Concurrent Entanglement Routing for Quantum Networks: Model and Designs

Shouqian Shi
sshi27@ucsc.edu
University of California, Santa Cruz

Chen Qian
qian@ucsc.edu
University of California, Santa Cruz

ABSTRACT

Quantum entanglement enables important computing applications such as quantum key distribution. Based on quantum entanglement, quantum networks are built to provide long-distance secret sharing between two remote communication parties. Establishing a multi-hop quantum entanglement exhibits a high failure rate, and existing quantum networks rely on trusted repeater nodes to transmit quantum bits. However, when the scale of a quantum network increases, it requires end-to-end multi-hop quantum entanglements in order to deliver secret bits without letting the repeaters know the secret bits. This work focuses on the entanglement routing problem, whose objective is to build long-distance entanglements via untrusted repeaters for concurrent source-destination pairs through multiple hops. Different from existing work that analyzes the traditional routing techniques on special network topologies, we present a comprehensive entanglement routing model that reflects the differences between quantum networks and classical networks as well as a new entanglement routing algorithm that utilizes the unique properties of quantum networks. Evaluation results show that the proposed algorithm Q-CAST increases the number of successful long-distance entanglements by a big margin compared to other methods. The model and simulator developed by this work may encourage more network researchers to study the entanglement routing problem.

CCS CONCEPTS

• **Networks** → **Network protocol design**; **Routing protocols**;
• **Hardware** → **Quantum communication and cryptography**;
• **Computer systems organization** → **Quantum computing**.

KEYWORDS

Quantum Internet; Quantum Networks; Entanglement Routing; Network Modeling

ACM Reference Format:
Shouqian Shi and Chen Qian. 2020. Concurrent Entanglement Routing for Quantum Networks: Model and Designs. In *Annual conference of the ACM Special Interest Group on Data Communication on the applications, technologies, architectures, and protocols for computer communication (SIGCOMM '20), August 10–14, 2020, Virtual Event, NY, USA.* ACM, New York, NY, USA, 14 pages. https://doi.org/10.1145/3387514.3405853

SIGCOMM '20, August 10–14, 2020, Virtual Event, NY, USA

ACM ISBN 978-1-4503-7955-7/20/08.
https://doi.org/10.1145/3387514.3405853

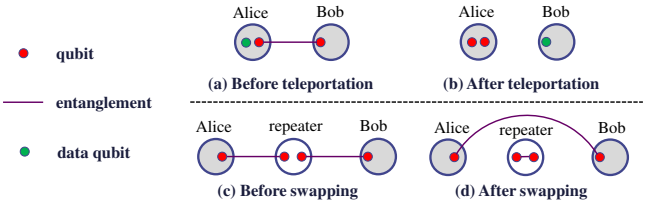


Figure 1: (a-b) Quantum teleportation to transmit a qubit (consuming a local or distant entanglement). (c-d) Entanglement swapping to build a long-distance entanglement.

1 INTRODUCTION

Secure information exchange via quantum networks has been proposed, studied, and validated since 1980s [5, 17–19, 35, 38, 48, 59] and many experimental studies have demonstrated that long-distance secrete sharing via quantum networks can become successful in reality, such as the DARPA quantum network [18, 19], SECOQC Vienna QKD network [38], the Tokyo QKD network [48], and the satellite quantum network in China [59]. A quantum network (also called a quantum Internet) is an interconnection of quantum processors and repeaters that can generate, exchange, and process quantum information [8, 10, 25, 57]. It transmits information in the form of quantum bits, called *qubits*, and stores qubits in quantum memories[1]. Quantum networks are not meant to replace the classical Internet communication. In fact, they supplement the classical Internet and enable a number of important applications such as quantum key distribution (QKD) [5, 17, 41], clock synchronization [26], secure remote computation [7], and distributed consensus [15], most of which cannot be easily achieved by the classical Internet.

Most applications of quantum networks are developed based on two important features of quantum entanglement. 1) Quantum entanglement is inherently private by the laws of quantum mechanics such as the "no-cloning theorem" [37] and hence prevents a third party from eavesdropping the communication [17]. Quantum entanglement enables a perfect solution to the fundamental problem of network security: key distribution (also known as key agreement) [16]. Compared to public key cryptography [46], quantum key distribution (QKD) has provable security in terms of information theory and forward secrecy [57], instead of relying on the computational complexity of certain functions such as factorization. 2) Quantum entanglement provides strong correlation and instantaneous coordination of the communication parties. Hence, quantum entanglement can achieve tasks that are difficult to coordinate in classical networks, and a well-known one is *quantum teleportation*, as shown in Fig. 1 (a) and (b). If a pair of entangled qubits are shared by Alice and Bob, then Alice can send one bit of secret information to Bob with the help of quantum measurement

[1] *e.g.*, transmitting a pair of entangled photons and storing the entanglement state into a pair of nitrogen-vacancy centers in two remote diamonds [11, 13]

and the classical Internet [45]. Hence, QKD can be achieved via quantum entanglement.

We note that quantum networks will become practical in the near future, and they do not rely on the success of well-functioning quantum computers. Both academia and industry have a time-to-time debate on when a practical quantum computer will be available with a sufficient amount of qubits to implement the proposed quantum algorithms, such as Shor's integer factorization [50]. It seems that well-functioning quantum computers might not become available in the near future. However, many applications of quantum networks can be implemented with one or two qubits. Considering the QKD example, we are able to distribute a secret bit with only one entanglement pair. By repeating the 1-pair QKD process we can generate secret keys with a sufficient length.

To generate a quantum entanglement between two parties Alice and Bob, an entangled pair of photons are created, and each photon is sent to a party through a channel, such as an optical fiber. However, the optical fiber is inherently lossy and the success rate p of establishing an entanglement pair decays exponentially with the physical distance between the two parties [42, 43]. Hence, to increase the success rate of *long-distance* quantum entanglement, a number of *quantum repeaters* need to be deployed between two long-distance communication parties [43, 57]. Many existing quantum networks [18, 38, 48, 59] rely on "*trusted repeaters*" to relay entanglements. Each trusted repeater gets the *actual data qubit* teleported from the sender and teleports the data qubit to the receiver, similar to the "store-and-forward" process in classical networks. A more attractive approach is to use quantum swapping [6, 34, 35]. As shown in Fig. 1 (c) and (d), via entanglement swapping, a quantum repeater that holds entanglements to both Alice and Bob can turn the two one-hop entanglements into one direct entanglement between Alice and Bob. Multi-hop swapping is also possible with a path of repeaters holding entanglements with their predecessors and successors. During quantum swapping, a repeater does not know the qubit information hence it does not have to be trusted.

This work focuses on a key problem called *entanglement routing*, whose objective is to build long-distance entanglements through multiple hops of quantum repeaters and entanglement swapping, even if the *repeaters may be untrusted or corrupted* [24, 36]. Entanglement routing has not been thoroughly investigated but is *necessary in future large-scale quantum networks*: When a quantum network scale increases, similar to the Internet, users do not always trust all forwarding devices between the source and destination or some trusted repeaters may be corrupted. In addition, a large number of trusted repeaters increase the attack space and the vulnerability of the whole system. Entanglement routing finds an end-to-end path of concurrent quantum entanglement through a number of repeaters and performs quantum swapping without letting the repeaters know the data bits. This can be considered on the network layer of a quantum network [13]. Existing works that investigate the entanglement routing problem of quantum networks are limited to analyzing the traditional routing algorithms (Dijkstra shortest paths, multipath routing, and greedy routing) on special network topologies (ring, sphere, or grid), such as the very recent ones [24, 36].

Similar to other network routing problems, entanglement routing is a distributed algorithm design problem to utilize the underlying link-layer models [13, 24, 36]. While the physical layer and link layer studies of quantum networks require experimental validations on special and expensive hardware, the entanglement routing algorithms can be comprehensively evaluated via simulations as long as the link-layer model reflects the practical physical facts, similar to prior studies for intra-/inter-domain routing, wireless multi-hop routing [20], data center routing [3, 51, 60], etc.

To our knowledge, this is the first work of a comprehensive protocol design specifically for entanglement routing in quantum networks, with new models, new metrics, and new algorithms, working on arbitrary network topologies. We present a comprehensive entanglement routing model that reflects the difference between quantum networks and classical networks and propose new entanglement routing designs that utilize the unique properties of quantum networks. We propose a few routing metrics that particularly fit quantum networks instead of using hop-count and physical distance. The proposed algorithms include realistic protocol-design considerations such as arbitrary network topologies, multiple concurrent sources and destinations to compete for resources, link state exchanges, and limited qubit capacity of each node, most of which have not been considered by prior studies. Evaluation results show that the proposed algorithm Q-CAST increases the number of successful long-distance entanglements by a big margin compared to other known methods. More importantly, this study may encourage more network researchers to study the entanglement routing problem. We present and clarify the models and problems of entanglement routing, with the comparison of similar terms and concepts used in classical network research. A simulator with algorithm implementation, topology generation, statistics, and network visualization functions is available on this link [1].

The rest of this paper is organized as follows. Section 2 presents the related work of quantum network routing and Section 3 introduces the network model. We present the algorithm designs in Section 4. The evaluation results are shown in Section 5. We discuss some related issues in Section 6 and conclude this work in Section 7.

2 RELATED WORK

Quantum information exchange has been proposed, studied, and validated for more than 20 years [5, 17, 18, 35, 38, 48, 59]. The concept of quantum networks is first introduced by the DARPA quantum network project aiming to implement secure communication in the early 2000s [18]. Recent implementations include the SECOQC Vienna QKD network [38], the Tokyo QKD network [48], and China's satellite quantum network [59]. These experimental works rely on trusted repeaters.

In order to design future large quantum networks in which repeaters may not trust each other, one fundamental problem is to route quantum entanglements with high reliability in quantum repeater networks [54]. Van Meter *et al.* studies applying Dijkstra algorithm to repeater network [31]. Pirandola *et al.* discuss the limits of repeaterless quantum communication [43] and propose multi-path routing in a diamond topology [40]. Schoute *et al.* [49] propose a framework to study quantum network routing. However, their discussion is only limited to ring or sphere topology. Das *et al.* [14] compare different special topologies for entanglement routing. Caleffi [9] studies the optimal routing problem in a chain of repeaters. Pant *et al.* [36] propose solutions for entanglement routing

in grid networks. [24] proposes virtual-path based greedy routing in ring and grid networks. Vardoyan *et al.* [55] study a quantum entanglement switch in a star topology. All these studies assume specialized network topologies such as a grid or ring, which may be over-simplified. The topologies of practical quantum networks may be arbitrary graphs because 1) the end hosts in quantum networks must exist on specified locations according to application requirements, instead of following certain topologies; 2) deploying unnecessary devices just to create a certain topology is a waste of resource.

The above studies are limited to analyzing the traditional routing algorithms on special network topologies. Compared to them, this paper is the first work of a comprehensive protocol design specifically for entanglement routing in quantum networks, with new models, new metrics, and new algorithms, working on arbitrary network topologies. It includes three unique improvements: 1) We present a practical network model that clearly specifies the network information that is locally known to each node, includes more practical network topologies such as arbitrary network graphs, and present locally executed protocols on every single node. 2) This work considers concurrent source-destination pairs that may cause contention on quantum links. Concurrent routing is one of the most important design challenges of quantum networks because each quantum link can only be used for one source-destination pair, unlike packet switching. We believe our solution matches practical quantum network applications. Prior methods are not specifically designed for concurrent source-destination pairs and might become sub-optimal in practical situations. 3) We propose a few routing metrics that particularly fit quantum networks instead of using hop-count and physical distance. These metrics are important to select good paths in quantum networks and can also be used for future studies.

Recently, Dahlberg *et al.* [13] provide a reference model of the quantum network stack, which contains the physical layer, link layer, network layer, and transport layer. Based on that, they provide a reliable physical and link layer protocol for quantum networks on the NV hardware platform. The routing algorithms proposed in our paper fit in the 'network layer' [13] to provide the concurrent entanglement routing solutions, leveraging the services in the quantum link layer.

3 NETWORK AND SECURITY MODELS

The network model used in this study follows the facts from existing physical experiments [6, 32, 34, 35] and the corresponding studies [14, 36, 49] to reflect a practical quantum network. Compared to prior models used in existing studies of quantum network performance [14, 24, 36, 49], this model includes many practical considerations, *e.g.* the dynamics of quantum links, definition and comparison of different routing metrics, concurrent source-destination pairs, limited qubit capacity of each node, clear differentiation of the network topology and link state information, and limited link state propagation in a time slot.

3.1 Network components

There are three main components in a quantum network [52, 57], explained as follows.

1. Quantum processors are similar to the end hosts in classical networks, which are connected to a certain number of other quantum processors by quantum channels to form a quantum network and run the network applications to communicate with each other. Different from classical end hosts, each quantum processor is equipped with a certain number of memory qubits and necessary hardware to perform quantum entanglement and teleportation on the qubits. All quantum processors are connected via the classical Internet and are able to freely exchange classical information.

2. Quantum repeaters. As it is difficult to directly establish an entanglement between two remote quantum processors, quantum repeaters are used as relays. Quantum repeaters support long-distances entanglement sharing via quantum swapping. A quantum repeater may also connect to other repeaters and quantum processors via the classical Internet to exchange the control messages.

Every quantum processor also includes the complete function of a repeater. Hereafter we call both quantum processors and repeaters as *nodes*.

3. Quantum channels. A quantum channel connecting two nodes supports the transmission of qubits. The physical material of quantum channels may be polarization-maintaining optical fibers. A quantum channel is inherently lossy: the success rate of each attempt to create an entanglement of a quantum channel c is p_c, which decreases exponentially with the physical length of the channel: $p_c = e^{-\alpha L}$, where L is the physical length of the channel and α is a constant depending on the physical media [36, 43, 53, 57].[2] If an attempt is successful, the two quantum processors share an entanglement pair, and there is a *quantum link* on this channel.

Network topology. Consider a network of quantum nodes described by a multigraph $G = \langle V, E, C \rangle$. V is the set of n nodes. Each node u is a quantum processor, equipped with a limited number Q_u of qubits to build quantum links. All nodes are connected via classical networks, *i.e.*, the Internet, and every node has a certain level of classical computing and storage capacity, such as a desktop server. E is the set of edges in the graph. An edge existing between two nodes means that the two nodes share one or more quantum channels. C is the set of all quantum channels, each of which is identified by its two end nodes. The number of channels on an edge is called the *width* W of the edge.

A node can assign/bind each of its quantum memory qubits to a quantum channel [28, 29], such that no qubit is assigned to more than one channel, and no channel is assigned more than one qubits at the same end of it. Channels that are assigned qubits at its both ends are *bound channels*, other channels are *unbound channels*. There could be more than one bound channels between two nodes. And two neighbor nodes may share multiple quantum links. To create a quantum entanglement, two neighbor nodes make a number of quantum entanglement attempts at the same time on the bound channels connecting them.

3.2 Communication and security model

For each round of communication, the source and destination are two mutually trusted quantum processors, but they may not trust other nodes. The source aims to deliver secret bits to the destination without letting other repeaters know, via a path of quantum

[2] The success rate of a link is determined by the physical layer and link layer, taking into account the channel transmissivity, fidelity of transmitted entanglements, number of permitted entanglement trials in one phase, and the link layer algorithm [13, 36]. In the link layer, a channel is allowed multiple attempts to build a link, and the link is established on the first successful attempt. The p_c here is the overall success rate.

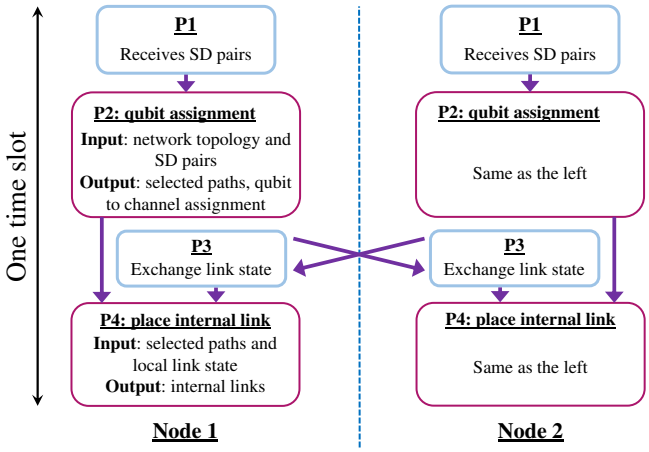
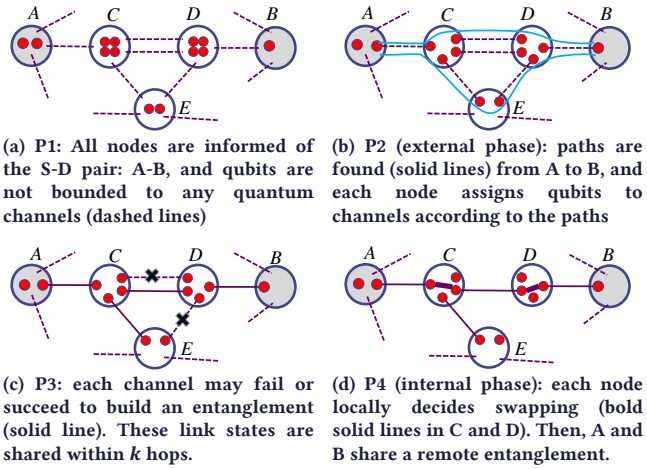


Figure 2: Phases in a time slot. Nodes 1 and 2 are two arbitrary neighbor nodes and run the same algorithm.



(a) P1: All nodes are informed of the S-D pair: A-B, and qubits are not bounded to any quantum channels (dashed lines)

(b) P2 (external phase): paths are found (solid lines) from A to B, and each node assigns qubits to channels according to the paths

(c) P3: each channel may fail or succeed to build an entanglement (solid line). These link states are shared within k hops.

(d) P4 (internal phase): each node locally decides swapping (bold solid lines in C and D). Then, A and B share a remote entanglement.

Figure 3: Phases in a time slot. Entanglement routing aims to build end-to-end paths for S-D pairs (A-B in this example).

swapping (explained in Sec. 3.3). All nodes will follow the protocol but may seek to get the secret information sent from the source to the destination, similar to the "honest-but-curious" model in classic network security. Once an intermediate node measures the information to perform passive eavesdropping, such behavior will be detected by the two endpoints according to the no-cloning theorem.

In addition, an external classical 'network information server' may be trusted to maintain the following information and send delta updates to all nodes in the network when necessary: 1) the network topology and 2) the current source-destination pairs (S-D pairs) that need to establish long-distance entanglements. The network information server may work in an honest-but-curious way and it will not know the communication content. Hence even if a network information server may be comprised – which can be detected by classical auditing methods – it will not hurt the confidentiality for previous, on-going, and future communications.

3.3 Quantum swapping via a path

Time slots. For multi-hop entanglement swapping, all nodes on the path need to establish and hold quantum entanglements with its predecessor and successor at the same time. Hence, some level of time synchronization among all nodes is necessary, which can be achieved by existing current synchronization protocols via Internet connections. Time is loosely synchronized in *time slots* [36]. Each time slot is a device-technology-dependent constant and set to an appropriate duration such that the established entanglements do not discohere within one time slot [36]. The global network topology $G = \langle V, E, C \rangle$, which is relatively stable, should be common knowledge for all nodes before any time slot.

As shown in Fig. 2 and Fig. 3, each time slot includes four phases as an extended model from [36]. In Phase One (**P1**), via the Internet, all nodes receive the information of the current S-D pairs that need to establish long-distance entanglements. As an example in Fig. 3(a), each node has a number of qubits (red dots) and multiple quantum channels (dashed lines) connecting neighbors. Two neighbors may share multiple channels. Suppose $\langle A, B \rangle$ is the only S-D pair for this time slot, and all nodes are informed of the S-D pair.

Phase Two (**P2**) is called the external phase [36]. In P2, paths are found for the S-D pairs, according to an identical routing algorithm running on all nodes that produces consistency results. Each node then binds its qubits to channels and attempts to generate quantum entanglements with neighbors on the bound channels [28, 28]. As an example in Fig. 3(b), two paths (solid curves) are calculated to connect A and B. A path is identified by the sequence of the nodes along the path $v_0, v_1, \cdots, v_h$ and the *path width* W, meaning each edge of the path has at least W parallel channels. The path $\langle (v_0, v_1, \cdots, v_h), W \rangle$ is also called a (W, h)-path, or a W-path. C, D, and E are nodes on the paths and work as repeaters. Since qubits are limited resources, some channels are not assigned qubits and thus not used in this time slot. During P2, each channel can make a number n_c of attempts [21], $n_c \geq 1$, until a link is built or timeout. After P2, some quantum links may be created as shown in Fig. 3(b). We call the information of these links as *link states*. Compared to the same term in link-state routing of classical networks [33], the quantum link states are *highly dynamic and nondeterministic*.

In Phase Three (**P3**), each node knows its own link states via classical communications with its neighbors [36] and shares its link states via the classical network, as shown in Fig. 3(c). Since entanglements will quickly decay, each node can only exchange the link states with a subset of other nodes. P3 only includes classical information exchange.

In Phase Four (**P4**), also called the internal phase [36], nodes perform entanglement swapping to establish long-distance quantum entanglement using the successful quantum links. Each node *locally* determines the swapping of successful entanglements, which can be considered as placing an *internal link* between two qubits as shown in Fig. 3(d). Each swapping succeeds at a device-dependent probability q. A and B can successfully share an entanglement qubit pair (*an ebit*) if there is an end-to-end path with both external and internal links as shown in Fig. 1(d).

After P4, the secret bit can be teleported from the source to the destination. Eavesdropping attempts at any repeater will be detected hence *the confidentiality is preserved.*

Local knowledge of link-state. P3 and P4 should be short such that the successful entanglements built in P2 do not decay. Hence, it is impractical for a node to know the global link states within such a short time as the classical network has latencies [36]. A

practical model is to allow each node to know the link states of its k-hop neighbors, $k \geq 1$ [24]. The swapping decisions in P4 thus include the k-hop link-state information as the input. It is obvious that the routing path selection could be sub-optimal without global link-state knowledge.

Exclusive qubit/channel reservation. In P2 of each time slot, to establish a single link on a channel, each end of the channel is assigned a qubit. This qubit-channel assignment is exclusive: one qubit cannot be shared by other channels, and no more qubits can be assigned to a channel. In P4, to generate distant entanglements from local ones, quantum swapping is performed on pairs of links. This quantum swapping is also exclusive and a single link cannot be used for more than one swapping. Hence, the qubits and channels are precious *routing resources* and should be carefully managed.

Physical parameters. We show the physical parameters of typical quantum networks, which provide several insights into our model and design. 1) The short entanglement persistent time determines the nodes should be synchronized to ensure all links are available simultaneously for selected paths. 2) The short entanglement persistent time T sets the limit that $t_2 + t_3 + t_4 < T$ in Fig. 3. 3) The local link state cannot be propagated to the whole network. 4) The qubit capacity is bounded in a node so that the dynamic binding of qubits and channels are necessary in P2. A most recent quantum processor can have up to 8 qubits [13]. The typical time for an entanglement to discohere is 1.46s [13]. The entanglement establishment time is ~165μs for concurrent trying [13]. The success rate of a *single* entanglement try is dependent on the length of the optical fiber and is typically ~0.01% [13]. Multiple concurrent entanglement tries are possible within a time slot to have a reasonable channel success rate in P2. A typical classical communication finishes at ~1ms, in a dedicated optical fiber network. Balancing the time for P2 and P3 are necessary to have both high channel success rate in P2 and a large enough local view of link states in P3. The entanglement readout time is typically $< 3\mu$s [13], negligible in routing algorithm design.

3.4 The entanglement routing problem

This work studies the entanglement routing problem: we are given a quantum network with an arbitrary network graph $G = \langle V, E, C \rangle$ and a number of source-destination (S-D) pairs $\langle s_1, d_1 \rangle$, $\langle s_2, d_2 \rangle$, $\cdots$, $\langle s_m, d_m \rangle$, where $s_i, d_i \in V$. The number of memory qubits of a node $u \in V$ is Q_u, and each edge $e \in E$ consists of one or more channels from C. For each bound channel c, a link is successfully built at a probability p_c in P2. In P3, each node gets the link-state information of its k-hop neighbors. Each node decides the swapping of its internal qubits in P4 *locally*, and each swapping succeeds in probability q. The objective of entanglement routing is to *maximize the number of ebits* delivered for all S-D pairs in each time slot. Each ebit must be delivered by a long-distance quantum entanglement, built by a path of successful quantum links from the source to the destination. Each S-D pair may share multiple ebits. The number of ebits for one S-D pair in one time slot is also called the *throughput* between the S-D pair. The objective can then be set to maximizing the overall throughput in the network.

This objective does not consider fairness among different S-D pairs, but we show the proposed algorithms achieve a certain level of fairness as in § 5. In addition, in § 6 we propose a simple extension to our designs to achieve better fairness among S-D pairs.

3.5 Compared with classic network routing

We summarize the differences between quantum entanglement routing and classic network routing. We show that existing routing techniques are **not sufficient** to solve the entanglement routing problem.

Term clarification. Edges, channels, and links have *different* definitions in this model, although they are used interchangeably in classic networks. Besides, the network topology and global link states may be considered as similar information in classic routing such as OSPF [33]. However, in a quantum network, while the network topology (nodes and channels) is stable and known to all nodes, the link states (whether the entanglements succeeded) are dynamic and only shared locally in P3 and P4 of each time slot.

Versus routing in wired packet-switching networks. Link-state and distance-vector are two main types of routing protocols for packet-switching networks. Main differences: **1)** Packet switching relies on either link-state broadcast or multi-round distance vector exchanges to compute the shortest paths. However, in a quantum network, link states are probabilistic and vary in different time slots. There is no time for global link-state broadcast or distance vector convergence, because entanglements on the links will quickly decay. **2)** Quantum links are highly unreliable while wired links are relatively reliable. **3)** A quantum link cannot be shared by multiple S-D pairs, which is allowed in classic packet switching. If a link is claimed by multiple S-D pairs, it can only satisfy one of them. Hence, the "shortest paths" computed by classic routing will not always be available. **4)** Classic packets can be buffered on any node for future transmission. In quantum networks, links on a path must be successful in the same time slot.

Versus routing in multi-hop wireless networks, such as mobile ad hoc networks [20] and wireless sensor networks [2]. Main differences: **1)** For an ad hoc wireless node, neither the network topology nor global link state is known. For a quantum node, although link state is local information, the network topology is known in advance via the Internet. **2)** An ad hoc wireless node can keep sending a packet until the transmission is successful or a preferred receiver moves close to it. Each quantum link can only be used once and all links on an end-to-end path must be available simultaneously. **3)** Existing wireless ad hoc routing methods such as DSR [22], AODV [39], and geographic routing [23, 27, 44] are all packet-switching protocols and do not fit quantum networks. Also, they do not take the global network topology information.

Versus circuit-switching, virtual circuit, and flow scheduling in SDN. Circuit switching, virtual circuit, and flow scheduling in software defined networks (SDNs) all need to pre-determine the end-to-end paths and reserve certain resources on the paths, such as [3, 4, 12, 47], which share similarity with entanglement routing. The main difference is that in a quantum network, though the topology (nodes and channels connecting them) is relatively stable, reserved paths for an S-D pair are not reliable because links may arbitrarily fail. Hence, more robust algorithms are required. Besides, to build a long-distance entanglement along a path, all hops of the path should have one or more success quantum link at the same time. Hence, time is divided into slots and phases for synchronization. Due to the two differences above, the algorithm of entanglement routing is very different from that of circuit-switching, in the following two novel designs: 1) multiple paths are selected in P2, based on global

and stable network topology; and 2) path recovery on P4, based on local and probabilistic link states. The recently proposed multipath routing for quantum entanglements [36] is a circuit-switching style protocol and will be compared to our work in Sec. 5.

4 ENTANGLEMENT ROUTING ALGORITHMS

The proposed entanglement routing algorithms utilize the unique properties of quantum networks that have not been explored in classic network routing. Compared to recent quantum network studies [14, 24, 36, 49], the proposed protocols follow a standard protocol-design approach and use more realistic network models: arbitrary network topologies, multiple concurrent S-D pairs to compete for links, link state exchanges, and limited qubit capacity of each node.

4.1 Main ideas

Our design is based on the following three innovative ideas to utilize the **unique features** of a quantum network:

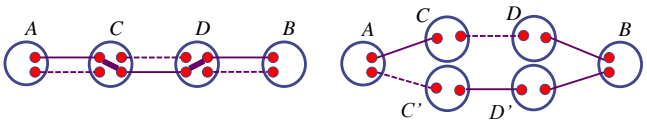
1. Path computation based on global topology and path recovery based on local link states. The quantum network graph $G = \langle V, E, C\rangle$ is relatively stable and hence can be known to every node. However, the link states are highly dynamic and probabilistic in each time slot. The frequent link state changes cannot be propagated throughout the whole network, especially when the entanglements decay quickly. Hence, nodes select and agree on the same list of paths based on global topology information in P2, and try to recover from link failures based on local link states in P4.

2. Wide paths are preferred. Recall that on a W-path, each edge has at least W parallel channels. Fig. 4(a) shows an example of a 2-path from A to B. Compared to two disjoint paths shown in Fig. 4(b), which cost the same amount of qubits and channels, the wide path is more reliable because it only fails when two links fail simultaneously at a single hop. To achieve high throughput on a path with $W > 1$, nodes should share a consensus on how to perform swapping (place internal links in Fig. 4) instead of making choice randomly. Each channel is assigned a globally unique ID. During P4, each node places an internal link between the link with the smallest ID to its predecessor and the link with the smallest ID to its successor. And it repeats this process until no internal link can be made for this path.

Formally, we may define a *routing metric*, called the expected number of ebits or expected throughput (**EXT**) E_t, to quantify an end-to-end path on the network topology. For a (W, h)-path P, suppose the success rate of a single channel on the i-th hop is p_i, where $i \in \{1, 2, \cdots, h\}$. We denote the probability of *the k-th hop on the path having exactly i successful links* as Q_k^i, and the probability of *each of the first k hops of P has $\geqslant i$ successful links* as P_k^i. Then we get the recursive formula set, for $i \in \{1, 2, \cdots, W\}$ and $k \in \{1, 2, \cdots, h\}$:

$$\begin{aligned} Q_k^i &= \binom{W}{i} p_k^i (1-p_k)^{W-i} \\ P_1^i &= Q_1^i \\ P_k^i &= P_{k-1}^i \cdot \sum_{l=i}^{W} Q_k^l + Q_k^i \cdot \sum_{l=i+1}^{W} P_{k-1}^l \end{aligned} \tag{1}$$

Further, considering the success probability q of each entanglement swapping, we get the EXT:



(a) One path with width two can deliver one qubit even under three failed links

(b) Two disjoint paths may easily fail with two unsuccessful links

Figure 4: A wide path (subfigure a) is more reliable than disjoint paths (subfigure b) using the same resource

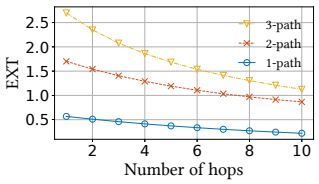


Figure 5: EXT, $p = 0.9$

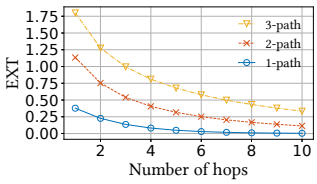


Figure 6: EXT, $p = 0.6$

$$E_t = q^h \cdot \sum_{i=1}^{W} i \cdot P_h^i \tag{2}$$

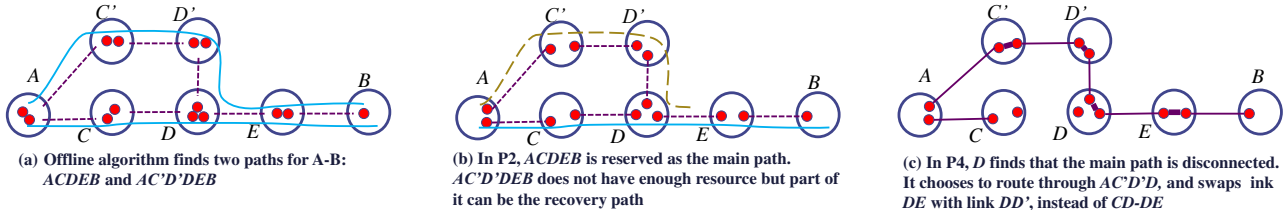
We show some numerical results. For simplicity, we set $p_1, p_2, \cdots, p_h = p$, and let p be 0.9 or 0.6. We vary the W from 1 to 3 and the h from 1 to 10, and the results of the EXT of a W-path are shown in Figures 5 and 6. It is obvious that a W-path has a significant improvement of EXT over a 1-path, for more than a factor of W.

3. Offline computation versus contention-aware online path selection. In different time slots, the S-D pairs may be different. We propose two approaches to select paths for S-D pairs in each time slot. The first approach utilizes *offline computation*, which happens before any time slot, *e.g.*, during system initialization. Multiple paths for *each* potential S-D pair are pre-computed and stored by all nodes as common knowledge. In P2 of each time slot, nodes select the pre-computed paths for current S-D pairs. The *contention-aware* online algorithm, however, does not pre-compute the paths for all S-D pairs. At each time slot, the algorithm finds contention-free paths for current S-D pairs. A set of paths are '*contention-free*' if the network can simultaneously satisfy the qubit and channel requirement for all the paths in full width.

4.2 Q-PASS: Pre-computed pAth Selection and Segment-based recovery

4.2.1 Algorithm overview. We present the algorithm Q-PASS, whose workflow follows the four-phase time slot model with an additional offline phase. The core idea of Q-PASS is to pre-compute potential 'good' paths between *all possible S-D pairs* based on the network topology G. Then in each time slot, every node uses an online algorithm to make qubit-to-channel assignments based on the pre-computed paths of *current S-D pairs* and make local swapping decisions based on local link states. The design includes both offline and online algorithms.

The offline computation happens at the system initialization and after the network topology changes. The results of an offline phase can be used by many succeeding time slots until a topology change happen. Hence, we may assume the time for an offline period is sufficiently long. The offline algorithm runs at the network information server, which is honest but curious, with replica servers for robustness. These servers connect to all quantum nodes via classical networks. The output of the offline algorithm is the "candidate paths" for all possible S-D pairs. The candidate paths of each



(a) Offline algorithm finds two paths for A-B: *ACDEB* and *AC'D'DEB*

(b) In P2, *ACDEB* is reserved as the main path. *AC'D'DEB* does not have enough resource but part of it can be the recovery path

(c) In P4, *D* finds that the main path is disconnected. It chooses to route through *AC'D'D*, and swaps ink *DE* with link *DD'*, instead of *CD-DE*

Figure 7: Example of Q-PASS. Suppose $\langle A, B\rangle$ is the only S-D pair.

S-D pair are paths connecting the S-D nodes and with the smallest values of the selected metric.

The algorithm of each time slot follows the four-phase time slot model shown in Fig. 2 and runs on each node in a distributed and concurrent manner. It should be fast and only use the k-hop link-state information. P1 and P3 only include standard processes and do not have special algorithmic designs. Q-PASS P2 takes the candidate paths from the offline algorithm and the S-D pairs as the input. It computes a number of selected paths for the S-D pairs and its local qubit-to-channel assignment. Note that the inputs are globally consistent on all nodes. Hence, the selected paths are also consistent on different nodes. The assignment will produce a number of successful links in P2. And in P3, nodes exchange the link states with their k-hop neighbors. Q-PASS P4 uses the selected paths and link state information as the input to compute the swapping decisions (i.e., internal links). After P4, possible long-distance entanglement can be built for S-D pairs. We present the algorithms in detail.

4.2.2 Offline path computation. The offline algorithm should find multiple paths for each S-D pair to provide multiple candidate paths in P2 of each time slot. We use Yen's algorithm [58] to get multiple paths for each pair. Note that the results of Yen's algorithm are not contention-free: the paths may overlap at nodes or channels, and in a single time slot, the network may not have enough qubits or channels to satisfy all the candidate paths for an S-D pair.

Yen's algorithm implicitly requires a selection of the routing metric. As shown in Equ. 1, computing the proposed routing metric EXT involves recursions, which may be prohibitively slow for multi-path computation for all possible S-D pairs. Hence, we propose three routing metrics, which are suboptimal in overall throughput but faster to compute. 1) **Sum of node distances (SumDist).** SumDist is computed as ΣL_i, where L_i is the length of any channel on the i-th hop of the path. As the success rate of a channel decreases exponentially with the physical distance L, SumDist can partially reflect the difficulty of a path. 2) **Creation Rate (CR).** CR is computed as $\Sigma 1/p_i$, where p_i is the success rate of any channel on the i-th hop of the path. Compared to SumDist, CR further considers the path width. 3) **Bottleneck capacity (BotCap).** From Figures 5 and 6, the path width W has a greater impact on the path quality. The BotCap metric is $-W$, prefers wider paths over narrower paths, and uses the CR to break ties for paths with the same width. We consider the routing metric as a design parameter, and their efficiency is compared in § 5.

For each possible S-D pair, the server running the offline algorithm will use Yen's algorithm to get $N = 25$ paths (*offline paths*) for the pair and tell *each node in the network* about the resulting paths. An example is shown in Fig. 7(a), the offline algorithm finds $ACDEB$ and $AC'D'DEB$ as two candidate paths. N will grow by 50% percent in the next offline phase if the paths happen to be not enough for a pair.

4.2.3 P2 algorithm of Q-PASS. The P2 algorithm runs on each node locally. The inputs are all the offline paths P (known before P1) and the S-D pairs (received in P1) $O = \{o_i\}$, where o_i is an S-D pair $\langle s_i, d_i\rangle$. The output is an ordered list of selected paths P', each of which connects a single S-D pair in O. According to the output path list, each node performs the local qubit-to-channel assignment and tries to establish entanglements on the bound channels with neighbor nodes to build quantum links on these paths. Since P and O are globally known for all nodes, the output P' is also consistent on different nodes, similar to the global consistency of classical link-state routing.

The algorithm consists of two steps. **Step 1)** The paths computed from the offline algorithm for all S-D pairs are retrieved and put into a priority queue, ordered by the selected routing cost metric. Then from the path with the lowest routing cost to the highest, channels and qubits taken by the path are reserved exclusively. If a path has width w by the offline algorithm, but currently available resource can only support width $0 \leqslant w' < w$, then the path is reinserted to the queue with an updated metric calculated from w'. This process ends until no paths can be fully satisfied. The paths selected in Step 1 are called *major paths*. **Step 2)** After Step 1, the queue contains all unsatisfiable paths in the ascending order of the routing metric. Each unsatisfiable path, however, may contain one or more satisfiable segments or 'partial paths'. The partial paths can be used to recover link failures for the major paths, and thus are called *recovery paths*. The qubits and channels for recovery paths are reserved in the order of its priority in the queue.

For example in Fig. 7, $ACDEB$ and $AC'D'DEB$ are two candidate paths. In Step 1, $ACDEB$ and $AC'D'DEB$ are put into a priority queue, and $ACDEB$ is more prioritized than the other and is reserved as the major path. Since D, E, and B do not have enough resources for $AC'D'DEB$, Step 1 stops. In Step 2, $AC'D'D$ is reserved as a recovery path. When the two steps finish, all nodes know the same set of selected major paths and recovery paths because they share the same set of inputs: network topology, S-D pairs, and offline paths. Hence, each node assigns its qubits to the corresponding channels and try to generate quantum links together with the neighbors. For example, node A in Fig. 7(b) will assign one qubit to the channel to C and another to the channel to C', and try to entangle with C and C' via channel AC and AC' respectively. The pseudocode for the P2 algorithm is shown in Alg. 1.

4.2.4 P4 algorithm of Q-PASS. If the entanglement attempts in P2 always succeed, each node just performs entanglement swapping to connect the links on the major paths, and the whole paths will be successful. In practice, however, link failures happen at a high probability and are not predictable. The P4 algorithm focuses on

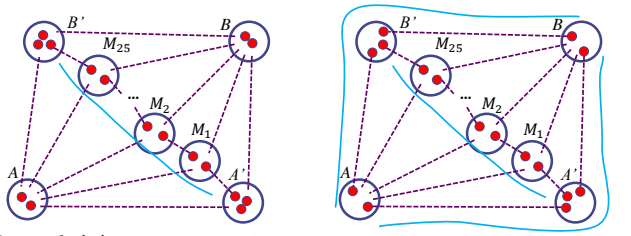
the recovery of broken major paths based on the recovery paths established in P2. The inputs of P4 algorithm are: 1) S-D pairs from P1, 2) a major path list and a recovery path list from P2, and 3) the k-hop link states of this node from P3.

Algorithm 1: Adaptive resource allocation

Input : $G = \langle V, E, C\rangle, O, P$
// O: list of S-D pairs
// P: mapping from any S-D pair to its offline paths
Output: $\langle L_C, L_P\rangle$
// L_C: list of channels to assign qubits
// L_P: ordered list of selected paths
1 $L_C \leftarrow \emptyset$
2 $L_P \leftarrow \emptyset$
3 $T_Q \leftarrow$ a table to map a node x to its qubit capacity Q_x
4 construct T_Q from current topology
5 $W \leftarrow \emptyset$
// empty table to map a path p to its width w_p
6 $q \leftarrow \emptyset$
// empty priority queue of paths, sorted by routing metric
7 **for** $o \in O$ **do**
8 **for** $p \in P[o]$ **do**
9 $T_W[p] \leftarrow Width(p, T_Q)$
10 $m \leftarrow$ routing metric of p with width W[p]
11 q.enqueue(p, m)
12 **while** q is not empty **do**
13 $p \leftarrow q$.dequeue()
14 **if** $Width(p, T_Q) < width[p]$ **then**
 // The width of p has changed
15 Update $width[p]$ and re-insert p to q
16 **continue**
17 **if** $Width(p, T_Q) = 0$ **then**
 // Even the best path is unsatisfiable
18 **break**
19 $L_P \leftarrow L_P + \langle p, width[p]\rangle$
20 **for** $\langle n1, n2\rangle \in$ edges of p **do**
21 $T_Q[n1] \leftarrow T_Q[n1] - width[p]$
22 $T_Q[n2] \leftarrow T_Q[n2] - width[p]$
23 $L_C \leftarrow L_C + width[p]$ unbound channels on $\langle n1, n2\rangle$
24 $partial \leftarrow L_P +$ (q **as** List)
25 **for** $p \in partial$ **do**
26 Update T_Q and L_C as line 21-23, only on available edges

We propose segment-based path recovery for P4. On each node, each major path given by P2 $\langle(v_0, v_1, \cdots, v_h), W\rangle$ is divided into $\lceil h/(k+1)\rceil$ segments, each with width W: $(v_0, v_1, \cdots, v_{k+1})$, $(v_{k+1}, v_{k+2}, \cdots, v_{2k+2})$, $\cdots$, $(v_{\lceil h/(k+1)-1\rceil(k+1)}, \cdots, v_{h-1}, v_h)$. The length of the segments is set to $k+1$ such that each node knows the states of all links on the segment containing it, via the k-hop link states received in P3. Then for segment $(v_{i_0}, v_{i_1}, \cdots, v_{i_{k+1}})$, each node on it finds paths connecting the two ends, v_{i_0} and $v_{i_{k+1}}$, using successful links in the k-hop neighborhood.

An example is shown in Fig. 7. Assume $k = 1$, and thus each node knows the link states of its 1-hop neighbors. The major path



(a) *A-B* and *A'-B'* are two S-D pairs. Q-PASS offline algorithm outputs 25 paths for *A-B*: $AM_iB, i \in \{1, 2, \ldots, 25\}$. However, path $A'M_1M_2\cdots M_{25}B'$ is better by metric than AM_iB and reserves all qubits on M_i. The pair *A-B* fails due to no more offline paths.

(b) Q-CAST also reserves all qubits on M_i for path $A'M_1M_2\cdots M_{25}B'$. However, in the residual graph, Q-CAST finds two more paths for *A-B*: $AB'B$ and $AA'B$. These paths are worse than AM_iB by metric, thus not recorded in the offline Q-PASS.

Figure 8: Comparison of Q-PASS and Q-CAST

$ACDEB$ is divided into two segments ACD and DEB, such that all nodes on a single segment know this segment is successful or not. If not, they will try to use a recovery path. In this example, A, C, and D know link C-D fails. Hence, the recovery path $AC'D'D$ is taken by D. The distributed recovery path selection is consistent among all nodes because recovery paths are found from local link states known to all involved nodes, and the recovery paths are ordered deterministically via the specified routing metric.

4.3 Q-CAST: Contention-free pAth Selection at runTime

The offline algorithm in Q-PASS has two fundamental disadvantages. 1) It has to compute candidate paths for $n(n-1)/2$ pairs because it does not know the runtime S-D pairs. 2) The candidate paths exhibit a low utilization rate due to severe resource contention among them. Q-CAST does not require any offline computation and always finds the paths if only paths exist in the residual graph. For example in Fig. 8(a), AB and $A'B'$ are two S-D pairs. The offline algorithm of Q-PASS finds 25 paths for AB, passing by nodes $M_1, \cdots, M_{25}$. But a single path $A', M_1, \cdots, M_{25}, B'$ takes all available qubits on M_i, and thus in the residual graph, all 25 candidate paths of AB fail, though paths $AA'B$ and $AB'B$ exist outside the offline paths, which are correctly found and reserved by Q-CAST online algorithm as shown in Fig. 8(b). Due to unpredictable combinations of S-D pairs and the resulting residual graphs, it is hard to pre-calculate and store the paths for all S-D pair combinations.

4.3.1 Algorithm overview. Q-CAST does not require any offline computation and follows the four-phase model in Fig. 2. Q-CAST P1 and P3 are standard procedures similar to those of Q-PASS. The inputs of Q-CAST P2 are the network topology and the S-D pairs. In P2, Q-CAST selects major paths for each S-D pair, *without resource contention*. Besides, contention-free recovery paths are also selected in P2. P4 takes the major paths and recovery paths from P2 and the link states from P3 to compute the swapping decisions.

4.3.2 P2 Algorithm of Q-CAST. The core task for Q-CAST P2 is to find multiple paths based on the knowledge of S-D pairs, and the paths should be contention-free on qubits and channels. Yen's algorithm [58] does not satisfy the requirements because its output paths are highly overlapped. Note, Q-PASS uses Yen's algorithm to find offline paths because the resulting overlapped path naturally provides small detours, serving as recovery paths for major paths. We propose to search multiple contention-free paths for online S-D pairs using a greedy algorithm, which runs as follows. **Step 1)** For every S-D pair, it uses the Extended Dijkstra's algorithm (described

later) to find the best path in terms of the routing metric EXT (Equation2) between this pair. **Step 2)** Among the best paths of all S-D pairs, it further selects the path with the highest EXT and reserve the resources (qubits and channels) of this path, and the network topology is updated to the residual graph by removing the reserved resources. Steps 1) and 2) are repeated with the residual graph, until no more path can be found, or the number of paths exceeds 200 – a value limiting the number of paths to avoid unnecessary computation. We call this algorithm as *Greedy EDA (G-EDA)*.

The above process aims to maximize the network throughput but does not consider fairness among S-D pairs. We will discuss how to balance throughput and fairness in § 6, and this could also be a future research topic.

The optimal routing metric. To find the optimal path under the EXT metric in a quantum network, the classical Dijkstra's algorithm fails because it only finds the shortest path when the routing metric is '*additive*'. Here, additive means the *sum* of the costs of all edges on the path is exactly the cost of the whole path. Obviously, the EXT E_t computed by Equation 2 is not additive. We propose the *Extended Dijkstra's algorithm (EDA)* to find the best path between any S-D pair for any non-additive but monotonic routing metric. The resulting path gives the maximum evaluation value among all possible paths between the S-D pair, with respect to a routing metric function e. The input of e is a path $\langle p, W\rangle$, and the output is the path quality evaluation value.

Similar to the original Dijkstra algorithm, EDA also constructs an optimal spanning tree rooted at the source node s. At the beginning, the *visited set* only includes s. The evaluation value from s to an unvisited node x is set as 0 or the evaluation value $e(s,x)$ of the edge (s,x) if s and x are neighbors. Each time, the node y with the maximum evaluation value to s is added to the visited set and the evaluation values from s to any other node x are updated if x and y are neighbors. The algorithm stops when the destination is visited. The pseudocode of EDA is shown in Alg. 2.

We skip the proof of the correctness of EDA due to space limit. Its correctness rely on a fact that the evaluation function e of a path $\langle p, W\rangle$ should *monotonically decrease* when extending p to a longer path p' by adding another node at the end of p. Since we use E_t as the evaluation function, we explain the monotonicity of E_t without a strict proof. As the W-path p grows, W may stay unchanged or decrease because the new edge may be narrower than W. In addition, adding one more hop means more hops to be transmitted. Neither of the above can increase E_t.

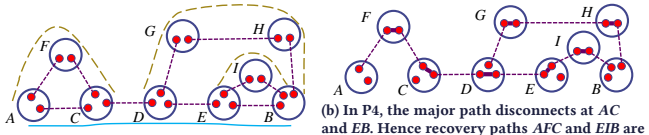
Different from the original Dijkstra algorithm, updating the path by adding one hop may cause a re-evaluation of the entire path, rather than simply adding the cost of a link. To avoid expensive recalculation for path updates, one optimization can be applied when using E_t as the evaluation function. If a (W,h)-path p grows by one hop with width $\geqslant W$, then the width of the new path p' stays unchanged to be W. Hence, in the calculation of $E_t(p') = q^{h+1} \cdot \sum_{i=1}^{W} i \cdot P_{h+1}^{i}$, the original values P_h^i when calculating $E_t(p)$ can be re-used, which significantly reduce the complexity by performing just one iteration.

Bound the path length. We set the upper-bound threshold h_m for the hopcount of major paths to ensure bounded searching in EDA. During EDA, any path with hopcount larger than h_m is ignored because it is unlikely to be a good path. The value of h_m can be determined at system initialization. For a new network G, 100 pairs of nodes are randomly selected. Then, multipath routing is performed via G-EDA for each pair with $h_m = \inf$. Then h_m is set to equal the largest hopcount of the resulting paths whose $E_t \geqslant 1$.

Algorithm 2: The Extended Dijkstra's algorithm

Input: $G = \langle V, E, C\rangle$, e, $\langle src, dst\rangle$
Output: The best path $\langle p, W\rangle$

```
// Initialize empty states
E ← an array of n elements, all set to −∞
prev ← an array of n elements, all set to null
visited ← an array of n elements, all set to false
width ← an array of n elements, all set to 0
q ← fibonacci-heap, highest E[·] first
// Initialize states of src
E[src] ← +∞
width[src] ← +∞
q.enqueue(src)
// Track the best path until dst
while q is not empty do
    // Get the current best end node
    u ← q.dequeue()
    if visited[u] then continue
    else visited[u] ← true
    if u = dst then
        ⟨p, W⟩ ← Construct path via prev and width
        return ⟨p, W⟩
    // Expand one hop based on u
    for v ∈ neighbors of u do
        if visited[v] then continue
        ⟨p, W⟩ ← Construct path via prev and width
        E′ ← e(p, W)
        if E[v] < E′ then
            E[v] ← E
            prev[v] ← u
            width[v] ← W
            q.reorder(v)
```



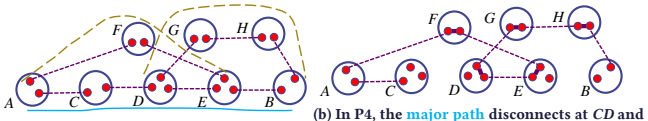
(a) In P2, The major path *ACDEB* and three recovery paths (in the order of being founded): *AFC*, *EIB*, and *DGHB*

(b) In P4, the major path disconnects at *AC* and *EB*. Hence recovery paths *AFC* and *EIB* are taken. *G* and *H* still swap along *DGHB*, but *D* swaps *CD* and *DE*, in favor of a smaller detour from the major path for higher ETX

Figure 9: Example of Q-CAST recovery algorithm

Recovery paths. After finding the major paths via G-EDA, the remaining qubits and channels can be utilized to construct *recovery paths*, each of which ends at two nodes (denote as *switch nodes*) on a single major path. The switch nodes should be no more than k hops away on a major path, where k is the link state range, because in P4, the two nodes should ensure consistent swapping decisions.

The recovery paths are found as following. For every node x on a major path, we use EDA to find $\leqslant R$ recovery paths between x and y in the residual graph, where y is the 1-hop ahead node on the



(a) In P2. Major path *ACDEB* and two recovery paths (in the order of being founded): *AFE* and *DGHB*

(b) In P4, the major path disconnects at *CD* and *EB*. Recovery paths *AFE* and *DGHB* are taken, because $\{AC, CD, DE, EB\} \oplus \{AF, FE, AC, CD, DE\} \oplus \{DG, GH, HB, DE, EB\}$ connects *A* and *B*.

Figure 10: Example of Q-CAST recovery via exclusive-or

major path, and R is a small constant parameter. When all nodes are processed, the algorithm will iterates further for the recovery paths that covers l hops on the major path, for $l = 2, 3, \cdots, k$. In Fig. 9(a), the major path is *ACDEB* and three recovery paths are found.

Every node will assign its qubits based on the reserved major paths and recovery paths, without qubit/channel contention.

4.3.3 P4 Algorithm of Q-CAST. In P4, each node knows the major paths, the recovery paths, and the k-hop link states. It then makes the swapping decisions locally. The challenges for Q-PASS P4 still present for Q-CAST P4: probabilistic link failures and non-interactive communication between nodes.

We propose an exclusive-or (*xor*, $\bigoplus$) based algorithm to recover from potential link failures. We define the xor operator of two set of edges E_1, E_2: $E_1 \bigoplus E_2 = E_1 \cup E_2 \setminus (E_1 \cap E_2)$. As both ends (switch nodes) of a recovery path p_r are on a single major path, a segment of the major path p_m is covered by p_r, where p_m and p_r form a loop in the graph, called a *recovery loop*. Then, the link recovery algorithm works as following. The major path list is traversed from beginning to end. Each visited major path p is treated as W separated 1-paths, where W is the width of p. For each separated 1-path, the set E collects the successful edges of it. K recovery paths are found, and the edges of the recovery loops of the recovery paths are collected as $E_{p_1}, E_{p_2}, \cdots, E_{p_K}$, such that the S-D pair is connected in the graph $\langle V, E \bigoplus E_{p_1} \bigoplus E_{p_2} \bigoplus \cdots \bigoplus E_{p_K} \rangle$, where V is the set of nodes on the major path and the K recovery paths. To break the tie, shorter recovery paths are preferred because shorter paths are more likely to succeed after swapping. The Q-CAST recovery algorithm is different from that of Q-PASS because each recovery path in Q-CAST is dedicated to a single major path, and they are contention-free.

As an example, in Fig. 9, the major path disconnects at *AC* and *EB*. Nodes *F*, *G*, and *H* swap along the recovery path no matter the recovery path is used or not. As switch nodes, *A* and *C* recover the broken edge *AC* by the recovery path *AFC*. Both *D* and *E* know the two recovery paths covering *EB*, namely *DGHB* and *EIB*. The shorter one *EIB* is used. *D* still swaps qubits on the major path and *E* switches to the recovery path.

As another example, in Fig. 10, the major path disconnects at *CD* and *EB*. Recovery paths *AFE* and *DGHB* are taken, because the xor of the major path *ACDEB* and two recovery loops *AFEDCA* and *DGHBED* connects A and B. Note the edge *DE* appears 3 times in the xor and is used "reversely" on the final path.

4.4 Time and space costs

We denote the number of S-D pairs as m, and the maximum width of paths as W_m, which is determined by node capacities and edge widths. We denote the maximum number of paths as K_m in EDA. The number of nodes is n. We summarize the results here and some details can be found in the Appendix.

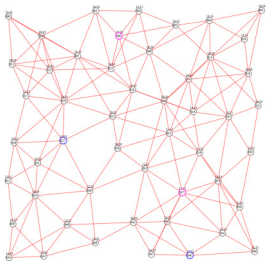


Figure 11: Visualized network with qubits and channels

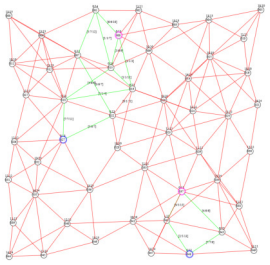


Figure 12: Visualized path selection and resource

Cost of routing metric evaluation. The time cost to calculate E_t for a (W, h)-path according to Equation 1 is $O(hW)$, and the space cost is $O(W)$.

Cost of Q-PASS P2. The space cost is $O(mK_m h_m + n)$ and the time cost is $O(mK_m(h_m + \log(mK_m)))$.

Cost of EDA. The space cost is $O(n)$. The time cost for EDA is $O(n \log n + |E|(h_m W_m))$.

5 PERFORMANCE EVALUATION

5.1 Simulator Implementation

We implement the proposed network models and algorithms on a custom-built time-based simulator, with additional supports for the topology generation, statistics, and network visualization. We do not use packet-based simulation because quantum networks do not use packet switching. As shown in Figures 11 and 12, the visualization tool shows the network topology, current qubit/channel occupation, and existing quantum links at simulation runtime, for protocol analysis and demonstration. The source code repository of the simulator can be found on this link [1].

We do not assume any specific topology and randomly generate quantum networks for simulations. We set the area A holding quantum networks is a 100K units by 100K units square, each unit may be considered as 1km. The network generation algorithm requires three input parameters: the number of nodes n, the average number of neighbors E_d, and the average success rate of all channels E_p. Nodes are randomly placed and the distance of any two nodes is at least $\leqslant 50/\sqrt{n}$ units. The edges are generated by the Waxman model [56] that was used for Internet topologies [30].

After the topology generation, a binary search on the model parameter α is further carried out to make the average channel success rate to be $E_p \pm 0.01$. The number of qubits Q for each node is independently uniformly picked from 10 to 14. The edge width W is independently uniformly generated from 3 to 7, for each edge. We pick the range for Q and W based on our conjecture of a well-functioning quantum network. Our designs should work on wider ranges, which we cannot cover due to enormous possibilities.

5.2 Methodology

We evaluate the throughput, scalability, and fairness of the proposed entanglement routing algorithms. To gain insight into the performance metrics and to provide a reference for future research, we show more simulation statistics: the resource efficiency towards high throughput, the contribution of recovery paths for both algorithms. Each data shown in the section is the average from 10 different network topologies.

We let the number of nodes n vary in set $\{50, 100, 200, 400, 800\}$, average channel success rate E_p vary in $\{0.6, 0.3, 0.1\}$, internal link

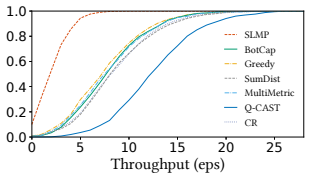


Figure 13: CDF of throughput under the reference setting

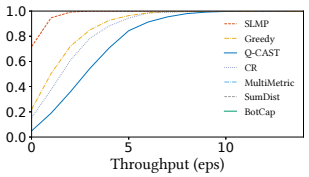


Figure 14: CDF of throughput, $E_p = 0.3$

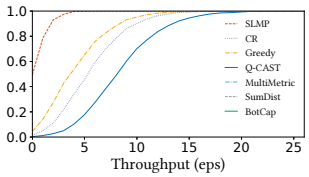


Figure 15: CDF of throughput, $n = 400$

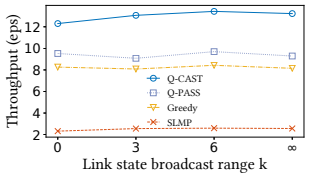


Figure 16: Throughput vs. LS sharing ranges

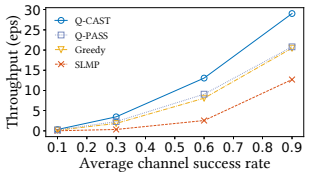


Figure 17: Throughput vs. channel success rates

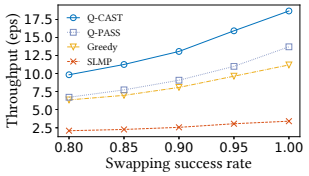


Figure 18: Throughput vs. swapping success rates

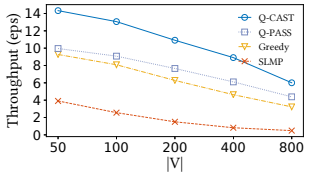


Figure 19: Throughput vs. network sizes

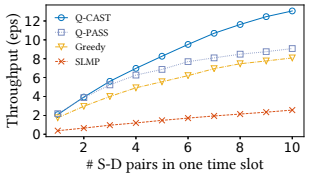


Figure 20: Throughput vs. # of S-D pairs

success rate q vary in $\{0.8, 0.9, 1.0\}$, link state range k vary in $\{0, 3, 6, \infty\}$, average degree E_d vary in $\{3, 4, 6\}$, and the number of S-D pairs m vary from 1 to 10. To control variable, we show the results under the **reference setting** $n = 100, E_p = 0.6, q = 0.9, k = 3, E_d = 6, m = 10$, unless explicitly changed to observe the data trend. For each setting of (n, E_p, q, k, E_d, m), 10 random networks are generated, and we simulate 1000 independent time slots on each of the networks.

We compare Q-PASS and Q-CAST with two existing routing algorithms that have been used in quantum network studies: single-link multipath routing (*SLMP*) [36] (a circuit-switching style protocol) and greedy routing [24] (a distributed protocol).

5.3 Evaluation results

Throughput. Figures 13 to 15 show the CDF of throughputs for Q-PASS, Q-CAST, Greedy, and SLMP, under the reference setting. The throughput results are calculated in terms of ebits per time slot (**eps**). The BotCap, CR, and SumDist are the routing metrics for the Q-PASS, and they are shown separately for better comparison. Despite the multipath routing, SLMP shows the lowest throughput because of the unreliability of a single channel/link. It fails to deliver any ebits in >10 percent of the time slots, and for 90 percent of the time slots, the total throughput between 10 S-D pairs are less than 5. The Greedy enjoys a high throughput, and for more than 90 percent of the time, it delivers more than 15 ebits for 10 random S-D pairs. For Q-PASS, all the three metrics of it exhibit similar throughput, and the CR metric gives the highest throughput among all metrics, which delivers about 2 eps more than the Greedy. Q-CAST shows great advantages over all other algorithms and outperforms the CR about 5 eps. Q-CAST is also the most reliable because it seldom delivers less than 5 eps. Since CR is slightly better than other metrics, we use CR to represent Q-PASS in the following results.

Vary link state range. In P3, each node shares its link states with its k-hop neighbors, and hence, k influences the path recovery performance. Fig. 16 shows the average throughput on different k. The Greedy algorithm does not rely on k and is shown for reference. k contributes little to the overall performance because most path failures are just one hop v_i-v_{i+1}, which can be recovered by v_i and v_{i+1} with their own link states. $k = 3$ is sufficient for Q-CAST, and larger k slightly degrades the throughput because longer and more unreliable recovery paths may be selected. This would occupy the routing resource which could have been allocated to other shorter and more reliable recovery paths.

Vary link success rates. Figures 17 and 18 show the average throughput of Q-PASS, Q-CAST, Greedy, and SLMP on different quantum device abilities by varying the average channel success rate and swapping (internal link) success rate. When the channel success rate p or the swapping success rate q is small, the overall throughput will be degraded. A robust routing algorithm should still perform well on low ability networks. From the figures, the swapping success rate also has big impact on the average throughput, because the link failure in the P2 can be mitigated by the recovery algorithms in P4, but there is no circumvention for swapping errors. And the Q-CAST performs the best among the four algorithms.

Scalability. We evaluate the scalability of routing algorithms on two dimensions: the size of the network n and the number of concurrent S-D pairs m. A larger network means the average distance of S-D pairs is longer; and more concurrent S-D pairs in one time slot introduce higher level of resource contention. Figures 19 and 20 show the average throughput on the two dimensions. All algorithms exhibit a logarithmic throughput decrease with the number of nodes in the network. Q-CAST outperforms others on all network sizes, and the throughput of Q-CAST is as high as 7.5eps when the network contains 800 nodes. The reason of lower throughput in larger networks is because the average path length is longer for the S-D pairs. Longer paths are more likely to fail in quantum networks. Besides, the throughput of all algorithms grow sub-linearly with the number of S-D pairs, due to resource contentions. Q-CAST outperforms others on most settings, and the advantage of Q-CAST over other algorithms grows rapidly with the number of S-D pairs. It is because Q-CAST actively resolves the resource contentions for the S-D pairs.

Fairness. Though we aim to maximize the throughput in the current designs, the fairness among the S-D pairs is evaluated. Fig. 21 shows the average number of successful S-D pairs under different numbers of concurrent requests. For a time slot, an S-D pair is successful (*epair*) when they establish at least one ebits after P4. Q-CAST outperforms others and all algorithms grow sub-linearly. Fig. 22 shows the CDF of the number of paths allocated

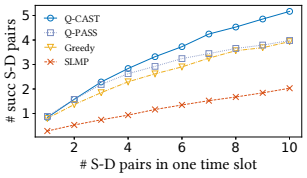


Figure 21: # of successful concurrent S-D pairs

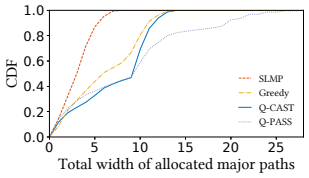


Figure 22: CDF of # of major paths

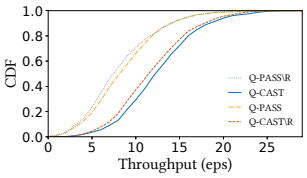


Figure 23: Contribution of recovery paths

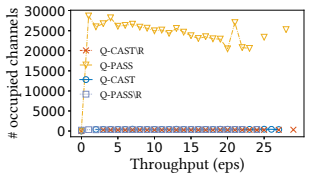


Figure 24: Overhead of recovery paths

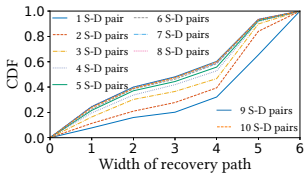


Figure 25: CDF of the width of recovery paths

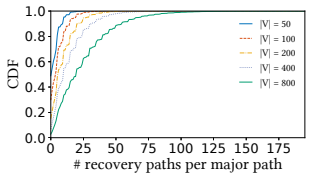


Figure 26: CDF of # of recovery paths on a single major path

to every S-D pair. A W-path is counted as W separate paths. As a baseline requirement, any S-D pair should be allocated at least one major path, which is fulfilled by all algorithms. The SLMP is the fairest. The Q-CAST has a turning point on the CDF figure, which means 40 percent of S-D pairs are allocated less than 9 paths, and the other pairs are allocated 10 to 14 paths, which is very fair. The Q-PASS is the most biased algorithm.

Recovery paths. We evaluate the contribution of recovery paths to the overall throughput for both Q-PASS and Q-CAST, by comparing their throughput with that of their recovery path-free versions Q-PAST\R and Q-CAST\R. The results are shown in Fig. 23. The recovery paths contribute about 0.5eps to Q-PASS and 1eps to Q-CAST. We further show the average number of occupied channels in one time slot for Q-CAST, Q-CAST\R, and Q-PASS in Fig. 24, where the x-axis shows the throughput of each case. Q-PASS is not shown in this figure because it takes way more channels in the recovery paths and the results are not in this range of y-axis. Q-PASS\R takes times fewer channels compared with Q-PASS, and Q-CAST\R saves 25% channels from the 400 channels taken by Q-CAST.

As the recovery paths are contention-free for Q-CAST, more interesting statistics are collected on Q-CAST recovery paths. The CDF of the width of recovery paths is shown in Fig. 25. The recovery paths can be wider when the number of S-D pairs is small, because of the low resource contention between S-D pairs. For most cases, the widths of recovery paths for a single S-D pair are larger than those of the 10 concurrent S-D pairs by 2. Besides, the CDF of the total number of recovery paths of a single major path is shown in Fig. 26. In larger networks, the major paths are longer, and more recovery paths can be found.

Summary of evaluations. Q-CAST exhibits much higher throughput, robustness, and scalability than other routing algorithms. Q-PASS also shows good throughput and the metric CR provides the highest throughput for Q-PASS. If the minimum resource utilization is a concern for some quantum networks, recovery paths for both algorithms can be disabled for better efficiency. Q-CAST\R is a good balance between throughput and resource efficiency.

6 DISCUSSION

Better fairness. The algorithms proposed in this paper aim to maximize throughput, and each time slot is considered totally separately. A simple extension, however, is available to both Q-PASS and Q-CAST to provide better fairness while maintaining high throughput. For any S-D pair that has failed to share an ebit in a slot T, the pair and the failing streak $\langle (s,d), 1\rangle$ are broadcast to all nodes in P1 in the slot $T+1$. The routing metric of all paths connecting this S-D pair is multiplied with a factor such as 1.1, which means their paths are slightly over-evaluated, and thus are more likely to be selected. If the pair still fails, the failing streak increases to a higher factor such as 2, and the related routing metric is multiplied with 1.1^2 in $T+2$. Eventually, this pair will succeed.

Prioritized routing. Both Q-PASS and Q-CAST are extendable to support simple prioritized routing. Suppose S-D pairs are in different priority classes, identified by the number $1, 2, \cdots, 10$, and the priority is 'hard' – a single S-D pair in priority class c is far more valuable than all S-D pairs in priority class $c-1$ and lower. In P2 of Q-PASS and Q-CAST, the offline paths (only Q-PASS) and online paths (both algorithms) of the highest priority S-D pair are selected until no more path is available. More paths are then selected in the residual graph for S-D pairs in lower priority classes. The P4 of Q-CAST is not modified because the selected paths have no contention. In P4 of Q-PASS, the paths of the highest priority S-D pair are recovered first.

7 CONCLUSION AND FUTURE WORK

This work presents a new entanglement routing model of quantum networks that reflects the differences compared to classical networks and new entanglement routing algorithms that utilize the unique properties of quantum networks. The proposed algorithm Q-CAST increases network throughput by a big margin compared to other methods. We expect more future research will be conducted on the entanglement routing problem and could contribute to the eventual success of quantum networks.

There could be a large amount of possible future work on the topic of routing in quantum networks. Among them, we identify three possible future research topics that are directly related to this work: 1) properly find offline paths such that generated paths are resilient to the runtime resource contention, 2) find an efficient algorithm to correctly select the recovery loops for Q-CAST P4, and 3) make use of the entangled but not used pairs in the previous time slot, rather than resetting the whole network at the beginning of every time slot.

This work does not raise any ethical issues.

8 ACKNOWLEDGEMENT

This work was partially supported by National Science Foundation Grants 1717948, 1750704, and 1932447. We thank Liang Jiang, Rui Li, Peter Young, our shepherd Hongqiang Harry Liu, and the anonymous reviewers for their suggestions and comments.

REFERENCES

[1] 2019. Source Code of the Quantum Routing Simulations. https://github.com/QianLabUCSC/QuantumRouting.

[2] Ian F Akyildiz, Weilian Su, Yogesh Sankarasubramaniam, and Erdal Cayirci. 2002. Wireless sensor networks: a survey. *Computer Networks* (2002).

[3] Mohammad Al-Fares, Sivasankar Radhakrishnan, Barath Raghavan, Nelson Huang, and Amin Vahdat. 2010. Hedera: dynamic flow scheduling for data center networks. In *Proceedings of USENIX NSDI*.

[4] James Aspnes, Yossi Azar, Amos Fiat, Serge Plotkin, and Orli Waarts. 1993. On-line load balancing with applications to machine scheduling and virtual circuit routing. In *Proceedings of the twenty-fifth annual ACM symposium on Theory of computing*.

[5] Charles H Bennett and Gilles Brassard. 1984. Quantum Cryptography: Public Key Distribution and Coin Tossing. In *Proceedings of the International Conference on Computers, Systems and Signal Processing*.

[6] Hannes Bernien, Bas Hensen, Wolfgang Pfaff, Gerwin Koolstra, Machiel S Blok, Lucio Robledo, TH Taminiau, Matthew Markham, Daniel J Twitchen, Lilian Childress, and R. Hanson. 2013. Heralded entanglement between solid-state qubits separated by three metres. *Nature* (2013).

[7] Anne Broadbent, Joseph Fitzsimons, and Elham Kashefi. 2009. Universal blind quantum computation. In *2009 50th Annual IEEE Symposium on Foundations of Computer Science*.

[8] Angela Sara Cacciapuoti, Marcello Caleffi, Francesco Tafuri, Francesco Saverio Cataliotti, Stefano Gherardini, and Giuseppe Bianchi. 2019. Quantum internet: networking challenges in distributed quantum computing. *IEEE Network* (2019).

[9] M. Caleffi. 2017. Optimal Routing for Quantum Networks. *IEEE Access* (2017).

[10] Marcello Caleffi, Angela Sara Cacciapuoti, and Giuseppe Bianchi. 2018. Quantum Internet: From Communication to Distributed Computing! *(Proceedings of NANOCOM '18)*.

[11] Lilian Childress and Ronald Hanson. 2013. Diamond NV centers for quantum computing and quantum networks. *MRS bulletin* 38, 2 (2013), 134–138.

[12] Richard Cole, Bruce M Maggs, Friedhelm Meyer auf der Heide, Michael Mitzenmacher, Andréa W Richa, Klaus Schröder, Ramesh K Sitaraman, and Berthold Vöcking. 1998. Randomized protocols for low-congestion circuit routing in multistage interconnection networks. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*.

[13] Axel Dahlberg, Matthew Skrzypczyk, Tim Coopmans, Leon Wubben, Filip Rozpedek, Matteo Pompili, Arian Stolk, Przemyslaw Pawelczak, Robert Knegjens, Julio de Oliveira Filho, Ronald Hanson, and Stephanie Wehner. 2019. A link layer protocol for quantum networks. In *Proceedings of ACM SIGCOMM*.

[14] S. Das, S. Khatri, and J. P. Dowling. 2018. Robust quantum network architectures and topologies for entanglement distribution. *Phys. Rev. A* (2018).

[15] Vasil S Denchev and Gopal Pandurangan. 2008. Distributed quantum computing: A new frontier in distributed systems or science fiction? *ACM SIGACT News* (2008).

[16] Whitfield Diffie and Martin Hellman. 1976. New directions in cryptography. *IEEE transactions on Information Theory* (1976).

[17] Artur K Ekert. 1991. Quantum cryptography based on Bell's theorem. *Physical review letters* (1991).

[18] Chip Elliot. 2002. Building the quantum network. *New Journal of Physics* (2002).

[19] Chip Elliott, David Pearson, and Gregory Troxel. 2003. Quantum Cryptography in Practice. In *Proceedings of ACM SIGCOMM*. Association for Computing Machinery.

[20] Xiaoyan Hong, Kaixin Xu, and Mario Gerla. 2002. Scalable routing protocols for mobile ad hoc networks. *IEEE network* (2002).

[21] Peter C. Humphreys, Norbert Kalb, Jaco P. J. Morits, Raymond N. Schouten, Raymond F. L. Vermeulen, Daniel J. Twitchen, Matthew Markham, and Ronald Hanson. 2018. Deterministic delivery of remote entanglement on a quantum network. *Nature* (2018).

[22] David B. Johnson and David A. Maltz. 1996. Dynamic Source Routing in Ad Hoc Wireless Networks. In *Mobile Computing*. Kluwer Academic Publishers, 153–181.

[23] Brad Karp and Hsiang-Tsung Kung. 2000. GPSR: Greedy perimeter stateless routing for wireless networks. In *Proceedings of the 6th annual international conference on Mobile computing and networking*.

[24] Axel Dahlberg Kaushik Chakraborty, Filip Rozpedek and Stephanie Wehner. 2019. Distributed Routing in a Quantum Internet. *arXiv:1907.11630* (2019).

[25] H Jeff Kimble. 2008. The quantum internet. *Nature* (2008).

[26] Peter Komar, Eric M Kessler, Michael Bishof, Liang Jiang, Anders S Sørensen, Jun Ye, and Mikhail D Lukin. 2014. A quantum network of clocks. *Nature Physics* (2014).

[27] Simon S. Lam and Chen Qian. 2011. Geographic Routing in d-dimensional Spaces with Guaranteed Delivery and Low Stretch. In *Proceedings of ACM SIGMETRICS*.

[28] Yuan Lee, Eric Bersin, Axel Dahlberg, Stephanie Wehner, and Dirk Englund. 2020. A Quantum Router Architecture for High-Fidelity Entanglement Flows in Multi-User Quantum Networks. arXiv:2005.01852 [quant-ph]

[29] David Luong, Liang Jiang, Jungsang Kim, and Norbert Lütkenhaus. 2016. Overcoming lossy channel bounds using a single quantum repeater node. *Applied Physics B* 122, 4 (Apr 2016). https://doi.org/10.1007/s00340-016-6373-4

[30] Alberto Medina, Anukool Lakhina, Ibrahim Matta, , and John Byers. 2001. BRITE: An Approach to Universal Topology Generation. In *International Workshop on Modeling, Analysis and Simulation of Computer and Telecommunications Systems*.

[31] Rodney Van Meter, Takahiko Satoh, Thaddeus D. Ladd, William J. Munro, and Kae Nemoto. 2013. Path Selection for Quantum Repeater Networks. *Networking Science* (2013).

[32] David L Moehring, Peter Maunz, Steve Olmschenk, Kelly C Younge, Dzmitry N Matsukevich, L-M Duan, and Christopher Monroe. 2007. Entanglement of single-atom quantum bits at a distance. *Nature* (2007).

[33] J. Moy. 1998. OSPF Version 2. RFC 2328.

[34] S Olmschenk, DN Matsukevich, P Maunz, D Hayes, L-M Duan, and C Monroe. 2009. Quantum teleportation between distant matter qubits. *Science* (2009).

[35] Jian-Wei Pan, Dik Bouwmeester, Harald Weinfurter, and Anton Zeilinger. 1998. Experimental entanglement swapping: entangling photons that never interacted. *Physical Review Letters* (1998).

[36] Mihir Pant, Hari Krovi, Don Towsley, Leandros Tassiulas, Liang Jiang, Prithwish Basu, Dirk Englund, and Saikat Guha. 2019. Routing Entanglement in the Quantum Internet. *npj Quantum Information* (2019).

[37] James L Park. 1970. The concept of transition in quantum mechanics. *Foundations of Physics* (1970).

[38] Momtchil Peev, Christoph Pacher, Romain Alléaume, Claudio Barreiro, Jan Bouda, W Boxleitner, Thierry Debuisschert, Eleni Diamanti, M Dianati, JF Dynes, S Fasel, S Fossier, M Fürst, J-D Gautier, O Gay, N Gisin, P Grangier, A Happe, Y Hasani, M Hentschel, H Hübel, G Humer, T Länger, M Legré, R Lieger, J Lodewyck, T Lorünser, N Lütkenhaus, A Marhold, T Matyus, O Maurhart, L Monat, S Nauerth, J-B Page, A Poppe, E Querasser, G Ribordy, S Robyr, L Salvail, A W Sharpe, A J Shields, D Stucki, M Suda, C Tamas, T Themel, R T Thew, Y Thoma, A Treiber, P Trinkler, R Tualle-Brouri, F Vannel, N Walenta, H Weier, H Weinfurter, I Wimberger, Z L Yuan, H Zbinden, and A Zeilinger. 2009. The SECOQC quantum key distribution network in Vienna. *New Journal of Physics* (2009).

[39] Charles E. Perkins and Elizabeth M. Royer. 1999. Ad-hoc On-demand Distance Vector Routing. In *IEEE WORKSHOP ON MOBILE COMPUTING SYSTEMS AND APPLICATIONS*. 90–100.

[40] Stefano Pirandola. 2019. End-to-end capacities of a quantum communication network. *Commun. Phys* 2 (2019), 51.

[41] Stefano Pirandola, Ulrik L Andersen, Leonardo Banchi, Mario Berta, Darius Bunandar, Roger Colbeck, Dirk Englund, Tobias Gehring, Cosmo Lupo, Carlo Ottaviani, J. Pereira, M. Razavi, J. S. Shaari, M. Tomamichel, V. C. Usenko, G. Vallone, P. Villoresi, and P. Wallden. 2019. Advances in Quantum Cryptography. arXiv:1906.01645 [quant-ph]

[42] Stefano Pirandola, Raul García-Patrón, Samuel L Braunstein, and Seth Lloyd. 2009. Direct and reverse secret-key capacities of a quantum channel. *Physical review letters* 102, 5 (2009), 050503.

[43] Stefano Pirandola, Riccardo Laurenza, Carlo Ottaviani, and Leonardo Banchi. 2017. Fundamental limits of repeaterless quantum communications. *Nature communications* (2017).

[44] Chen Qian and Simon Lam. 2011. Greedy distance vector routing. In *Proceedings of IEEE ICDCS*.

[45] Mark Riebe, H Häffner, CF Roos, W Hänsel, J Benhelm, GPT Lancaster, TW Körber, C Becher, F Schmidt-Kaler, DFV James, et al. 2004. Deterministic quantum teleportation with atoms. *Nature* 429, 6993 (2004).

[46] Ronald L Rivest, Adi Shamir, and Leonard Adleman. 1978. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM* (1978).

[47] Reza Rooholamini, Vladimir Cherkassky, and Mark Garver. 1997. Finding the Right ATM Switch for the Market. *IEEE Computer* (1997).

[48] M. Sasaki, M. Fujiwara, H. Ishizuka, W. Klaus, K. Wakui, M. Takeoka, S. Miki, T. Yamashita, Z. Wang, A. Tanaka, K. Yoshino, Y. Nambu, S. Takahashi, A. Tajima, A. Tomita, T. Domeki, T. Hasegawa, Y. Sakai, H. Kobayashi, T. Asai, K. Shimizu, T. Tokura, T. Tsurumaru, M. Matsui, T. Honjo, K. Tamaki, H. Takesue, Y. Tokura, J. F. Dynes, A. R. Dixon, A. W. Sharpe, Z. L. Yuan, A. J. Shields, S. Uchikoga, M. Legré, S. Robyr, P. Trinkler, L. Monat, J.-B. Page, G. Ribordy, A. Poppe, A. Allacher, O. Maurhart, T. Länger, M. Peev, and A. Zeilinger. 2012. Field test of quantum key distribution in the Tokyo QKD Network. *Optics Express* (2012).

[49] Eddie Schoute, Laura Mancinska, Tanvirul Islam, Iordanis Kerenidis, and Stephanie Wehner. 2016. Shortcuts to quantum network routing. *arXiv preprint arXiv:1610.05238* (2016).

[50] Peter W Shor. 1994. Algorithms for quantum computation: discrete logarithms and factoring. In *Proceedings 35th annual symposium on foundations of computer science*.

[51] Ankit Singla, Chi-Yao Hong, Lucian Popa, and P. Brighten Godfrey. 2012. Jellyfish: Networking Data Centers Randomly. In *Proceedings of USENIX NSDI*.

[52] Pirandola Stefano and Braunstein Samuel Leon. 2016. Unite to build a quantum internet. *Nature* 532 (2016), 169–171.

[53] Masahiro Takeoka, Saikat Guha, and Mark M Wilde. 2014. Fundamental rate-loss tradeoff for optical quantum key distribution. *Nature communications* (2014).

[54] Rodney Van Meter and Joe Touch. 2013. Designing quantum repeater networks. *IEEE Communications Magazine* (2013).

[55] Gayane Vardoyan, Saikat Guha, Philippe Nain, and Don Towsley. 2019. On the Stochastic Analysis of a Quantum Entanglement Switch. In *ACM SIGMETRICS Performance Evaluation Review*.
[56] Bernard M Waxman. 1988. Routing of multipoint connections. *IEEE journal on selected areas in communications* (1988).
[57] Stephanie Wehner, David Elkouss, and Ronald Hanson. 2018. Quantum internet: A vision for the road ahead. *Science* (2018).
[58] Jin Y Yen. 1971. Finding the k-shortest loopless paths in a network. *Manegement Science* (1971).
[59] Juan Yin, Yuan Cao, Yu-Huai Li, Sheng-Kai Liao, Zhang, Ji-Gang Ren, Wen-Qi Cai, Wei-Yue Liu, Bo Li, Hui Dai, Guang-Bing Li, Qi-Ming Lu, Yun-Hong Gong, Yu Xu, Shuang-Lin Li, Feng-Zhi Li, Ya-Yun Yin, Zi-Qing Jiang, Ming Li, Jian-Jun Jia, Ge Ren, Dong He, Yi-Lin Zhou, Xiao-Xiang Zhang, Na Wang, Xiang Chang, Zhen-Cai Zhu, Nai-Le Liu, Yu-Ao Chen, Chao-Yang Lu, Rong Shu, Cheng-Zhi Peng, Jian-Yu Wang, and Jian-Wei Pan. 2017. Satellite-based entanglement distribution over 1200 kilometers. *Science* (2017).
[60] Ye Yu and Chen Qian. 2014. Space Shuffle: A Scalable, Flexible, and High-Bandwidth Data Center Network. In *Proceedings of IEEE ICNP*.

A APPENDIX

Appendices are supporting material that has not been peer-reviewed.

A.1 Finding the optimal path selection for Q-CAST

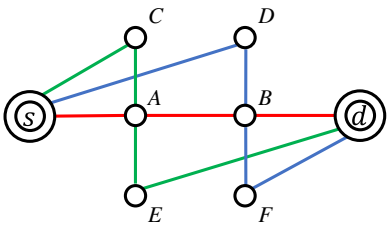


Figure 27: Counterexample for two possible algorithms

We summarize the hardness of the contention-free path selection problem without classifying it into a certain complexity class, and show its hardness in three examples. On one hand, because of the resource constraints (qubits/channels), path selection depends highly on the link states and hence the search space is much more than the classical algorithms which only depends on the weighted graph while edges and nodes have unlimited capacity; on the other hand, E_t is non-linear, which invalids many existing proofs based on the linear additivity of the routing metric and thus degrades the efficiency of classical algorithms.

Example 1. Despite its good performance (shown in § 5), we prove G-EDA is not the optimal. An example graph is shown in Fig. 27 [3]. Suppose all the edges have width 3, all channels have creation rate $p = 0.99$, the swapping success rate $q = 1$, s and d have qubit capacity 6, and all other nodes have capacity 3. Then the optimal contention-free paths are the blue path plus the green path. But the G-EDA will output only the red path. The reason of the failure of G-EDA is it falls in a local minimum and fails to give the max-flow – the width of the red path is 3, as opposed to 6 for the blue path plus the green path.

Example 2. Though the classical max-flow algorithm gives the optimal solution, it performs worse than G-EDA in some other cases. Consider the same topology in Fig. 27 with changed parameters. Suppose all blue and green edges have width 1, red edges have width 2, all channels have creation rate $p = 0.6$, the swapping success rate $q = 1$, s and d have qubit capacity 3, and all other nodes have capacity 2. From Fig. 6, we know when $p = 0.6$, one $(2, 3)$-path is better than three $(1, 4)$-paths. Hence, the optimal solution is the red path with $W = 2$, which can be found via G-EDA. The max-flow algorithm, however, gives the green path, the blue path, and the red path – all paths are single – which is the sub-optimal solution.

Example 3. Due to the enormous search space, we failed to find the optimal strategy via brute-force even in a 10-node network. Suppose $|V| = 10$, every node has 15 qubits and 6 edges, and each edge is composed of 5 quantum channels. In the brute-force searching, we do not assume the P2 and P4 are carried out based on 'paths', but just try all possible assignments of qubits to channels, perform the swapping, calculate the E_t between the given S-D pair, and record the highest result. For any S-D pair, the search space for P2 is $\sim 2.3 \cdot 10^{364}$. Even worse, the entanglement swapping in P4 depends on local states, which is prohibitively hard to enumerate all possible swapping combinations.

A.2 Time and space cost analysis

To avoid unbound computation and space cost in P2, we set the maximum number of multipath $K_m = 200$. We set the maximum path hopcount according to the network itself. For any input G, 100 S-D pairs are randomly selected, and then multipath routing is performed via G-EDA between each S-D pair. The largest hopcount of selected paths whose $E_t > 1$ is the maximum hopcount h_m of all selected paths. We denote the number of nodes as n, the number of S-D pairs as m, and the maximum width of paths as W_m, which is determined by node capacities and edge widths.

A.2.1 Cost of routing metric evaluation. The calculation of E_t can be performed by following the recursive formula set 1. For an h-hop path with width W, the calculation of E_t goes as following. Iterate on k, from 1 to h and further iterate on i, from W to 1: calculate Q^i_k, $\sum_{l=i}^{W} Q^l_k$, P^i_{k-1}, $\sum_{l=i+1}^{W} P^l_{k-1}$, and P^i_k. Five W-element arrays are allocated to store the values. After that, $E_t = q^h \cdot \sum_{i=1}^{W} i \cdot P^i_h$ is calculated in $W + h$ time. Hence, the time cost is $O(hW)$, and the space cost is $O(W)$.

A.2.2 Cost of P2 algorithm of Q-PASS. The initialization costs $O(n)$ time. The double-**for** loop costs $O(mK_m(h_m + h_m + \log(mK_m)))$ time. The **while** loop costs $O(mK_m(h_m + \log(mK_m) + h_m))$ time. Hence, the overall time cost is $O(mK_m(h_m + \log(mK_m)))$.

Each of the L_C, L_P, and q costs $O(mK_m h_m)$ space, the T_Q costs $O(n)$ space, the $width$ costs $O(mK_m)$ space. Hence, the overall space cost is $O(mK_m h_m + n)$.

A.2.3 Cost of EDA. For a classical network $\langle V, E\rangle$, the Dijkstra's algorithm costs $O(n \log n + |E|)$ time because the dequeue operation costs $O(\log n)$ time, the reorder operation of the Fibonacci heap costs $O(1)$ time, and each edge is visited at most once. Similarly, the time cost for EDA is $O(n \log n + h_m W_m + |E|(h_m W_m)) = O(n \log n + |E|(h_m W_m))$. The space cost is $O(n)$.

[3] Red path: (s, A, B, d). Green path: (s, C, A, E, d). Blue path: (s, D, B, F, d).

[4] This number is got via a recursive algorithm instead of mathematical derivation. Consider the number of unique combinations of 15 indistinguishable balls put into 6 different buckets, each with capacity 5.