^{(s-1)}$ when k is odd. At the bottom level $\lceil \log n \rceil - 1$, we would have the sub-chains at all the levels for the chain $L(0,n)$. The set of these sub-chains is defined as $\mathfrak{L} = \bigcup_{s=0}^{\lceil \log n \rceil - 1} \mathcal{L}^{(s)}$. Fig. 7(d) and Fig. 7(e) show two instances of $\mathfrak{L}$ for a 16-hop chain $L(0,16)$ and a 256-hop chain $L(0,256)$, respectively, where the sub-chains in $\mathfrak{L}$ in the figures are colored in dark blue.

Take a 16-hop chain as an example, as shown in Fig. 7(b)-7(d), the small square at the i-row and the j-column represents the sub-chain $L(i,j)$. The collection of the squares (marked in dark blue) in the upper part of Fig. 7(b) represents $\mathcal{L}_1^{(1)}$, and the collection of the squares (marked in dark blue) in the right part of Fig. 7(b) represents $\mathcal{L}_2^{(1)}$.

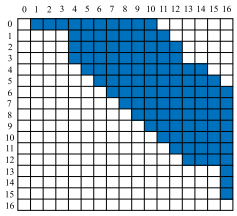
Now we explain the relation of children sets between any two adjacent levels. Essentially, $\mathcal{L}^{(s+1)}$ is the children set of $\mathcal{L}^{(s)}$. Hence, $\mathcal{L}^{(s+1)}$ can be obtained from $\mathcal{L}^{(s)}$ using Algorithm 3. Fig. 7(c) shows $\mathcal{L}^{(2)}$ of a 16-hop chain, which can be obtained from $\mathcal{L}^{(1)}$ shown in Fig. 7(b). Finding the near-optimal repeater for any sub-chain $L(i,j)$ at level s requires the knowledge of the EDRs of all its children sub-chains at level $s+1$. In other words, the EDRs of sub-chains in $\{L(i,k), L(k,j) | k \in [(\frac{i+j}{2}) - \log(j-i), (\frac{i+j}{2}) + \log(j-i)]\}$ should be computed before the EDR of $L(i,j)$. Once the EDRs of the sub-chains in $\mathcal{L}^{(s+1)}$ are available, we can derive the near-optimal repeater indices and EDRs of the sub-chains in $\mathcal{L}^{(s)}$ using (6). These sub-chains can be repeatedly used to calculate the EDRs of their parent sub-chains until reaching the ancestor chain $L(0,n)$. Therefore,



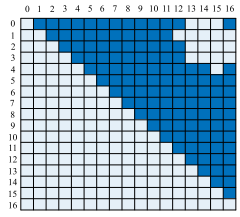
(a) A parent sub-chain (marked with striped shading) in level s and its two child sub-sub-chain sets (blue) in level $s+1$ using (4).



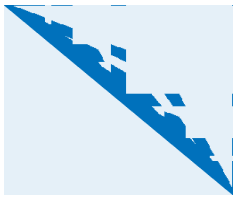
(b) Level 1 of 16-hop



(c) Level 2 of 16-hop



(d) $\mathbf{C}$ of 16-hop



(e) $\mathbf{C}$ of 256-hop

Fig. 7. Example of the C-loop pruning array construction for 16-hop and 256-hop. The level 2 array in Fig. 7(c) can be obtained from level 1 array in Fig. 7(b) following the rule as shown in Fig. 7(a).

finding the near-optimal swapping structure for the ancestor chain involves computing the EDRs of the descendant sub-chains.

In contrast, NOPTS-DP needs to compute the sub-chains marked in light blue (except the sub-chains in the lower left triangular region) in Fig. 7(d) and Fig. 7(e). These sub-chains have no contribution in finding the near-optimal repeater in $L(0,n)$ since they are outside the searching interval in (6).

We use a 2D boolean array $\mathbf{C} \in \mathbb{Z}^{(n+1)\times(n+1)}$ to indicate the existence of a sub-chain in $\mathfrak{L}$ and let $C(i,j)$ of $\mathbf{C}$ be 1 (0) if $L(i,j) \in \mathfrak{L}$ ($L(i,j) \notin \mathfrak{L}$). For example, for $\mathfrak{L}$ shown Fig. 7(e), we set the entries of $\mathbf{C}$ that correspond to the sub-chains marked in blue and other entries to 0. Thus, $\mathbf{C}$ contains the information of all sub-chains that need to be computed for finding the near-optimal swapping structure in $L(0,n)$ with the heuristic in (5). As FastHED prunes the redundant computation in NOPTS-DP for the input chain, the searching processes of the two algorithms are identical, resulting in the same output swapping structure.

2) Construction of the C-Loop Pruning Array $\mathbf{C}$*:* The sub-chains in $\mathfrak{L}$ between two adjacent levels have overlaps as they are created by dividing the chain $L(0,n)$ for multiple times. For example, Fig. 7(b) and Fig. 7(c) show the 1st-level sub-chains $\mathcal{L}^{(1)}$ and 2nd level sub-chains $\mathcal{L}^{(2)}$ of the 16-hop chain $L(0,16)$, respectively. $\mathcal{L}^{(1)}$ overlaps with $\mathcal{L}^{(2)}$ as the sub-chains $L(0,4), L(0,5), \ldots, L(0,10)$ in the first row

Algorithm 4 Construction of C-Loop Pruning Array $\mathbf{C}$

Input: Chain size n, K-loop pruning array $\mathbf{K}$
Output: C-loop pruning array $\mathbf{C}$
1 Initialize $\mathcal{L}$ as an set $\{(0,n)\}$; *// Representing $L(0,n)$ at 0-th level*
2 Initialize $\mathfrak{L}$ as an empty set and $\mathfrak{L} \leftarrow \mathfrak{L} \bigcup \mathcal{L}$;
3 Mark $(0,n)$ as *IsOverlapped = 1* and set *IsOverlapped = 0* for all (i,j) by default;
4 **for** $s \in [0, \lceil \log n \rceil - 1]$ **do**
5 Initialize a new set $\mathcal{L}'$; *// The next level*
6 $\tilde{\mathcal{L}} \leftarrow$ FINDCHILDREN$(\mathcal{L}, \mathbf{K})$; *// see Algorithm 3*
7 **for** $(i,j) \in \tilde{\mathcal{L}}$ **do**
8 **if** (i,j) *with IsOverlapped = 0* **then**
9 Mark (i,j) as *IsOverlapped = 1*;
10 Insert (i,j) to $\mathcal{L}'$;
11 **end**
12 **end**
13 $\mathcal{L} \leftarrow \mathcal{L}'$;
14 $\mathfrak{L} \leftarrow \mathfrak{L} \bigcup \mathcal{L}$;
15 **end**
16 $\mathbf{C} \leftarrow \mathfrak{L}$; *// Transform $\mathfrak{L}$ into a 2D array*

and $L(4,16), L(5,16), \ldots, L(10,16)$ appear in both $\mathcal{L}^{(1)}$ and $\mathcal{L}^{(2)}$. Such overlaps between the children sets at the adjacent levels result in redundant search of near-optimal repeaters in the same sub-chain.

To address this issue, we introduce a C-loop pruning construction algorithm in Algorithm 4. Let the tuple $(i,j) \in \mathfrak{L}$ represent the sub-chain $L(i,j)$ that must be computed for finding the near-optimal swapping structure for $L(0,n)$. To construct $\mathbf{C}$, we would first construct $\mathfrak{L}$.

The construction of $\mathfrak{L}$ starts with $\mathfrak{L} = \mathcal{L} = \{(0,n)\}$ in Line 2. Each tuple is associated with a Boolean variable, *IsOverlapped*, used to determine whether a tuple overlaps with others. We first create the children set $\tilde{\mathcal{L}}$ of $\mathcal{L}$ using FINDCHILDREN in Line 6. Note that the new tuples in $\tilde{\mathcal{L}}$ may have overlaps with those in $\mathcal{L}$. Hence, we need to eliminate the overlapped tuples using *IsOverlapped* from Line 7 to Line 12. The set without overlapped tuples is saved in $\mathcal{L}'$. We thus insert it to $\mathfrak{L}$ and update $\mathcal{L}$ and $\mathfrak{L}$ with $\mathcal{L} \leftarrow \mathcal{L}'$ in Line 13 and 14, respectively. Specifically, if a new tuple is not overlapped with previous tuples, i.e., *IsOverlapped = 0*, we mark it as *IsOverlapped = 1* and insert it to the non-overlap set $\mathcal{L}'$ in Line 9 and 10. Thus, the overlaps are eliminated. At last, we construct $\mathbf{C}$ using $\mathfrak{L}$ in Line 16: we set $C(i,j)$ to 1 (0), if the tuple $(i,j) \in \mathfrak{L}$ ($(i,j) \notin \mathfrak{L}$).

3) Further Discussions on $\mathbf{K}$ *and* $\mathbf{C}$*:* We can apply the K-loop and C-loop pruning techniques to Pure-DP in Algorithm 1 with $\mathbf{K}$ and $\mathbf{C}$, resulting in our algorithm FastHED. The utilization of the two pruning techniques in FastHED is shown in Algorithm 5. Algorithm 5 is a modification of Algorithm 1 and incorporates an additional input of the K-loop pruning array $\mathbf{K}$. We first construct the corresponding C-loop pruning array $\mathbf{C}$ of the K-loop pruning array $\mathbf{K}$ in Line 1. Note that the construction of $\mathbf{C}$ is before the DP search following a top-down manner starting with the ancestor $L(0,n)$. Then we start the DP search. In Line 8,

Algorithm 5 Fast Heuristic Entanglement Distribution (FastHED, Modified From Algorithm 1)

Input: The same as Algorithm 1 and pruning array $\mathbf{K}$
Output: The same as Algorithm 1

$\mathbf{C} \leftarrow$ Algorithm 4 (n, $\mathbf{K}$); *// Construct C-loop pruning array*
for $i \in [0, n-1]$ **do**
  $R(i, i+1) \leftarrow 1/p_{gen}^{(i,i+1)}$;
end
for $d \in [2, n]$ **do**
  for $i \in [0, n-d]$ **do**
    $j = i + d$;
    if $C(i,j) == 1$ *// C-loop pruning* **then**
      Determine $I(i,j)$ with $K(i,j)$ using (7);
      for $k \in I(i,j)$ *// K-loop pruning* **do**
        Find the optimal repeater N_{k^*};
      end
      $S(i,j) \leftarrow k^*$;
      $R(i,j) \leftarrow \min\{R(i,k^*), R(k^*,j)\} \cdot p_{swap}^{(k)}$;
    end
  end
end
$\mathfrak{O} \leftarrow$ FINDORDER($\mathbf{S}$, 0, n);
return $\mathbf{S}$, $\mathbf{R}$ and $\mathfrak{O}$

using the C-loop pruning technique, we only need to search the sub-chains with $C(i,j) == 1$ as other sub-chains are redundant for $L(0,n)$. In Line 9, using the K-loop pruning technique, we can narrow the search interval with $K(i,j)$ using (7).

The relation between $\mathbf{K}$ and $\mathbf{C}$ is as follows. $\mathbf{K}$ determines the search intervals of K-loop in NOPTS-DP. $\mathbf{C}$ represents all the useful sub-chains needed to be computed for $L(0,n)$ by leveraging $\mathbf{K}$. Algorithm 4 shows how to obtain $\mathbf{C}$ from any given $\mathbf{K}$. $\mathbf{C}$ depends on the specific pruning array $\mathbf{K}$. For instance, one can set the value of K-loop pruning array as $K(i,j) = \lceil w \log(j-i) \rceil$, where $1 \leq w \leq 2$. In particular, one would narrow the search interval of K-loop by selecting a small w to search for the optimal repeaters, especially for long chains, as it is more likely that the optimal repeater appears in the middle of the chain when the chain is long.

D. Analysis

We present a performance analysis of FastHED, including computational complexity, classical communication complexity and the function of EDR over the length of a chain.

We first theoretically analyze the computational complexity of Algorithm 5 for FastHED as the following Theorem. The proof is presented in Appendix A.

Theorem 1: For an n-hop repeater chain, the computational complexity of FastHED is upper bounded by $O(n(\log n)^2 \log\log n)$, if $\log n - 2\log\log n \leq (2\log n)/3$ ($n \leq 2^{29}$), and $O(n(\log n)^3)$, otherwise.

We then analyze the complexity of Algorithm 4. Since it requires $O(\log n)$ time to find the near-optimal repeater in the K-loop in Algorithm 5, we have the following corollary,

Corollary 1: The number of sub-chains to be computed is $O(n(\log n)^2)$ (precisely, $O(n \log n \log\log n)$ when $n \leq 2^{29}$). FINDCHILDREN in Algorithm 3 finds $2\log(j-i)$ children sub-chains for each parent sub-chain $L(i,j)$. Thus, the total computational complexity of Algorithm 4 can be expressed as the product of the size of $\mathfrak{L}$ and the complexity of FINDCHILDREN. By Corollary 1, the total number of sub-chains (the size of $\mathfrak{L}$) is $O(n(\log n)^2)$ ($O(n \log n \log\log n)$ when $n \leq 2^{29}$). Therefore, the computational complexity for the construction of $\mathbf{C}$ (Algorithm 4) is $O(n(\log n)^3)$ ($O(n(\log n)^2 \log\log n)$ when $n \leq 2^{29}$).

In addition, we discuss the classical communication complexity and how the EDR falls as a function of the length of a repeater chain for FastHED in our supplementary materials.

V. BEST-FIRST ENTANGLEMENT DISTRIBUTION FRAMEWORK

In this section, we first introduce a best-first entanglement distribution framework aimed at evaluating the average EDR performance of algorithms mentioned in Sec. IV. Then, the framework is integrated to tackle the swapping structure optimization in a chain with time varying parameters.

In the practical distribution of entanglements, the conducted swapping structure has a great impact on the EDR. Therefore, a good swapping structure should be conducted as soon as possible for high EDR.

Heuristic DP algorithms (NOPTS-DP and our FastHED algorithm) leverage $K(i,j) = \lceil w \log(j-i) \rceil$, where $1 \leq w \leq 2$, to reduce computational complexity. Moreover, the values for w are not necessarily the same for all the sub-chains. This leads to distinct swapping structures and different EDRs when using heuristic DP search algorithm with different w (different $\mathbf{K}$). Hence, we randomly select m different $\mathbf{K}$ by randomly selecting m different values of w for each sub-chain. With these m different $\mathbf{K}$, we run the heuristic DP search (NOPTS-DP or ours) for m times, where the i-th run takes T_i time. We assume that the parameters of the repeater chain (such as p_{gen} and p_{swap}) remain unchanged throughout all runs.

Now let us describe our best-first framework. The entanglements are distributed while the search algorithms are running. During the first run (T_1), we use the swapping structure of BDCZ to distribute entanglement and we denote its corresponding EDR as λ_0. By the end of T_1, we complete the 1st run and obtain a new swapping structure. The corresponding EDR is λ_1 and we start the second run. At the time $t = T_1 + T_2$, according to the obtained EDRs (λ_1 and λ_2), we select the better swapping structure that results the higher EDR to be conducted in T_2. This procedure continues and we keep selecting the best swapping structure among the obtained ones so far for the subsequent run, thereby calling it best-first. Therefore, during the distribution process, we can obtain an EDR of λ_0 in T_1, $\max\{\lambda_0, \lambda_1\}$ in T_2, and $\max\limits_{0 \leq j < i}\{\lambda_j\}$ in T_i. We can obtain the average EDR over the m runs:

$$\overline{\lambda} = \frac{\sum_{i=1}^{m} T_i \max_{0 \leq j < i}\{\lambda_j\}}{\sum_{i=1}^{m} T_i}. \tag{8}$$

The details of our best-first framework for distributing entanglement are outlined in Algorithm 6 and illustrated in

Algorithm 6 The Best-first Framework

Output: The desired entanglements
1 Select m K-loop pruning arrays $\{\mathbf{K}_i | 1 \leq i \leq m\}$;
2 $\mathfrak{Q}_{curr} \leftarrow BDCZ$; *// BDCZ swapping structure*
3 **for** i *in* $[1, m]$ **do**
4 *// At the beginning of the i-th run (T_i).*
5 Conduct $\mathfrak{Q}_{curr}$ to distribute entanglement;
6 Select $\mathbf{K}_i$ as current K-loop pruning array;
7 $(\mathfrak{Q}, \lambda_i) \leftarrow$ Heuristic-DP-search($\mathbf{K}_i$);
8 *// Update $\mathfrak{Q}_{curr}$ in the end of the i-th run.*
9 **if** $\mathfrak{Q}$ *is better than* $\mathfrak{Q}_{curr}$ *(i.e.,* $\lambda_i \geq \max_{0 \leq j < i} \{\lambda_j\}$*)* **then**
10 $\mathfrak{Q}_{curr} \leftarrow \mathfrak{Q}$;
11 **end**
12 **end**

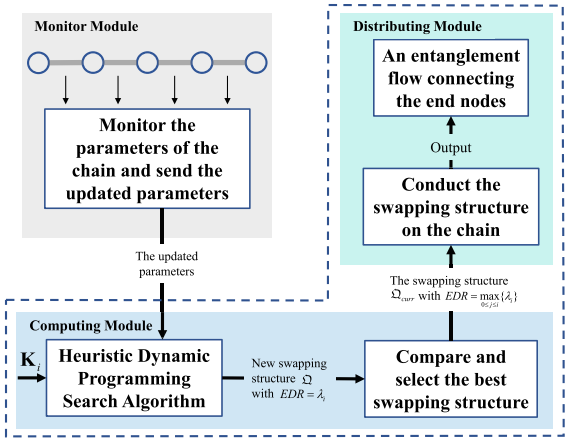
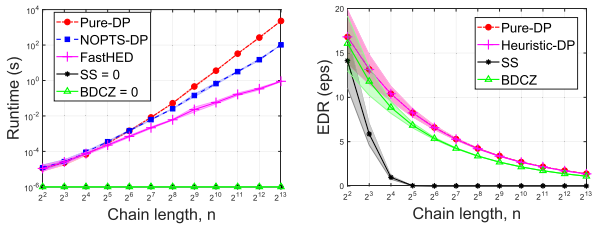


Fig. 8. The best-first framework for entanglement distribution.

Fig. 8. Specifically, We first select a series of K-loop pruning arrays in Line 1. At the beginning of each run (T_i), we conduct the selected swapping structure to distribute entanglement in Line 5, which corresponds to the Distributing Module in Fig. 8. Simultaneously, we begin running Heuristic DP search algorithm with the newly selected K-loop pruning array $\mathbf{K}_i$ to find the swapping structure and compute its corresponding EDR in Line 6 and 7. Subsequently (Line 9 to 11), we compare the newly obtained swapping structure $\mathfrak{Q}$ with the current conducted one $\mathfrak{Q}_{curr}$ based on their EDRs and select the better one as the structure for the subsequent run, as shown in the Computing Module in Fig. 8. In this framework, it is clear that the average EDR can be enhanced by conducting the better swapping structure once it is identified. This motivates the development of a fast algorithm.

The proposed best-first framework can be integrated to tackle the swapping structure optimization problem in a chain with time varying parameters. A chain with time varying parameters refers to a scenario where the success probabilities fluctuate over time. This is because that quantum devices may be not always stable (i.e., influenced by circumstances) and the quantum memory suffers from decoherence [52], [53]. For entanglement distribution on a chain with varying parameters, we incorporate a Monitor Module (marked in gray in Fig. 8) into our best-first framework. The Monitor Module captures the varying parameters and updates them as soon as possible as the input for input into the Computing Module. Upon receiving new parameters, the Computing Module terminates the current run and initiates a new run with new parameters. Hence, a fast algorithm is required to quickly respond to varying parameters, avoiding outputting an outdated swapping structure with outdated input parameters.



(a) Runtime vs. chain length (b) EDR vs. chain length

Fig. 9. Runtime and EDR under different chain lengths.

VI. Numerical Results

In this section, we first present the EDRs and runtimes of the three algorithms mentioned in Sec. IV. Second, we examine the effects of different K-loop pruning arrays on the EDR. In addition, we evaluate the EDRs of the three algorithms using our best-first framework. At last, we compare the three algorithms under a chain with time varying parameters.

A. EDR and Runtime of Different Algorithms

We conduct experiments of entanglement distribution on repeater chains of varying lengths and examine the EDRs of swapping structures computed by different algorithms. We consider a non-uniform repeater chain where the elementary chains have different entanglement generation probabilities. According to [8], [9], and [21], we randomly vary $p_{gen}^{(i,i+1)}$ from 0.2 to 0.5, set the operation frequencies f of the elementary chains to 100 Hz, and set $p_{swap}^{(k)}$ to 0.8.[6] We compare FastHED (Algorithm 5) with two existing algorithms: Pure-DP (Algorithm 1) and NOPTS-DP [17]. Both NOPTS-DP and FastHED are heuristic dynamic programming search algorithms, and we leverage the same heuristic: $K(i,j) = \log(j-i), \forall i < j \in [0, n]$, resulting in an identical EDR of the two algorithms. Hence, when discussing the EDR, we unify NOPTS-DP and FastHED under a single notation, Heuristic-DP We also conduct experiments on the strict parallel symmetric swapping structure (BDCZ) and the strict Serial Swapping structure (SS). SS is a swapping structure that sequentially performs swapping operations from the left end node to the right end node. We run our experiments on a machine with an Intel i7-12700@2.1GHz processor and 16GB RAM. The results are the average of 1000 Monte Carlo trials (10 Monte Carlo trials for $n \geq 2^{11}$).

Fig. 9 shows the runtimes and EDRs of the three algorithms on repeater chains with length n ranging from 2^2 (4) to 2^{13} (8,192) hops. Note that the DP-based algorithms can

[6] Although 0.5 is an upper bound for the success probability of photonic swapping, the probability can be increased to exceed 0.5 in [54] and [55].

handle repeater chains with arbitrary lengths, while BDCZ is specifically designed for chains with lengths that are powers of 2. To benchmark with BDCZ, we conduct experiments using chains with lengths that are powers of 2.

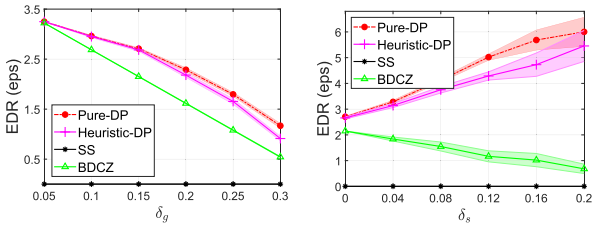
As shown in Fig. 9(a), on short chains with no more than 2^5-hops, NOPTS-DP and FastHED exhibit longer runtimes than Pure-DP as they both require additional runtime to compute the K-loop pruning arrays. However, such additional runtimes in NOPTS-DP and FastHED are insignificant compared to the runtime of searching the near-optimal swapping structures in long repeater chains. This can be verified by the trend that the runtime of FastHED is much lower than other algorithms when the length of the chain exceeds 2^6 (64) hops. Especially, for an 8,192-hop chain, Pure-DP requires average 2,297 seconds (about 0.64 hours) to find the optimal swapping structure, NOPTS-DP finds a near-optimal structure within a few minutes, and FastHED finds the same near-optimal structure in 1 second. The reason is that NOPTS-DP reduces the search space for the optimal swapping structure by leveraging the K-loop pruning technique and FastHED can further prune the search space by using both C-loop and K-loop pruning techniques. Following specific rules (strict parallel symmetric or serial), the constructions of swapping structures in BDCZ or SS do not require optimal swapping structure searching algorithms. Hence, the runtime of BDCZ or S is 0, as shown at the bottom of Fig. 9(a).

As can be observed in Fig. 9(b), FastHED, NOPTS-DP, and Pure-DP all have larger EDRs than the SS structure. The reason is that a near-optimal or optimal swapping structure resulting from FastHED, NOPTS-DP, and Pure-DP would significantly increase the EDR compared to the serial swapping structure. Moreover, FastHED and NOPTS-DP have almost the same EDRs as the Pure-DP since FastHED and NOPTS-DP both can find near-optimal structures that perform closely to the optimal one computed by Pure-DP.

In addition, we conduct experiments to show the effects on EDR of standard deviations of success probabilities. Our FastHED can adapt to success probabilities across all regions. In the following, we demonstrate how effectively FastHED responds to different parameters. We conduct two experiments for different standard deviations of link success probabilities and swap success probabilities, respectively.

First, we fix $p_{swap}^{(k)}$ at 0.8, randomly take the link success probability from $[0.35-\delta_g, 0.35+\delta_g]$, and vary the standard deviation δ_g from 0.05 to 0.3, for a 1024-chain. As illustrated in Fig. 10(a), with increasing δ_g, the EDRs decrease of all algorithms and the advantage obtained from optimizing becomes lager. This is because the chain becomes less homogeneous with a larger δ_g. Due to using a fixed heuristic, the heuristic DP exhibits a lower EDR compared to Pure-DP as δ_g increases. This observation further motivates the development of different heuristics to better accommodate such heterogeneous chains. The following Sec. VI-B shows how different heuristics used in DP-based algorithms influence the EDR.

Second, the main results of different standard deviations of swap success probabilities are illustrated in Fig. 10(b). In this experiment, the $p_{swap}^{(k)}$ of each repeater is randomly taken from $0.8-\delta_s$ to $0.8+\delta_s$. We vary the standard deviation δ_s from 0 to 0.2 for a 1024-hop chain, where the link success



(a) EDR vs. δ_g. (b) EDR vs. δ_s.

Fig. 10. Effects of standard deviations of success probabilities.

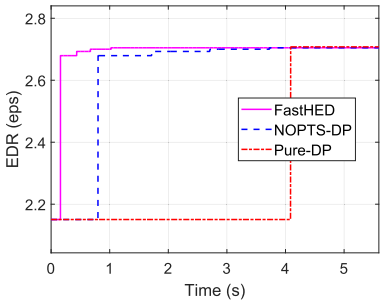


Fig. 11. EDRs of different algorithms in our best-first framework.

probability is randomly taken from $[0.2, 0.5]$. Due to the fixed swapping structure, BDCZ cannot adapt to the heterogeneous swap success probabilities, resulting in a decreasing EDR as the chain becomes less homogeneous, while the DP-based algorithms adapt well after swapping structure optimization.

B. Effects of K-Loop Pruning Arrays on EDR and Runtime

We set 4 different $\mathbf{K}$s and compute the corresponding swapping structures using the heuristic DP search algorithms (NOPTS-DP and FastHED) to examine the effects of K-loop pruning arrays on EDR. Specifically, for each $\mathbf{K}$, we set $K(i,j)$ $(\forall i \in [0, n-2], j \in [i+1, n])$ for an n-hop chain as follows,

- $K_1(i,j) = \log(j-i)$. $\mathbf{K}_1$ serves as the baseline, where we set the search intervals according to (6) using the basic heuristic in (5).
- $K_2(i,j) = \begin{cases} 2\log(j-i), & \text{if } j-i > n/3, \\ \log(j-i), & \text{otherwise.} \end{cases}$

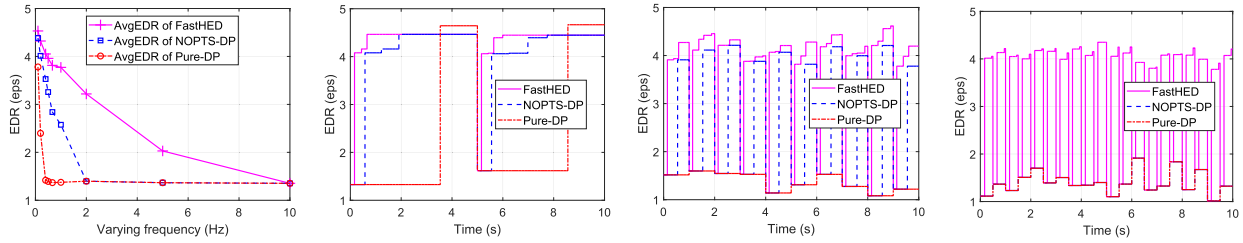
  $\mathbf{K}_2$ extends the search intervals for the sub-chains with lengths longer than $n/3$ to twice the length in $\mathbf{K}_1$, aiming to improve the swapping structures for long sub-chains.
- $K_3(i,j) = \begin{cases} 2\log(j-i), & \text{if } j-i < 2n/3, \\ \log(j-i), & \text{otherwise.} \end{cases}$

  $\mathbf{K}_3$ extends the search intervals for the sub-chains with lengths shorter than $2n/3$ to twice the length in $\mathbf{K}_1$, trying to find better swapping structures for short sub-chains.
- $K_4(i,j) = 2\log(j-i)$. $\mathbf{K}_4$ extends all search intervals to twice the length in $\mathbf{K}_1$ to improve swapping structures for all the sub-chains.

Using the same parameter setup as in Sec. VI-A, we study the EDRs and runtimes of different algorithms with different $\mathbf{K}$s on a $1,024$-hop chain. We run Monte Carlo experiments

TABLE II
RUNTIME, EDR AND OPTIMAL RATIO OF DIFFERENT ALGORITHMS WITH DIFFERENT $\mathbf{K}$S

	Pure-DP	NOPTS-DP ($\mathbf{K}_1$)	NOPTS-DP ($\mathbf{K}_2$)	NOPTS-DP ($\mathbf{K}_3$)	NOPTS-DP ($\mathbf{K}_4$)	FastHED ($\mathbf{K}_1$)	FastHED ($\mathbf{K}_2$)	FastHED ($\mathbf{K}_3$)	FastHED ($\mathbf{K}_4$)
C-loop runtime (s)	-	-	-	-	-	0.0837	0.1025	0.1777	0.1924
DP search time (s)	4.0881	0.8013	0.9172	0.9938	1.0165	0.0761	0.0989	0.1320	0.1597
Total runtime (s)	4.0881	0.8013	0.9172	0.9938	1.0165	**0.1598**	**0.2014**	**0.3097**	**0.3521**
EDR (eps)	2.7077	2.6794	2.6929	2.7001	2.7048	2.6794	2.6929	2.7001	2.7048
Optimal ratio	100%	61.6%	85.4%	92%	97.5%	61.6%	85.4%	92%	97.5%



(a) EDRs vs. varying frequency (b) Varying period = 5 s (0.2 Hz) (c) Varying period = 1 s (1 Hz) (d) Varying period = 0.5 s (2 Hz)

Fig. 12. EDRs on a chain with varying parameters.

where we repeatedly conduct entanglement distribution experiments 1,000 times and present average results in Table II.

Table II shows the runtimes, EDRs, and the optimal ratios of different algorithms with different $\mathbf{K}$s. The optimal ratio is the ratio of the number of experiments with errors less than 1% from the optimal EDR to the total number of experiments. We use the optimal ratio to measure the difference between the EDR of the near-optimal swapping structure and the EDR of the optimal one. In Table II, the C-loop runtime represents the additional time needed for computing the C-loop pruning array (Algorithm 4) in FastHED. The DP search time corresponds to the time for searching the near-optimal swapping structures in NOPTS-DP and FastHED with given $\mathbf{K}$. The total runtime of FastHED is the sum of the C-loop runtime and the DP search time. The total runtimes of NOPTS-DP are the same as its corresponding DP search times as it does not need to compute $\mathbf{C}$. As shown in Table II, the total runtime of FastHED is lower than that of other algorithms. This validates our analysis that our FastHED algorithm has a much lower computational complexity in Sec. A. The optimal ratios of both NOPTS-DP and FastHED both increase as the search intervals extend from $\mathbf{K}_1$ to $\mathbf{K}_4$. The reason is that extending the search intervals could increase the probability of finding the optimal repeater within the extended interval and therefore result in larger EDRs.

C. EDR Under Best-first Framework

We use the same parameter setup as in Sec. VI-A and run different algorithms in our best-first framework on a $1,024$-hop repeater chain with timespan $T = 5.5$ seconds. In the timespan T, we use the four $\mathbf{K}$s described in Sec. VI-B over four runs sequentially to compute the swapping structures, namely, $\mathbf{K}_1$ for the 1st run, $\mathbf{K}_2$ for the 2nd run, $\mathbf{K}_3$ for the 3rd run, and $\mathbf{K}_4$ for the 4th run.

As shown in Fig. 11, the better swapping structure is always selected first to be conducted by each algorithm and FastHED converges to the optimal EDR faster than NOPTS-DP and Pure-DP. Thus, the total average EDR of FastHED is higher than that of NOPTS-DP. After the 4th run, the EDRs of FastHED and NOPTS-DP closely approach the optimal EDR using the extended search interval specified in $\mathbf{K}_4$. Thus, the average EDR of FastHED is much higher than Pure-DP since Pure-DP requires a much longer time to compute the optimal swapping structure.

D. EDR on a Chain With Time Varying Parameters

To simulate the time varying nature of a chain, we assume success probabilities fluctuate over time following a specific period. We conduct experiments on a 1024-hop repeater chain and randomly take the entanglement generation success probability from $[0.2, 0.5]$ and swap success probability from $[0.7, 0.9]$. We leverage heuristics ($\mathbf{K}$s) identical to those in Sec. VI-C, and the timespan is set to 10 s. The typical value of the decoherence time is about 1.46 s [52], [53]. According to this, we set the varying period of success probabilities to range from 0.1 s to 10 s (i.e., the varying frequency from 10 Hz to 0.1 Hz). The comparison of the average EDRs of the three algorithms is illustrated in Fig. 12(a). Specifically, we examine how the EDR changes over time for different varying periods of 0.5, 1, and 5 s in Figs. 12(b), 12(c) and 12(d).

As can be observed in Fig. 12(a), the average EDR decreases with the increasing varying frequency for each algorithm. Our FastHED performs better than the other two algorithms when the varying frequency is below 10 Hz. Due to the high computational complexity, NOPTS-DP and Pure-DP degenerate to perform similarly as BDCZ when the varying frequency exceeds 0.4 and 2 Hz, respectively. As shown in Figs. 12(b), 12(c) and 12(d), when an algorithm's runtime exceeds the varying period, it always selects BDCZ structure for entanglement distribution, as the output swapping structure becomes outdated. Therefore, it is crucial to design a fast algorithm to quickly respond to the time varying parameters. Our FastHED outperforms both NOPTS-DP and Pure-DP

since it exhibits the lowest computational complexity among the three algorithms.

VII. Conclusion and Future Work

In this paper, we propose a fast heuristic entanglement distribution algorithm (FastHED) that finds the near-optimal swapping structure for a repeater chain with a low computational complexity. We design a novel heuristic pruning technique in FastHED and significantly reduce the redundant computations in existing algorithms. Meanwhile, we establish an upper bound on the complexity of FastHED, $O(n(\log n)^3)$ (more precisely, $O(n(\log n)^2 \log\log n)$ when $n \leq 2^{29}$), representing a significant improvement over the state-of-the-art algorithm with $O(n^2 \log n)$. In addition, we develop a best-first entanglement distribution framework to select the best swapping structure for distributing entanglement, with the objective of increasing average EDR. We conduct experiments to show that our algorithm FastHED has a lower computational complexity and higher average EDR than existing algorithms on chains with both static and time varying parameters.

Last but not least, we hope that our results may inspire some future work. For example, it would be interesting to extend our results to accommodate multiple entanglement demands in a repeater chain. It may also be worthwhile to devise a fast algorithm for an arbitrary network topology.

Appendix A
Proofs of Theorem 1

In this appendix, we analyze the computational complexity of FastHED.

We define the sub-chain with the minimum left node index and the minimum right node index within the children set $\mathcal{L}_k^{(s)}$ at level s as the *leftmost sub-chain*, and the sub-chain with the maximum left node index and the maximum right node index within the children set $\mathcal{L}_k^{(s)}$ at level s as the *rightmost sub-chain*. Fig. 13 gives an illustration of both the leftmost and the rightmost sub-chains.

We define $\alpha_k^{(s)}$ as the index of the left node of the leftmost sub-chain in $\mathcal{L}_{k+1}^{(s)}, \forall k \in [1, 2^s - 1]$. Similarly, we define $\beta_k^{(s)}$ as the index of the right node of the rightmost sub-chain in $\mathcal{L}_k^{(s)}, \forall k \in [1, 2^s - 1]$. The subscript k, ranging from 1 to $2^s - 1$, denotes the index of $\alpha_k^{(s)}$ ($\beta_k^{(s)}$) at level s as we have $2^s - 1$ child sets. Let $\alpha_0^{(s)} = \beta_0^{(s)} = 0$ and $\beta_{2^s}^{(s)} = \alpha_{2^s}^{(s)} = n$. The $\alpha_k^{(s)}$ and $\beta_k^{(s)}$ are defined using different child sets, $\mathcal{L}_{k+1}^{(s)}$ and $\mathcal{L}_k^{(s)}$, respectively, to offer a simplified definition for the segments, as detailed below.

Definition 2 (Segment): A segment, denoted as $[\alpha_k^{(s)}; \beta_k^{(s)}]$, represents a group of adjacent nodes within an n-hop chain. Here, $\alpha_k^{(s)}$ is the index of the left node of the leftmost sub-chain in $\mathcal{L}_{k+1}^{(s)}$ and $\beta_k^{(s)}$ is the index of the right node of the rightmost sub-chain in $\mathcal{L}_k^{(s)}$.

We can use the segments to identify the child sets at each level. Each pair of two adjacent segments ($[\alpha_k^{(s)}, \beta_k^{(s)}]$ and $[\alpha_{k+1}^{(s)}, \beta_{k+1}^{(s)}]$) at level s determines a sub-chain $L(\alpha_k^{(s)}, \beta_{k+1}^{(s)}), k \in [0, 2^s - 1]$. For $k \in [0, 2^s - 1]$, $\mathcal{L}_{k+1}^{(s)}$ is a subset of the set of all sub-chains that belong to $L(\alpha_k^{(s)}, \beta_{k+1}^{(s)})$). For instance, in Fig 13, the sub-chains (represented by the green double arrowed lines) in $\mathcal{L}_3^{(2)}$ all lie in $L(\alpha_2^{(2)}, \beta_3^{(2)})$.

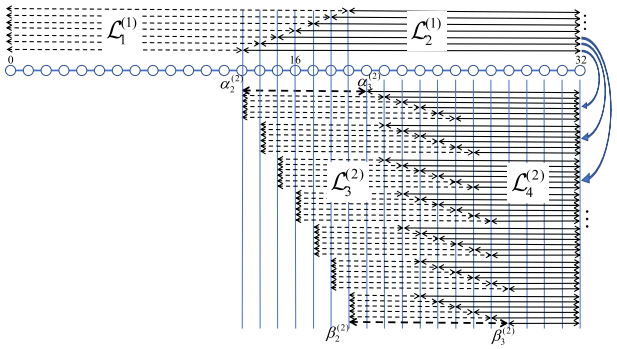


Fig. 13. An illustration of children sets and segments for a 32-hop chain. The collection of the dashed arrowed lines under the chain denotes $\mathcal{L}_3^{(2)}$, while the collection of the solid arrowed lines represents $\mathcal{L}_4^{(2)}$. The two thick arrowed lines represent the leftmost and the rightmost sub-chains in $\mathcal{L}_3^{(2)}$.

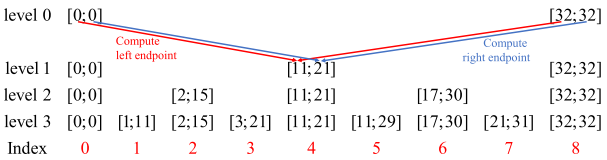


Fig. 14. Example of segment construction of 32-hop chain. The colored arrows mean the computation of a new segment.

For example, at level 0, there are two segments $[\alpha_0^{(0)}; \beta_0^{(0)}] = [0; 0]$ and $[\alpha_1^{(0)}; \beta_1^{(0)}] = [n; n]$. At level 1, a new segment $[\alpha_1^{(1)}; \beta_1^{(1)}] = [n/2 - \log n; n/2 + \log n]$ can be generated using the division technique. This segment represents the group of the intermediate repeaters for a n-hop chain, fulfilling the condition specified in (5). As depicted in Fig. 14, the red numbers under level 3 indicate the indices of the segments in level 3, ranging from 0 to 2^3.

The segments at level s can be derived from those at the upper level $s-1$. Using the division technique, the left node of the leftmost sub-chain in $\mathcal{L}_{2k-1}^{(s)}$ is just the left node of the leftmost sub-chain in $\mathcal{L}_k^{(s-1)}$. Similarly, the right node of the rightmost sub-chain in $\mathcal{L}_{2k}^{(s)}$ is the right node of the rightmost sub-chain in $\mathcal{L}_k^{(s-1)}$. Therefore, a segment at level s with an even index $2k, \forall k \in [0, 2^{s-1}]$ can be written as:

$$[\alpha_{2k}^{(s)}; \beta_{2k}^{(s)}] = [\alpha_k^{(s-1)}; \beta_k^{(s-1)}]. \tag{9}$$

For a segment at level s with an odd index $2k+1, \forall k \in [0, 2^{s-1} - 1]$, as defined earlier, $\alpha_k^{(s)}$ represents the left node of the leftmost sub-chain in a children set. The leftmost sub-chain in $\mathcal{L}_{k+1}^{(s-1)}$ is $L(\alpha_k^{(s-1)}, \alpha_{k+1}^{(s-1)})$. This leftmost sub-chain determines the leftmost sub-chain in $\mathcal{L}_{2k+2}^{(s)}$, and its left node index can be written as:

$$\alpha_{2k+1}^{(s)} = (\alpha_{k+1}^{(s-1)} + \alpha_k^{(s-1)})/2 - \log(\alpha_{k+1}^{(s-1)} - \alpha_k^{(s-1)}), \tag{10}$$

Similarly, the index of the right node of the rightmost sub-chain in $\mathcal{L}_{2k+1}^{(s)}$ can be written as:

$$\beta_{2k+1}^{(s)} = (\beta_{k+1}^{(s-1)} + \beta_k^{(s-1)})/2 + \log(\beta_{k+1}^{(s-1)} - \beta_k^{(s-1)}), \tag{11}$$

For example, Fig. 14 shows the computation process for generating a new segment level by level, where (10) and (11) are shown as the red and blue arrows, respectively.

Using (9) to (11), we can prove the following lemma that reveals a consistent pattern of the left endpoint and the right endpoint in all the segments.

Lemma 1: The k-th segment $[\alpha_k^{(s)}; \beta_k^{(s)}]$ at level s contains a specific node, referred to as the regular node, positioned at index $\lceil kn/2^s \rceil$ within an n-hop chain. In other words, $\alpha_k^{(s)} \leq \lceil kn/2^s \rceil \leq \beta_k^{(s)}$.

Using Lemma 1, we can establish an upper bound for the length of the sub-chain $L(\alpha_k^{(s)}, \beta_{k+1}^{(s)})$ as follows.

Lemma 2: Denote the length of the sub-chain $L(\alpha_k^{(s)}, \beta_{k+1}^{(s)})$ as $l_{k+1}^{(s)} = \beta_{k+1}^{(s)} - \alpha_k^{(s)}$, and $l_{k+1}^{(s)}$ is upper bounded by $l^{(s)} = (n/2^s) + 2s\log n - s(s-1)$ for $k \in [0, 2^s - 1]$.

Due to page limits, we refer the reader to our supplementary materials for the proofs of Lemma 1 and Lemma 2.

Remark 1: Lemma 2 implies that the length $\gamma^{(s)} = n/2^s + 2s\log n - s(s-1)$ of segments at any level s is upper bounded by $\max_{0 \leq s \leq \log n} \{\gamma^{(s)}\} = O((\log n)^2)$. Now we analyze the computational complexity of FastHED.

To proceed, we first devise an algorithm modified from FastHED, which, due to page limits, is detailed in our supplementary materials and yields a higher computational complexity than FastHED. The modified algorithm includes two steps. First, at level s, we employ NOPTS-DP to compute the swapping structures and EDRs for sub-chains $L(\alpha_k^{(s)}, \beta_{k+1}^{(s)}), k \in [0, 2^s - 1]$, as determined by (9)-(11). Note that all the sub-problems of $L(\alpha_k^{(s)}, \beta_{k+1}^{(s)})$ have been solved using NOPTS-DP, thereby covering $\mathcal{L}_{k+1}^{(s)}$. Second, leveraging these results, we can compute the swapping structures and EDRs for sub-chains at level $s-1$ (e.g., $\mathcal{L}^{(s-1)}$). Thus, we can compute the swapping structures and EDRs the n-hop chain. This computation proceeds in a bottom-up manner, similar to Algorithm 5, and the search for sub-chains that have already been computed by NOPTS-DP can be omitted. The primary difference between the modified algorithm and Algorithm 5 is the utilization of NOPTS-DP. Due to $|\mathcal{L}_{k+1}^{(s)}| \leq |L(\alpha_k^{(s)}, \beta_{k+1}^{(s)})|^2/2$, the computational complexity of the modified algorithm upper bounds Algorithm 5 in the first step. The remaining details of the modified algorithm are identical to Algorithm 5.

Thus, the modified algorithm has a higher complexity than FastHED. For the modified algorithm, its computational complexity $\phi(n, s)$ can be expressed as:

$$\phi(n, s) \leq 2^s \psi(l^{(s)}) + \sum_{h=0}^{s-1} c_h \log(l^{(h)}), \tag{12}$$

where the right side of (12) can be split into two parts: $\phi_A(n, s) = 2^s \psi(l^{(s)})$ represents the computational complexity of running NOPTS-DP for the 2^s sub-chains at level s, and $\phi_B(n, s) = \sum_{h=0}^{s-1} c_h \log(l^{(h)})$ represents the remaining computational complexity of Algorithm 5 based on the results of NOPTS-DP. At each level h, there are $c_h = |\mathcal{L}^{(h)}|$ sub-chains and each sub-chain requires at most $\log(l^{(h)})$ time in the K-loop as its length is upper bounded by $l^{(h)}$ by Lemma 2.

Using Lemma 1 to Lemma 2 and (12), we present the proof for Theorem 1 as follows.

Proof: Initially, we analyze the computational complexity $\phi_A(n, s)$. Note that $\phi_A(n, s)$ is a convex function with respect to s. We can utilize the minimum of $\phi_A(n, s)$ (w.r.t. s) to analyze the complexity upper bound of FastHED. It is challenging to derive a closed-form expression to find the minimizer s^* of $\phi_A(n, s)$ as s is discrete. Thus, we introduce a continuous variable $\tilde{s}$ and consider $\phi_A(n, \tilde{s})$ to determine $\tilde{s}^*$. Then we can estimate s^* by discretizing $\tilde{s}^*$. Take the first derivative of $\phi_A(n, \tilde{s})$ with respect to $\tilde{s}$:

$$\frac{d\phi_A(n, \tilde{s})}{d\tilde{s}} = 2^{\tilde{s}}[(2\log(l^{(\tilde{s})}) + 1/\ln 2) \cdot l^{(\tilde{s})} \cdot \frac{dl^{(\tilde{s})}}{d\tilde{s}} + \ln 2 \cdot \psi(l^{(\tilde{s})})], \tag{13}$$

where $\frac{dl^{(\tilde{s})}}{d\tilde{s}} = -\ln 2 \cdot n/2^{\tilde{s}} + 2\log n - 2\tilde{s} + 1$. We choose specific values of $\tilde{s}$ to identify the zero point of the derivative. Let $\tilde{s} = \log n - k\log(\log n)$ for $k \in \mathbb{Z}$. That is, $2^{\tilde{s}} = n/(\log n)^k$. We can find the optimal k that minimizes $\phi_A(n, \tilde{s})$.

It can be verified that $\frac{d\phi_A(n,\tilde{s})}{d\tilde{s}}$ is negative when $k = 3$ and positive when $k = 1$. The zero point occurs between $k = 3$ and $k = 1$, i.e., around $s = \log n - 2\log\log n$.

Next, we analyze the computational complexity of $\phi_B(n, s)$. We have:

$$\begin{aligned}\phi_B(n, s) &= \sum_{h=0}^{s-1} c_h \log(l^{(h)}) \\ &\leq \sum_{h=0}^{s-1} 2^h (\gamma^{(h)})^2 \log(l^{(h)}) \\ &\leq 2^s (\max_{0 \leq h \leq s-1} \{(\gamma^{(h)})^2 \log(l^{(h)})\}).\end{aligned} \tag{14}$$

The first inequality holds since the number of the child sets at level h is 2^h and the number of sub-chains in each child set is upper bound by $(\gamma^{(h)})^2$. This is because the number of the left (or right) nodes of sub-chains in a children set at level h is less than $\gamma^{(h)}$. The second inequality holds since $(\gamma^{(h)})^2 \log(l^{(h)}) \leq \max_{0 \leq h \leq s-1} \{(\gamma^{(h)})^2 \log(l^{(h)})\}$ and $\sum_{h=0}^{s-1} 2^h \leq 2^s$. We have:

$$\begin{aligned}&(\gamma^{(h)})^2 \log(l^{(h)}) \\ &\leq 4h^2 (\log n)^2 (\log(n/2^h) + \log(2h\log n)) \\ &= O((\log n)^2 h^2 (\log(n/2^h)) + O((\log n)^4 \log\log n),\end{aligned} \tag{15}$$

where, the inequality holds when $h \leq \log n$. For ease of description, we denote $\theta(h) = h^2(\log(n/2^h)$. From (14) and (15), we can derive the upper bound of $\phi_B(n, s)$ as follows,

$$\begin{aligned}\phi_B(n, s) &= O(2^s) \cdot O(\max_{0 \leq h \leq s-1} \{(\gamma^{(h)})^2 \log(l^{(h)})\}) \\ &= O(2^s) \cdot O((\log n)^2 \max_{0 \leq h \leq s-1} \theta(h)) \\ &\quad + O(2^s) \cdot O((\log n)^4 \log\log n).\end{aligned} \tag{16}$$

It can be verified that $h^* = (2\log n)/3$ is the maximum point for $\theta(h)$, i.e., $O(\max_{0 \leq h \leq \log n} \theta(h)) = O((\log n)^3)$.

For $\log n - 2\log\log n \leq (2\log n)/3$, we take $s = \log n - 2\log\log n$. We have $\phi_A(n, s) = O(n(\log n)^2 \log\log n)$. As $s \leq h^*$, we have $O(\max_{0 \leq h \leq s-1} \theta(h)) = O((\log n)^2 \log\log n)$, resulting in $\phi_B(n, s) = O(n(\log n)^2 \log\log n)$. Thus,

$O(n(\log n)^2 \log\log n)$ is an upper bound of FastHED. The condition of $\log n - 2\log\log n \leq (2\log n)/3$ also can be rewritten as $n \leq 2^{29}$.

For $\log n - 2\log\log n > (2\log n)/3$, we use the maximum of $\theta(h)$ to obtain $\phi_B(n,s) = O(n(\log n)^3)$. We take $s = \log n - 2\log\log n$ for $\phi_A(n,s) = O(n(\log n)^2 \log\log n)$. Hence, $O(n(\log n)^3)$ is an upper bound of FastHED.

This completes the proof of Theorem 1. ■

REFERENCES

[1] A. Einstein, B. Podolsky, and N. Rosen, "Can quantum-mechanical description of physical reality be considered complete?" *Phys. Rev.*, vol. 47, pp. 777–780, May 1935.

[2] R. Horodecki, P. Horodecki, M. Horodecki, and K. Horodecki, "Quantum entanglement," *Rev. Mod. Phys.*, vol. 81, no. 2, pp. 865–942, Jun. 2009.

[3] K. Chen and H.-K. Lo, "Conference key agreement and quantum sharing of classical secrets with noisy GHZ states," in *Proc. Int. Symp. Inf. Theory*, 2005, pp. 1607–1611.

[4] J. Ribeiro, G. Murta, and S. Wehner, "Fully device-independent conference key agreement," *Phys. Rev. A, Gen. Phys.*, vol. 97, no. 2, Feb. 2018, Art. no. 022307.

[5] R. Jozsa, D. S. Abrams, J. P. Dowling, and C. P. Williams, "Quantum clock synchronization based on shared prior entanglement," *Phys. Rev. Lett.*, vol. 85, no. 9, pp. 2010–2013, Aug. 2000.

[6] R. Ben-Av and I. Exman, "Optimized multiparty quantum clock synchronization," *Phys. Rev. A, Gen. Phys.*, vol. 84, no. 1, Jul. 2011, Art. no. 014301.

[7] C. Crépeau, D. Gottesman, and A. Smith, "Secure multi-party quantum computation," in *Proc. 34th Annu. ACM Symp. Theory Comput.* New York, NY, USA: Association for Computing Machinery, May 2002, pp. 643–652.

[8] B. He, Y. Ren, and J. A. Bergou, "Creation of high-quality long-distance entanglement with flexible resources," *Phys. Rev. A, Gen. Phys.*, vol. 79, no. 5, May 2009, Art. no. 052323.

[9] S. Pirandola, R. Laurenza, C. Ottaviani, and L. Banchi, "Fundamental limits of repeaterless quantum communications," *Nature Commun.*, vol. 8, no. 1, pp. 1–15, Apr. 2017.

[10] S. Pirandola, "End-to-end capacities of a quantum communication network," *Commun. Phys.*, vol. 2, no. 1, p. 51, May 2019.

[11] 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.

[12] 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.*, 2020, pp. 62–75.

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

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

[15] S. Zhang, S. Shi, C. Qian, and K. L. Yeung, "Fragmentation-aware entanglement routing for quantum networks," *J. Lightw. Technol.*, vol. 39, no. 14, pp. 4584–4591, Jul. 15, 2021.

[16] L. Jiang, J. M. Taylor, N. Khaneja, and M. D. Lukin, "Optimal approach to quantum communication using dynamic programming," *Proc. Nat. Acad. Sci. USA*, vol. 104, no. 44, pp. 17291–17296, Oct. 2007.

[17] K. Goodenough, D. Elkouss, and S. Wehner, "Optimizing repeater schemes for the quantum internet," *Phys. Rev. A, Gen. Phys.*, vol. 103, no. 3, Mar. 2021, Art. no. 032610.

[18] K. Azuma et al., "Quantum repeaters: From quantum networks to the quantum internet," *Rev. Modern Phys.*, vol. 95, no. 4, Dec. 2023, Art. no. 045006.

[19] E. Shchukin, F. Schmidt, and P. van Loock, "Waiting time in quantum repeaters with probabilistic entanglement swapping," *Phys. Rev. A, Gen. Phys.*, vol. 100, no. 3, Sep. 2019, Art. no. 032322.

[20] 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.

[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. L. Braunstein and A. Mann, "Measurement of the bell operator and quantum teleportation," *Phys. Rev. A, Gen. Phys.*, vol. 51, no. 3, pp. 1727–1730, Mar. 1995.

[23] Y.-S. Kim et al., "Informationally symmetrical bell state preparation and measurement," *Opt. Exp.*, vol. 26, no. 22, pp. 29539–29549, 2018.

[24] Á. G. Iñesta, G. Vardoyan, L. Scavuzzo, and S. Wehner, "Optimal entanglement distribution policies in homogeneous repeater chains with cutoffs," *Npj Quantum Inf.*, vol. 9, no. 1, p. 46, May 2023.

[25] S. Haldar, P. J. Barge, S. Khatri, and H. Lee, "Fast and reliable entanglement distribution with quantum repeaters: Principles for improving protocols using reinforcement learning," *Phys. Rev. Appl.*, vol. 21, no. 2, Feb. 2024, Art. no. 024041.

[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] C. Li, T. Li, Y.-X. Liu, and P. Cappellaro, "Effective routing design for remote entanglement generation on quantum networks," *Npj Quant. Inf.*, vol. 7, no. 1, p. 10, Dec. 2021.

[28] K. Chakraborty, F. Rozpedek, A. Dahlberg, and S. Wehner, "Distributed routing in a quantum internet," 2019, *arXiv:1907.11630*.

[29] S. Das, S. Khatri, and J. P. Dowling, "Robust quantum network architectures and topologies for entanglement distribution," *Phys. Rev. A, Gen. Phys.*, vol. 97, no. 1, Jan. 2018, Art. no. 012335.

[30] 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.

[31] T. N. Nguyen, D. H. P. Nguyen, D. H. Pham, B.-H. Liu, and H. N. Nguyen, "LP relaxation-based approximation algorithms for maximizing entangled quantum routing rate," in *Proc. IEEE Int. Conf. Commun.*, May 2022, pp. 3269–3274.

[32] Y. Zeng, J. Zhang, J. Liu, Z. Liu, and Y. Yang, "Entanglement routing design over quantum networks," *IEEE/ACM Trans. Netw.*, vol. 32, no. 1, pp. 352–367, Feb. 2024.

[33] A. Chang and G. Xue, "Order matters: On the impact of swapping order on an entanglement path in a quantum network," in *Proc. IEEE INFOCOM Conf. Comput. Commun. Workshops (INFOCOM WKSHPS)*, May 2022, pp. 1–6.

[34] E. Shchukin and P. van Loock, "Optimal entanglement swapping in quantum repeaters," *Phys. Rev. Lett.*, vol. 128, no. 15, Apr. 2022, Art. no. 150502.

[35] R. Bellman, "Dynamic programming," *Science*, vol. 153, pp. 34–37, Jul. 1966.

[36] W. Dür and H. J. Briegel, "Entanglement purification and quantum error correction," *Rep. Prog. Phys.*, vol. 70, no. 8, pp. 1381–1424, Aug. 2007.

[37] W. Dür, H.-J. Briegel, J. I. Cirac, and P. Zoller, "Quantum repeaters based on entanglement purification," *Phys. Rev. A, Gen. Phys.*, vol. 59, no. 1, pp. 169–181, 1999.

[38] L. Childress, J. M. Taylor, A. S. Sørensen, and M. D. Lukin, "Fault-tolerant quantum repeaters with minimal physical resources and implementations based on single-photon emitters," *Phys. Rev. A, Gen. Phys.*, vol. 72, no. 5, Nov. 2005, Art. no. 052330.

[39] L. Childress, J. M. Taylor, A. S. Sørensen, and M. D. Lukin, "Fault-tolerant quantum communication based on solid-state photon emitters," *Phys. Rev. Lett.*, vol. 96, no. 7, Feb. 2006, Art. no. 070504.

[40] 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.

[41] T. Coopmans, S. Brand, and D. Elkouss, "Improved analytical bounds on delivery times of long-distance entanglement," *Phys. Rev. A, Gen. Phys.*, vol. 105, no. 1, Jan. 2022, Art. no. 012608.

[42] S. E. Vinay and P. Kok, "Statistical analysis of quantum-entangled-network generation," *Phys. Rev. A, Gen. Phys.*, vol. 99, no. 4, Apr. 2019, Art. no. 042313.

[43] A. Patil, M. Pant, D. Englund, D. Towsley, and S. Guha, "Entanglement generation in a quantum network at distance-independent rate," *Npj Quant. Inf.*, vol. 8, no. 1, p. 51, Dec. 2022.

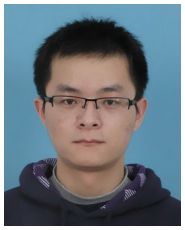
[44] A. Patil, J. I. Jacobson, E. Van Milligen, D. Towsley, and S. Guha, "Distance-independent entanglement generation in a quantum network using space-time multiplexed Greenberger–Horne–Zeilinger (GHZ) measurements," in *Proc. IEEE Int. Conf. Quantum Comput. Eng. (QCE)*, Oct. 2021, pp. 334–345.

[45] Y. Zeng, J. Zhang, J. Liu, Z. Liu, and Y. Yang, "Entanglement routing over quantum networks using Greenberger–Horne–Zeilinger measurements," in *Proc. IEEE 43rd Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Jul. 2023, pp. 350–360.

[46] T. Bacinoglu, B. Gulbahar, and O. B. Akan, "Constant fidelity entanglement flow in quantum communication networks," in *Proc. IEEE Global Telecommun. Conf.*, Dec. 2010, pp. 1–5.
[47] C. Langer et al., "Long-lived qubit memory using atomic ions," *Phys. Rev. Lett.*, vol. 95, no. 6, Aug. 2005, Art. no. 060502.
[48] H. Häffner et al., "Robust entanglement," *Appl. Phys. B, Lasers Opt.*, vol. 81, pp. 151–153, Mar. 2005.
[49] T. P. Harty et al., "High-fidelity preparation, gates, memory, and readout of a trapped-ion quantum bit," *Phys. Rev. Lett.*, vol. 113, no. 22, Nov. 2014, Art. no. 220501.
[50] P. Wang et al., "Single ion qubit with estimated coherence time exceeding one hour," *Nature Commun.*, vol. 12, no. 1, p. 233, Jan. 2021.
[51] L. Ruan, W. Dai, and M. Z. Win, "Adaptive recurrence quantum entanglement distillation for two-Kraus-operator channels," *Phys. Rev. A, Gen. Phys.*, vol. 97, no. 5, May 2018, Art. no. 052332.
[52] 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)*. New York, NY, USA: Association for Computing Machinery, 2019, pp. 159–173.
[53] J. Li et al., "Fidelity-guaranteed entanglement routing in quantum networks," *IEEE Trans. Commun.*, vol. 70, no. 10, pp. 6748–6763, Oct. 2022.
[54] F. Ewert and P. van Loock, "3/4-efficient bell measurement with passive linear optics and unentangled ancillae," *Phys. Rev. Lett.*, vol. 113, no. 14, Sep. 2014, Art. no. 140403.
[55] H. A. Zaidi and P. van Loock, "Beating the one-half limit of ancilla-free linear optics bell measurements," *Phys. Rev. Lett.*, vol. 110, no. 26, Jun. 2013, Art. no. 260501.

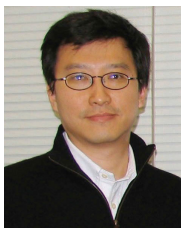
Wenkang Cen received the B.E. degree in communication engineering from Sun Yat-sen University, Shenzhen, China, in 2023, where he is currently pursuing the Ph.D. degree with the School of Electronics and Communication Engineering. His current research interests include quantum information and networking.

Jinbei Zhang (Member, IEEE) received the B.S. degree in electronic engineering from Xidian University, Xi'an, China, in 2010, and the Ph.D. degree in electronic engineering from Shanghai Jiao Tong University, Shanghai, China, in 2016. From 2016 to 2018, he was a Post-Doctoral Researcher with The Chinese University of Hong Kong. Since 2018, he has been an Associate Professor with Sun Yat-sen University, Shenzhen, China. His current research interests include network information theory and quantum networks.



Kechao Cai (Member, IEEE) received the Ph.D. degree from The Chinese University of Hong Kong in 2019. He is currently an Assistant Professor with the School of Electronics and Communication Engineering, Sun Yat-sen University, China. His current research area includes distributed protocol design, online learning algorithms, and quantum internet.

Shihai Sun received the Ph.D. degree in physics from the National University of Defense Technology, Changsha, China, in 2012. Since 2019, he has been a Professor with the School of Electronics and Communication Engineering, Sun Yet-sen University, China. He is a member of China Institute of Communications, and Chinese Association for Cryptologic Research. He has published about 60 papers about quantum communication and quantum information processing in *Optica* and PRL. His current research interests include quantum communication, quantum networks, and quantum cryptography.



John C. S. Lui (Fellow, IEEE) received the Ph.D. degree in computer science from UCLA. He is currently the Choh-Ming Li Chair Professor with the Department of Computer Science and Engineering, The Chinese University of Hong Kong. His current research interests include communication networks, system security (e.g., cloud security and mobile security), network economics, network sciences, large-scale distributed systems, and performance evaluation theory. His personal interests include films and general reading. He is an Elected Member of the IFIP WG 7.3, a Fellow of ACM, and a Croucher Senior Research Fellow. He received various departmental teaching awards and the CUHK Vice-Chancellors Exemplary Teaching Award. He was a co-recipient of the IFIP WG 7.3 Performance 2005 and IEEEIFIP NOMS 2006 Best Student Paper Awards. He was the Chairman of the CSE Department, from 2005 to 2011. He serves in the editorial board for IEEE/ACM TRANSACTIONS ON NETWORKING, IEEE TRANSACTIONS ON COMPUTERS, IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, *Journal of Performance Evaluation*, and *International Journal of Network Security*.