

Quantum Network Routing based on Surface Code Error Correction

Tianjie Hu, Jindi Wu, and Qun Li

Department of Computer Science, William & Mary, Williamsburg, VA 23185, USA

Abstract— Quantum networks encounter unavoidable channel noises and erasure errors, presenting a huge obstacle in designing protocols that attain both high reliability and efficiency. Typically, quantum networks fall into two categories: those utilize quantum entanglements for quantum teleportation, and those directly transfer the actual quantum messages. In this paper, we present SurfNet, a quantum network that inherits the main advantages from both categories. It employs surface codes as logical qubits for encoding messages, and utilizes two parallel communication channels to fault-tolerantly transfer each surface code in a modular manner. Our approach of using surface codes can timely correct both operational and photon loss errors within the network, and the integration of the two channels within the network can greatly improve network throughput. For the implementation of SurfNet, we propose a novel network architecture, designed to better integrate surface codes into quantum networks. We also propose a novel error correction decoder, designed to fully utilize the modular characteristic of surface codes within our network. Simulation results demonstrate that SurfNet with its decoder significantly enhances the communication fidelity within quantum networks.

Index Terms— Quantum Network; Network Routing; Surface Code; Error Correction

I. INTRODUCTION

Quantum networks find applications in various domains such as distributed quantum machine learning, quantum cryptography, quantum key distribution, and more [1]–[4]. Due to the intricacies of quantum mechanics, designing quantum networks requires carefully managing both reliability and efficiency. It has been a prominent area of research for several decades, with numerous research groups proposing various quantum network designs. These designs can be broadly classified into two schemes based on how they transmit quantum messages within the network.

The first quantum network scheme [5]–[12] utilizes quantum entanglements for quantum teleportation across nodes. As quantum data are inherently unstable and highly susceptible to environmental noises, in this scheme, a series of quantum entangling and swapping operations are performed before the actual data are transferred. The sender and receiver will then attain a pair of entangled qubits, which can be utilized for quantum teleportation to transfer the actual data. However, in practice, this scheme faces inefficiency issues due to the short lifespan of entangled pairs. Even worse, generation of entanglements across adjacent nodes is a probabilistic process, which significantly hinders distant quantum teleportation, making it highly time-consuming. In addition, a large amount of classical communications across nodes are required for both

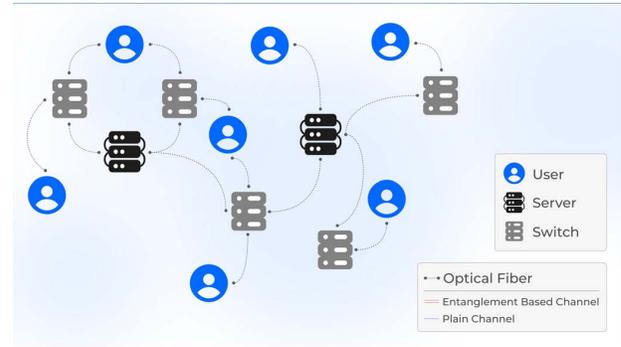


Fig. 1. Example quantum network of SurfNet, where users communicate with each other through switches and servers, interconnected by entanglement-based channels and plain channels within optical fibers.

entanglement swapping and quantum teleportation, frequently constituting a large portion of the total communication time.

The second quantum network scheme [13]–[15] transfers the actual quantum message through optical fibers, and often utilizes logical qubits to carry and preserve the message data. In this scheme, the actual transmission of quantum message can effectively address the inefficiency observed in the previous scheme, and logical qubits are utilized to address the instability of quantum data. A logical qubit is typically represented by multiple physical qubits connected with quantum gates, and its state is determined collectively by these physical qubits. It can encode any single-qubit quantum information into a geometric layout of quantum entanglements. This encoding provides resilience against random noise, through its inherent redundancy provided by the additional physical qubits and the stabilizer operators formed by the quantum gates. However, it comes with the drawback of significantly increasing the traffic in networks since to transmit a single logical qubit, multiple physical qubits must be transmitted.

In this paper, we present *SurfNet*, a new quantum networking strategy that combines techniques from both quantum network schemes above, aiming to simultaneously achieve high network throughput and communication fidelity. Throughout this paper, *throughput* is defined as the number of concurrent communications, and *fidelity* is defined as the probability of occurring without any errors. SurfNet employs surface codes as logical qubits for encoding communication messages, to effectively cope with errors induced by random noise within networks. The choice of surface codes is motivated by their

compact code size and high error threshold, making them a promising quantum networking technique. Incorporating them into quantum networks offers a solution to promptly address operational errors and photon losses within the network via timely performing error corrections on surface codes.

And to minimize the traffic within networks, SurfNet chooses the surface codes with low code distances, which have fewer physical qubits to be transmitted but are normally with lower error thresholds. In SurfNet, fidelity of these surface codes is maintained high with the dual-channel design within SurfNet. Each surface code in SurfNet is transferred in a modular manner as two parts: the *Core* part, consisting of qubits critical for the success rate of error corrections, and the *Support* part, containing the remaining qubits which are less critical but still essential for error corrections. These two parts are transmitted separately using two parallel communication channels: the *entanglement-based channel* that transfers the Core part utilizing quantum teleportation, and the *plain channel* that transfers the Support part itself as photons. This dual-channel approach strikes for a balance between the reliability and efficiency of quantum networks, and addresses the limitations posed by the low entanglement generation rates. As demonstrated later in the routing protocol design, a coordinated use of both channels can effectively achieve high network throughput and communication fidelity.

We also present a new error correction decoder specifically designed for SurfNet. It is based on the finding that physical qubits within the same surface code may experience distinct error rates, and exploiting these variations can significantly improve error correction accuracy. The modular structure of the surface codes in SurfNet utilizes and amplifies this phenomenon by maintaining different error rates at the Core and Support parts of the same surface code. And the *SurfNet Decoder* is designed to effectively incorporate these controlled error rates with the observed syndromes for efficiently decoding the most probable error pattern. In this paper, we focus solely on the Pauli errors and erasure errors. All measurements are assumed to be error-free, and decoherence errors are handled separately via error mitigation techniques at each node within the network.

Our main contributions are summarized as follows:

- We present the first dual-channel quantum network, incorporating both the entanglement-based channel and the plain channel. The end-to-end communication procedure described in this paper efficiently utilizes both channels for transferring surface codes with high fidelity.
- We present a novel error correction decoder that analyzes surface codes in a modular manner and performs the decoding accordingly. And we evaluate its performance against the popular Union-Find decoder. We also illustrate how conventional error correction decoders can be adapted for SurfNet.
- We implement the routing protocol of SurfNet as an integer programming problem. And we evaluate its performance against our proposed baseline model and the mainstream entanglement-based networks.

II. RELATED WORK

Quantum networks have emerged as a promising solution for connecting distant quantum devices. Current physical quantum networks [16]–[18] are still in their early stages, constrained by the current capabilities of qubit processing and storage, as well as the high costs associated with establishing photonic channels and repeater nodes. Efforts have been directed towards leveraging entanglement between nodes for quantum teleportation [19]. Specifically, techniques such as entanglement purification [11], [12] have been employed to enhance the quality of entangled pairs shared across inferior or distant optical fibers.

Extensive research has been conducted in the field of quantum network routing [6]–[10], [20]–[22]. Various research groups are actively exploring general architecture designs for quantum networks, including repeaters or routers [23]–[25], as well as the development of photonic channels [26]–[28]. In the realm of fault tolerant communication, efforts have been made in [13]–[15], [29]. For surface code, different decoders [30]–[32] have been proposed to enhance the error threshold of surface codes, thereby reducing the fidelity requirements for optical fiber constructions. These advancements contribute to the overall efficiency and reliability of quantum networks.

III. BACKGROUND

A. Qubits

Different from classical computing bits, which can only be either 0 or 1, qubits in quantum computing exist as linear combinations of the basis states $|0\rangle$ and $|1\rangle$. This property, known as superposition, allows a qubit to exist in a continuous space between these two basis states, commonly visualized as the Bloch sphere. The specific value of a qubit as either $|0\rangle$ or $|1\rangle$ is only determined upon a quantum measurement. Exploiting superposition, each qubit can simultaneously represent both classical values (0 and 1) of information. Moreover, when n qubits are employed together as an input, this advantage grows exponentially, enabling them to encode 2^n pieces of classical information simultaneously. However, the inherent nature of qubits being in superposition also makes them highly susceptible to instability [33], [34], vulnerable to environmental noise, and prone to random collapse. Common errors affecting single qubits are Pauli errors related to the three quantum Pauli gates $\{X, Y, Z\}$. Each of the three Pauli gates flips a qubit based on the corresponding $X/Y/Z$ -axis on the Bloch sphere. And Pauli errors are defined as situations where random Pauli gates are unintentionally applied to qubits, often due to environmental noise or crosstalk within circuits. Thus, instead of directly utilizing the unstable physical qubits, numerous papers have proposed designs for quantum code designs. These quantum codes are composed of clusters of physical qubits and function as logical qubits [35].

B. Surface Codes

Among various quantum error correction codes, surface code stands out as the most promising one due to its compact

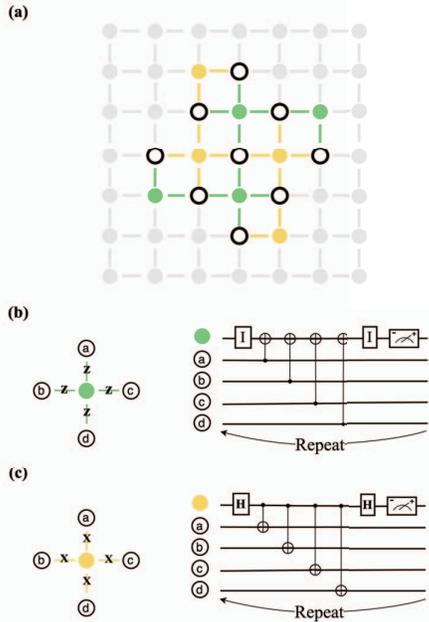


Fig. 2. (a) **Example distance-3 surface code**, for which distance refers to the minimum number of data qubits required to perform a logical operation. Open circles represent data qubits, and colored dots represent measurement qubits (green for measure-Z, yellow for measure-X). (b)(c) Quantum circuits of measure-Z and measure-X.

size and high error threshold. Illustrated in Fig. 2(a), each surface code consists of two groups of physical qubits: data qubits and measurement qubits. Both groups are essentially interconnected physical qubits through quantum gates, yet they play distinct roles within the surface code. The logical qubit information is encoded and stored within data qubits, while measurement qubits are utilized to preserve information integrity. Each data qubit is connected to its neighboring measurement qubits, comprising two measure-X qubits and two measure-Z qubits, except on the boundaries. And if an error occurs in a specific data qubit, its neighboring measurement qubits would reflect the error: measure-X qubits reflect Y or Z types errors, and measure-Z qubits reflect X or Y types errors. Hence, measurement qubits are also known as syndrome qubits or, in code theory, ancilla qubits.

Serving as logical qubits within quantum networks, each surface code encodes a single logical qubit carrying a one-qubit quantum state. There also exist variants on surface codes such as X-cut, Z-cut, or multi-cuts surface codes [36] that can encode multiple logical qubits in a single topology, but for the purpose of this paper, we focus on the use of surface codes serving as single logical qubits. The mechanism underlying surface codes relies on quantum entanglements and reduction on the degrees of freedom. For a comprehensive introduction into Quantum Computation and Quantum Information, we refer readers to [1], and for a detailed explanation on Surface Codes, [36] is recommended.

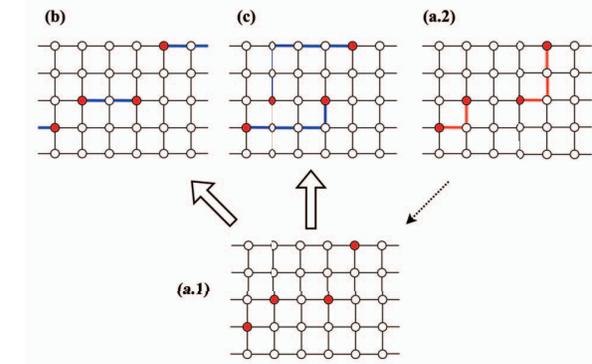


Fig. 3. **Example Surface Code Error Correction**. (a.1) Example syndrome pattern of a surface code, induced by the (a.2) example error pattern. The edges in red represent data qubits with errors, and the vertices in red represent their induced syndromes. (b)(c) Example decoding results from decoders, where (c) is equivalent to (a.2) upon stabilizers, and (b) is not. The edges in blue represent potential erroneous data qubits indicated by decoders.

C. Error Correction

Error correction can be performed on surface codes to effectively mitigate the environmental noise. During initialization, each surface code begins with only the data qubits. As we commonly initialize surface codes in their logical $|0\rangle$, all data qubits are commonly prepared in the state $|0\rangle$. Next, we put on the measurement qubits and connect them to their neighboring data qubits using quantum CNOT gates, as depicted in Fig.2(b)(c). For the ease of syndrome readout, all measurement qubits are commonly prepared in their $|0\rangle$ states. After all the CNOT gates are established, we need to perform at least one round of error correction cycle to complete the setup of a surface code. As in Fig.2(b)(c), each cycle involves a sequence of CNOT operations and a measurement on each measurement qubit, which forces its neighboring data qubits to be in a mutual eigenstate based on its initial measurement outcome. For example, we initialize all data qubits in $|0\rangle$, and via the circuit in Fig.2(b), the measurement outcome of each measure-Z qubit is determined to be $|0\rangle$. So we call the state $|0000\rangle$ of qubits abcd to be a valid eigenstate in the $+1$ eigenspace of the measure-Z qubit, for which $+1$ is associated with the $|0\rangle$ measurement outcome. However, the initialized state $|0000\rangle$ is not a valid eigenstate for the measure-X qubit as shown in Fig.2(c). Thus, the measurement outcome of each measure-X qubit can be either $|0\rangle$ or $|1\rangle$. Yet, once the outcome of each measure-X qubit is determined, its neighboring data qubits are forced to be in one of the possible eigenstates measured in $|+\rangle$'s or $|-\rangle$'s, which are the two basis states along the X-axis of the Bloch sphere.

At this point, we complete the initialization of a surface code, with all data qubits and measurement qubits becoming highly entangled. We name the result of the first round measurement outcomes of all measurement qubits as the quiescent state. And if no error occurs in the future, all measurement results in subsequent error correction cycles would stay in

the quiescent state. If the measurement outcome of any measurement qubit changes, then at least one of its neighboring data qubits has an error, and we mark this measurement qubit as a *syndrome*. In each error correction cycle, all syndromes collectively form a syndrome pattern, as in Fig. 3(a.1), which is then inputted into an error correction decoder. The decoder aims to connect all syndromes either into pairs or to the boundary, and any data qubit along the connected path will be identified as erroneous. Due to the nature of qubits, the real error pattern, as in Fig. 3(a.2), cannot be directly observed, since measuring the data qubits would destroy their encoded logical information and collapse them to basis states.

This decoding problem can be approximated as a minimum weight perfect matching problem, which has been widely studied. Each minimum weight perfect matching decoder connects syndromes with the minimum sum of weights. In our example of Fig. 3, assuming the weight on each edge is 1, the decoder is likely to output either (a.2) or (b), both of which have a sum of weights being 5. However, these two error patterns differ by a logical operator of the surface code, as the combination of these two patterns traverse across the decoding graph. Meanwhile, the error patterns (a.2) and (c) are equivalent in surface codes, as their combination forms a cycle within the decoding graph. Thus, if the decoder outputs (b), a logical error will arise in this surface code; and if the decoder outputs (a.2) or (c), the errors in this surface code can be successfully corrected. Other decoding methods, such as Union-Find decoders, tensor network decoders, and machine learning models, have also been proposed, each demonstrating competitive performances in different noise models. Error correction cycles are typically scheduled in a regular basis, and more frequently when the noise level is high.

IV. NETWORK BASED ON ERROR CORRECTION

As illustrated in the example above, all contemporary error correction decoders suffer from logical errors, which arise when the decoder incorrectly connects two syndromes into a pair instead of individually to the boundary, or vice versa. In fact, along any axis of a logical operator on a surface code, if we can ensure a single data qubit is error-proof, then we can avoid the logical error on this axis. Since for each two syndromes on this axis, the decoder can only connect them into a pair if the path does not pass through the error-proof data qubit. Unfortunately, we can never guarantee a single data qubit to be error-proof, but to achieve a similar effect, we can maintain a high fidelity for such a qubit. Once again, fidelity in this paper is defined as the probability of occurring without errors, and thus, calculated as one minus the error rate.

Considering a complete surface code, distance-3 surface code has 3 such axes, distance-5 surface code has 7 such axes, and a distance- k surface code generally has $(k-1) + (k-2)$ such axes. Along each axis, we ensure the high fidelity of at least one data qubit for preventing the logical error, and we define the minimal subset of such data qubits to be the **Core part** of each surface code, which is critical to the logical error

rate upon decoding; and the remaining data qubits constitute the **Support part** of each surface code, which is less critical to logical error rate, but is still essential for decoding syndromes at routine error correction cycles. The specific selection of data qubits and geometry for the Core part depends on the implementation of various surface codes and is also influenced by the fidelity of the designated routing path, which will be analyzed as a future improvement on SurfNet. For the purpose of this paper, we employ a fixed topology for the Core part, which is latter shown in Section IV-C.

SurfNet is based on this finding, and preserves the fidelity difference of the two parts via two communication channels: the entanglement-based channel and the plain channel. We assume error-free measurements, and only consider random Pauli errors and erasure errors, as these two are the predominant errors in quantum networks. When a data qubit is erased, we substitute it with a maximally mixed state, initialized as $|0\rangle$ and randomly subjected to a Pauli gate chosen uniformly from $\{I, X, Y, Z\}$. Thus, on each transmitted surface code, locations with erasures exhibit a much higher error rate. Leveraging this observation, the SurfNet Decoder prioritizes locations with erasures, identifying these replaced data qubits with a higher likelihood for decoding paths to travel through. Similarly, data qubits within the Support part are prioritized than those in the Core part, given their relatively higher error rates. In the following, we introduce the components within our quantum network, our proposed one-way communication procedure, and our new error correction decoder.

A. Components

Users User nodes generate communication requests in the network, which are to transfer quantum messages from one user to another. In SurfNet, all messages are encoded into surface codes before transmission, and each surface code contains two parts, the Core and the Support.

Optical Fibers Optical fibers serve as edges in quantum networks that interconnect users and switches. Each optical fiber in SurfNet is equipped with two communication channels: an entanglement-based channel that utilizes teleportation to transmit the Core parts of surface codes, and a plain channel that actually transmits the Support parts themselves as photons down the optical fiber.

Switches Switch nodes in SurfNet serve as intermediate stations along each communication path. Each switch operates two routines, corresponding to the two communication channels. One routine continuously generates entangled qubit pairs, which are then shared with adjacent switches through the entanglement-based channels. The other routine re-encodes any incoming Support parts of surface codes from the plain channels into new photons, aiming to reduce the chance of random decoherence.

Servers Server nodes are essentially switch nodes with larger quantum memories. Similar to switches, servers operate the same two routines corresponding to the two communication channels. Additionally, servers are capable of performing error correction when a complete surface code arrives.

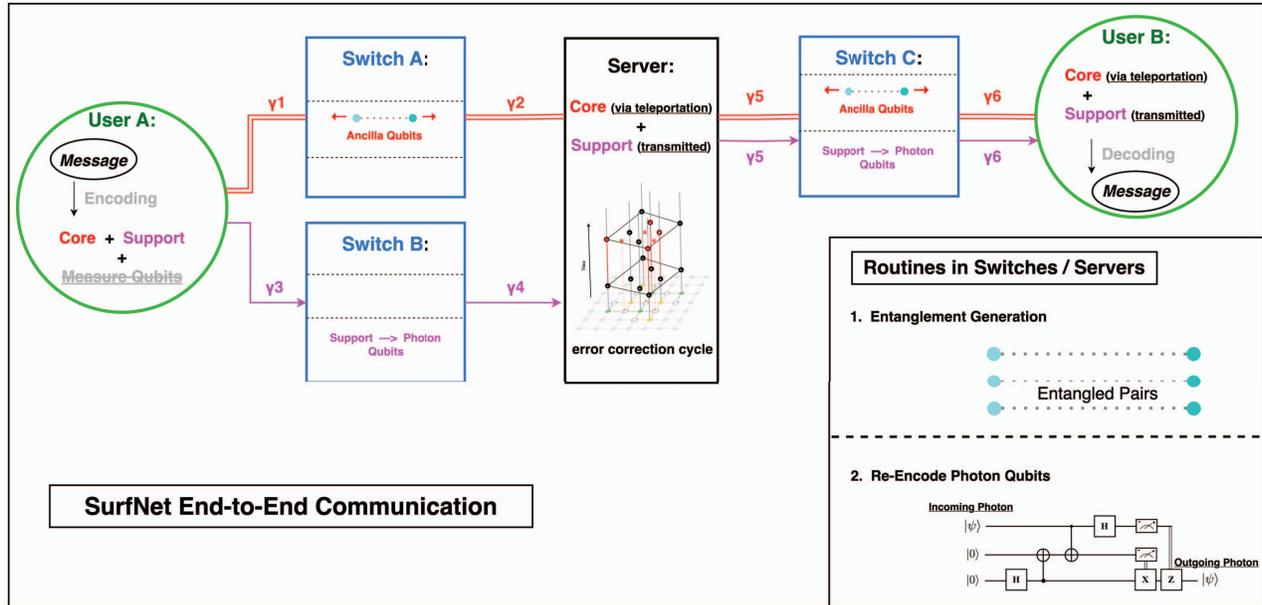


Fig. 4. **Example one-way communication** of transferring a surface code from user A to B, where the green circles represent Users, blue squares represent Switches, black square represents Server, red double lines represent Optical Fibers for transmitting ancillary entangled pairs, and pink arrows represent Optical Fibers for transmitting the Support parts as photon qubits. Each optical fiber is labeled with its fidelity $\gamma_i \in [0, 1]$.

B. One-way Communication

We assume that the entire quantum network is connected, so each pair of users can communicate either directly or through intermediate switches and servers. In SurfNet, transmitting each surface code involves the transmission of the Core part and the Support part. Notably, each part can be further subdivided into smaller parts to enable efficient parallel transmission.

Transmission of the Core parts is through the entanglement-based channel, which guarantees high fidelity but consumes more computing resources. Once the routing path is scheduled, all switches or servers involved in the transmission of the Core parts start to share its generated entangled pairs through the entanglement-based channels with its adjacent nodes along the routing path, preparing for quantum teleportation. Once the adjacent entanglements along the whole path are all established, each intermediate node performs entanglement swapping to transform the chain of adjacent entanglements into a single entangled pair shared between the sender node and receiver node. As the process of entanglement is highly probabilistic, the procedure of entanglement generation and swapping needs to be repeated for several times. Subsequently, entanglement purification is deployed to combine the multiple low-quality entangled pairs into a single pair of high-quality entangled qubits. Finally, this entangled pair is consumed in quantum teleportation to transfer the Core part from the sender node to the receiver node.

Meanwhile, the Support parts are encoded into individual photons and physically transmitted through the plain channels.

Since error correction requires the complete surface code to be present in the same server, the Support part(s) normally arrives earlier in the server and keeps being refreshed via error mitigation circuits until the Core part(s) arrives.

Example one-way communication is illustrated in Fig. 4. For ease of illustration, here we do not subdivide the Core part and Support part into smaller parts. Before the error correction in Server, the Core part is teleported to Server by consuming the ancillary entangled pair generated in Switch A and shared through the entanglement-based channel. Meanwhile, the Support part is physically transmitted as photons to Server through the plain channel, bypassing Switch B. After the error correction, both routes bypass Switch C, yet once again, the route for the Core part is through the entanglement-based channel, and the route for the Support part is through the plain channel. It is important to note that the two routes do not necessarily need to take the same path after error correction, and the routes in Fig. 4 are just for illustrative purposes.

C. Error Correction Decoder

Before we introduce our SurfNet Decoder, we firstly illustrate how mainstream decoders can be adapted to solve the error corrections in SurfNet. We begin by representing each surface code as a *decoding graph* $G = \{V, E, W\}$, where V and E are vertices and edges, and W represents a set of edge weights correlated to the error rates. The desired graph comes directly from the square lattice representation of the surface code, where each vertex on the square lattice is a measurement qubit, and each edge is a data qubit. Computing the weights needs some extra work. Notice in SurfNet, data

Algorithm 1 Modified MWPM Decoder

Input: set of Syndromes σ , estimated data qubit fidelity $\{\rho_i\}$ **Output:** estimated error pattern Γ

- 1: Construct decoding graph $G = \{V, E, W\}$
 - 2: Create path graph $G' = \{\sigma, E', W'\}$, initialize $G' = \{\sigma, \{\}, \{\}\}$
 - 3: **for** $u, v \in \sigma$ with $u \neq v$ **do**
 - 4: Find $P(u, v)$, the shortest path from u to v in G
 - 5: Compute $w'(u, v) = \sum_{e \in P(u, v)} w_e$
 - 6: Add (u, v) and $w'(u, v)$ to E' and W' of G'
 - 7: **end for**
 - 8: Apply Blossom to G' and get the minimum weight perfect matching M'
 - 9: Initialize Γ as empty
 - 10: **for** (u, v) in M' **do**
 - 11: add the edges of $P(u, v)$ to Γ
 - 12: **end for**
-

qubits are transferred via different channels and different routes, then data qubits are with different fidelity and should be given different weights. The *estimated fidelity* for each data qubit is calculated by multiplying the fidelity of all optical fibers it travels through, denoted as $\rho = \prod_{i \in \text{path}} \gamma_i$, where γ_i is the fidelity of the optical fiber. For a data qubit in the Core part, after each time of entanglement purification, its new $\rho = \frac{\rho_1 \rho_2}{\rho_1 \rho_2 + (1 - \rho_1)(1 - \rho_2)}$ [11], for which ρ_1 and ρ_2 are the estimated fidelity of the consumed entangled pairs. For erasures, the estimated fidelity equals to 0.5. The weight at each edge is then computed as $w = -\ln(1 - \rho)$.

Theorem 1. *Error corrections in SurfNet can be solved using a minimum weight perfect matching (MWPM) algorithm.*

Proof. Following the above construction, let $G = \{V, E, W\}$ be an arbitrary decoding graph, and $\sigma \subseteq V$ be the set of syndromes. As mentioned previously, the error correction decoder aims to connect all syndromes into pairs, and each syndrome should be connected exactly once. Each connected path made by the decoder is associated with a *likelihood*, which correlates with the fidelity of each data qubit along the connected path. And the performance of the decoder is evaluated by summing the likelihoods of all connected paths it made: the higher the likelihood is, the better the decoder performs. This decoding problem can be efficiently formulated as the minimum weight perfect matching problem. A *perfect matching* M of graph G is defined as $M \subseteq E$ such that each vertex in V is incident to one and only one edge in M . Notice in the decoding graph G , there exist vertices $V \setminus \sigma$ that are without syndromes and should not be connected. Thus, before we apply the minimum weight perfect matching algorithm, we need to construct a path graph G' . As in Algorithm 1, syndromes σ constitute the vertices in G' , and they are interconnected via the shortest paths found in the original graph G . These paths are added as edges into G' . Then a minimum weight perfect matching algorithm (for example, as in Algorithm 1, the

Algorithm 2 SurfNet Decoder

Input: set of Syndromes σ , set of Erasures δ , estimated data qubit fidelity $\{\rho_i\}$, decoder step size $r = 2/3$ **Output:** estimated error pattern Γ

- 1: Create list of odd clusters $\{C_1, C_2, \dots\}$ that each contains a single syndrome
 - 2: **while** exists odd cluster C_i **do**
 - 3: **for** all odd Clusters $\{C_i\}$ **do**
 - 4: **Grow** C_i by corresponding **speed**:
 - 5: $-r / \ln(1 - 0.5)$ edge if on δ
 - 6: $-r / \ln(1 - \rho_i)$ edge if else
 - 7: If C_i meets another cluster, fuse together
 - 8: If C_i becomes even, remove from odd list
 - 9: **end for**
 - 10: **end while**
 - 11: Find a spanning tree for each cluster
 - 12: Apply peeling decoder to the spanning forest to find Γ
-

blossom algorithm [37]) is applied to G' to get the minimum weight perfect matching M' . The requirement of minimum weight ensures that such M' represents the decoding result with a high likelihood. Notably, this result is not guaranteed to be with the maximal likelihood, since the minimum weight perfect matching formulation serves as an approximation to the actual decoding problem. The proof on the correctness of this approximation is accomplished as a joint result from [30] and [38], which handles the error correction for surface codes in two cases: surface codes without boundaries, and surface codes with boundaries. \square

Corollary 1.1. *Error corrections in SurfNet can be solved via the sparse blossom algorithm in $O(n^2)$.*

Proof. The proof follows as a direct result from Theorem 1, since the sparse blossom algorithm [31] is a solver for the minimum weight perfect matching problem. Notably, different from other conventional minimum weight perfect matching decoders, the sparse blossom algorithm solves for the *embedded matching* M_E , which is defined as $M_E \subseteq E$ such that each vertex in σ is incident to an odd number of edges in M_E , and each vertex in $V \setminus \sigma$ is incident to an even number of edges in M_E . The sparse blossom algorithm aims to find a minimum weight embedding matching of the decoding graph, and this problem proves in [31] to be a reduction from the original minimum weight perfect matching problem. As shown in [31], the sparse blossom algorithm outputs results with an observed complexity of $O(n^{1.32})$. And updating the decoding graph with our calculated fidelity requires $O(n^2)$. Thus, the total time for error corrections in SurfNet using the sparse blossom algorithm is $O(n^2)$. \square

So far, we have proven that conventional minimum weight perfect matching algorithms can be modified for solving the error corrections in SurfNet, achieving a $O(n^2)$ running time. In addition, as noted in [31], employing the sparse blossom

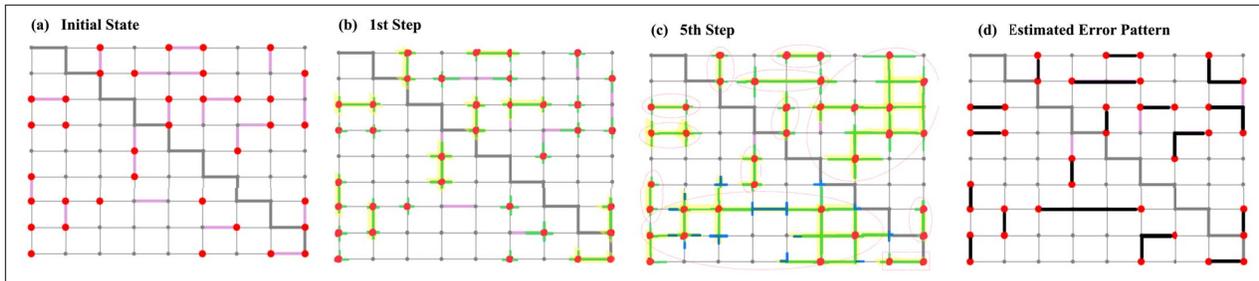


Fig. 5. **Example decoding graph of the SurfNet Decoder**, where each vertex is a measurement qubit and each edge is a data qubit. For illustration, the growth speeds on Erasures, Core, Support are $\{1/2, 1/8, 1/4\}$. (a) **Initial State**. Red vertices denote Syndromes, Purple lines denote Erasures, and Solid Grey lines denote the Core part. (b) **Growth Step 1**. Green lines denote newly added edges for each cluster. Even clusters are marked in Yellow. (c) **Growth Step 5**. Blue lines denote newly added edges for each cluster, and Green lines denote existing edges. After this step, all clusters become even and finish growing. (d) **Estimated Error Pattern**, where Black lines denote data qubits with errors.

algorithm in our Algorithm 1 can be adapted for a parallel implementation, with a theoretical amortized complexity of linear time. However, employing the sparse blossom algorithm also suffers a worst time complexity of $O(nq^3 + mq^2)$, in which n is the number of vertices in the graph, m is the number of edges, and q is the number of syndromes.

Thus, it motivates us to also design a new decoder that works better in the worst case scenarios, which is our SurfNet Decoder. Overall, the SurfNet Decoder is based on the definition of “growth speed” of clusters. Its routine is adapted from the Union-Find Decoder [32] with the peeling decoder [39], and is tailored explicitly for our dual-channel network architecture. The algorithm is outlined in Algorithm 2, with an illustrated procedure in Fig. 5. Among the inputs to the SurfNet Decoder, syndromes and erasures are derived from the measurement outcomes in Fig.2 (b)(c), and the estimated fidelity is computed as previously described. The decoder step size can be further adjusted to optimize between the decoding speed and accuracy, with the default $2/3$ generally achieving a good balance. During the decoding, the initial step involves creating a singleton set for each syndrome vertex, denoted as a *cluster*. Additionally, we define odd clusters to be clusters containing an odd number of syndromes, and even clusters to be clusters containing an even number of syndromes. In this step, each syndrome is an odd cluster and its cluster solely contains itself. Next we grow the exteriors of each cluster at different speeds, calculated as $-r/\ln(1 - \rho_i)$, where r is the decoder step size and $(1 - \rho_i)$ is the probability of errors occurring at the corresponding edge. The speed is maximized at erasures, as the replaced qubits with maximally mixed states exhibit the highest error rate. When two expanded edges touch or one edge reaches the exterior of another cluster, the two clusters fuse together. If the combined cluster becomes an even cluster, we remove it from the list of odd clusters, as it no longer needs to grow.

Theorem 2. *Error corrections in SurfNet can be solved via the SurfNet Decoder with a worst case complexity of $O(n\alpha(n))$, where α is the inverse Ackermann function.*

Proof. First we prove the near-maximal likelihood of the output Γ from the SurfNet Decoder. Let Γ' be arbitrary error pattern that is with a different likelihood from Γ . Denote $\{C_1, C_2, \dots, C_k\}$ as the set of even clusters at the end of the SurfNet Decoder. As the optimality of the peeling decoder is proven in [39], then there exists at least one path in Γ' that travels from one even cluster to another, which spans extra edges that were not grown by any clusters. Also by Algorithm 2, each cluster suffices to be a shape with constant radius, with each point on its boundary reachable by the same likelihood, and any exterior edges can only be reached with a lower likelihood, which determines Γ' to be with a lower likelihood than the algorithm output Γ . The worst time complexity of $O(n\alpha(n))$ is guaranteed by deploying an union-find implementation as in [32]. In this implementation, each cluster is recorded as a tree data structure and its boundary vertices are recorded in a separate list. The boundary lists are updated at each round of growth, and cluster fusion is accomplished using the union-find algorithm. Several subroutines for tracing the states of each cluster are also employed, and the implementation details can be found in [32]. \square

V. NETWORK ROUTING

Given the limited capacity and low entanglement generation rates at each switch and server, we aim to design a reliable and efficient network that achieves a high network throughput while maintaining a high successful rate for each communication. And we make the following assumptions:

- The network is bidirectional and connected, ensuring the existence of at least one path from one user to another.
- The fidelity of each optical fiber can be measured and remains constant during routing.
- Switch nodes, server nodes, and optical fibers have finite capacities that are not excessively large.

The routing procedure consists of two stages: scheduling and execution. To ensure simplicity and efficiency, we adopt offline scheduling to allocate available resources. And we employ online execution to promptly recover routing paths from errors or failures, minimizing the waste of utilized resources.

A. Offline Scheduling

First, we introduce the offline scheduling in SurfNet. Before each round of routing, the routing protocol collects requests from user nodes and operating information from switches, including their current capacity and entanglement status. For each request, the Support part consumes capacity in switches, and the Core part additionally consumes the prepared entangled pairs. The routing problem is then formulated as an optimization problem with capacity and entanglement resources acting as constraints. To provide a formal mathematical representation of our routing protocol, we define key terms as outlined in Table I. As listed in its last column, the variables in our formulation consist of four sets of integer variables: Y_k to decide whether each request will be scheduled or partially scheduled, a_e^k and b_e^k to decide routes for each request, and x_r^k to decide the number of scheduled error corrections for each request.

Computing fidelity for surface codes involves multiple exponentiations and multiplications, leading to a non-linear problem formulation. Thus, we translate the fidelity at each edge as its *noise*. The noise at each edge is calculated as $\mu = \log(1/\gamma)$, where γ is its fidelity measured in the range $[0, 1]$. This allows noise to accumulate as summations instead of multiplications, and lower values are better. Correspondingly, we model the effect of each error correction as a decrease in the current noise to the surface code. And when computing the total noise to a surface code, we separately sum the noise for the Core and Support parts, to account for their different influences in logical error rates. For example, if we transmit a surface code of 25 data qubits, with 7 data qubits in the Core part, through the route in Fig. 4, then its two expected noises are:

$$\begin{aligned} & \frac{7}{7}(\mu_1 + \mu_2 + \mu_5 + \mu_6) - \omega \\ & \frac{7}{25} * \frac{1}{2}(\mu_1 + \mu_2 + \mu_5 + \mu_6) + \frac{18}{25}(\mu_3 + \mu_4 + \mu_5 + \mu_6) - \omega \end{aligned}$$

correspondingly for the Core part and the entire surface code. Notice in the second equation, when summing the noise for the entire surface code, the noise for the Core part is halved to account for the effect of entanglement purification through the entanglement-based channel. As later shown in Eq. 6, we require both calculated noises to be below their corresponding thresholds to ensure the high fidelity of each communication.

Our integer programming for the routing problem states as follows. The objective function is to sum the number of concurrent communications, with the goal of maximizing the total throughput in our network:

$$\max \sum_{k \in \mathbb{K}} Y_k \quad (1)$$

with the following constraints of Eq. [2]-[6]:

$$Y_k \in [0, i_k], x_r^k \in [0, i_k], a_e^k \geq 0, b_e^k \geq 0 \quad (2)$$

which sets up the feasible regions of variables.

TABLE I
TABLE OF NOTATIONS USED IN ROUTING FORMULATION

Notation	Definition
\mathbb{U}	The set of all Users
\mathbb{R}	The set of all Switches (including Servers)
$\mathbb{R}\mathbb{R}$	The set of all Servers
E	The set of all Edges
η_r	Storage capacity in each switch $r \in \mathbb{R}$
η_e	Number of prepared entanglements across each edge $e \in E$
μ_e	Noise at each edge e
\mathbb{K}	The set of all Communication Requests $k = [(s_k, d_k), i_k]$
i_k	Number of Surface Codes in Request k
n	Number of Core data qubits in each surface code
m	Number of Support data qubits in each surface code
ω	Noise Reduction for performing error correction in a server
W_c	Noise Threshold for the Core part
W	Noise Threshold for the entire surface code
Y_k	Integer variable of determining Execution of request k
a_e^k	Integer variable of determining number of Core qubits for request k travelling through edge e
b_e^k	Integer variable of determining number of Support qubits for request k travelling through edge e
x_r^k	Integer variable of determining number of Error Corrections scheduled for request k in each server r

$$\begin{aligned} & \sum_{e \in E} a_{(d,j)}^k + b_{(d,j)}^k + a_{(i,s)}^k + b_{(i,s)}^k = 0 \quad \forall_{k=(s,d)} \in \mathbb{K} \\ & \sum_{(i,d) \in E} a_{(i,d)}^k = \sum_{(s,j) \in E} a_{(s,j)}^k = n * Y_k \quad \forall_{k=(s,d)} \in \mathbb{K} \quad (3) \\ & \sum_{(i,d) \in E} b_{(i,d)}^k = \sum_{(s,j) \in E} b_{(s,j)}^k = m * Y_k \quad \forall_{k=(s,d)} \in \mathbb{K} \end{aligned}$$

which are the Initialization and Termination constraints that apply to the sender nodes and destination nodes.

$$\begin{aligned} & \frac{1}{n} \sum_{(i,r) \in E} a_{(i,r)}^k = \frac{1}{m} \sum_{(i,r) \in E} b_{(i,r)}^k = x_r^k \quad \forall_{r \in \mathbb{R}\mathbb{R}, k \in \mathbb{K}} \\ & \sum_{(i,r) \in E} a_{(i,r)}^k = \sum_{(r,j) \in E} a_{(r,j)}^k \quad \forall_{r \in \mathbb{R}, k \in \mathbb{K}} \quad (4) \\ & \sum_{(i,r) \in E} b_{(i,r)}^k = \sum_{(r,j) \in E} b_{(r,j)}^k \quad \forall_{r \in \mathbb{R}, k \in \mathbb{K}} \end{aligned}$$

which are the Conservation constraints that apply to the switch and server nodes.

$$\begin{aligned} & \sum_{k \in \mathbb{K}} \sum_{(i,r) \in E} a_{(i,r)}^k + b_{(i,r)}^k \leq \eta_r \quad \forall_{r \in \mathbb{R}} \\ & \sum_{k \in \mathbb{K}} a_{(i,j)}^k + \sum_{k \in \mathbb{K}} a_{(j,i)}^k \leq \eta_e \quad \forall_{e=(i,j) \in E} \end{aligned} \quad (5)$$

which are the Capacity constraints that ensure the computing resources within the network cannot be over-consumed. The first set of constraints sum the number of data qubits scheduled to be stored in each switch and server, and ensure they do not exceed the quantum memory capacity at each location. The second set of constraints sum the number of teleportation

scheduled across each optical fiber, and ensure they do not exceed the number of prepared entangled pairs in the corresponding optical fiber.

$$\begin{aligned}
0 &\leq \sum_{e \in E} (\mu_e * a_e^k) - \omega \sum_{r \in \mathbb{R}\mathbb{R}} x_r^k \leq W_c * Y_k \quad \forall k \in \mathbb{K} \\
\sum_{e \in E} [\mu_e * (\frac{1}{2}a_e^k + b_e^k)] - \omega \sum_{r \in \mathbb{R}\mathbb{R}} x_r^k &\leq W * Y_k \quad \forall k \in \mathbb{K}
\end{aligned} \tag{6}$$

which are the Noise constraints that ensure high-fidelity communication, computed as summing the noise along the routing path and subtracting the number of error corrections performed. The first set of constraints sum the noise only for the Core parts, to account for their critical influences in logical error rates of surface codes. The summation is required to be below the threshold W_c to prevent excessive noise accumulation that can drift up the logical error rates. We also force this summation to be above 0 to prevent the inclusion of consecutive servers in a single route to conserve the computing resources. The second set of constraints sum the noise for the entire surface code, and is required to be below the threshold W . As demonstrated later in Section VI-C, fine-tuning the two thresholds W_c and W allows for balancing communication fidelity and network throughput.

Note that $\eta_r, \eta_e, \mu_e, i_k$ are collected from the network, and n, m, ω, W_c, W are pre-defined parameters, so the above constraints remain linear. However, as some of these variables are integers, the above integer programming formulation does not guarantee a polynomial-time solution. Feasible solutions can be adapted from existing algorithms for solving the integer minimum-cost Multi-Commodity Network Flow (MCNF) problems [40] using the Dantzig-Wolfe decomposition principle as in [41], [42]. For simplicity, in this paper, we employ a relaxed Linear Programming version with rounding in the Evaluation section to simulate the performance of SurfNet.

B. Online Execution

Next, we introduce the online execution in SurfNet. Upon receiving the routing schedules from the routing protocol, users directly transfer their Support parts of surface codes to the designated next node, while retaining the Core parts until reliable entanglements are established. Switches and servers operate two simultaneous routines: preparing entanglements for the Core parts and directly transferring the Support parts. In case an error correction is scheduled in a server, the server initiates a third routine: preparing error correction circuits and awaits the arrival of a complete surface code.

Common challenges arise from unpredictable situations like transmission delays or crashes in incoming/outgoing ports. And these issues are effectively addressed in our network. Each qubit has the option to remain in its current node and can be preserved in error mitigation circuits such as dynamic decoupling (DD) sequences. Additionally, if abundant resources are available in the local neighborhood, a node can locally replace a failed route with a recovery path leading to the

next designated node. Thus, even without a centralized routing protocol, SurfNet can still operate in a hierarchical manner. In scenarios where only a few requests are generated in the network, centralized and hierarchical routing demonstrate similar performances.

Another main mechanism in our network is the parallelism between data transfer and error correction. Thanks to the intrinsic redundancy of surface codes, servers do not need to wait for all data qubits to be present before performing error correction. In SurfNet, error correction begins as soon as all Core parts and sufficient Support parts are collected, with all missing qubits marked as erasures. The tolerance for missing Support parts depends on their specific locations on the square lattice and is influenced by the overall fidelity of collected data qubits. Generally, in SurfNet, the parallelism between data transfer and error correction presents another trade-off between efficiency and reliability: if the SurfNet Decoder always waits for the complete surface code to arrive, the reliability of error correction improves, but the message delivery efficiency is compromised.

In addition, we deploy opportunistic routing to accelerate the transmission through entanglement-based communication channels. Recall during the transmission of the Core parts, we need to wait for reliable entanglements to be established between the sender and receiver nodes. It requires multiple rounds of pairwise entanglements along the entire path, and its difficulty scales exponentially with the distance between the sender and receiver nodes. Thus, instead of waiting for the distant entanglement to be established, we allow the Core parts to move forward along the path as soon as a reliable entanglement is established across the next few nodes, even if the movement is only across a single optical fiber. This approach allows us to divide a long routing path into segments, significantly improving both communication efficiency and fidelity. Based on simulation experiments, we fix the minimum distance for the movement to be two consecutive optical fibers.

VI. PERFORMANCE EVALUATION

In this section, we conduct a series of simulations to evaluate the performance of the SurfNet network design and the SurfNet Decoder. We implement our proposed routing protocol and decoder algorithm for different network scenarios, and compare their performance against baseline and mainstream models.

A. Evaluation Methodology

Considering the randomness in real-world network structures, we randomly generate networks without a predefined topology. Among these, three representative networks are chosen, each falling into a distinct scenario: a network with abundant facilities of switches and servers, one with sufficient facilities, and one with insufficient facilities. For each of the three networks, we randomly assign two fidelity to each optical fiber, representing scenarios of good or poor qualities of

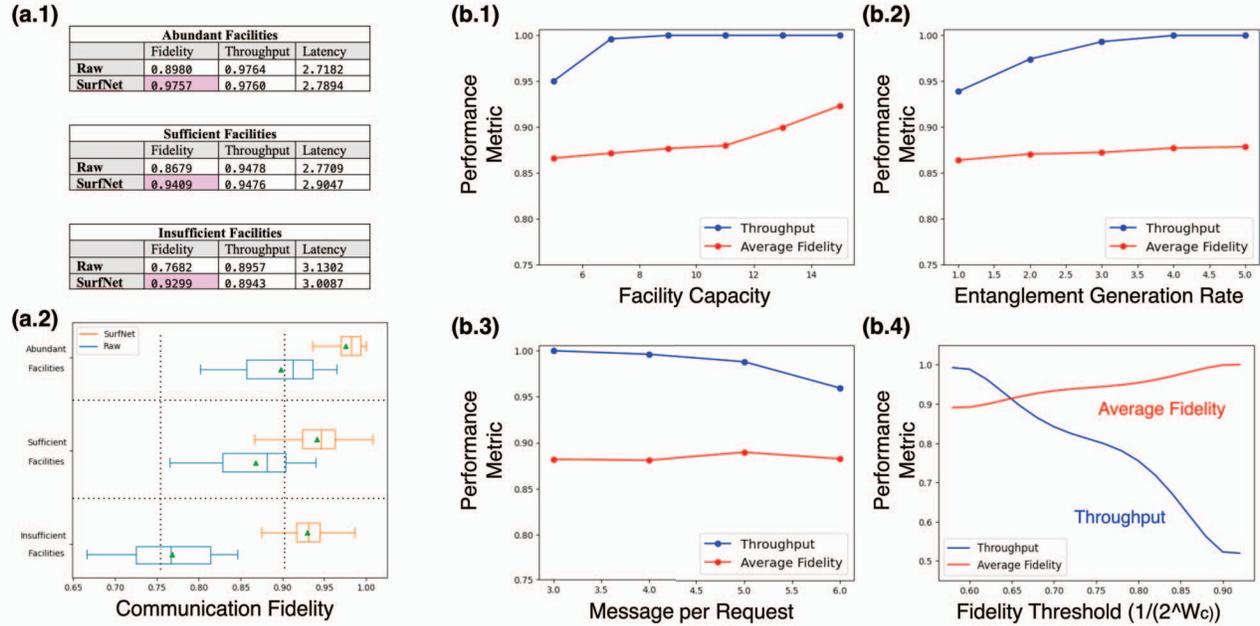


Fig. 6. (a) Comparison between Raw and SurfNet in different network scenarios. In each scenario, the comparison is over three evaluation metrics shown in table, and the comparison over communication fidelity is detailed in the plots. (b) Performance of SurfNet with respect to different (b.1-3) network parameters and (b.4) routing parameter. The performance is evaluated using two metrics: fidelity and throughput.

service. In addition, to evaluate the performance of the SurfNet Decoder, we implement it on surface codes with four different sizes: distance-9, distance-11, distance-13, and distance-15, for which distance refers to the minimum number of data qubits for a logical operator. Then the decoder is evaluated at different rates of Pauli errors, for the calculation of error threshold. We also include a fixed erasure error rate to mimic real scenarios within a network. Both erasure errors and Pauli errors randomly occur on each data qubit, without following a specific error pattern.

B. Benchmarks

Our proposed network design is denoted as **SurfNet**, and is compared to the following network designs:

- **Raw**: the baseline version of SurfNet, which does not divide surface codes into Core and Support parts, and all data qubits are transmitted through plain channels. Error corrections are available in each server, and all switches and servers have increased capacity as they no longer need to prepare entanglements.
- **Purification $N=1,2,9$** : the mainstream quantum networks that purely utilize quantum teleportation to transfer data qubits, and employ entanglement purification to improve communication fidelity. In a straightforward implementation, purification is carried out at each optical fiber in the network. Three scenarios are considered, denoted as “ $N=1$ ”, “ $N=2$ ”, “ $N=9$ ”, where N indicates the number of additional entanglement pairs consumed for purification at each optical fiber.

The above network designs are tested in three network scenarios, which are generated using the Barabasi-Albert model with over 20 nodes, with the most connected nodes chosen to be the servers and switches. Fidelity at each optical fiber is randomly assigned within the range of $[0.75, 1]$ representing good-connection scenarios and within $[0.5, 1]$ representing poor-connection scenarios. Each network design undergoes 1080 trials with different sets of network parameters, including the capacity for switches and servers, entanglement generation rate, number of requests, and number of messages in each request.

Our proposed error correction decoder is denoted as the *SurfNet Decoder*, and is compared against the baseline *Union-Find Decoder* implemented as in [32]. Each input surface code is subjected to both erasure errors and Pauli errors, with the erasure rate fixed at 15%, and the Pauli error rate ranging between 5.0-8.5%. These error rates are halved at the Core part of each surface code.

C. Results

In this section, evaluation is conducted over three evaluation metrics: *fidelity*, *latency*, and *throughput*. Fidelity and latency are defined as the success rate and waiting time for each communication, and are computed as the average of all communications executed in the network. Throughput is calculated as the number of executed communications divided by the total number of requested communications.

From the tables (a.1) in Fig. 6, throughput and latency in both networks are similar, while SurfNet exhibits much higher

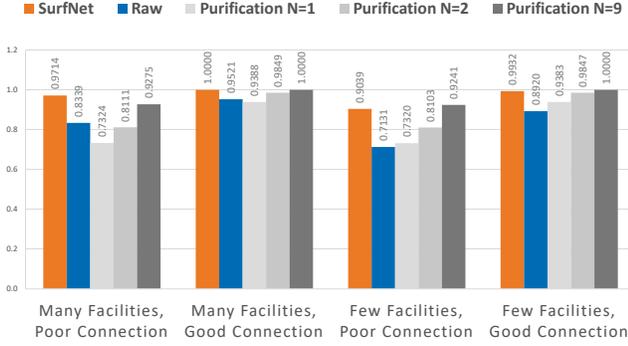


Fig. 7. Comparison on averaged communication fidelity between the five routing designs in four network scenarios. In each scenario, the result is computed as the average from all 1080 trials.

average communication fidelity in all three network scenarios as in (a.2). This improvement is mainly due to the dual-channel characteristic of SurfNet, and the partitioning of each surface code into the Core and the Support parts proves to be effective in maintaining communication fidelity.

Fig. 6 (b.1-3) provide a detailed analysis of SurfNet with respect to different network parameters. Facility capacity in (b1) and entanglement generation rate in (b2) are the two main parameters controlling the total computing resources within the network. Their direct impacts on both throughput and fidelity are evident since, naturally, more resources can support more communications. Message per request in (b3) limits the maximum number of surface codes allowed in each request, thus controlling the consumption rate of the computing resources. From the graph, it is observed that it has little impact on fidelity but seemingly affects throughput, as the supply of computing resources can no longer meet the growing demand. And in Fig. 6 (b.4), we analyze the effect of fidelity threshold in the routing protocol of SurfNet, computed as $1/(2^{W_C})$ where W_C is the noise threshold. As mentioned earlier, the noise thresholds in SurfNet can be used to strike for a balance between communication fidelity and network throughput. As in the graph, a low fidelity threshold allows for low-fidelity communications, resulting in high network throughput but low average fidelity for each communication. Conversely, a higher fidelity threshold is more selective for communication quality, leading to lower network throughput but higher average fidelity.

An overall comparison of all five network designs is shown in Fig. 7. We configure the routing protocols in all networks to yield similar throughputs, and focus only on the average fidelity in each network design. SurfNet achieves high fidelity across all four scenarios. It demonstrates significant advantage in networks with abundant facilities, while its drawbacks become more apparent in scenarios with limited facilities and poor connections. Improving the online execution stage of SurfNet, such as finding better recovery paths or incorporating adaptive code sizes based on quality of service, are potential

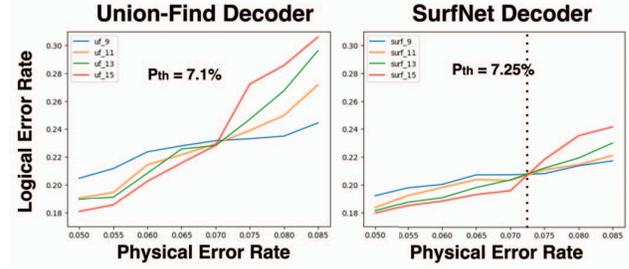


Fig. 8. Pauli error threshold of surface codes using Union-Find decoder (left) and SurfNet Decoder (right).

directions for enhancing SurfNet in such scenarios.

Fig. 8 compares error correction performance between the Union-Find decoder and the SurfNet Decoder. In general, below the pauli error threshold, a surface code with a larger size tends to have a lower logical error rate. Conversely, above the pauli error threshold, a surface code with a smaller size tends to have a lower logical error rate. The SurfNet Decoder exhibits a higher threshold of 7.25% compared with the 7.1% of Union-Find decoder. This advantage can be further enhanced if the Core part of the surface code is configured to be larger or has a more optimized geometry, which presents potential future directions for improving the SurfNet Decoder.

VII. CONCLUSION

In this paper, we introduced our SurfNet, an innovative quantum network utilizing surface codes as logical qubits for quantum message transmission. We proposed a novel end-to-end communication procedure employing two parallel communication channels within SurfNet: the entanglement-based channel and the plain channel. This dual-channel approach effectively addresses the limitation posed by the low entanglement generation rate at each node, and offers a way to balance the reliability and efficiency of quantum networks. To enhance communication fidelity, error corrections can be performed at servers along the routing path. The routing protocol in our network is designed to coordinate the balance between the network throughput and average communication fidelity. Additionally, we introduced the SurfNet Decoder, which can fully leverage the modular characteristic of surface codes within our network. For evaluation, we conducted simulations for various network scenarios, comparing SurfNet against its baseline and other mainstream quantum networks. We also separately compared our SurfNet Decoder against its baseline Union-Find decoder. The results demonstrated that both SurfNet and its decoder can significantly improve the overall fidelity of quantum networks.

ACKNOWLEDGMENTS

The authors would like to thank all the reviewers for their helpful comments. This work was supported in part by the Commonwealth Cyber Initiative (CCI, cyberinitiative.org).

REFERENCES

- [1] M. A. Nielsen and I. Chuang, *Quantum computation and quantum information*. Cambridge University Press, 2002.
- [2] M. Tengner, *Photonic Qubits for Quantum Communication: Exploiting photon-pair correlations; from theory to applications*. PhD thesis, KTH, Stockholm, Sweden, 2008.
- [3] J. Wu, Z. Tao, and Q. Li, "Scalable quantum neural networks for classification," in *2022 IEEE International Conference on Quantum Computing and Engineering (QCE)*, pp. 38–48, IEEE, 2022.
- [4] J. Wu, T. Hu, and Q. Li, "Distributed quantum machine learning: Federated and model-parallel approaches," *IEEE Internet Computing*, vol. 28, no. 2, pp. 65–72, 2024.
- [5] H.-J. Briegel, W. Dür, J. I. Cirac, and P. Zoller, "Quantum repeaters: the role of imperfect local operations in quantum communication," *Physical Review Letters*, vol. 81, no. 26, p. 5932, 1998.
- [6] Y. Zeng, J. Zhang, J. Liu, Z. Liu, and Y. Yang, "Multi-entanglement routing design over quantum networks," in *IEEE INFOCOM 2022-IEEE Conference on Computer Communications*, pp. 510–519, IEEE, 2022.
- [7] S. Shi and C. Qian, "Concurrent entanglement routing for quantum networks: Model and designs," in *Proceedings of the Annual conference of the ACM Special Interest Group on Data Communication on the applications, technologies, architectures, and protocols for computer communication*, pp. 62–75, 2020.
- [8] C. Qiao, Y. Zhao, G. Zhao, and H. Xu, "Quantum data networking for distributed quantum computing: Opportunities and challenges," in *IEEE INFOCOM 2022-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPs)*, pp. 1–6, IEEE, 2022.
- [9] L. Chen, K. Xue, J. Li, N. Yu, R. Li, J. Liu, Q. Sun, and J. Lu, "A heuristic remote entanglement distribution algorithm on memory-limited quantum paths," *IEEE Transactions on Communications*, vol. 70, no. 11, pp. 7491–7504, 2022.
- [10] A. Farahbakhsh and C. Feng, "Opportunistic routing in quantum networks," in *IEEE INFOCOM 2022-IEEE Conference on Computer Communications*, pp. 490–499, IEEE, 2022.
- [11] J. Li, M. Wang, K. Xue, R. Li, N. Yu, Q. Sun, and J. Lu, "Fidelity-guaranteed entanglement routing in quantum networks," *IEEE Transactions on Communications*, vol. 70, no. 10, pp. 6748–6763, 2022.
- [12] Y. Zhao, G. Zhao, and C. Qiao, "E2e fidelity aware routing and purification for throughput maximization in quantum networks," in *IEEE INFOCOM 2022-IEEE Conference on Computer Communications*, pp. 480–489, IEEE, 2022.
- [13] A. G. Fowler, D. S. Wang, C. D. Hill, T. D. Ladd, R. Van Meter, and L. C. Hollenberg, "Surface code quantum communication," *Physical review letters*, vol. 104, no. 18, p. 180503, 2010.
- [14] S. Muralidharan, J. Kim, N. Lütkenhaus, M. D. Lukin, and L. Jiang, "Ultrafast and fault-tolerant quantum communication across long distances," *Physical review letters*, vol. 112, no. 25, p. 250501, 2014.
- [15] T. Hu, J. Wu, and Q. Li, "Surfacenet: Fault-tolerant quantum networks with surface codes," *IEEE Network*, 2023.
- [16] S.-K. Liao, W.-Q. Cai, J. Handsteiner, B. Liu, J. Yin, L. Zhang, D. Rauch, M. Fink, J.-G. Ren, W.-Y. Liu, *et al.*, "Satellite-relayed intercontinental quantum network," *Physical review letters*, vol. 120, no. 3, p. 030501, 2018.
- [17] E. Diamanti, H.-K. Lo, B. Qi, and Z. Yuan, "Practical challenges in quantum key distribution," *npj Quantum Information*, vol. 2, no. 1, pp. 1–12, 2016.
- [18] S. Wehner, D. Elkouss, and R. Hanson, "Quantum internet: A vision for the road ahead," *Science*, vol. 362, p. eaam9288, oct 2018.
- [19] C. H. Bennett and S. J. Wiesner, "Communication via one- and two-particle operators on einstein-podolsky-rosen states," *Physical review letters*, vol. 69, no. 20, p. 2881, 1992.
- [20] N. K. Panigrahy, M. G. De Andrade, S. Pouryousof, D. Towsley, and L. Tassiulas, "Scalable multipartite entanglement distribution in quantum networks," in *2023 IEEE International Conference on Quantum Computing and Engineering (QCE)*, vol. 2, pp. 391–392, IEEE, 2023.
- [21] N. Yu, C.-Y. Lai, and L. Zhou, "Protocols for packet quantum network intercommunication," *IEEE Transactions on Quantum Engineering*, vol. 2, pp. 1–9, 2021.
- [22] H. Gu, Z. Li, R. Yu, X. Wang, F. Zhou, and J. Liu, "Fendi: High-fidelity entanglement distribution in the quantum internet," *arXiv preprint arXiv:2301.08269*, 2023.
- [23] W. J. Munro, K. Azuma, K. Tamaki, and K. Nemoto, "Inside quantum repeaters," *IEEE Journal of Selected Topics in Quantum Electronics*, vol. 21, no. 3, pp. 78–90, 2015.
- [24] T. Vasantam and D. Towsley, "A throughput optimal scheduling policy for a quantum switch," in *Quantum Computing, Communication, and Simulation II*, vol. 12015, pp. 14–23, SPIE, 2022.
- [25] Y. Lee, E. Bersin, A. Dahlberg, S. Wehner, and D. Englund, "A quantum router architecture for high-fidelity entanglement flows in quantum networks," *npj Quantum Information*, vol. 8, no. 1, p. 75, 2022.
- [26] R. Zhou, X. Lai, Y. Gan, K. Obraczka, S. Du, and C. Qian, "A simulator of atom-atom entanglement with atomic ensembles and quantum optics," in *2023 IEEE International Conference on Quantum Computing and Engineering (QCE)*, vol. 1, pp. 1271–1277, IEEE, 2023.
- [27] Q. Sun, Y. Mao, S. Chen, W. Zhang, Y. Jiang, Y. Zhang, W. Zhang, S. Miki, T. Yamashita, H. Terai, *et al.*, "Quantum teleportation with independent sources over an optical fibre network," *arXiv preprint arXiv:1602.07081*, 2016.
- [28] M. G. De Andrade, J. Diaz, J. Navas, S. Guha, I. Montañó, B. Smith, M. Raymer, and D. Towsley, "Quantum network tomography with multi-party state distribution," in *2022 IEEE International Conference on Quantum Computing and Engineering (QCE)*, pp. 400–409, IEEE, 2022.
- [29] F. Hua, Y. Chen, Y. Jin, C. Zhang, A. Hayes, Y. Zhang, and E. Z. Zhang, "Autobraid: A framework for enabling efficient surface code communication in quantum computing," in *MICRO-54: 54th Annual IEEE/ACM International Symposium on Microarchitecture*, pp. 925–936, 2021.
- [30] A. G. Fowler, "Minimum weight perfect matching of fault-tolerant topological quantum error correction in average $o(1)$ parallel time," *arXiv preprint arXiv:1307.1740*, 2013.
- [31] O. Higgott and C. Gidney, "Sparse blossom: correcting a million errors per core second with minimum-weight matching," *arXiv preprint arXiv:2303.15933*, 2023.
- [32] N. Delfosse and N. H. Nickerson, "Almost-linear time decoding algorithm for topological codes," *Quantum*, vol. 5, p. 595, 2021.
- [33] Google Quantum AI, "Suppressing quantum errors by scaling a surface code logical qubit," *Nature*, vol. 614, no. 7949, pp. 676–681, 2023.
- [34] Y. Kim, A. Eddins, S. Anand, K. X. Wei, E. Van Den Berg, S. Rosenblatt, H. Nayfeh, Y. Wu, M. Zaletel, K. Temme, *et al.*, "Evidence for the utility of quantum computing before fault tolerance," *Nature*, vol. 618, no. 7965, pp. 500–505, 2023.
- [35] D. Gottesman, *Stabilizer codes and quantum error correction*. PhD thesis, California Institute of Technology, 1997.
- [36] A. G. Fowler, M. Mariantoni, J. M. Martinis, and A. N. Cleland, "Surface codes: Towards practical large-scale quantum computation," *Physical Review A*, vol. 86, no. 3, p. 032324, 2012.
- [37] J. Edmonds and E. L. Johnson, "Matching, euler tours and the chinese postman," *Mathematical programming*, vol. 5, pp. 88–124, 1973.
- [38] B. H. Korte, J. Vygen, B. Korte, and J. Vygen, *Combinatorial optimization*, vol. 1. Springer, 2011.
- [39] N. Delfosse and G. Zémor, "Linear-time maximum likelihood decoding of surface codes over the quantum erasure channel," *Physical Review Research*, vol. 2, no. 3, p. 033042, 2020.
- [40] K. Salimifard and S. Bigharaz, "The multicommodity network flow problem: state of the art classification, applications, and solution methods," *Operational Research*, vol. 22, no. 1, pp. 1–47, 2022.
- [41] T. C. Hu, "Multi-commodity network flows," *Operations research*, vol. 11, no. 3, pp. 344–360, 1963.
- [42] J. A. Tomlin, "Minimum-cost multicommodity network flows," *Operations Research*, vol. 14, no. 1, pp. 45–51, 1966.