Decoherence-Aware Entangling and Swapping Strategy Optimization for Entanglement Routing in Quantum Networks

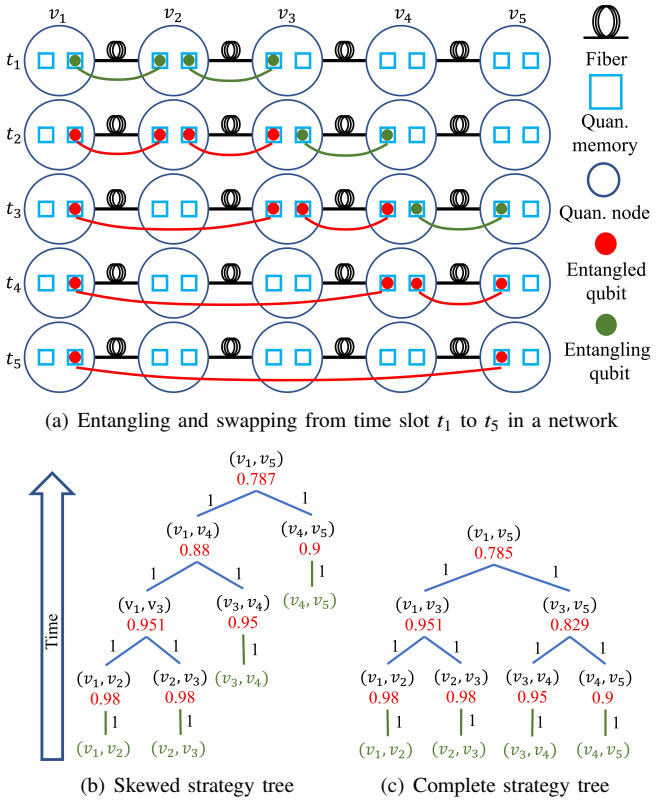
Shao-Min Huang‖, Cheng-Yang Cheng‖, Ming-Huang Chien‖, Jian-Jhih Kuo, and Chih-Yu Wang

***Abstract*—Quantum teleportation enables high-security communications through end-to-end quantum entangled pairs. End-to-end entangled pairs are created by using swapping processes to consume short entangled pairs and generate long pairs. However, due to environmental interference, entangled pairs decohere over time, resulting in low fidelity. Thus, generating entangled pairs at the right time is crucial. Moreover, the swapping process also causes additional fidelity loss. To this end, this paper presents a short time slot protocol, where a time slot can only accommodate a process. It has a more flexible arrangement of entangling and swapping processes than the traditional long time slot protocol. It raises a new optimization problem TETRIS for finding strategies of entangling and swapping for each request to maximize the fidelity sum of all accepted requests. To solve the TETRIS, we design two novel algorithms with different optimization techniques. Finally, the simulation results manifest that our algorithms can outperform the existing methods by up to 60 ~ 78% in general, and by 20 ~ 75% even under low entangling probabilities.**

***Index Terms*—Quantum networks, fidelity, decoherence, entanglement routing, scheduling, resource management, optimization problem, NP-hardness, inapproximability, bi-criteria approximation algorithm**

I. Introduction

As the frontiers of modern technology, quantum networks (QNs) have been developed and implemented to evaluate their practicability for information transmission [1]–[5]. QNs connect quantum nodes to transmit quantum bits (qubits) using end-to-end entangled pairs [5] (i.e., quantum teleportation). These networks serve as the foundation of quantum services such as quantum key distribution (QKD) [2], distributed quantum computing [3], and clock synchronization [4]. As illustrated in Fig. 1(a), each quantum node in the QN has a specific amount of quantum memory (blue squares) to store entangled qubits, mitigating rapid decoherence [5]–[8]. Adjacent nodes are interconnected by optical fibers (black lines) that facilitate the entanglement process. However, an end-to-end entangled pair is not directly available between two non-adjacent nodes. In such a case, their end-to-end entangled pair can be achieved via one or more entanglement swapping processes, each of which consumes two short entangled pairs to create a long one. For example, at time slot t_2 in Fig. 1(a), there are two entangled pairs (v_1, v_2) and (v_2, v_3). After performing a swapping at the end of t_2, we obtain a long entangled pair (v_1, v_3) at time slot t_3 and free two quantum memory units at v_2 [8].



(a) Entangling and swapping from time slot t_1 to t_5 in a network

(b) Skewed strategy tree (c) Complete strategy tree

Fig. 1. Scheduling of entangling and swapping in QNs.

However, entangled pairs stored in the quantum memory decohere over time, resulting in a loss of quality, i.e., fidelity loss. Thus, finding the perfect timing for entangling is crucial; an improper strategy may cause entangled pairs to idle and

S.-M. Huang is with the Research Center for Information Technology Innovation, Academia Sinica, Taiwan, and also with the Department of Computer Science and Information Engineering, National Chung Cheng University, Taiwan (e-mail: shaominhuang2019@alum.ccu.edu.tw).

C.-Y. Cheng is with the Institute of Data Science and Engineering, National Yang Ming Chiao Tung University, Taiwan, and also with the Department of Computer Science and Information Engineering, National Chung Cheng University, Taiwan (e-mail: yang890912.cs12@nycu.edu.tw).

M.-H. Chien is with the Institute of Computer Science and Engineering, National Yang Ming Chiao Tung University, Taiwan, and also with the Department of Computer Science and Information Engineering, National Chung Cheng University, Taiwan (e-mail: qq11123334.cs12@nycu.edu.tw).

J.-J. Kuo is with the Department of Computer Science and Information Engineering, National Chung Cheng University, Taiwan, and also with the Advanced Institute of Manufacturing with High-tech Innovations, National Chung Cheng University, Taiwan (e-mail: lajacky@cs.ccu.edu.tw).

C.-Y. Wang is with the Research Center for Information Technology Innovation, Academia Sinica, Taiwan (e-mail: cywang@citi.sinica.edu.tw).

‖ denotes the equal contributions. Corresponding author: Jian-Jhih Kuo

unnecessary fidelity loss. Moreover, swapping also causes fidelity loss as the entangled link generated by swapping exhibits lower fidelity than the two consumed entangled links, and the loss is exacerbated when the fidelity difference between the two pairs is significant. Thus, the entangling and swapping strategy should be properly designed to improve fidelity.

Existing QN protocols, studied in the literature, often employ a standard long time slot protocol setting [9]–[22], which may result in unnecessary fidelity loss when handling multiple requests. Specifically, in traditional long time slot protocols, the entangling processes of each request (i.e., source-destination (SD) pair) take place during the entangling phase, while swapping processes occur during the swapping phase. As a result, shorter requests might have to wait for longer requests to complete because the latter involve more processes to perform. Further, traditional long time slot protocols bind resources for the entire time slot, even though a swapping process frees two quantum memory units, which originally could be used for other requests but remain blocked by the protocol.

In light of the shortcomings, we propose a short time slot protocol where either entangling or swapping can occur in any given time slot. Specifically, in our short time slot protocol, a node is free to perform one process within a single time slot, and the process could be entangling, swapping, teleporting, or idling. Thus, short requests no longer need to wait for other requests to establish their entangled pairs, as they can perform processes separately, while longer requests can utilize the resource released by the nodes performing swapping processes.

The adoption of a short time slot protocol enables more efficient entangling and swapping strategies, minimizing unnecessary fidelity loss by precisely timing entangled pair generation for subsequent swapping. Take Figs. 1(a) and 1(b) for example, we have a request from v_1 to v_5.[1] At time slot t_1, pairs (v_1, v_2) and (v_2, v_3) start entangling, taking one time slot. At time slot t_2, both the generated pairs (v_1, v_2) and (v_2, v_3) have the initial fidelity 0.98 and start decoherence while doing the swapping process. One may observe that we can choose to start the entangling of pair (v_3, v_4) at either t_1 or t_2. Clearly, entangling at t_2 is better than t_1 since they will have less wait time for subsequent swapping processes, leading to less decoherence and better fidelity. On the other hand, swapping processes cause extra fidelity loss, and the fidelity of (v_1, v_3) is 0.951, which is lower than the fidelity before swapping (i.e., the two links with fidelity 0.98 after decoherence for 1 time slot will have fidelity 0.975). At time slot t_3, the generated pair (v_1, v_3) with fidelity 0.951 and (v_3, v_4) with fidelity 0.95 start decoherence while swapping, leading to pair (v_1, v_4) with fidelity 0.88 after swapping. Meanwhile, pair (v_4, v_5) start entangling. Finally, (v_1, v_5) are generated at t_5 and has fidelity 0.787. Unlike the skewed strategy tree in Fig. 1(b), the strategy in Fig. 1(c) requires only four slots. Intuitively, it should have better fidelity because it suffers less decoherence time. However, the result is counter-intuitive; Fig. 1(b) has better fidelity than Fig. 1(c) because most of the swapping processes in Fig. 1(b) consume two entangled pairs with similar fidelity and thus suffer less fidelity loss due to swapping processes.[2] From the perspective above, we can guess that if all the links on the path have similar initial fidelity, the complete strategy tree will perform better than others. For example, if all the generated pairs (v_1, v_2), (v_2, v_3), (v_3, v_4), and (v_4, v_5) have initial fidelity 0.98, then the final fidelity of the strategy trees in Figs. 1(b) and 1(c) will be 0.889 and 0.891, respectively.

We have two observations from the above examples: 1) swapping two entangled pairs with similar fidelity performs better, and 2) the more skewed the strategy tree, the less memory is required within a time slot. Moreover, a different strategy leads to varying fidelity and distribution of resource consumption. In this paper, such a distribution is referred to as *numerology*, representing how the resources (i.e., quantum memory in this paper) are consumed for the request. The numerology provides a direct representation of the corresponding strategy to examine its feasibility. The quality of the strategy, which is the resulting fidelity of the request, can also be derived directly through the corresponding numerology. Thus, the entangling and swapping strategy formation can be transformed into the numerology selection problem.

In this paper, we aim to choose a numerology for each request to maximize the fidelity sum for all accepted requests while meeting the fidelity threshold within a batch of time slots, i.e., the Op**t**imized D**e**coherence-aware S**tr**ategy for Entangl**i**ng and **S**wapping Scheduling Problem (TETRIS). The TETRIS introduces four novel challenges: 1) Entangled pairs will decohere over time. How can we choose the right time to generate entangled pairs to reduce their wait time? 2) The swapping process may cause additional fidelity loss. How can we achieve an intelligent swapping strategy to minimize this loss? 3) Numerologies determine the resulting fidelity. How can we identify the feasible numerologies for requests to meet the fidelity threshold, guaranteeing the quality of teleportation? 4) Quantum memory is limited, and its availability will be affected by simultaneous requests. How can we find efficient numerologies to mitigate resource contention and satisfy more requests while ensuring a better fidelity sum?

It can be shown that the TETRIS with no fidelity threshold constraint (i.e., TETRIS-N) is already NP-hard and cannot be approximated within any factor of $|I|^{1-\epsilon}$, where $|I|$ denotes the number of SD pairs. To conquer the above challenges, we propose two novel algorithms. 1) The first one is **F**ractional **N**umerology **P**acking and **R**ounding Algorithm (FNPR). FNPR aims to maximize the number of accepted requests. With linear programming (LP) rounding, FNPR can achieve a bi-criteria approximation ratio for the TETRIS-N as the actual duration of a time slot and the number of time slots are sufficiently small. Then, we extend FNPR to solve the TETRIS. 2) The second one is **F**idelity-**L**oad **T**rade-**O**ff Algorithm (FLTO). FLTO utilizes two efficient dynamic programming (DP) algorithms to find two candidate numerologies for each request with a given

[1]In this example, all the parameters follow the default settings in Section VII. In addition, the fidelity after decoherence over time and swapping is calculated using Eqs. (1) and (2), respectively, in Section III.

[2]More simulations and clarifications of the reasons behind this counter-intuitive example are provided in Appendix A of the supplementary material.

path. One maximizes fidelity, and the other considers resource utilization. Then, FLTO iteratively invokes them to allocate the resource for each request based on a subtly-designed index called resource efficiency index (REI), inspired by [23].

In sum, the contributions of this paper are as follows: 1) To the best of our knowledge, this is the first attempt to consider the decoherence of entangled pairs over time for entangling and swapping scheduling while applying a short time slot protocol that offers better resource allocation. 2) We prove that the TETRIS is an NP-hard problem, and even its special case, the TETRIS-N, cannot be approximated within any factor of $|I|^{1-\epsilon}$. 3) We propose a combinatorial algorithm with a clever separation oracle to achieve a bi-criteria approximation. Meanwhile, we develop two DP algorithms to find candidate numerologies with different goals and design an index to choose numerologies adaptively.

II. Related Work

Before discussing the related works on optimization problems for entangling and swapping, we review previous works that examine feasibility and realization of QNs [1], [24]–[28]. Early foundational studies have laid the groundwork for secure communication protocols and robust architectures in quantum networks. Elliott *et al.* introduced QN to realize secure communications [1]. Meter *et al.* proposed a large QN architecture with layered recursive repeaters, where repeaters may not trust each other, and then designed new protocol layers to support quantum sessions to ensure robustness and interoperable communication [24]. Muralidharan *et al.* presented new quantum nodes that can execute the quantum error correction (QEC) processes and classified the theoretically feasible technologies of quantum nodes into three generations [25]. Pirandola *et al.* discussed the limits of repeater-less quantum communications and provided general benchmarks for repeaters [26]. Caleffi *et al.* designed a routing protocol to ensure a high end-to-end entanglement rate between any two nodes [27]. Zhao *et al.* proposed two transport layer protocols for quantum data networks that achieve high throughput and fairness [28].

Building on these foundational works, subsequent research has focused on path selection and entangling and swapping process optimization for long time slot system-based QNs [9]–[22]. Pant *et al.* presented a greedy algorithm to determine the paths for each request [9]. Shi *et al.* designed a routing method Q-CAST based on the Dijkstra algorithm to find primary paths and recovery paths to mitigate the effect of entanglement failures [10]. Zhao *et al.* presented an LP-based algorithm REPS to maximize throughput in SDN-based QNs [11]. Zhao *et al.* considered entangled links' fidelity and exploited quantum purification to enhance link fidelity [12]. Chen *et al.* proposed two heuristic algorithms for entangling and swapping phases separately [13]. Farahbakhsh *et al.* developed an add-up scheme to store and forward data qubits as much as possible [14]. Zeng *et al.* studied how to simultaneously maximize the number of quantum-user pairs and their expected throughput [15]. Zeng *et al.* utilized the properties of n-fusion to help increase the success probability of constructing a long entangled pair for quantum networks [16]. Li *et al.* exploited purification to meet the fidelity requirements for multiple SD pairs as many as possible [17]. Chakraborty *et al.* gave an LP formulation to compute the maximum total entanglement distribution rate [18]. Pouryousef *et al.* attempted to stockpile entangled pairs in advance when the traffic demand is low while using them otherwise [19]. Zhao *et al.* considered a room-size network scenario, where a longer entangled pair can be established by sending one of its photons via all-optical switching without swapping at intermediate nodes [20]. However, none of the above works considers the effect of decoherence on fidelity. To this end, Cicconetti *et al.* studied different policies of path selection and request scheduling and then measured the resulting link fidelity under the influence of dephasing and depolarizing noises (i.e., decoherence) [21]. Ghaderibaneh *et al.* further considered time decoherence when determining the swapping order for each SD pair to minimize entangled pair generation latency [22]. However, all of the above works use long time slot systems and conduct entangling and swapping sequentially, which may block short requests to wait for long requests, resulting in more memory idle time.

To address the limitations of long time slot systems, some studies [29], [30] have explored short time slot systems or even asynchronous protocols to improve efficiency. Huang *et al.* employed a short time slot protocol while opportunistically forwarding data qubits via social nodes to ensure security. However, they neglected decoherence over time, fidelity loss due to swapping processes, and freed qubits after swapping processes [29]. Yang *et al.* proposed an asynchronous model to freely generate entangled pairs or conduct swapping processes without time slot limitations. It helps establish end-to-end connections more quickly while utilizing more resources, as resources can be freed immediately [30]. However, it did not consider decoherence or fidelity loss. In contrast, our short time slot system considers more practical fidelity losses due to time decoherence and swapping processes. It further leverages diverse numerologies for requests to maximize the fidelity sum, achieving efficient entangling and swapping scheduling.

Although some short time slot approaches have been proposed to enhance efficiency, they still overlook fidelity degradation over time due to decoherence and swapping processes. To this end, the following works [31]–[34] consider decoherence when constructing remote entangled pairs along a specific path. Haldar *et al.* employed the Q-learning reinforcement-learning (RL) algorithm to discover policies that optimize both average waiting time and fidelity for a single request [31]. Iñesta *et al.* proposed a method based on Markov decision processes-based method with value and policy iteration to minimize the expected needed time to achieve end-to-end entanglement for a single request [32]. Haldar *et al.* proposed the quasi-local multiplexing policies, following the SWAP-ASAP approach and incorporating entanglement purification, to optimize both average waiting time and fidelity for a single request in linear chain quantum networks [33]. Goodenough *et al.* analyzed the value of fidelity up to 25 segments over time using a generating function. The approach can be applied to find cut-off policies in

TABLE I
COMPARISON OF RELATED WORKS

Literature	Time slot	Fidelity decohere over time	Multi-request scheduling	Path selection	Objective	Solution strategy
[9]	Long	No	No	Yes	Max throughput	Greedy
[10]	Long	No	No	Yes	Max throughput	Dijkstra-based
[11]	Long	No	No	Yes	Max throughput	LP rounding + Heuristic
[12]	Long	No	No	Yes	Max throughput	Dijkstra-based + LP rounding
[13]	Long	No	No	Yes	Max throughput	Heuristic + DP
[14]	Long	No	No	No	Min waiting time	Greedy
[15]	Long	No	No	Yes	Max user pairs & Max throughput	Dijkstra-based + LP rounding + Branch and bound algorithm
[16]	Long	No	No	Yes	Max throughput	Dijkstra-based
[17]	Long	No	No	Yes	Max throughput	Dijkstra-based + Greedy
[18]	Long	No	No	Yes	Max throughput	Multicommodity flow-based algorithm
[19]	Long	No	No	Yes	Max throughput & Min delay	LP
[20]	Long	No	No	Yes	Max throughput	LP rounding + Heuristic
[21]	Long	No	Yes	Yes	Max throughput	Heuristic + Dijkstra-based
[22]	Long	Yes	No	No	Min latency	DP
[29]	Short	No	No	Yes	Min waiting time	Greedy
[30]	Async	No	No	Yes	Max network efficiency	Dijkstra-based + Primal-Dual-based algorithm
[31]	Short	Yes	No	No	Min waiting time & Max fidelity	Q-learning reinforcement-learning
[32]	Short	Yes	No	No	Min delivery time	Markov decision process-based algorithm
[33]	Short	Yes	No	No	Min waiting time & Max fidelity	SWAP-ASAP-based algorithm
Ours	Short	Yes	Yes	Yes	Max fidelity & Max throughput	Combinatorial algorithm with DP-based separation oracle + LP rounding & Greedy + DP

QKD [34]. Since the above studies focused on handling single requests, their policies will bind sufficient resources for each accepted request, enabling re-entangling and swapping until a successful end-to-end entangled link is established. Thus, they may perform well in single-request scenarios but are less effective when dealing with multiple requests.

Table I summarizes the related works based on their setups (e.g., time slot length, fidelity decoherence, scheduling method, path selection) and optimization objectives (e.g., throughput, fidelity, latency, waiting time).

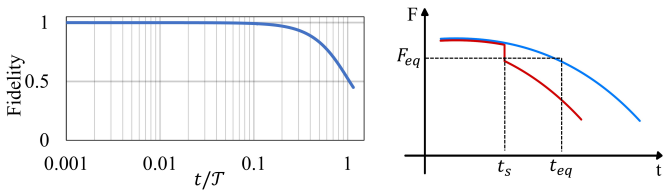
III. Background and Assumptions

A. Fidelity and Decoherence of Entangled Pairs

Fidelity is a classic index to qualify whether an entangled pair is good. An entangled pair with fidelity F can be written in a density matrix as $\rho = F|\Phi^+\rangle\langle\Phi^+| + b|\Phi^-\rangle\langle\Phi^-| + c|\Psi^+\rangle\langle\Psi^+| + d|\Psi^-\rangle\langle\Psi^-|$, where $F + b + c + d = 1$ and $|\Phi^+\rangle$ is our wanted state. In this paper, we consider our entangled pairs are Werner state [8], [35], [36], which has the relation $b = c = d$. A quantum state will interact with the environment and gradually lose its quantum properties (i.e., the fidelity decreases), called decoherence. As $F = b = c = d = 0.25$, it loses all its quantum properties and behaves like a classic random behavior [8]. The decoherence speed depends on the type of quantum memory. The general decoherence can be described as follows:

$$F_d(t) = A + B \times e^{-(t/\mathcal{T})^{\kappa}}, \tag{1}$$

which is an empirical formula fitting to the experimental data [37], [38], as plotted in Fig. 2(a), while the unit of t is second. Note that A, B, $\mathcal{T}$, and κ are the constants according to the adopted technique. In addition, there is a one-to-one mapping between wait time and fidelity because $F_d(t)$ is invertible.



(a) Decoherence curve (b) Fidelity of swapping

Fig. 2. Fidelity loss due to elapsed time and swapping.

B. Entanglement Swapping

Entanglement swapping causes extra fidelity loss [8]. Swapping two Werner states with the fidelity of F_1 and F_2 will lead to the fidelity as follows:

$$F_s(F_1, F_2) = F_1 \times F_2 + \frac{1}{3}(1 - F_1) \times (1 - F_2). \tag{2}$$

Fig. 2(b) shows how the fidelity changes after swapping. The blue curve represents the fidelity change of an entangled pair due to decoherence; on the other hand, the red curve depicts the fidelity of the entangled pair that conducts swapping at t_s. After swapping, there is a sudden drop of fidelity to F_{eq}, and it keeps decohering after the drop. It can be viewed as a shift from the blue curve, i.e., the red curve after t_s behaves the same as the blue curve after t_{eq}. We call t_{eq} the equivalent wait time after swapping, which can be calculated by $t_{eq} = F_d^{-1}(F_{eq})$.

The success probability of entangling decreases exponentially with channel distance [10]. Specifically, the entangling probability between any *adjacent* nodes u and v is defined as:

$$\Pr(u,v) = 1 - (1 - e^{-\lambda \cdot l(u,v)})^{\xi}, \tag{3}$$

where $\xi = \lfloor \frac{\tau}{\mathfrak{T}} \rfloor$ represents the number of entangling attempts, τ is the length of a short time slot, $\mathfrak{T}$ is the entangling time, $l(u,v)$ is the length of the quantum channel between u and v, and λ is a constant determined by the optical fiber material [7]. Then, swapping also has a different success probability at each node v, denoted as $\Pr(v)$ [10]. Thus, the success probability of a path p between the source s and the destination d can be expressed as follows:

$$\Pr(p) = \prod_{(u,v)\in p} \Pr(u,v) \prod_{v\in p\setminus\{s,d\}} \Pr(v). \tag{4}$$

C. Assumptions

For ease of presentation, this paper has the following assumptions: 1) Each entangled pair is described by a Werner state [8], [35], [36], a mixture of a specific pure state and white noise. The white noise is represented by the normalized identity operator in the Hilbert space corresponding to the state [39]. 2) This paper assumes that the fiber resource is always sufficient because it is relatively cheaper than quantum memory. In other words, we focus on quantum memory consumed by the entangling process while assuming all entangling processes will not be blocked by fiber availability. 3) Following [11], all nodes communicate with a central controller and share global knowledge. 4) The entangled pairs between the same pair of two nodes have identical initial fidelity. Besides, all the entangled pairs decohere as the same relation, as shown in Fig. 2(a), because of using the same quantum memory technique (i.e., constants A, B, $\mathcal{T}$, and κ are the same for all nodes). 5) For synchronization, the time slot length should be at least the duration required for a single swapping process (e.g., around 1.1 ms [40]). Since an entangling process generally requires less time than swapping (e.g., 0.25 ms [40]), it can repeatedly attempt entangling within a time slot to improve the entangling success probability [10], as described in Eq. (3).

IV. System Model & Problem Formulation

A. System Model

We consider a QN with multiple quantum nodes, each with limited quantum memory. Nodes connected through a bunch of fibers are adjacent nodes. Data qubits are transmitted via end-to-end entangled pairs in QN. A one-hop entangled pair can be directly constructed via the fiber between two adjacent nodes u and v with the success probability of $\Pr(u,v)$ and the initial fidelity of $F(u,v)$. Plus, a long entangled pair (u,w) can be constructed by performing the swapping process at a repeater v to consume two short entangled pairs (u,v) and (v,w) with the success probability of $\Pr(v)$. The quality of an entangled pair is measured by fidelity. A generated entangled pair will decohere over time, as defined by Eq. (1) in Section III-A.

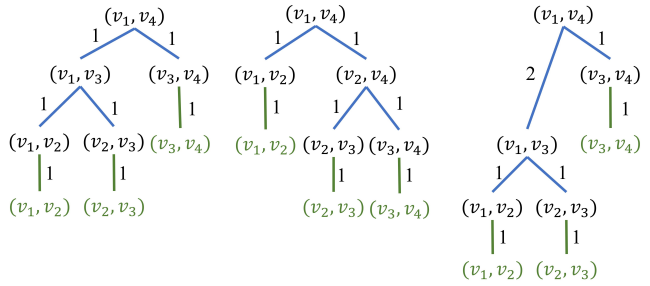
A node can perform an entanglement swapping process when it possesses two qubits, each from different entangled pairs. Subsequently, the memory occupied by these two qubits after swapping will be freed. The fidelity of the resulting longer entangled pair after swapping follows Eq. (2).

We consider a short time slot protocol where a single time slot only accommodates a single process for each memory unit. Each node can freely execute one process within a time slot for each memory unit, such as entangling, swapping, or even idling. The protocol periodically performs all necessary computations in advance, producing a complete operation plan before any entangling or swapping begins. The controller then instructs all quantum nodes to synchronously execute these operations within a fixed batch of time slots. Any request that fails within a batch is deferred to the next batch for reattempt. For clarity, let T be the set of time slots in the current batch.

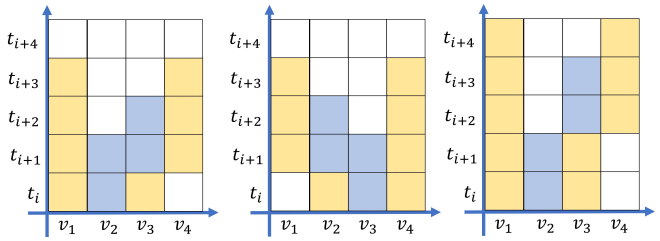
To consider the resource distribution, we define the strategy tree, numerology, and fidelity formulas in our system.

Definition 1. A strategy tree γ is an activity-on-vertex tree structure, describing an entangling and swapping strategy between node v_1 and node v_n on path $p = \{v_1, v_2, ..., v_n\}$. It consists of two parts; one is the entangling part (γ_e), and the other is the swapping part (γ_s), as shown by the green and blue lines in Fig. 1(b), respectively. The entangling part γ_e includes all the external pairs. Each external pair denotes a quantum pair (v_i, v_{i+1}) being entangled, where $i \in \{1, 2, ..., n-1\}$, as shown by the green pair in Fig. 1(b). On the other hand, the swapping part γ_s is an edge-weighted binary tree with a root pair (v_1, v_n), where each leaf pair represents an entangled pair (v_i, v_{i+1}) for each $i \in \{1, 2, ..., n-1\}$ and each edge's weight denotes the number of elapsed time slots for the corresponding entangling or swapping processes. Besides, each non-leaf pair (v_i, v_j) has exactly two child pairs, (v_i, v_k) and (v_k, v_j), where $i < k < j$, denoting a long pair created by consuming two pairs. Last, each leaf pair (v_i, v_{i+1}) in γ_s is connected to a corresponding external pair (v_i, v_{i+1}) in γ_e by an edge with a cost to form γ.

For a given strategy tree γ, each edge in γ has a positive cost (i.e., the number of elapsed time slots for the corresponding entangling or swapping processes). Let ρ_r denote the root pair of the strategy tree γ, and ρ_a denote any pair in γ. We define the cost sum $\delta(\rho_r, \rho_a)$ as the total cost along the simple path from ρ_a to ρ_r in γ, i.e., the total number of elapsed time slots from ρ_a to ρ_r. Following the definition above, for a given strategy tree γ, the root pair ρ_r is assigned a designated time slot $t_r \in T$, where $T = \{1, 2, \ldots, |T|\}$ represents the set of time slots in the current batch. Therefore, the corresponding time slot for a pair ρ_a in γ can be expressed as $t_a = t_r - \delta(\rho_r, \rho_a)$.



(a) Three possible strategy trees for the path $v_1 \to v_2 \to v_3 \to v_4$



(b) The corresponding numerologies for the strategy trees in (a)

Fig. 3. Numerologies induced by different strategy trees.

A strategy tree γ is *feasible* if satisfying the condition: The time slot t_a of each pair $\rho_a \in \gamma$ is within T (i.e., $t_a \in T$). This condition is essential; otherwise, t_a may fall outside the set T.

For example, consider the right strategy tree in Fig. 3(a). Suppose the root pair $\rho_r = (v_1, v_4)$ is with $t_r = 3$. Now, taking the external pair (v_1, v_2) as ρ_a, the corresponding time slot t_a is calculated as $t_a = 3 - \delta(\rho_r, \rho_a) = 3 - 4 = -1$, which means $t_a \notin T$. Thus, if $t_r = 3$, the right strategy tree is infeasible.

Definition 2. A numerology m represents the resource distribution of a specific strategy tree γ. Specifically, let $\rho_p = (v_i, v_j)$ denote a pair at time slot t_p in γ, with a left child pair $\rho_{c_l} = (v_i, v_k)$ at time slot t_{c_l} and a right child pair $\rho_{c_r} = (v_k, v_j)$ at time slot t_{c_r}, where $t_p > t_{c_l}$ and $t_p > t_{c_r}$. Consequently, the left child pair ρ_{c_l} occupies one memory unit of v_i and v_k during time slots $T(\rho_{c_l}) = \{t \in \mathbb{Z} \mid t_{c_l} \le t < t_p\}$, while the right child pair ρ_{c_r} occupies one memory unit of v_k and v_j during time slots $T(\rho_{c_r}) = \{t \in \mathbb{Z} \mid t_{c_r} \le t < t_p\}$. Furthermore, if ρ_p has only a child pair $\rho_c = (v_i, v_j)$ (i.e., the external pair in γ) at time slot t_c, this child pair occupies one memory unit of v_i and v_j during time slots $T(\rho_c) = \{t \in \mathbb{Z} \mid t_c \le t < t_p\}$. Plus, $T(\rho_r) = \{t_r\}$ for the root pair ρ_r at time slot t_r in γ. Therefore, numerology m captures the resource distribution across the nodes in the strategy tree γ and can be formally defined as $m = \{(\rho_a, T(\rho_a)) \mid \rho_a \in \gamma\}$, where each pair $\rho_a \in \gamma$ and its associated time slots are grouped in a tuple $(\rho_a, T(\rho_a))$. The amount of memory on node v occupied by numerology m at time slot t is $\theta_m(t, v) = \left|\{(\rho_a, T(\rho_a)) \in m \mid v \in \rho_a, t \in T(\rho_a)\}\right|$. Note that $\theta_m(t, v) \in \{0, 1, 2\}$.

We further illustrate the example in Fig. 3. Each strategy tree in Fig. 3(a) corresponds to a distinct numerology shown in Fig. 3(b). In this figure, yellow squares indicate a node using one memory unit during that time slot, while blue squares represent a node using two memory units. For the numerology m on the right in Fig. 3(b), we observe that $\theta_m(t_{i+1}, v_2)$ equals 2, as node v_2 simultaneously uses one memory unit for each of the pairs (v_1, v_2) and (v_2, v_3) during time slot t_{i+1}. Additionally, it is worth noting that for any given path, feasible strategy trees have a one-to-one and onto mapping to feasible numerologies.

Lemma 1. For any given path p, there exists a one-to-one and onto mapping f, which maps a feasible strategy tree γ to a feasible numerology m, while f^{-1} denotes the inverse mapping, i.e., $f(\gamma) = m$ and $f^{-1}(m) = \gamma$.

Proof. Definition 2 indicates that each strategy tree has a corresponding numerology, i.e., $f(\gamma) = m$. Then, function f is *onto* since only the numerologies derived from feasible strategy trees are feasible. It suffices to show that for any two different strategy trees γ_1 and γ_2, their corresponding numerologies m_1 and m_2 are different, i.e., $m_1 = f(\gamma_1) \neq f(\gamma_2) = m_2$. We prove it by contradiction. Assume that there exist two different strategy trees mapping to the same numerology for the given path $p = \{v_1, v_2, v_3, ..., v_n\}$, i.e., $f(\gamma_1) = f(\gamma_2) = m$. Let t_r denote the time slot of root pair r in γ_1. Therefore, m uses one memory unit of nodes v_1 and v_n at time t_r, respectively, denoted by (v_1, v_n) in the strategy tree. By Definition 1, we know that a non-leaf node (v_i, v_j) in the swapping part must have exactly two children (v_i, v_k) and (v_k, v_j), where $i < k < j$. This means that if we start scanning m from time slot t_r in decreasing order, we can find exactly one node k which spends two memory units at some time slot. Take Fig. 4(a) as an example. We start scanning from root pair (v_1, v_6) at time t_5. Then, when reaching t_4, we can find node v_4 with two occupied memory units, implying two children (v_1, v_4) and (v_4, v_6). Similarly, we keep scanning the remaining time slots from t_4 to t_3. Note that the resource distribution at t_4 is the combination of resource distributions of pairs (v_1, v_4) and (v_4, v_6), each of which also meets Definition 1, as shown in Fig. 4(b). Thus, we can divide m into two sub-distributions from v_4, find their children individually, and construct the sub-strategy tree recursively. By doing so, we can reconstruct only one possible strategy tree, implying a contradiction. Then, f is a one-to-one and onto mapping, and the lemma follows. □

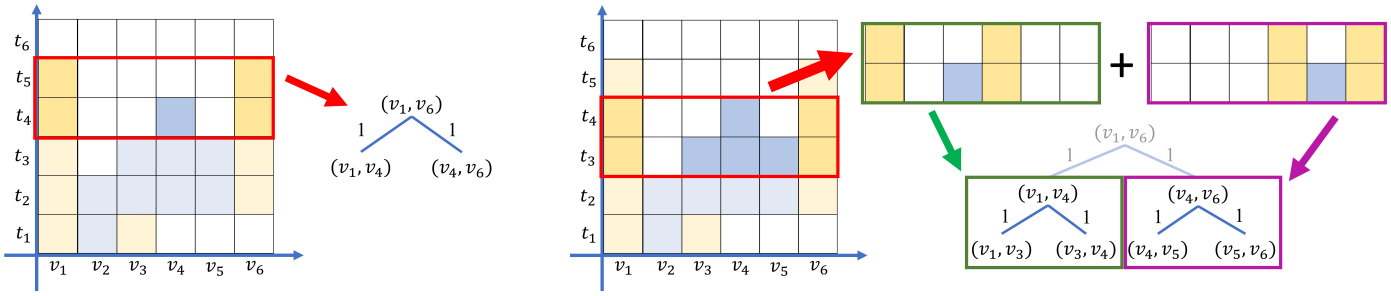
Definition 3. In the short time slot protocol, with Eq. (1), the fidelity of an entangled pair after one-time-slot idling becomes

$$F^{\tau}(F) = F_d(F_d^{-1}(F) + \tau), \tag{5}$$

where τ is the actual duration (i.e., length) of one time slot and F is the fidelity before decay. As two short entangled pairs with fidelity F_1 and F_2 conduct swapping, with Eqs. (2) and (5), the fidelity of the long entangled pair becomes

$$F_s^{\tau}(F_1, F_2) = F_s(F^{\tau}(F_1), F^{\tau}(F_2)). \tag{6}$$

We continue to demonstrate the example in Fig. 3. The left strategy tree in Fig. 3(a) (or the left numerology in Fig. 3(b)) and its mirror strategy tree (i.e., the middle case) will have the same fidelity if all the entangled pairs have the same initial fidelity. The right case is similar to the left one but useful as v_2 and v_4 have no available memory at t_{i+2} and t_{i+1}, respectively (i.e., the left and middle cases cannot be chosen). However, it costs more resources and has lower fidelity than the left case.



(a) Scanning numerology from time slot t_5 to t_4

(b) Scanning numerology from time slot t_4 to t_3

Fig. 4. An illustrating example of transforming a numerology into a strategy tree.

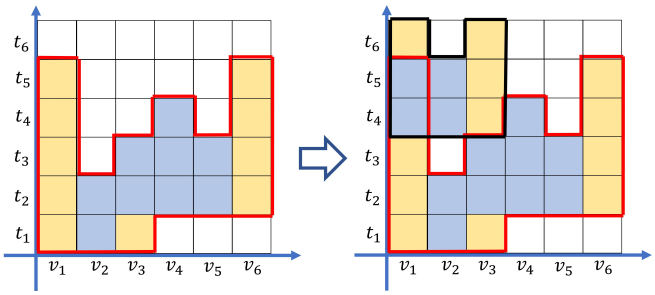


Fig. 5. Overlapping numerologies for two accepted requests.

Note that requests' numerologies could overlap, as shown in Fig. 5, where two requests from v_1 to v_6 and from v_1 to v_3 are denoted by the red- and black-frame numerologies.

As the numerology indicates the strategy tree in the form of quantum resource distribution, the QN strategy tree selection problem can map to a quantum resource allocation problem.

B. Problem Formulation

We formulate the problem based on the above scenario.

Definition 4. Consider a network $G = (V, E)$, where each node $v \in V$ has a memory limit $c^t(v) \in \mathbb{Z}^+$ for each time slot $t \in T = \{1, 2, ..., |T|\}$. The network also includes a set of SD pairs I. Each pair $i \in I$ has a predefined path set $P(i)$ derived by an unspecified routing algorithm.[3] Each path $p \in P(i)$ has the success probability $\Pr(p)$. Given the fidelity threshold $\widehat{F}$, the Optimized Decoherence-aware Strategy for Entangling and Swapping Scheduling Problem (TETRIS) aims to maximize the expected fidelity sum of the chosen numerology for each SD pair, subject to the following constraints.

1) At most, a numerology associated with one path from the path set $P(i)$ can be selected for each SD pair.[3]
2) The amount of occupied memory on each node v does not exceed its memory limit $c^t(v)$ at each time slot t.
3) The selected numerology's fidelity should be no less than the threshold $\widehat{F}$ to ensure high-quality teleportation.

Let $M_p(i)$ denote the set of all numerologies that can be implemented from the path $p \in P(i)$ within T (i.e., the period from time slot 1 to time slot $|T|$) for an SD pair $i \in I$, and let $F(m)$ denote the fidelity of an entangled pair constructed with numerology $m \in M_p(i)$. Following Definition 2, let $\theta_m(t, v) \in \{0, 1, 2\}$ denote the required amount of memory on node $v \in V$ at time slot $t \in T$ when implementing a numerology $m \in M_p(i)$ for SD pair $i \in I$. In this way, TETRIS can be formulated as the following integer linear programming (ILP), where the binary variable x_m^{ip} indicates whether a numerology $m \in M_p(i)$ is chosen ($x_m^{ip} = 1$) or not ($x_m^{ip} = 0$) for SD pair $i \in I$ and path $p \in P(i)$.

$$\text{maximize} \quad \sum_{i \in I} \sum_{p \in P(i)} \sum_{m \in M_p(i)} \Pr(p) \cdot F(m) \cdot x_m^{ip} \tag{7a}$$

$$\text{subject to} \quad \sum_{p \in P(i)} \sum_{m \in M_p(i)} x_m^{ip} \le 1, \quad \forall i \in I \tag{7b}$$

$$\sum_{i \in I} \sum_{p \in P(i)} \sum_{m \in M_p(i)} \theta_m(t, v) \cdot x_m^{ip} \le c^t(v), \quad \forall v \in V, \forall t \in T \tag{7c}$$

$$(F(m) - \widehat{F}) \cdot x_m^{ip} \ge 0, \quad \forall i \in I, \forall p \in P(i), \forall m \in M_p(i) \tag{7d}$$

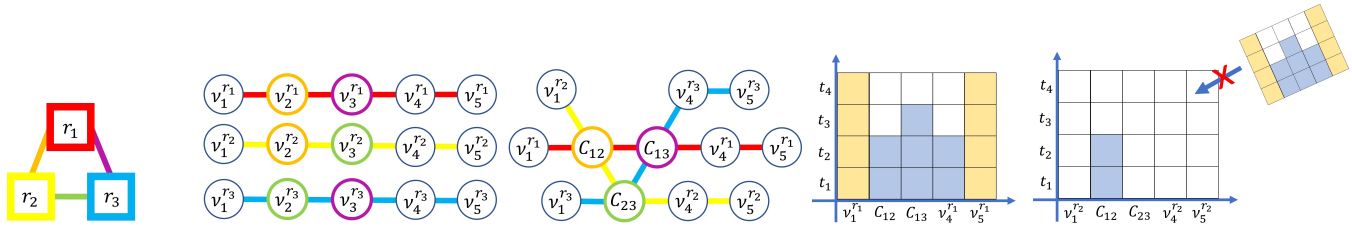
$$x_m^{ip} \in \{0, 1\}, \quad \forall i \in I, \forall p \in P(i), \forall m \in M_p(i) \tag{7e}$$

The objective (7a) maximizes the expected fidelity sum of all SD pairs, where $\Pr(p)$ is the success probability of path p. Constraint (7b) ensures that at most one numerology can be chosen from a single path p within the path set $P(i)$ for each SD pair $i \in I$. Constraint (7c) guarantees that the total amount of memory occupied on each node $v \in V$ at every time slot $t \in T$ does not exceed its memory limit. Constraint (7d) ensures that the selected numerology fidelity should be no less than the threshold $\widehat{F}$ the controller sets based on its policy.

C. NP-hardness and Inapproximability

We prove the hardness of TETRIS by demonstrating that even without the constraint (7d), the problem, termed TETRIS-N, is very challenging. Specifically, we prove the NP-hardness and inapproximability of TETRIS-N in Theorem 1 by reducing the well-known NP-hard problem, the MIS [41], to the TETRIS-N. Therefore, the TETRIS, as a general case of the TETRIS-N, must be at least as hard as TETRIS-N, thereby implying Corollary 1. Note that the MIS asks for a maximum set of vertices in a graph, no two of which are adjacent.

[3]There are many algorithms for finding routing paths for SD pairs [9]–[11] to derive the predefined path set $P(i)$ for each pair $i \in I$.



(a) MIS instance $\mathcal{G}$ (b) Path construction for G (c) TETRIS-N instance G (d) Numerology of r_1 (e) Resource contention for r_2

Fig. 6. An example of the reduction from the MIS to the TETRIS.

Theorem 1. The TETRIS-N is NP-hard and cannot be approximated within any factor of $|I|^{1-\epsilon}$ unless $NP = P$ for any fixed $\epsilon > 0$, where $|I|$ is the number of SD pairs.

Proof. The proof idea is to create an SD pair and a dedicated path and then add them to the TETRIS-N instance for each node in the MIS instance such that the corresponding SD pairs of any two neighboring nodes in the MIS instance cannot be satisfied simultaneously within T. In the following, for any given MIS instance $\{\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}\}$, we show how to construct the corresponding TETRIS-N instance in detail.

For each node in $\mathcal{V}$, we create a corresponding SD pair and add it to the TETRIS-N instance. Subsequently, for each created SD pair, we construct a dedicated path that consists of $2^{\lceil \log |\mathcal{V}| \rceil} + 1$ nodes with $2^{\lceil \log |\mathcal{V}| \rceil}$ edges and then add the path to G of the TETRIS-N instance. Afterward, $|T|$ is set to $\lceil \log |\mathcal{V}| \rceil + 2$. In addition, the amounts of memory of the source and destination on the path are set to 1, and those of the other nodes (i.e., intermediate nodes) on the path are set to 2. Then, the probability of each path is uniformly set to 1 (i.e., $\Pr(p) = 1$). Last, for each pair of neighboring nodes in $\mathcal{G}$, we pick an intermediate node that has not been picked yet from each of their corresponding paths in G and then merge the two picked nodes into a single node. Note that the node induced by merging cannot be picked for merging again. The construction can be done in polynomial time.

Fig. 6 illustrates an example of instance construction. The MIS instance $\mathcal{G}$ includes a clique with three nodes, r_1, r_2, and r_3, which are drawn in the red, yellow, and blue frames, respectively. Then, for each node in $\mathcal{G}$, we create an SD pair and construct a dedicated path that consists of $2^{\lceil \log 3 \rceil} + 1 = 5$ nodes, as shown in Fig. 6(b). Note that the color of the edges on each path corresponds to the color of relevant node in $\mathcal{G}$, and thus the path $\{v_1^{r_1}, v_2^{r_1}, ..., v_5^{r_1}\}$ of SD pair r_1 in G represents the node r_1 in $\mathcal{G}$. In $\mathcal{G}$, an orange edge is between r_1 and r_2. Therefore, $v_2^{r_1}$ and $v_2^{r_2}$ are selected from the two corresponding paths, respectively, and merged into an orange node denoted by C_{12} in G, as shown in Fig. 6(c). Similarly, C_{23} and C_{13} are derived due to the edges (r_2, r_3) and (r_1, r_3), respectively.

We then show that the solution of the MIS instance is one-to-one mapping to that of the constructed TETRIS-N instance. In this TETRIS-N instance, every SD pair has only one path with scarce memory, and its candidate strategy trees include only the full binary tree due to the subtle setting of $|T|$, as shown in Fig. 6(d), where $|T|$ is set to $\lceil \log 3 \rceil + 2 = 4$. Therefore, for any two neighboring nodes in $\mathcal{G}$, the numerologies of the two corresponding pairs in the TETRIS-N instance cannot be selected at the same time since the merged node on the two paths in G has no sufficient memory to serve the two pairs within T. In contrast, they can be selected simultaneously if the two corresponding nodes in $\mathcal{G}$ are not adjacent. Figs. 6(d) and 6(e) continue to show the example. Assume that SD pair r_1 is admitted to construct numerology on path $\{v_1^{r_1}, C_{12}, C_{13}, v_4^{r_1}, v_5^{r_1}\}$. In such a case, r_2 cannot construct the numerology on the path $\{v_1^{r_2}, C_{12}, C_{23}, v_4^{r_2}, v_5^{r_2}\}$ since the memory of C_{12} has been occupied by r_1 (i.e., resource contention), as shown in Fig. 6(e). Thus, the solutions of any MIS instance and its corresponding TETRIS-N instance are one-to-one mapping. Thereby, TETRIS-N is NP-hard.

We continue to show that the TETRIS-N does not admit any approximation algorithm within a factor of $|I|^{1-\epsilon}$ for any fixed $\epsilon > 0$ by contradiction. To this end, suppose there exists an $|I|^{1-\epsilon}$-approximation algorithm A for the TETRIS-N. Following the above-mentioned instance construction, any arbitrary MIS instance has a corresponding TETRIS-N instance. Assume the optimal solution for the MIS instance has k nodes, implying the optimal solution for the TETRIS-N instance maximizes the expected fidelity sum by satisfying the corresponding k pairs. Then, we can employ algorithm A to find a solution that satisfies at least $\frac{k}{|I|^{1-\epsilon}}$ pairs, and the found solution corresponds to a solution with at least $\frac{k}{n^{1-\epsilon}}$ nodes for the MIS instance, where n denotes the number of nodes in the corresponding MIS instance. This is because $|I|$ is set to n during the instance construction. However, unless $NP = P$, the MIS does not admit any approximation ratio within $n^{1-\epsilon}$ for any fixed $\epsilon > 0$ [41]. Thus, algorithm A must not exist; otherwise, A could be employed to derive an $n^{1-\epsilon}$-approximation algorithm for the MIS. The theorem follows. □

Corollary 1. The TETRIS is at least as hard as the TETRIS-N.

V. The Design of Fractional Numerology Packing and Rounding Algorithm (FNPR)

In this section, we first design a bi-criteria approximation algorithm FNPR for the TETRIS-N (i.e., it has an approximation ratio while relaxing the memory limit by a bounded ratio), then remedy the relaxation on the memory limit, and finally extend it to support the TETRIS. In the beginning, we observe the case where τ and $|T|$ are small. In this case, the impact of time slot decoherence diminishes, therefore, all feasible numerologies within T achieve similar fidelity. To better illustrate this perspective, we conduct the simulation to observe the results, as shown in Fig. 7, where the average path

length of each SD pair is about seven. This figure shows that as the time slot length τ and the batch size of time slots T decrease, the ratio of $\frac{F_{\max}}{F_{\min}}$ approaches 1, where $F_{\max}$ denotes maximum fidelity that can be achieved among all feasible numerologies within T, while $F_{\min}$ denotes the minimum one. Specifically, this is because: 1) A smaller time slot length τ reduces decoherence and fidelity loss within each time slot, thereby increasing $F_{\min}$. 2) With a smaller batch size of time slots T, the overall processing time decreases and further limits the feasible numerologies available for each request. Thus, these feasible numerologies yield comparable fidelity levels, as T restricts the strategies for entangling and swapping.

Therefore, in this case, we can temporarily neglect $F(m)$ in the TETRIS-N and consider it later when deriving the bi-criteria approximation ratio. Then, the modified problem formulation has the objective as Eq. (8) and the constraints (7b), (7c), and (7e).

$$\text{maximize} \sum_{i\in I}\sum_{p\in P(i)}\sum_{m\in M_p(i)} \Pr(p)\cdot x_m^{ip}. \tag{8}$$

Specifically, the FNPR employs a combinatorial algorithm with a cleverly-designed DP-based separation oracle, a randomized rounding technique, and two heuristics for ensuring feasibility, as detailed in Sections V-A, V-B, V-C, and V-E, respectively. Furthermore, in Section V-E, we extend the FNPR to address the constraint (7d), guaranteeing the solution feasibility for the TETRIS. Overall, the FNPR can approximate the optimum solution of the TETRIS-N within an approximation ratio of $O(\frac{F_{\max}}{F_{\min}})$ in the price of a bounded relaxation ratio of $O(\log(|V|\cdot|T|))$ on the memory limit (i.e., bi-criteria approximation). Then, with the two heuristics, FNPR's solution can be improved to meet the fidelity threshold and approach the optimum solution as τ and $|T|$ are small enough, i.e., $\frac{F_{\max}}{F_{\min}} \approx 1$.

A. The Combinatorial Algorithm

We first obtain the relaxed LP by replacing constraint (7e) with constraint (9) as follows:

$$x_m^{ip} \geq 0, \qquad \forall i \in I, \forall p \in P(i), \forall m \in M_p(i). \tag{9}$$

However, the number of variables in our ILP grows exponentially with the input size since the number of feasible numerologies for each SD pair may be exponential (i.e., the number of possible binary trees). Thus, the relaxed LP (8), (7b), (7c), (9) cannot be solved by an LP solver in polynomial time. On the other hand, by [42], if an LP is in standard form and has a polynomial number of constraints (except for non-negativity constraints of variables), the number of variables in its dual LP is also polynomial, as described in Definition 5. Then, by combinatorial algorithms, we can obtain a near-optimum solution to the primal LP in polynomial time if the following properties are satisfied [43].

1) Each coefficient on the left-hand side is not greater than the constant on the right-hand side for every inequality of the constraint in the primal LP (except for the non-negativity constraints of variables).

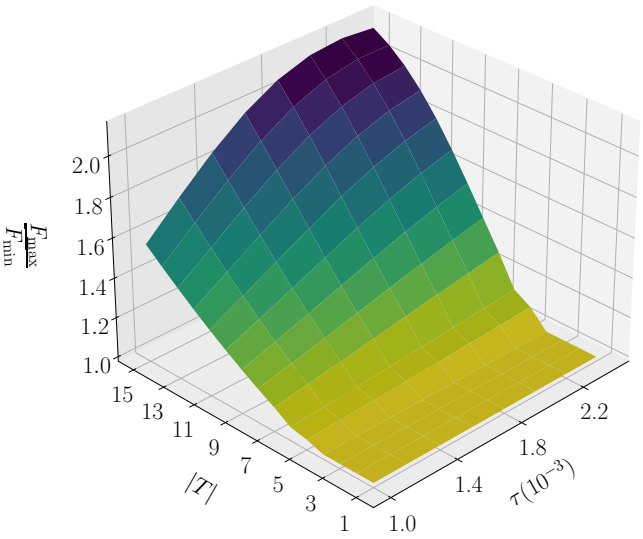


Fig. 7. The ratio of $\frac{F_{\max}}{F_{\min}}$ for different values of τ and $|T|$.

2) In the dual LP, all coefficients on the left-hand-side of inequality and the constant on the right-hand-side of inequality are positive in each constraint (except for the non-negativity constraints of variables).
3) A separation oracle exists to tell whether there is a violated constraint for the dual LP.

Definition 5. [42] The LP standard form is $\max\{c^T x | Ax \leq b, x \geq 0\}$, where x is an $n\times 1$ variable vector, and c, b, and A denote $n\times 1$, $m\times 1$, $m\times n$ constant vectors, respectively. The corresponding dual LP is $\min\{b^T y | A^T y \geq c, y \geq 0\}$, where y is an $m\times 1$ variable vector. Note that $x \geq 0$ and $y \geq 0$ are the non-negativity constraints of x and y.

Clearly, the relaxed primal LP (8), (7b), (7c), and (9) meets the first property. By Definition 5, we associate dual variables α_i and β_v^t with each constraint in (7b) and (7c), respectively. Thus, we obtain the corresponding dual LP (10a)–(10c). It can be seen that the dual LP also satisfies the second property.

$$\text{minimize} \quad \sum_{i\in I}\alpha_i + \sum_{v\in V}\sum_{t\in T} c^t(v)\cdot\beta_v^t \tag{10a}$$

$$\text{subject to} \quad \alpha_i + \sum_{v\in V}\sum_{t\in T}\theta_m(t,v)\cdot\beta_v^t \geq \Pr(p), \quad \forall i\in I, \forall p\in P(i), \forall m\in M_p(i) \tag{10b}$$

$$\alpha_i, \beta_v^t \geq 0, \qquad \forall i\in I, \forall v\in V, \forall t\in T \tag{10c}$$

Then, we design a separation oracle for the dual LP and get the fractional solution for the relaxed primal LP in Section V-B. However, the fractional solution may be infeasible for the ILP (8), (7b), (7c), and (7e). To this end, we propose an LP rounding algorithm and improve the solution in Sections V-C and V-E. Last, we analyze its performance in Section V-D.

B. The Separation Oracle

Given an arbitrary (fractional) solution (α, β) to the dual LP, the separation oracle tells whether a violated constraint exists.

$$\hat{g}(t,(s,d),(\sigma_s,\sigma_d)) = \begin{cases} \beta_s^t + \beta_d^t + \beta_s^{t-1} + \beta_d^{t-1} & \text{if } \substack{(s,d)\in E,\, t\geq 2,\\ c^t(s) \text{ and } c^{t-1}(s)\geq\sigma_s,\\ c^t(d) \text{ and } c^{t-1}(d)\geq\sigma_d;} \\ \beta_s^t + \beta_d^t + \min\left\{\hat{g}(t-1,(s,d),(\sigma_s,\sigma_d)), \min\limits_{k\in\mathcal{I}(s,d)}\left\{\hat{h}(t-1,(s,k,d),(\sigma_s,\sigma_d))\right\}\right\} & \text{else if } \substack{t\geq 2, c^t(s)\geq\sigma_s,\\ c^t(d)\geq\sigma_d;} \\ \infty & \text{otherwise.} \end{cases} \tag{13}$$

$$\hat{h}(t,(s,k,d),(\sigma_s,\sigma_d)) = \min\left\{\hat{g}(t,(s,k),(\sigma_s,2)) + \hat{g}(t,(k,d),(1,\sigma_d)),\ \hat{g}(t,(s,k),(\sigma_s,1)) + \hat{g}(t,(k,d),(2,\sigma_d))\right\}. \tag{14}$$

It is easy to examine every constraint in (10c) in polynomial time, but the challenge arises in identifying whether a violated constraint exists in (10b). Note that the number of SD pairs and the size of path set for each SD pair are both polynomial. Therefore, it suffices to identify the most violated constraint in (10b) for each pair $i \in I$ and each path $p \in P(i)$. This can be done by computing

$$\min_{m\in M_p(i)} \sum_{v\in V}\sum_{t\in T} \theta_m(t,v)\cdot\beta_v^t. \tag{11}$$

To this end, we subtly design a DP algorithm to iteratively solve a larger subproblem by examining its smaller subproblems. Specifically, any numerology $m \in M_p(i)$ has a one-to-one and onto mapping to a strategy tree through path p with a root represented by the SD pair (s_i, d_i), as shown in Fig. 3. We introduce the function $\hat{f}(T,(s,d))$ to find a numerology m that can generate an entangled pair from node s to node d through the path p within T, while minimizing $\sum_{v\in V}\sum_{t\in T}\theta_m(t,v)\cdot\beta_v^t$. In this way, the desired numerology for a given SD pair i and path $p \in P(i)$ can be found by using $\hat{f}(T,(s_i,d_i))$.

To compute $\hat{f}(T,(s,d))$, we examine all local minimum numerologies, each corresponding to a strategy tree rooted at a time slot $t \in T$. We know that any feasible numerology m requires $\theta_m(t,v)$ memory units on node v at time slot t, and a strategy tree can be built by combining two sub-strategy trees. Thus, we define a function $\hat{g}(t,(s,d),(\sigma_s,\sigma_d))$ to find a numerology m with a (sub-)strategy tree rooted at (s,d) at time slot t, which can minimize $\sum_{v\in V}\sum_{t'\in T}\theta_m(t',v)\cdot\beta_v^{t'}$ while ensuring the memory limit $c^t(s) \geq \sigma_s$ and $c^t(d) \geq \sigma_d$, where $\sigma_s, \sigma_d \in \{1,2\}$. Then, Eq. (12) derives $\hat{f}(T,(s,d))$.

$$\hat{f}(T,(s,d)) = \min_{t\in T}\left\{\hat{g}(t,(s,d),(1,1))\right\} \tag{12}$$

We derive $\hat{g}(t,(s,d),(\sigma_s,\sigma_d))$ according to the three cases.

1) *Leaves of strategy tree*. If $(s,d) \in E$ and $t \geq 2$, then we reach a pair of leaves in a strategy tree. Thus, it returns the sum of values of these two leaves and their external nodes as long as the memory limits of s and d are sufficient.
2) *Non-leaves of strategy tree*. If $(s,d) \notin E$ and $t \geq 2$, then we have two possible cases at time slot $t-1$. 1) Both s and d idle. 2) s and d conduct swapping to consume two entangled links (s,k) and (k,d) to acquire an entangled link (s,d), where k is an intermediate node between s and d on path p. It examines the values of s and d plus the value of every case and then returns the minimum if the memory limits of s and d are sufficient.
3) *Wrong time or no capacity*. If the time slot $t \leq 1$, it is impossible to have an entangled link (s,d) at time slot t. Moreover, if the memory limits of s and d do not meet the requirements, the numerology does not exist. Thus, these cases result in an infeasible solution.

Based on the above cases, to derive $\hat{g}(t,(s,d),(\sigma_s,\sigma_d))$, we additionally define the function $\hat{h}(t,(s,k,d),(\sigma_s,\sigma_d))$ in Eq. (14) to represent the minimum $\sum_{v\in V}\sum_{t'\in T}\theta_m(t',v)\cdot\beta_v^{t'}$ of the entangled pair (s,d) that can be generated by limiting the amount of used memory on node k in either the left or right subproblem. Specifically, if the amount of memory on node k is limited in the left (or right) subproblem, then we set $\sigma_d = 2$ (or $\sigma_s = 2$) to check whether the amount of memory of k is at least 2. Otherwise, the left (or right) subproblem has no limit, and we set $\sigma_d = 1$ (or $\sigma_s = 1$). Then, the recurrence relation of $\hat{g}(t,(s,d),(\sigma_s,\sigma_d))$ can be expressed as Eq. (13), where $\mathcal{I}(s,d)$ denotes the set of intermediate nodes on the path p between s and d (i.e., the nodes on p but excluding s and d). Eqs. (12)–(14) can help find the numerology m that minimize $\sum_{v\in V}\sum_{t\in T}\theta_m(t,v)\cdot\beta_v^t$ for any given $i \in I$ and $p \in P(i)$.

With Eqs. (12)–(14), we can efficiently identify the most violated constraint in (10b), satisfying the third property. As all three properties are satisfied, we can solve the relaxed primal LP and obtain a fractional solution $\hat{x}_m^{ip}$ using the combinatorial algorithm in [43] that incorporates our DP separation oracle.

C. The Randomized Rounding

Given a fractional solution $\hat{x}_m^{ip}$ of our primal LP, we design a randomized rounding algorithm to obtain the solution for TETRIS-N. Specifically, let $\hat{M}_p(i)$ denote the set of numerologies with $\hat{x}_m^{ip} > 0$ for each SD pair $i \in I$ and path $p \in P(i)$. Then, let $\hat{M}(i) = \hat{M}_{p_1}(i) \cup \hat{M}_{p_2}(i) \cup \cdots \cup \hat{M}_{p_n}(i)$, where $\{p_1, p_2, \ldots, p_n\} = P(i)$. It is worth noting that the size of $\hat{M}(i)$ is polynomial since the combinatorial algorithm terminates in polynomial time [43]. We then interpret $\hat{x}_m^{ip}$ as the probability of selecting the numerology m for each SD pair i. For example, assume that SD pair i has three numerologies m_1, m_2, and m_3 in $\hat{M}(i)$, and they are $\hat{x}_{m_1}^{ip_1} = 0.1$, $\hat{x}_{m_2}^{ip_1} = 0.2$, and $\hat{x}_{m_3}^{ip_2} = 0.3$, respectively. Thus, the probabilities of selecting m_1, m_2, and m_3 are 0.1, 0.2, and 0.3, respectively, while the probability of not selecting any numerology for SD pair i is 0.4.

D. Bi-Criteria Approximation Ratio and Time Complexity

We first analyze the feasibility and the relaxation bound of Eqs. (7b) and (7c) after randomized rounding, and the time complexity in Lemmas 2, 3, and 4, respectively. We then derive the bi-criteria approximation ratio in Theorem 3. To this end, we use the Chernoff bound in Theorem 2 as follows.

Theorem 2 (Chernoff bound [44])**.** There is a set of n independent random variables $x_1, ..., x_n$, where $x_i \in [0, 1]$ for each $i \in [1, n]$. Let $X = \sum_{i=1}^{n} x_i$ and $\mu = \mathbb{E}[X]$. Then,

$$\Pr\left[\sum_{i=1}^{n} x_i \geq (1+\epsilon)\mu\right] \leq e^{\frac{-\epsilon^2\mu}{2+\epsilon}}. \tag{15}$$

Lemma 2. The FNPR will satisfy constraint (7b).

Proof. The FNPR selects at most one numerology from the union of all sets $M_p(i)$ for each SD pair i using the randomized rounding in Section V-C. Thus, the lemma holds. □

Lemma 3. The probability that the FNPR relaxes constraint (7c) for any node v at any time slot t by more than a factor of $(1 + 4\ln(|V| \cdot |T|))$ is at most $\frac{1}{|V|^2 \cdot |T|^2}$, which is negligible.

Proof. For each node v at time slot t, we define a random variable $z^i_{t,v}$ that denotes the amount of memory occupied on node v at time slot t for each SD pair i. Note that a quantum node at time slot t may require $\theta_m(t, v)$ units of memory to perform numerology m. Thus, $z^i_{t,v}$ can be expressed as:

$$z^i_{t,v} = \begin{cases} \theta_m(t,v) & \text{with probability } \hat{x}^{ip}_m; \\ 0 & \text{otherwise.} \end{cases}$$

Note that their sum $Z_{t,v} = \sum_{i \in I} z^i_{t,v}$ is exactly the amount of memory needed on v at time slot t after rounding. Then, we derive the upper bound of the expectation of the amount of memory occupied on node v at time slot t as follows:

$$\begin{aligned}\mathbb{E}[Z_{t,v}] &= \mathbb{E}\left[\sum_{i \in I} z^i_{t,v}\right] = \sum_{i \in I} \mathbb{E}[z^i_{t,v}] \\ &= \sum_{i \in I}\left(\sum_{p \in P(i)} \sum_{m \in M_p(i)} \theta_m(t,v) \cdot \hat{x}^{ip}_m\right) \leq c^t(v).\end{aligned}$$

Note that the last inequality directly follows the memory limit constraint (7c). Afterward, we can prove the lemma by Chernoff bound as follows:

$$\begin{aligned}&\Pr\big(\exists v \in V, \exists t \in T : Z_{t,v} \geq (1 + 4\ln(|V| \cdot |T|)) \cdot c^t(v)\big) \\ \leq\ & \sum_{v \in V}\sum_{t \in T} \Pr\left(\frac{Z_{t,v}}{c^t(v)} \geq 1 + 4\ln(|V| \cdot |T|)\right) \\ \leq\ & \sum_{v \in V}\sum_{t \in T} \Pr\left(\sum_{i \in I} \frac{z^i_{t,v}}{c^t(v)} \geq 1 + 4\ln(|V| \cdot |T|)\right) \\ \leq\ & |V| \cdot |T| \cdot e^{\frac{-(4\ln(|V| \cdot |T|))^2}{2+4\ln(|V| \cdot |T|)}} \leq |V|^{-2} \cdot |T|^{-2}.\end{aligned}$$

Note that $\frac{z^i_{t,v}}{c^t(v)} \leq 1$ since the separation oracle adopts Eq. (13) to examine the memory limit, meeting the condition of the Chernoff bound (i.e., every variable $\frac{z^i_{t,v}}{c^t(v)}$ ranges from 0 to 1). Besides, the last inequality holds when $|V| \cdot |T| \geq 5$. □

Lemma 4. The FNPR can be executed in polynomial time.

Proof. According to [43], the combinatorial algorithm executes $O(\omega^{-2}\eta_1 \log \eta_1)$ times of separation oracle, where ω is a constant error bound of primal LP solution and η_1 is the number of constraints (except for the non-negativity constraints), i.e., $\eta_1 = (|V| \cdot |T| + |I|)$. Thus, it outputs $O(\omega^{-2}\eta_1 \log \eta_1)$ numerologies for randomized rounding. In addition, each time of separation oracle takes $\eta_2 = O(|V|^3 \cdot |T| \cdot |I|)$. Then, the combinatorial algorithm takes $O(\omega^{-2}\eta_1\eta_2 \log \eta_1)$, and thus the overall time complexity of FNPR is $O(\omega^{-2}\eta_1\eta_2 \log \eta_1)$. □

Theorem 3. The bi-criteria approximation ratio of the FNPR for the TETRIS-N is $(O(\frac{F_{\max}}{F_{\min}}), O(\log(|V| \cdot |T|))$.

Proof. With Lemmas 2, 3, and 4, we can know that FNPR relaxes the memory limit by a ratio of at most $O(\log(|V| \cdot |T|))$ with high probability and runs in polynomial time. It suffices to focus on the gap between the optimal solution and the FNPR's solution. For ease of presentation, let OPT, OPT^{nF}, and OPT^{nF}_{PL} be the optimum values of the TETRIS-N (i.e., the ILP (7) with no constraint (7d)), the TETRIS-N without considering fidelity (i.e., the ILP (8), (7b), (7c), (7e)), and the primal LP (8), (7b), (7c), (9), respectively.

Clearly, $OPT \leq OPT^{nF} \cdot F_{\max}$ because OPT^{nF} has the maximum number of SD pairs with an allocated numerology. Besides, $OPT^{nF} \leq OPT^{nF}_{PL}$ since the primal LP (8), (7b), (7c), (9) is a LP relaxation of the ILP (8), (7b), (7c), (7e). In addition, let $\hat{x}^{ip}_m$ denote the solution output by the combinatorial algorithm. Then, let $\bar{x}^{ip}_m$ be the rounded solution. Since the expected value of the rounded solution is equal to the optimal solution of the primal LP, the expected fidelity sum of the rounded solution can be bound as follows:

$$\begin{aligned}&\mathbb{E}\left[\sum_{i \in I}\sum_{p \in P(i)}\sum_{m \in M_p(i)} \Pr(p) \cdot F(m) \cdot \bar{x}^{ip}_m\right] \\ \geq\ & \sum_{i \in I}\sum_{p \in P(i)}\sum_{m \in M_p(i)} \mathbb{E}\left[\Pr(p) \cdot F_{\min} \cdot \bar{x}^{ip}_m\right] \\ =\ & F_{\min} \cdot \sum_{i \in I}\sum_{p \in P(i)}\sum_{m \in M_p(i)} \Pr(p) \cdot \hat{x}^{ip}_m \\ =\ & F_{\min} \cdot APP^{nF}_{PL} \geq \frac{F_{\min}}{1+\omega} \cdot OPT^{nF}_{PL} \\ \geq\ & \frac{F_{\min}}{1+\omega} \cdot OPT^{nF} \geq \frac{F_{\min}}{F_{\max}} \cdot \frac{OPT}{1+\omega},\end{aligned}$$

where APP^{nF}_{PL} is the near-optimal value of primal LP from the combinatorial algorithm with a user-defined constant error bound $\omega > 0$, and the second inequality holds due to [43]. □

E. Heuristic Algorithms for Solution Improvement

1) Remedy for the Memory Limit Relaxation: Since the solution acquired by the FNPR after rounding may relax the memory limit on nodes with a bounded ratio, the following steps are adopted to deal with the memory limit relaxation. Let $\bar{x}^{ip}_m$ and $\bar{M}_p(i)$ denote the variables in the rounded solution and the set of numerologies with $\bar{x}^{ip}_m > 0$ for each $i \in I$ and $p \in P(i)$, respectively. The heuristic has two phases as follows.

In the first phase, for each node v and each time slot t where the memory limit is exceeded, we sort the numerologies occupying v's memory at time slot t (i.e., $\bar{M}(v,t) = \{m \mid \theta_m(t,v) \cdot \bar{x}^{ip}_m > 0, \forall m \in \bar{M}_p(i), \forall p \in P(i), \forall i \in I\}$) in non-decreasing order of their expected fidelity, and then iteratively

remove numerologies in $\bar{M}(v,t)$ based on this order until the memory limit of node v at time slot t is satisfied to make the solution feasible.

In the second phase, we utilize the residual resources as much as possible. We iteratively allocate the numerology m for each pair i in non-increasing order of the value of those positive variables $\hat{x}_m^{ip}$ (i.e., the fractional solution of LP (8), (7b), (7c), (9)) until no numerology can be chosen or accommodated by the residual network. The heuristic is a polynomial-time algorithm since the number of positive variables $\hat{x}_m^{ip}$ is polynomial.

2) Extension to Support the TETRIS: We design the FNPR based on the observation of numerology characteristics in the context of small τ and $|T|$. However, as τ and $|T|$ increase, the FNPR may not guarantee solution feasibility for the TETRIS. To address this limitation, we extend the FNPR to support the TETRIS by introducing a heuristic as follows. Following the heuristic proposed in Section V-E1, we additionally add one step to remove the numerology from $\bar{M}(v,t)$ if its fidelity is less than the threshold $\widehat{F}$ in the first phase. In the second phase, we only add the numerology m with adequate fidelity for each pair i in non-increasing order of the value of those positive variables $\hat{x}_m^{ip}$ until no m can be chosen or accommodated. This heuristic can ensure solution feasibility, and later, the numerical results show the modified solution can approach the optimum.

VI. The Design of Fidelity-Load Trade-Off Algorithm (FLTO)

Although the FNPR can approximate the optimum when τ and $|T|$ are small, its performance might decrease when τ or $|T|$ increases since it neglects fidelity loss. However, it is non-trivial to find the optimal solution for the TETRIS since every SD pair could have an exponential number of numerologies for selection. Fortunately, the separation oracle in the FNPR serves as inspiration for determining the optimal numerology. Following this inspiration and the intrinsic properties of TETRIS, we first design a DP algorithm that can find the optimal numerology with the maximum expected fidelity on the selected path $p \in P(i)$ for any given *single* SD pair i.

Intuitively, we can derive a solution for the TETRIS by iteratively adding a new request based on the available resource by invoking the DP algorithm until no more request can be satisfied. Nonetheless, such a greedy algorithm tends to fall into a local optimum trap since it lacks an overview of current resource utilization for load balancing unless a proper index is provided for the greedy algorithm to estimate the efficiency of the current allocation. Inspired by [23], we subtly design the resource efficiency index (REI) to evaluate a numerology. With REI, our greedy algorithm can mitigate severe network congestion and increase the fidelity sum as τ or $|T|$ grows.

A. Numerology with the Max. Exp. Fidelity for Single Request

Similar to Section V-B, we exploit DP to iteratively solve a larger subproblem by examining the optimum solution of each smaller subproblem to derive the optimal solution for ILP (7a)−(7e) when the SD pair i is single. Hence, we introduce the recursive function $f(T,(s,d))$ to return the maximum fidelity that can be achieved between node s and node d on a specific path p within T as well as the corresponding numerology. Similar to Eq. (12), $f(T,(s,d))$ requires an additional function $g(t,(s,d),(\sigma_s,\sigma_d))$ to derive its value, as shown in Eq. (16).

$$f(T,(s,d)) = \max_{t \in T} \big\{ g(t,(s,d),(1,1)) \mid g(t,(s,d),(1,1)) \geq \widehat{F} \big\}. \tag{16}$$

Note that if the derived maximum fidelity is less than the fidelity threshold $\widehat{F}$, the corresponding path will not be considered. Following Eq. (13), to derive $g(t,(s,d),(\sigma_s,\sigma_d))$, we define the function $h(t,(s,k,d),(\sigma_s,\sigma_d))$ in Eq. (18), which represents the maximum fidelity of the entangled pair (s,d) that can be generated by limiting the amount of used memory on node k in either the left or right subproblem. The details are omitted due to the similarity, and the recurrence relation of $g(t,(s,d),(\sigma_s,\sigma_d))$ can be expressed as Eq. (17). In this way, we can identify the numerology with the maximum fidelity on any specific path for any given single SD pair. Subsequently, it is possible to find the numerology with the maximum expected fidelity among all paths in $P(i)$ since the size of predefined path set $P(i)$ is polynomial. To this end, we calculate $\Pr(p) \cdot F(\hat{m})$ of the numerology $\hat{m}$ returned by Eq. (16) for each $p \in P(i)$ and choose the numerology with the highest value.

B. Greedy Algorithm with DP Technique

We can design a greedy algorithm to solve the ILP (7a)−(7e) by utilizing the DP algorithm in Section VI-A. That is, we iteratively choose the numerology with the maximum expected fidelity by the DP algorithm among all SD pairs and then allocate the resources to the corresponding SD pair until no more numerologies can be chosen. However, such a naive algorithm may cause congestion. To remedy the side effect, we design the REI to help us choose a numerology while avoiding hot spots. Specifically, the REI is defined as follows:

$$\mathfrak{R}(i,m) = \frac{\Pr(p) \cdot F(m)}{\sum_{v \in V} \sum_{t \in T} \frac{\theta_m(t,v)}{c^t(v)}}, \tag{22}$$

where $p \in P(i)$ and $m \in M_p(i)$ for SD pair i. Note that the numerator and denominator denote the expected fidelity and resource cost of numerology m, respectively. However, finding the numerology with the maximum $\mathfrak{R}(i,m)$ among all possible numerologies is computationally-intensive. Thus, FLTO focuses on two types of candidate numerologies for each SD pair i as follows: 1) the numerology with the maximum expected fidelity on each path $p \in P(i)$, denoted by m_1^{ip}, and 2) the numerology with the minimum resource cost on each path $p \in P(i)$, denoted by m_2^{ip}. Subsequently, FLTO selects the numerology with the highest $\mathfrak{R}(i,m)$ from all candidates across all SD pairs, i.e., $\max_{i \in I, p \in P(i)} \{\mathfrak{R}(i,m_1^{ip}), \mathfrak{R}(i,m_2^{ip})\}$.

Specifically, FLTO invokes the DP algorithm in Section VI-A to determine the m_1^{ip} for each path $p \in P(i)$. In addition, m_2^{ip} for each p can be found using a similar DP algorithm designed in Eqs. (19)−(21). These equations describe the recurrence relation to search the numerology with the minimum

$$g(t,(s,d),(\sigma_s,\sigma_d)) = \begin{cases} F(s,d) & \text{if } \substack{(s,d)\in E,\, t\geq 2,\\ c^t(s) \text{ and } c^{t-1}(s)\geq\sigma_s,\\ c^t(d) \text{ and } c^{t-1}(d)\geq\sigma_d;} \\ \max\left\{F^\tau\left(g(t-1,(s,d),(\sigma_s,\sigma_d))\right), \max\limits_{k\in\bar{I}(s,d)}\left\{h(t-1,(s,k,d),(\sigma_s,\sigma_d))\right\}\right\} & \text{else if } \substack{t\geq 2,\, c^t(s)\geq\sigma_s,\\ c^t(d)\geq\sigma_d;} \\ -\infty & \text{otherwise.} \end{cases} \tag{17}$$

$$h(t,(s,k,d),(\sigma_s,\sigma_d)) = \max\left\{F_s^\tau\left(g(t,(s,k),(\sigma_s,2)),g(t,(k,d),(1,\sigma_d))\right),\ F_s^\tau\left(g(t,(s,k),(\sigma_s,1)),g(t,(k,d),(2,\sigma_d))\right)\right\}. \tag{18}$$

$$\bar{f}(T,(s,d)) = \min_{t\in T}\left\{\bar{g}(t,(s,d),(1,1))\right\}. \tag{19}$$

$$\bar{g}(t,(s,d),(\sigma_s,\sigma_d)) = \begin{cases} r_s^t + r_d^t + r_s^{t-1} + r_d^{t-1} & \text{if } \substack{(s,d)\in E,\, t\geq 2,\\ c^t(s) \text{ and } c^{t-1}(s)\geq\sigma_s,\\ c^t(d) \text{ and } c^{t-1}(d)\geq\sigma_d;} \\ r_s^t + r_d^t + \min\left\{\bar{g}(t-1,(s,d),(\sigma_s,\sigma_d)), \min\limits_{k\in\bar{I}(s,d)}\left\{\bar{h}(t-1,(s,k,d),(\sigma_s,\sigma_d))\right\}\right\} & \text{else if } \substack{t\geq 2,\, c^t(s)\geq\sigma_s,\\ c^t(d)\geq\sigma_d;} \\ \infty & \text{otherwise.} \end{cases} \tag{20}$$

$$\bar{h}(t,(s,k,d),(\sigma_s,\sigma_d)) = \min\left\{\bar{g}(t,(s,k),(\sigma_s,2)) + \bar{g}(t,(k,d),(1,\sigma_d)),\ \bar{g}(t,(s,k),(\sigma_s,1)) + \bar{g}(t,(k,d),(2,\sigma_d))\right\}. \tag{21}$$

resource cost, where r_v^t denotes the resource cost of node v at time slot t and is set to $\frac{1}{c^t(v)}$ based on Eq. (22). In brief, $\bar{f}(T,(s,d))$ finds the numerology that can generate an entangled pair from node s to node d on specific path p with the minimum resource cost within T, while $\bar{g}(t,(s,d),(\sigma_s,\sigma_d))$ and $\bar{h}(t,(s,k,d),(\sigma_s,\sigma_d))$ are additional functions similar to Eqs. (17) and (18). Note that the numerology will not be considered if the fidelity is less than the fidelity threshold $\widehat{F}$, and the DP details are omitted due to the similarity.

Overall, FLTO iteratively chooses an SD pair with the numerology that has the maximum REI among all unsaturated SD pairs and then allocates the resources asked by that numerology to saturate the SD pair until no more SD pairs can be served. Then, we analyze the time complexity of FLTO. Both the DP algorithm Eqs. (16) and (19) take $O(|V|^3\cdot|T|)$ for each SD pair $i\in I$ on specific path $p\in P(i)$. Thus, it needs $O(|V|^3\cdot|T|\cdot|P|)$ for each SD pair $i\in I$ calculating for all paths, where $|P|$ denotes the maximum size of path set $P(i)$ between all SD pairs. Since FLTO chooses one SD pair among all pairs for each round and the number of chosen SD pairs is at most $|I|$, the overall time complexity is $O(|V|^3\cdot|T|\cdot|P|\cdot|I|^2)$.

VII. Performance Evaluation

A. Simulation Settings

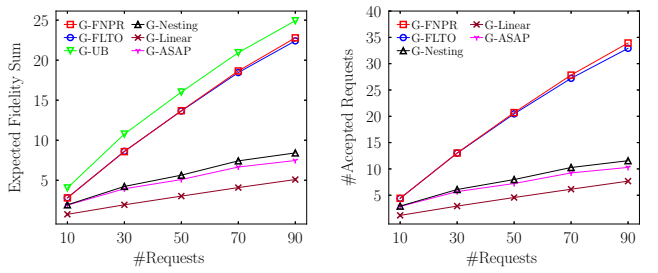
The model and *default* parameters are as follows. The QN topology is generated using the Waxman model [45], with $|V|=100$ nodes deployed over a 150×300 km region, and an average edge length of 30 km. For clarity, a network topology example is given in Appendix B of the supplementary material. We select $|I|=50$ random SD pairs within a batch of $|T|=13$ time slots, where each time slot length is $\tau=2$ ms. This setup ensures that the total duration of a batch ($\tau\times|T|$) does not exceed 40 ms, consistent with recent advances in quantum memory [38]. The choice of $\tau=2$ ms is reasonable since a single swapping process requires approximately 1.1 ms [40]. For each edge, we set $\lambda=0.045$/km [7] and the entangling time $\mathfrak{T}=0.25$ ms [40]. Thus, the average entangling probability is around 0.9, with the number of entangling attempts per time slot given by $\xi=\lfloor\frac{\tau}{\mathfrak{T}}\rfloor=8$. The initial fidelity for each edge is randomly sampled from the range $[0.7,0.98]$ [12]. The average memory limit of a node (i.e., $c^t(v)$) is set between 6 to 14 units, aligned with [33] (10 to 16 per node). The swapping probability is set to 0.9 [10], [11]. Following [38], the parameters in Eq. (1) are set as $A=0.25$, $B=0.75$, $\mathcal{T}=40$ ms, and $\kappa=2$. The fidelity threshold $\widehat{F}$ is set to 0.5. Then, we apply GREEDY [9], Q-CAST [10], and REPS [11] to generate the predefined path set $P(i)$ for each pair $i\in I$. For simplicity, they are denoted by G, Q, and R. Each result is averaged over 50 trials.

We compare FNPR and FLTO with the following methods.

1) Nesting [7] first conducts entangling for any two adjacent nodes until all entangled links on the path are created. After that, it greedily maximizes the number of swapping processes for each time slot until the end-to-end entangled link is built. Thus, Nesting has a near-balanced tree structure.
2) Linear [14] performs swapping operations one by one, starting from the source towards the destination, after all entangled links on the path are created. Thus, the numerology that Linear tends to use is a near-biased tree structure.
3) ASAP [33] performs swapping as soon as two adjacent entangled links are successfully generated. ASAP needs to bind the resources for re-entangling and swapping until a successful end-to-end entangled link is achieved.
4) UB is the fractional solution of LP (8), (7b), (7c), (9), derived by the combinatorial algorithm with the separation oracle in FNPR to know the upper bound of the optimum. Note that it is not necessarily feasible for TETRIS.

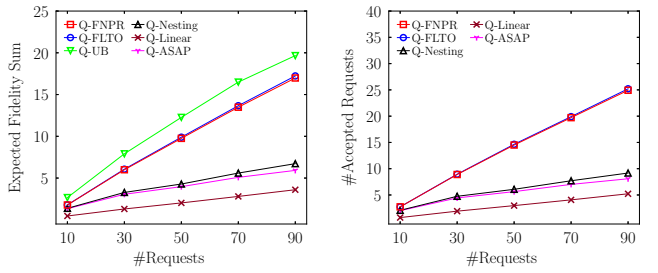
B. Numerical Results

Figs. 8–11 illustrate the expected fidelity sum and the number of accepted requests under different parameters. Overall, both FNPR and FLTO consistently demonstrate superior performance across all parameter settings, effectively enhancing



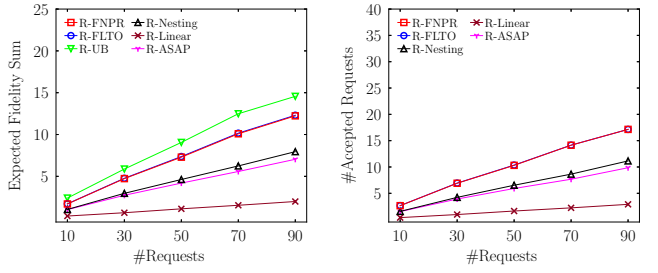
(a) #Requests vs Exp. Fid. Sum

(b) #Requests vs #Acc. Requests



(c) #Requests vs Exp. Fid. Sum

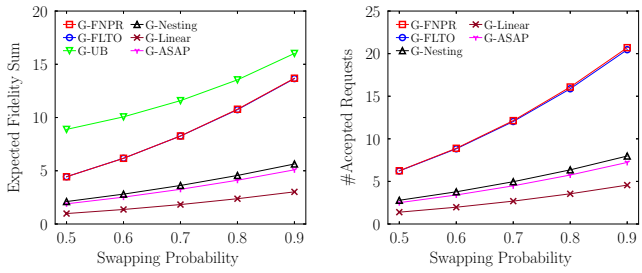
(d) #Requests vs #Acc. Requests



(e) #Requests vs Exp. Fid. Sum

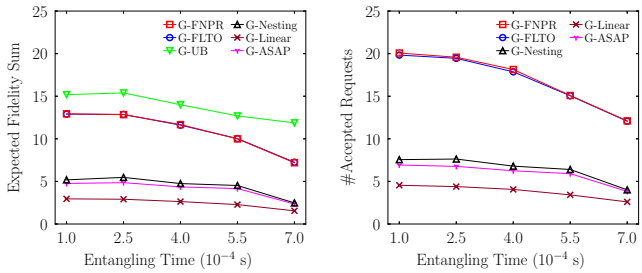
(f) #Requests vs #Acc. Requests

Fig. 8. Effect of different parameters and predefined paths on various metrics.



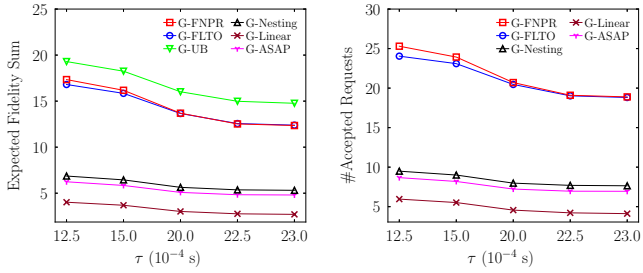
(a) Swap. Prob. vs Exp. Fid. Sum

(b) Swap. Prob. vs #Acc. Requests



(c) Ent. Time vs Exp. Fid. Sum

(d) Ent. Time vs #Acc. Requests



(e) τ vs Exp. Fid. Sum

(f) τ vs #Acc. Requests

Fig. 9. Effect of different parameters on various metrics.

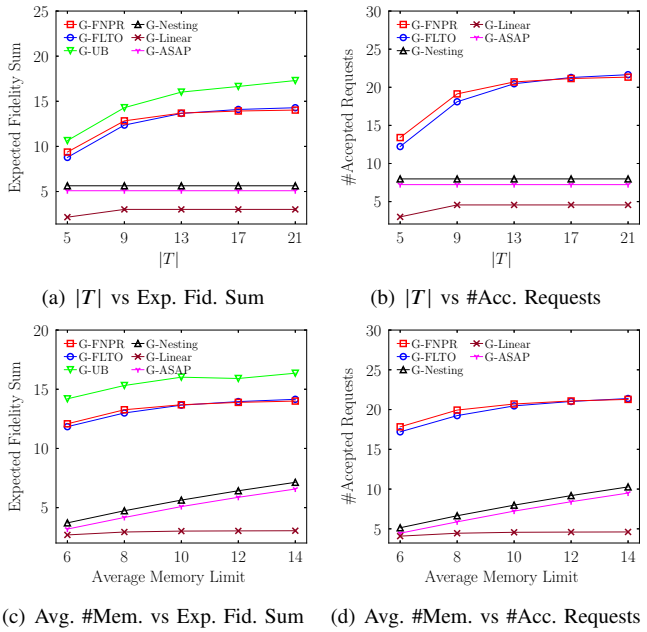
fidelity and throughput while efficiently utilizing the resources of the quantum network.

1) Effect of Number of Requests: Fig. 8 shows the expected fidelity sum and number of accepted requests across varying numbers of requests. As the number of requests increases, both the expected fidelity sum and the number of accepted requests generally exhibit an upward trend since more numerologies can be considered for selection to fully utilize the network resources. No matter which routing algorithm is employed, FNPR and FLTO significantly outperform Nesting, Linear, and ASAP. The results show that FNPR (and FLTO) averagely outperforms Nesting, Linear, and ASAP on expected fidelity sum by up to 60% (60%), 78% (77%), and 63% (62%), respectively. This is because FNPR can achieve near-optimum under a moderate τ and $|T|$ by the combinatorial algorithm with the separation oracle, while FLTO uses REI to choose numerologies, balancing resource utilization and fidelity.

2) Effect of Swapping Probability: In the literature, the swapping probability for simulations is typically between 0.5 and 1. To make the model more comprehensive and realistic, we compare the performance of algorithms under different swapping probabilities ranging from 0.5 to 0.9. Figs. 9(a) and 9(b) show the expected fidelity sum and the number of accepted requests under a different swapping probability. Generally, as the swapping probability increases, both the expected fidelity sum and the number of accepted requests tend to rise. Additionally, FNPR and FLTO consistently outperform the other algorithms across all swapping probabilities, indicating they are more robust and efficient in managing resources, leading to superior performance regardless of the swapping probability.

3) Effect of Entangling Time: The timing relationship between entangling and swapping may be affected by distance in reality, causing ξ to vary under certain conditions. To achieve more comprehensive comparisons, we consider variations in the entangling time length ranging from 0.1 ms to 0.7 ms, with corresponding ξ values ranging from 8 to 2. Figs. 9(c) and 9(d) show the expected fidelity sum and the number of accepted requests for different entangling time lengths. Generally, as entangling time length increases, both the expected fidelity sum and the number of accepted requests tend to decrease. This is because a smaller ξ allows for fewer entangling attempts, leading to a lower entangling probability. Additionally, FNPR and FLTO outperform the other algorithms.

4) Effect of τ: Figs. 9(e) and 9(f) illustrate the expected fidelity sum and the number of accepted requests under a different τ. Note that varying τ affects ξ, as the number of entangling attempts depends on the time slot length. Although a larger τ allows more entangling attempts, leading to higher entangling



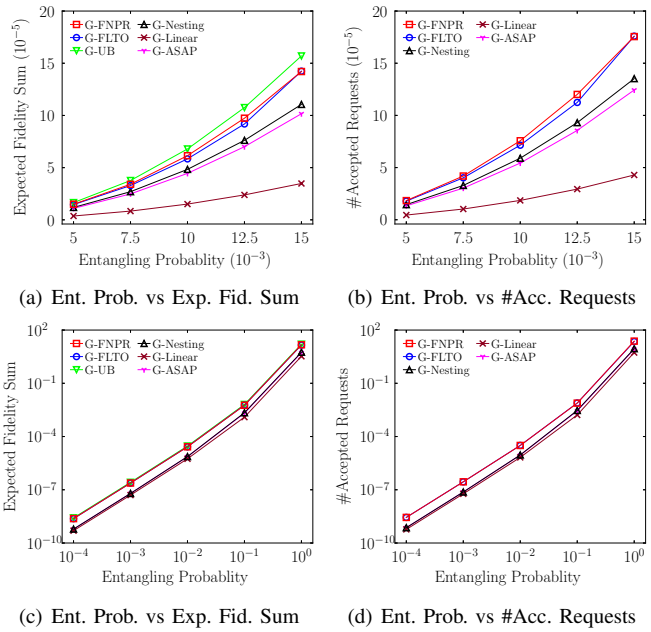
(a) $|T|$ vs Exp. Fid. Sum (b) $|T|$ vs #Acc. Requests

(c) Avg. #Mem. vs Exp. Fid. Sum (d) Avg. #Mem. vs #Acc. Requests

(e) Min. Init. Fid. vs Exp. Fid. Sum (f) Min. Init. Fid. vs #Acc. Requests

(g) Fid. Thr. vs Exp. Fid. Sum (h) Fid. Thr. vs #Acc. Requests

Fig. 10. Effect of different parameters on various metrics.



(a) Ent. Prob. vs Exp. Fid. Sum (b) Ent. Prob. vs #Acc. Requests

(c) Ent. Prob. vs Exp. Fid. Sum (d) Ent. Prob. vs #Acc. Requests

Fig. 11. Effect of different entangling probabilities on various metrics.

probability, the results show that a larger τ conversely results in a lower expected fidelity sum and fewer accepted requests. This is because the impact of decoherence over a longer time slot outweighs the benefits of increased entangling probability in our setting. Thus, some numerologies will not be admitted due to their low fidelity. Fig. 9(e) further confirms that FNPR outperforms FLTO as τ is small, while FLTO is slightly better when τ is large, as described in Section VI.

5) Effect of Resource Allocation: Figs. 10(a)−10(d) show the expected fidelity sum and the number of accepted requests under different $|T|$ and average memory limits. Generally, as $|T|$ or the amount of memory increases, both the fidelity sum and the number of accepted requests tend to increase, benefiting from sufficient resources. In Figs. 10(a) and 10(b), FNPR slightly outperforms FLTO as $|T|$ is small since FNPR can be close to the optimum solution as $\frac{F_{\max}}{F_{\min}} \approx 1$. However, in scenarios with a large $|T|$, FLTO proves more suitable since it can avoid selecting inefficient numerologies via our DP algorithm with REI when the resources are sufficient. The above results show that each has its own merits.

6) Effect of Initial Fidelity and Threshold: Figs. 10(e)−10(f) and 10(g)−10(h) show the expected fidelity sum and the number of accepted requests at varying minimum initial fidelity levels and fidelity thresholds. Generally, as the minimum initial fidelity increases, both the expected fidelity sum and the number of accepted requests increase since most numerologies can achieve high fidelity. FNPR performs the best under higher minimum initial fidelity because it can approach the optimum solution as $\frac{F_{\max}}{F_{\min}} \approx 1$. Besides, as the fidelity threshold increases, the expected fidelity sum tends to decrease inevitably; however, FNPR and FLTO still outperform others in all cases.

7) Effect of Entangling Probability: To further observe the effect of low entangling probability, we conduct experiments by treating the entangling probability (i.e., $\Pr(u, v)$) as an abstract input parameter, as shown in Fig. 11. Figs. 11(a) and 11(b) show the expected fidelity sum and number of accepted requests across varying entangling probabilities. In this simulation, the entangling probability is set to range from 0.005 to 0.015, which covers the realistic parameter range discussed in [46]. Due to the low entangling probability, the fidelity sum and number of accepted requests are low under this setting for all simulated methods. Even under such adverse conditions, the proposed FNPR and FLTO still outperform the other algorithms by 20% (18%) to 75% (75%) on average. This is because FNPR utilizes the combinatorial algorithm with randomized rounding to achieve near-optimal solutions, while FLTO invokes the DP techniques to design an effective greedy algorithm. Both algorithms are designed to solve the TETRIS

suitably. To more deeply observe the performance changes under different entangling probabilities, we conducted simulations across a wider range of entangling probabilities (i.e., from 0.0001 to 1), as shown in Figs. 11(c) and 11(d). The results show that our proposed algorithms consistently outperform the other algorithms across all entangling probability levels, which ensures the robustness of the proposed algorithms.

VIII. Conclusion

This paper proposes a promising short time slot protocol for entangling and swapping scheduling with a novel optimization problem TETRIS. The TETRIS asks for the solution that selects the numerology (strategy tree) for each accepted request while maximizing the fidelity sum of all accepted requests. To solve the TETRIS, we design two new algorithms. FNPR is a bi-criteria approximation algorithm by solving an LP with a separation oracle and rounding the solution for the cases where the time slot length and the number of time slots in a batch are small. FLTO is a greedy algorithm with a proper index to call two DP-based algorithms adaptively for the other cases. Finally, the simulation results manifest that our algorithms can outperform the existing methods by up to $60 \sim 78\%$ in general, and by $20 \sim 75\%$ even under low entangling probabilities.

References

[1] C. Elliott, "Building the quantum network," *New J. Phys.*, vol. 4, no. 1, p. 46, 2002.

[2] Y.-A. Chen *et al.*, "An integrated space-to-ground quantum communication network over 4,600 kilometres," *Nature*, vol. 589, no. 7841, pp. 214–219, 2021.

[3] S. Daiss *et al.*, "A quantum-logic gate between distant quantum-network modules," *Science*, vol. 371, no. 6529, pp. 614–617, 2021.

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

[5] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information: 10th Anniversary Edition*. Cambridge university press, 2010.

[6] M. Schlosshauer, "Decoherence, the measurement problem, and interpretations of quantum mechanics," *Rev. Mod. Phys.*, vol. 76, no. 4, p. 1267, 2005.

[7] N. Sangouard *et al.*, "Quantum repeaters based on atomic ensembles and linear optics," *Rev. Mod. Phys.*, vol. 83, no. 1, p. 33, 2011.

[8] R. Van Meter, *Quantum Networking*. John Wiley & Sons, Ltd, 2014.

[9] M. Pant *et al.*, "Routing entanglement in the quantum internet," *npj Quantum Inf.*, vol. 5, no. 1, p. 25, 2019.

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

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

[12] Y. Zhao *et al.*, "E2E fidelity aware routing and purification for throughput maximization in quantum networks," in *IEEE INFOCOM*, 2022.

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

[14] A. Farahbakhsh and C. Feng, "Opportunistic routing in quantum networks," in *IEEE INFOCOM*, 2022.

[15] Y. Zeng *et al.*, "Entanglement routing design over quantum networks," *IEEE/ACM Trans. Netw.*, vol. 32, no. 1, pp. 352–367, 2024.

[16] Z. Yiming *et al.*, "Entanglement routing over quantum networks using greenberger-horne-zeilinger measurements," in *IEEE ICDCS*, 2023.

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

[18] K. Chakraborty *et al.*, "Entanglement distribution in a quantum network: A multicommodity flow-based approach," *IEEE Trans. Quantum Eng.*, vol. 1, pp. 1–21, 2020.

[19] S. Pouryousef *et al.*, "A quantum overlay network for efficient entanglement distribution," in *IEEE INFOCOM*, 2023.

[20] G. Zhao *et al.*, "Segmented entanglement establishment with all-optical switching in quantum networks," *IEEE/ACM Trans. Netw.*, vol. 32, no. 1, pp. 268–282, 2024.

[21] C. Cicconetti, M. Conti, and A. Passarella, "Request scheduling in quantum networks," *IEEE Trans. Quantum Eng.*, vol. 2, pp. 2–17, 2021.

[22] M. Ghaderibaneh *et al.*, "Efficient quantum network communication using optimized entanglement swapping trees," *IEEE Trans. Quantum Eng.*, vol. 3, pp. 1–20, 2022.

[23] Y. Azar and O. Regev, "Strongly polynomial algorithms for the unsplittable flow problem," in *IPCO*, 2001.

[24] R. Van Meter and J. Touch, "Designing quantum repeater networks," *IEEE Commun. Mag.*, vol. 51, no. 8, pp. 64–71, 2013.

[25] S. Muralidharan *et al.*, "Optimal architectures for long distance quantum communication," *Sci. Rep.*, vol. 6, no. 1, p. 20463, 2016.

[26] S. Pirandola *et al.*, "Fundamental limits of repeaterless quantum communications," *Nat. Commun.*, vol. 8, no. 1, p. 15043, 2017.

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

[28] Y. Zhao and C. Qiao, "Distributed transport protocols for quantum data networks," *IEEE/ACM Trans. Netw.*, vol. 31, no. 6, pp. 2777–2792, 2023.

[29] S.-M. Huang *et al.*, "Socially-aware opportunistic routing with path segment selection in quantum networks," in *IEEE GLOBECOM*, 2023.

[30] L. Yang *et al.*, "Asynchronous entanglement provisioning and routing for distributed quantum computing," in *IEEE INFOCOM*, 2023.

[31] S. Haldar *et al.*, "Fast and reliable entanglement distribution with quantum repeaters: Principles for improving protocols using reinforcement learning," *Phys. Rev. Appl.*, vol. 21, p. 024041, 2024.

[32] Á. G. Iñesta *et al.*, "Optimal entanglement distribution policies in homogeneous repeater chains with cutoffs," *npj Quantum Inf.*, vol. 9, p. 46, 2023.

[33] S. Haldar *et al.*, "Reducing classical communication costs in multiplexed quantum repeaters using hardware-aware quasi-local policies," *arXiv preprint arXiv:2401.13168*, 2024.

[34] K. Goodenough, T. Coopmans, and D. Towsley, "On noise in swap asap repeater chains: exact analytics, distributions and tight approximations," *arXiv preprint arXiv:2404.07146*, 2024.

[35] R. F. Werner, "Quantum states with einstein-podolsky-rosen correlations admitting a hidden-variable model," *Phys. Rev. A*, vol. 40, no. 8, p. 4277, 1989.

[36] N. K. Panigrahy *et al.*, "On the capacity region of a quantum switch with entanglement purification," in *IEEE INFOCOM*, 2023.

[37] M. H. Abobeih *et al.*, "One-second coherence for a single electron spin coupled to a multi-qubit nuclear-spin environment," *Nat. Commun.*, vol. 9, no. 1, p. 2552, 2018.

[38] C. E. Bradley *et al.*, "A ten-qubit solid-state spin register with quantum memory up to one minute," *Phys. Rev. X.*, vol. 9, no. 3, p. 031045, 2019.

[39] A. Sen, U. Sen, Č. Brukner, V. Bužek, and M. Żukowski, "Entanglement swapping of noisy states: A kind of superadditivity in nonclassicality," *Phys. Rev. A: At., Mol., Opt. Phys.*, vol. 72, no. 4, p. 042310, 2005.

[40] M. Pompili *et al.*, "Realization of a multinode quantum network of remote solid-state qubits," *Science*, vol. 372, no. 6539, pp. 259–264, 2021.

[41] J. Hastad, "Clique is hard to approximate within $n^{1-\epsilon}$," in *IEEE FOCS*, 1996.

[42] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms, Third Edition*. The MIT Press, 2009.

[43] N. Garg and J. Könemann, "Faster and simpler algorithms for multicommodity flow and other fractional packing problems," *SIAM J. Comput.*, vol. 37, no. 2, pp. 630–652, 2007.

[44] M. Mitzenmacher and E. Upfal, *Probability and Computing: Randomization and Probabilistic Techniques in Algorithms and Data Analysis*, 2nd ed. USA: Cambridge University Press, 2017.

[45] B. Waxman, "Routing of multipoint connections," *IEEE J. Sel. Areas Commun.*, vol. 6, no. 9, pp. 1617–1622, 1988.

[46] A. J. Stolk *et al.*, "Metropolitan-scale heralded entanglement of solid-state qubits," *Sci. Adv.*, vol. 10, no. 44, p. eadp6442, 2024.

Appendix A
Discussion of the Counter-Intuitive Example

We conducted additional simulations and clarified the reasons behind this counter-intuitive example. There are two main

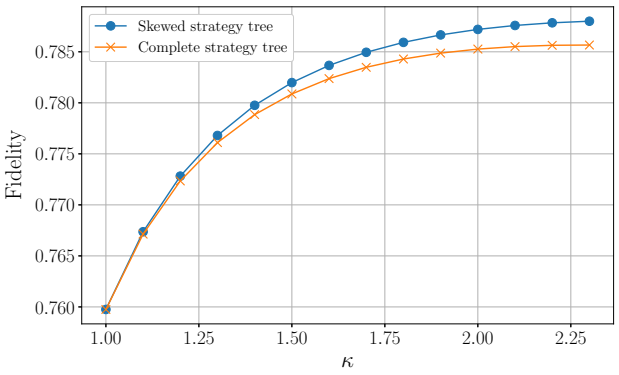


Fig. 12. Comparison of fidelity between the strategy trees in Figs. 1(b) and 1(c) under different values of κ.

factors explaining why Fig. 1(b) has better fidelity than Fig. 1(c): 1) In our decoherence model (i.e., Eq. (1)), we set $\kappa = 2$. Note that κ controls the decoherence speed. For example, in Fig. 12, we compare the two strategy trees in Figs. 1(b) and 1(c) under different values of κ ranging from 1 to 2.3. As κ increases, the fidelity of the skewed strategy tree in Fig. 1(b) tends to surpass that of the complete strategy tree in Fig. 1(c) because a higher κ reduces decoherence rate. 2) Most of the swapping processes in Fig. 1(b) consume two entangled pairs with similar fidelity and thus suffer less fidelity loss due to swapping processes. From the perspective above, we can also conclude that if all the links along the path have similar initial fidelity, the complete strategy tree will conversely outperform the other. For example, if all the generated pairs (v_1, v_2), (v_2, v_3), (v_3, v_4), and (v_4, v_5) have initial fidelity 0.98, then the final fidelity of the strategy trees in Figs. 1(b) and 1(c) will be 0.889 and 0.891, respectively.

Appendix B
Example of Network topology

Fig. 13 shows a representative illustration of the network topology generated using the Waxman model.

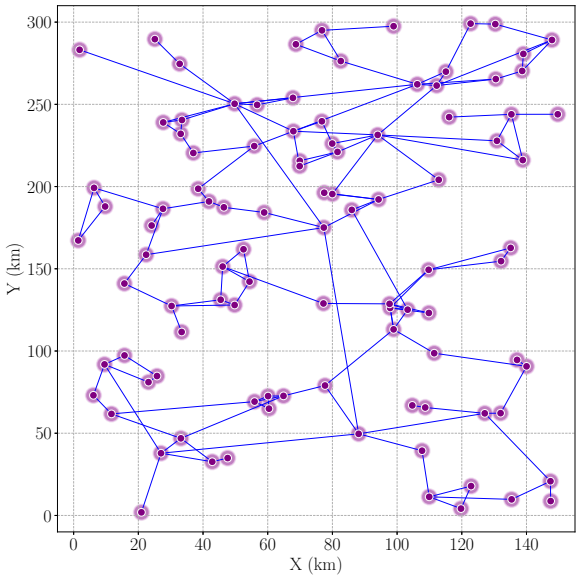


Fig. 13. Network topology sample generated using the Waxman model.