# Performance Analysis of Interconnection Networks for Distributed Quantum Computing

Yingling Mao*, Yu Liu*, Xu Xu*, Xiaojun Shang†, and Yuanyuan Yang*, *Fellow, IEEE*
*Department of Electrical and Computer Engineering, Stony Brook University, USA
†Department of Applied Mathematics and Statistics, University of Texas at Arlington, USA

***Abstract*—Quantum computing has the potential to solve complicated problems that are impossible for classical servers. Nevertheless, the applications of current quantum processors are restricted by their limited qubit capacity. Distributed Quantum Computing (DQC) is promising to scale up the computing capability by interconnecting quantum processors and performing computing collectively. The network interconnecting quantum processors can impact the efficiency of DQC. In this paper, we analyze and compare the performance of various interconnection networks for DQC. First, we meticulously derive the success probabilities of entanglement generation and the fidelity of shared Bell states generated within three typical static networks: line, ring, and grid. In addition, we propose a switching network with a minimal number of switch stages and evaluate its performance in terms of probability and fidelity. Moreover, we conduct extensive simulations based on real-world parameters to compare the static and switching networks, and the results reveal that the switching network performs better and is more scalable.**

## I. Introduction

Quantum computing has demonstrated significant promise in addressing critical computational challenges, e.g., large integer factorization, that prove highly complex for conventional computing systems [1]. Thus, the development of proficient quantum computing systems has garnered attention from both industrial and academic sectors, such as IBM (Osprey) and Google (Sycamore) [2]. The market size of the global quantum computing industry is projected to surpass 4.456 billion USD by the year 2030 [3]. Despite the considerable potential and rapid development of quantum computing, there remains a significant journey ahead before quantum computers can effectively tackle practical problems. One pressing technological challenge revolves around the restricted number of qubits within a singular processor, constraining the scope of problems quantum processors can effectively tackle. For instance, the decryption of RSA-2048 encryption demands millions of qubits, i.e., the fundamental components of quantum computing akin to classical bits, surpassing the computational capacity of contemporary quantum systems equipped with hundreds or thousands of qubits.

While developing a single Quantum Processor Unit (QPU) with a sufficient number of qubits presents significant challenges, i.e., the pronounced decrease in coherence time observed in superconducting-based quantum systems when scaling up the qubit count [4], numerous experiments have demonstrated the viability of interconnecting multiple mild-scale quantum processors across various platforms through quantum communication methodologies [5]. This leads to the advent of distributed quantum computing (DQC), which offers the potential to expand quantum computing systems by interconnecting QPUs and performing computing collaboratively. In particular, shared entangled qubit pairs between QPUs generated within the interconnection network are exploited to perform remote quantum gate operations applied to qubits on distinct QPUs.

Under the DQC paradigm, the network interconnecting QPUs may impact the computing efficiency [6] via the success probability of entanglement generation between two randomly selected QPUs and the resulting entanglement fidelity. There are two ways to interconnect QPUs: static connection networks and switching networks. Static interconnection networks are fixed, e.g., mesh, grid, ring, and line networks, where static links between QPUs allow shared entangled qubit pair generation between them. In these static networks, entanglement swapping is required to generate shared entangled qubit pairs between two non-directed QPUs. Despite the success of static networks in quantum internet [7], [8], they are not without their limitations. As the network scale increases, static networks may lead to performance degradation in terms of both probability and fertility due to error accumulation via entanglement swapping. On the other hand, QPUs can be interconnected by switching networks using optimal switches [9], which can offer dynamic connections between QPUs by configuring switch states. However, the insertion loss of optimal switches will lead to photon loss during the entanglement generation process and affect the success probability of entanglement generation. In particular, as the number of switch stages increases, the success probability of entanglement generation will decrease, and computing efficiency will decrease. Existing works mainly focus on feasibility analysis or performance improvement of static or switching quantum computing networks without considering potential risks of performance degradation when the network scale increases. Consequently, analyzing and comparing the performance of various interconnection networks for DQC holds significant value.

Hence, in this paper, we first quantitatively analyze the performance of three popular static network topologies for DQC, i.e., line, ring, and grid, regarding the network scalability. Specifically, we calculate the closed-form average success probability of entanglement generation between two randomly selected

This work was supported in part by the National Science Foundation under grant number 2231040.

QPUs, along with the closed-form average entanglement fidelity, within the three static networks. In addition, we construct a multistage switching network employing optical Mach-Zehnder Interferometer (MZI) switches for DQC, with a minimal number of switch stages. We also show the success probability of entanglement generation and the resulting entanglement fidelity in the proposed switching network. Furthermore, we conduct extensive comparisons between static and switching quantum networks, evaluating them in terms of both probability and fidelity based on real-world parameters from experiments. The outcomes indicate that the switching quantum network exhibits better performance and scalability than static networks.

## II. Preliminaries

### A. Remote Quantum Gate

The DQC paradigm promises great potential for quantum computing with increased qubit capacity. However, it also poses the challenge of executing gate operations on qubits located in separate QPUs, namely remote gates. Since all quantum circuits can be implemented using single-quit gates and two-qubit CNOT gates [10], our discussion will focus on the implementation of remote two-qubit CNOT gates. Figure 1 depicts the circuit for implementing a remote CNOT gate on two qubits, $q_1$ and $q_4$, located in different QPUs. These two QPUs have a pair of shared entangled qubits, $q_2$ and $q_3$, with a shared Bell state $|\Phi^+\rangle = \frac{|00\rangle+|11\rangle}{\sqrt{2}}$. After local operations in each QPU, the measurement of the qubits $Q_2$ and $Q_2'$ will determine further operations on $Q_0$ and $Q_0'$, thus realizing the remote CNOT gate.

### B. Herald Entanglement Generation

Since the remote gate on qubits in two QPUs necessitates a pair of shared entangled qubits, we will give a brief introduction to the generation of entanglement between two communication qubits in different QPUs. Since trapped ion quantum computers offer advantages such as high quantum volume, long coherence time, high gate fidelity, and extensive qubit connectivity [11], we will employ the trapped ion here as an example of qubits to elucidate the detailed entanglement generation process, as shown in Figure 2. The Bell State Analyzer (BSA) is a quantum component for detecting the entanglement between qubits. The protocol begins with split laser pulses from the same source simultaneously exciting qubit A in QPU1 and qubit B in QPU2. The successful excitation leads to the generation of two entangled ion-photon pairs, i.e. ion A and photon A, ion B and photon B. These photons are then collected and coupled into fibers directed towards the BSA. After detecting the two photons in the BSA, it will determine if their pattern matches one of the two specific patterns that indicate successful Bell state entanglement, out of the four equally probable patterns. In other words, the probability of a successful Bell state generation is 0.5, given that the two photons are detected by the BSA. Once the expected pattern is detected, the BSA would send a confirmation signal to the QPUs to proceed with the remote gate operations. Otherwise, the BSA would prompt for another entanglement attempt with a retry signal.

TABLE I: Notations

| Definition | Notation |
|---|---|
| Prob. of successful photon generation, collection, and coupling | $p_p$ |
| Prob. of a photon successfully traversing the fiber | $p_f$ |
| Prob. of BSA successfully detecting two photons | $p_d$ |
| Success prob. of Bell pair generation per attempt on each link | $p_a$ |
| Duration of the external phase | $T_e$ |
| Maximum photon production rate | $\gamma_{max}$ |
| Maximum entanglement generation attempts per slot | $n_e$ |
| Success prob. of Bell pair generation on each link along a route | $p_e$ |
| Success prob. of entanglement swapping | $p_d$ |
| Success Prob. of Bell pair generation between S-D QPUs per slot | $p_r$ |
| Average success prob. of Bell pair generation between S-D QPUs | $\bar{p}_r$ |
| Prob. that each qubit of a Bell pair will flip | $p_b$ |

We use $p_p$ to denote the probability that every time we excite an ion, the entangled ion-photon pair is generated successfully, and the photon is successfully collected and coupled into the fiber. The probability of a photon successfully traversing the fiber and reaching the BSA is represented as $p_f$. The overall efficiency of the BSA is quantified by $p_d$, reflecting the chance that the BSA will successfully identify the photon pairs. The aggregate success rate for each entanglement attempt is denoted as $p_a$, as follows:

$$p_a = 0.5 \cdot p_p^2 p_f^2 p_d. \tag{1}$$

### C. Static and Switching Networks

There are two ways to establish interconnections between QPUs: direct connection networks (a.k.a. static networks) and switching networking, as shown in Figure 3. Direct connection networks involve positioning a BSA at the midpoint of each link, as seen in the line network example in Figure 3.a. The second way to interconnect QPUs and BSAs is using optical MZI switches [9], as shown in Figure 3.b, where we use switches to route the photons generated by the QPUs requesting a shared Bell state to the BSA.

## III. Performance Analysis of Static Networks

In this section, we delve into an analysis of the performance of static networks as employed in quantum internet. In particular, important notations are listed in Table I.

### A. Static Networks

In the context of quantum internet [7], [8], a prevalent method for establishing interconnections between QPUs is through the creation of static links between them. This process involves placing a BSA at the midpoint of each link, as illustrated in the line network example depicted in Figure 3.a. Given that mesh topologies, requiring $(N-1)!$ links and $N$ collection modules per QPU, lack essential scalability, non-mesh designs such as line, grid, or ring networks are preferred in such static networks. In these networks, entanglement swapping plays a pivotal role in facilitating the generation of shared Bell states between indirectly linked QPUs. A typical shortest-path-based entanglement routing protocol [7], [8] for generating a shared Bell state between any two selected distinct QPUs, i.e., Source-Destination (S-D) QPUs, in the static topology, is as follows. First, find the shortest path route between the S-D QPUs Then, we will perform entanglement generation along this route. In

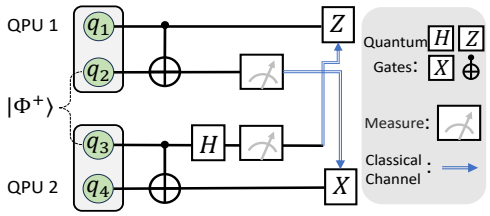


Fig. 1: Remote CNOT Gate.

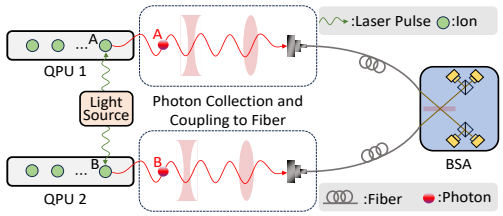


Fig. 2: Herald Entanglement Generation.

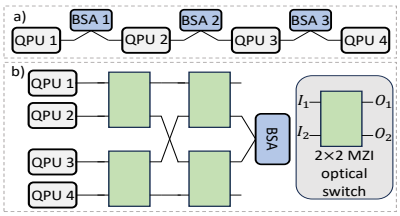


Fig. 3: Static and Switching Net.

particular, the system operates in slotted time, where each time slot must be shorter than the coherence time of the qubits. Each time slot comprises two phases: the external and internal phases. Then, in the external phase, each link along the route independently attempts to generate shared entangled pairs using the protocol outlined in Section II-B. Let $T_e$ be the duration of the external phase, and $\gamma_{max}$ be the maximum photon production rate of each QPU for entanglement generation [12]. Therefore, the maximum number of link-level entanglement generation attempts during the external phase is $n_e \triangleq T_e \cdot \gamma_{max}$. If shared Bell states are successfully generated for all links along the route, we then perform entanglement swapping on each intermediate QPU on the route during the internal phase.

### B. Success Probability Analysis

First, we analyze the probability of successfully generating a shared Bell state on a route with $N$ links between the S-D QPUs within a given time slot, denoted by $p_r(N)$. Let $p_e$ represent the probability of successfully generating a shared Bell state on a link during the external phase, and $p_s$ denote the success probability of the entanglement swapping process at each intermediate QPU. Since there are $N$ links and $N-1$ intermediate QPUs in the route between the S-D QPUs, the success entanglement generation probability at each time slot, denoted by $p_r(N)$, is as follows:

$$p_r(N) = (p_e)^N {p_s}^{N-1}. \tag{2}$$

Here, $p_e = 1-(1-p_a)^{n_e}$, where $p_a$ from (1) represents the success probability of each entanglement generation attempt.

Next, we derive the average probability of successfully generating a shared Bell state between two randomly selected QPUs in three typical static networks: line, ring, and grid.

**Theorem 1.** *The average probability of successfully generating a shared Bell state between two randomly selected QPUs at each time slot, denoted by $\bar{p}_r$, in line, ring, and grid networks are:*

- *Line network with $N$ QPUs*

$$\bar{p}_r = \frac{2p_e}{(N-1)(1-p_e p_s)} - \frac{2p_e[1-(p_e p_s)^N]}{N(N-1)(1-p_e p_s)^2}. \tag{3}$$

- *Ring network with $N$ QPUs*

$$\bar{p}_r = \begin{cases} \frac{2p_e[1-(p_e p_s)^{\frac{N-1}{2}}]}{(N-1)(1-p_e p_s)}, & \text{if } N \text{ is odd;} \\ \frac{p_e[2-(p_e p_s)^{\frac{N}{2}-1}(1+p_e p_s)]}{(N-1)(1-p_e p_s)}, & \text{if } N \text{ is even.} \end{cases} \tag{4}$$

- *$N_1$ by $N_2$ grid network with $N \triangleq N_1 N_2$ QPUs*

$$\bar{p}_r = \frac{4p_e}{(N-1)(1-p_e p_s)^2} + \frac{4p_e^2 p_s[1-(p_e p_s)^{N_1}][1-(p_e p_s)^{N_2}]}{N(N-1)(1-p_e p_s)^4} - \frac{2p_e(1+p_e p_s)[1-(p_e p_s)^{N_1}]}{N_1(N-1)(1-p_e p_s)^3} - \frac{2p_e(1+p_e p_s)[1-(p_e p_s)^{N_2}]}{N_2(N-1)(1-p_e p_s)^3}. \tag{5}$$

*Proof.* Under the line network, we have

$$\begin{aligned} \bar{p}_r &= \sum_{k=1}^{N-1} \frac{(N-k)\cdot p_r(k)}{C(N,2)} = 2p_e \sum_{k=1}^{N-1} \frac{(N-k)(p_e p_s)^{k-1}}{N\cdot(N-1)} \\ &= \frac{2p_e}{N-1} \sum_{k=1}^{\infty} (p_e p_s)^{k-1} - \frac{2p_e(1-(p_e p_s)^N)}{N(N-1)} \sum_{k=1}^{\infty} k(p_e p_s)^{k-1} \\ &\overset{(a)}{=} \frac{2p_e}{(N-1)(1-p_e p_s)} - \frac{2p_e[1-(p_e p_s)^N]}{N(N-1)(1-p_e p_s)^2}, \end{aligned}$$

where $C(N,2)$ is the combination formula, and equation (a) holds because $\sum_{k=1}^{\infty} x^{k-1} = 1/1-x$ and $\sum_{k=1}^{\infty} kx^{k-1} = 1/(1-x)^2$.

Under the ring network, the main idea of calculating (4) is as follows.

$$\bar{p}_r = \begin{cases} \frac{N}{C(N,2)} \sum_{k=1}^{\frac{N-1}{2}} p_r(k), & \text{if } N \text{ is odd;} \\ \frac{N}{C(N,2)} \sum_{k=1}^{\frac{N}{2}-1} p_r(k) + \frac{N}{2C(N,2)} p_r(\frac{N}{2}), & \text{if } N \text{ is even.} \end{cases}$$

Then, by exploiting the finite geometric series formula, we can obtain (4).

Under the $N_1$ by $N_2$ grid network ($N = N_1 N_2$), we have $\bar{p}_r = \frac{2}{C(N,2)} \sum_{i=1}^{N_2-1} \sum_{j=1}^{N_1-1} (N_2-i)\cdot(N_1-j)\cdot p_r(i+j) + \frac{N_2}{C(N,2)} \sum_{j=1}^{N_1-1} (N_1-j)\cdot p_r(j) + \frac{N_1}{C(N,2)} \sum_{i=1}^{N_2-1} (N_2-i)\cdot p_r(i)$. Then, by exploiting the Taylor series, we can get (5). The detailed proof is omitted due to space limitations. □

### C. Bit Flip Errors and Fidelity Analysis

Qubits may experience bit flip errors [13] caused by quantum logic gate operations such as CNOT gate in the entanglement swapping process. Thus, we need to analyze the "closeness" of the obtained and expected states, i.e., the fidelity of the generated state.

During the external phase, we want to generate the shared qubit pair $|\Phi^+\rangle = \frac{|00\rangle+|11\rangle}{2}$ on each link along the route between the S-D QPUs. If the first qubit of a Bell pair $|\Phi^+\rangle$ experiences a bit flip error, the state of the qubit pair changes to $|\Psi^+\rangle = \frac{|01\rangle+|10\rangle}{2}$, which is not our expected one. Assuming each qubit of a Bell pair will flip independently with the same probability $p_b$ [1], the density matrix of the generated qubit pair will be $\rho = (p_b^2 + (1-p_b)^2)|\Phi^+\rangle\langle\Phi^+| + 2p_b(1-$

$p_b)|\Psi^+\rangle\langle\Psi^+| = F|\Phi^+\rangle\langle\Phi^+| + (1-F)|\Psi^+\rangle\langle\Psi^+|$, and its fidelity is $F = p_b^2 + (1-p_b)^2$.

Next, we consider the entanglement-swapping process during the internal phase. If the states of generated Bell pairs over two adjacent links are both ideal $|\Phi^+\rangle$, the final shared Bell state after an entanglement-swapping process on the intermediate QPU is the expected $|\Phi^+\rangle$, i.e., $|\Phi^+\rangle \otimes |\Phi^+\rangle \xrightarrow{\text{entg-swap}} |\Phi^+\rangle$. But considering the bit flip errors, the state of each generated Bell pair may be $|\Psi^+\rangle$ rather than $|\Phi^+\rangle$. Thus, it is necessary to consider the following two cases further. (1) If the states of generated Bell pairs over two adjacent links are both $|\Psi^+\rangle$, the final shared Bell state after entanglement swapping is still the expected $|\Phi^+\rangle$, i.e., $|\Psi^+\rangle \otimes |\Psi^+\rangle \xrightarrow{\text{entg-swap}} |\Phi^+\rangle$. (2) If one state of generated Bell pairs over two adjacent links is $|\Phi^+\rangle$ and the other is $|\Psi^+\rangle$, the final shared Bell state after entanglement swapping is $|\Psi^+\rangle$, i.e., $|\Phi^+\rangle \otimes |\Psi^+\rangle \xrightarrow{\text{entg-swap}} |\Psi^+\rangle$ and $|\Psi^+\rangle \otimes |\Phi^+\rangle \xrightarrow{\text{entg-swap}} |\Psi^+\rangle$.

If the fidelity of the generated Bell pairs over two adjacent links are respectively $F_A$ and $F_B$, the fidelity of the final shared Bell state after entanglement swapping is

$$F_{new} = F_A \cdot F_B + (1-F_A)\cdot(1-F_B). \tag{6}$$

For quantitative analysis, we assume the fidelity of all the Bell pairs generated over each link in the static network (without any purification) is identical, noted as $F$, and their density matrix is $\rho = F|\Phi^+\rangle\langle\Phi^+| + (1-F)|\Psi^+\rangle\langle\Psi^+|$. To reveal the fidelity of the generated shared Bell state between the S-D QPUs with $N-1$ intermediate QPUs, noted as $f_N$, we first show Lemma 1.

**Lemma 1.** The order of each entanglement swapping process over the intermediate QPUs does NOT impact the final shared Bell state between the S-D QPUs.

The proof is omitted due to space limitations. The main idea of the proof is to employ mathematical induction. Based on Lemma 1, we perform entanglement swapping on each intermediate QPU following the order from source to destination along the route and compute the fidelity of the resulting shared Bell state between the S-D QPUs of this route, i.e., $f_N$. According to (6), we can obtain $f_N = F \cdot f_{N-1} + (1-F)\cdot(1-f_{N-1})$. Solving this recursion formula, we can reach the conclusion that

$$f_N = 2^{N-1}\cdot(F-\frac{1}{2})^N + \frac{1}{2}.$$

Based on it, we can prove the average fidelity of the shared Bell generated between two randomly selected QPUs.

**Theorem 2.** *The average fidelity of the successfully generated shared Bell state between two randomly selected QPUs at each time slot, denoted by $\bar{F}$, in line, ring, and grid networks are:*

- *Line network with $N$ QPUs*

$$\bar{F} = \frac{2F-1}{2(N-1)(1-F)} - \frac{(2F-1)[1-(2F-1)^N]}{4N(N-1)(1-F)^2} + \frac{1}{2} \tag{7}$$

- *Ring network with $N$ QPUs*

$$\bar{F} = \begin{cases} \frac{(2F-1)[1-(2F-1)^{\frac{N-1}{2}}]}{2(N-1)(1-F)} + \frac{1}{2}, & \text{if } N \text{ is odd;} \\ \frac{(2F-1)[1-F(2F-1)^{\frac{N}{2}-1}]}{2(N-1)(1-F)} + \frac{1}{2}, & \text{if } N \text{ is even;} \end{cases} \tag{8}$$

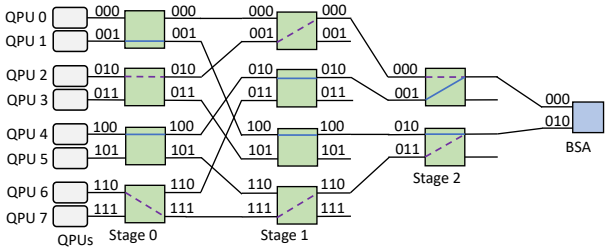


Fig. 4: 8-input QTS.

- *$N_1$ by $N_2$ grid network with $N \triangleq N_1 N_2$ QPUs*

$$\bar{F} = \frac{2F-1}{2(N-1)(1-F)^2} - \sum_{k\in\{N_1,N_2\}} \frac{F(2F-1)[1-(2F-1)^k]}{4k(N-1)(1-F)^3} + \frac{(2F-1)^2[1-(2F-1)^{N_1}][1-(2F-1)^{N_2}]}{8N(N-1)(1-F)^4} + \frac{1}{2}. \tag{9}$$

The proof of Theorem 2 is omitted due to space limitations.

## IV. Performance Analysis of Switching Networks

In this section, we design a switching network for DQC, and analyze its performance.

### A. Switching Network Design

We consider building a multistage switching network that utilizes $2\times 2$ binary switches. Our goal is centered on creating a network that interconnects (any two of the) $N$ QPUs using a single BSA, where each QPU has a singular output, and the BSA has two inputs, as illustrated in Figure 2.

To switch any pair of QPUs to the BSA, we propose a structure called Quantum Tree-Like Switching (QTS). Figure 4 shows an instance of an 8-input QTS. The $N$-input QTS consists of $\log(N)$ stages of switches. In particular, stage 0 and stage 1 have $N/2$ switches, and stage $n>1$ has $N/2^n$ switches. The two input ports of the $i$-th switch in each stage are labeled by $2i$ and $2i+1$. Similarly, The two output ports of the $i$-th switch in each stage are labeled by $2i$ and $2i+1$. Next, we illustrate the interconnections between the QPUs, switching stages, and BSAs. For any given $i \in [N]$, QPU $i$ is connected to input port $i$ of stage 0. Then, the $i$-th output port of stage 0 is connected to the $S_r(i,N)$-th[1] input port of stage 1. For instance, in Figure 4, output ports $0,1,\cdots,7$ of stage 0 are connected to input ports $(S_r(0,8), S_r(1,8),\cdots,S_r(7,8)) = (0,4,1,5,2,6,3,7)$ of stage 1, respectively. For each stage $j \in \{1,2,\cdots,\log(N)-1\}$, the $2i$-th output port connects to the $i$-th input port of the next stage (or BSA stage if $j=\log(N)-1$), and the $(2i+1)$-th output port is idle and connects to nowhere. For example, in Figure 4, output ports $(0,2,4,6)$ of stage 1 are connected to input ports $(0,1,2,3)$ of stage 2, and output ports $(1,3,5,7)$ of stage 1 are idle.

The routing algorithm for this QTS system is straightforward. Assuming a QPU pair denoted as $(i,j)$ is requesting a shared Bell state. Then, for stage 0, if both $i$ and $j$ are either even

[1] $S_r(i,N)$ is the right circular shift operation. For an integer $i$, $S_r(i,N)$ involves converting $i$ into its binary representation of $\log(N)$ bits length, then performing a right circular shift.

or odd, route one input to the upper output port of the corresponding switch, and the other input to the lower output port of the corresponding switch. Otherwise, route inputs $i$ and $j$ to output ports $i$ and $j$, respectively. For stage $n > 0$, route each input to the upper output port of the switch. For example, the routing of the QPU pair $(1, 4)$ is represented by solid lines in Figure 4. In stage 0, since 1 is odd and 4 is even, input 1 is routed to output port 1, and input 4 is routed to output port 4. In stage 1 and 2, the inputs are all routed to the upper output port in the corresponding switch. The dashed lines in Figure 4 represent the routing decisions for the QPU pair $(2, 6)$. In stage 0, input 2 is routed to the output port 2, and input 6 is routed to the output port 7 because they are both even numbers. The inputs are then routed to the upper output ports, as in the case above. Note that QTS can only support one pair of QPUs requesting a shared Bell state at any given time. This limitation of sequential entanglement generation is manageable for DQC applications because quantum gate operations usually require applying to qubits in a specific sequence. Consequently, we can sequentially generate shared qubit pairs for remote gates. Next, we show $\log_2(N)$ is the minimum number of stages.

**Theorem 3.** *To connect $N = 2^n$ QPUs with a BSA using $2 \times 2$ switches such that any two inputs can be routed to the BSA, the minimum number of switch stages required is $\log_2(N)$.*

The proof of Theorem 3 is omitted due to space limitations. The main idea of the proof is to show that the network with the fewest stages that can connect $N$ QPUs with one BSA is a tree network with $\log_2(N) - 1$ stages, which cannot ensure that every pair of inputs can be routed the BSA. In the QTS, the connectivity pattern between the initial two stages (stages 0 and 1) is aligned with that of the Beneš network with $2\log_2(N) - 1$ stages. Specifically, each switch in stage 0 is linked to one switch on the upper side and another on the lower side in the subsequent stage. The network structure after stage 1 is a tree network, that is why we call it a tree-like network.

### B. Performance Analysis

Next, we analyze the success probability of each entanglement generation attempt in the proposed QTS. Let $p_i$ denote the probability of a photon successfully passing through a switch, determined by the switch's insertion loss. For a QPU pair requesting a shared Bell state, each of their emitted two photons must traverse through $\log_2(N)$ switches. The probability that both photons successfully reach the BSA is given by $p_i^{2\log_2(N)}$. Thus, the success probability of each entanglement generation attempt, denoted by $p_a^\star$, is as follows:

$$p_a^\star = 0.5 p_p^2 p_f^2 p_i^{2\log_2(N)} p_d. \tag{10}$$

The probability of successfully generating at least one entangled pair over $n$ attempts, denoted by $p_r^\star(n)$, is expressed as:

$$p_r^\star(n) = 1 - (1 - p_a^\star)^n. \tag{11}$$

Since no entanglement swapping in the intermediate QPU is involved in the switching network, there is no fidelity degradation caused by entanglement swapping. Assuming there is only bit-flip error, the fidelity of the generated shared qubit pair, denoted by $F_{QTS}$, is a function of the bit-flip probability as follows:

$$F_{QTS} = p_b^2 + (1 - p_b)^2. \tag{12}$$

## V. Performance Evaluation

In this section, we compare the performance of the three static networks and the proposed switching network.

### A. Simulation Settings

Based on [15], we assume the external phase of the static networks lasts for $T_e = 2.5$ microseconds. Based on [12], we set the maximum photon production rate $\gamma_{max}$ as 2 megahertz. That is, we set $n_e$ as $\gamma_{max} \cdot T_e = 5000$. For fairness, we compare the performance of the three static networks in each time slot with $n_e = 5000$ and that of the proposed switching network with $n = 5000$ entanglement generation attempts. Since entanglement swapping in the static networks takes tens of microseconds [16], we neglect the duration of the internal phase, which favors the static networks. As reported in [17], a typical value of $p_p$ is $0.021$. Based on [18], [19], the insertion loss can be as low as $-0.1$ dB ($p_i = 0.977$) and $-0.35$ dB ($p_i = 0.922$), and in this paper, we evaluate the performance of QTS under $p_i = \{0.977, 0.922, 0.891\}$. Based on [12], [20], we set $p_f = 0.98$, $p_d = 0.8$, and $p_s = 0.95$. In terms of the grid network with $N = 2^n$ QPUs, we set $N_1 = 2^{\lceil \frac{n}{2} \rceil}$ and $N_2 = 2^{\lfloor \frac{n}{2} \rfloor}$.

### B. Simulation Results

We first compare the average success probability of generating a shared Bell state between two randomly selected QPUs at each time slot with $n_e = 5000$ in the three statics networks and that in the proposed switching network with $n = 5000$ and $p_i = \{0.891, 0.922, 0.977\}$, as shown in Figure 5. Among the three static networks, the grid network performs the best while the line network performs the worst, indicating that network topology significantly influences static network performance. Regarding the QTS network, as $p_i$ increases, the average success probability increases. In addition, the average success probability under a larger $p_i$ decreases more slowly as $N$ increases. This is because the larger $p_i$ means the smaller switch's insertion loss. Besides, compared with the static networks, the QTS network generally performs better. Although for small network scales $N$ and $p_i$, i.e., $N = 4$ and $p_i = 0.891$, the static networks exhibit superior average success probability performance compared to the QTS network, the average success probability downtrend observed in our QTS network is considerably smaller than those observed in the three static networks. This discrepancy arises from the fact that the downtrends in average success probability for these static networks are all mainly governed by $O(\frac{1}{N})$, whereas the downtrend in our QTS network is determined by $O(p_i^{2\log_2(N)}) = O(\frac{1}{N^{-2\log_2(p_i)}})$. That is, as $N$ increases, our proposed QTS demonstrates better performance compared to the three static networks when $p_i \geq 1/\sqrt{2}$, which is generally ture [18], [19]. Consequently, QTS holds greater potential than the static networks for large-scale DQC.

Next, we compare the average fidelity of the successfully generated shared Bell state between two randomly selected QPUs at each time slot in the three statics networks and that in the proposed switching network under $F = \{0.99, 0.9\}$,

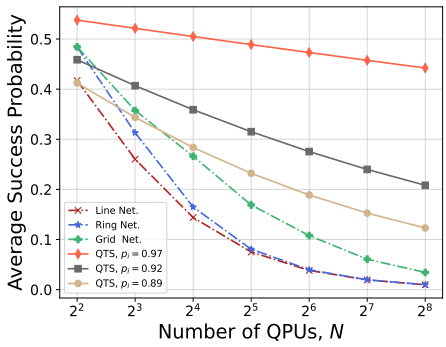


Fig. 5: Success probability of entanglement generation vs. number of QPUs.

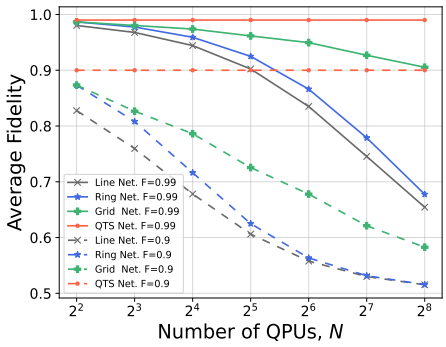


Fig. 6: Average fidelity of generated shared Bell states vs. number of QPUs.

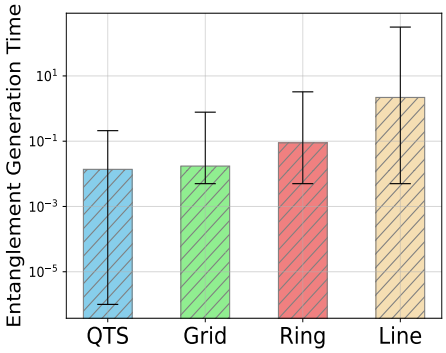


Fig. 7: Average entanglement generation time for a real-world circuit from [14].

as shown in Figure 6. The figure demonstrates that our QTS network consistently outperforms three static networks in terms of average fidelity. As $N$ increases, the average fidelity of the three static networks decreases, whereas that of QTS remains unchanged. This superiority primarily stems from the absence of fidelity degradation caused by entanglement swapping in the QTS network, unlike in static networks. Consequently, the switching network exhibits superior fidelity performance compared to static networks.

In addition, we compare the average time required for a successful entanglement pair generation in the three static networks and the proposed switching network. In particular, compare the average time in the networks for a real-world quantum circuit from [14] with 16 qubits. For static networks, we randomly map the 16 logical qubits to 16 QPUs in line, ring, and grid networks, assuming the physical qubits in each QPU represent a single logical qubit for quantum error correction. We employ the entanglement routing protocol outlined in [8] for static networks, enabling simultaneous entanglement generation with one, two, and up to four routes between S-D QPU pairs in line, ring, and grid networks, respectively. The remote CNOT gates are generated sequentially based on their positioning within the quantum circuit. Figure 7 shows that the average entanglement generation times within the grid, ring, and line networks are $2.0\times$, $7.3\times$, and $1.5e{+}02\times$ of that in QTS, respectively. The black error bars show the minimum and maximum entanglement generation times.

## VI. Conclusion

Distributed quantum computing is a promising paradigm to increase the scale of quantum computing systems. In this paper, we analyze the performance of three static networks and a proposed switching network regarding the success probability of entanglement generation and the fidelity of generated shared entangled qubit pairs. In addition, we conducted extensive real-world data-driven simulations. Results show that the proposed switching network demonstrates better scalability. Our future work will be proposing multistage switching networks that can generate shared qubit pares between QPUs simultaneously.

## References


[1] M. A. Nielsen and I. L. Chuang, *Quantum computation and quantum information*. Cambridge university press, 2010.

[2] L. Hour, S. Heng, S. Heng, M. Go, and Y. Han, "Context-aware coupler reconfiguration for tunable coupler-based superconducting quantum computers," *arXiv preprint arXiv:2401.03817*, 2024.

[3] MarketDigits, "Quantum computing market," https://www.marketdigits.com/quantum-computing-market, accessed: 204-03-31.

[4] C. D. Bruzewicz, J. Chiaverini, R. McConnell, and J. M. Sage, "Trapped-ion quantum computing: Progress and challenges," *Applied Physics Reviews*, vol. 6, no. 2, p. 021314, 2019.

[5] L. Stephenson, D. Nadlinger, B. Nichol, S. An, P. Drmota, T. Ballance, K. Thirumalai, J. Goodwin, D. Lucas, and C. Ballance, "High-rate, high-fidelity entanglement of qubits across an elementary quantum network," *Physical review letters*, vol. 124, no. 11, p. 110501, 2020.

[6] Y. Mao, Y. Liu, and Y. Yang, "Qubit allocation for distributed quantum computing," in *IEEE INFOCOM 2023-IEEE Conference on Computer Communications*. IEEE, 2023, pp. 1–10.

[7] S. Shi and C. Qian, "Concurrent entanglement routing for quantum networks: Model and designs," in *ACM SIGCOMM*, 2020, pp. 62–75.

[8] M. Pant, H. Krovi, D. Towsley, L. Tassiulas, L. Jiang, P. Basu, D. Englund, and S. Guha, "Routing entanglement in the quantum internet," *npj Quantum Information*, vol. 5, no. 1, p. 25, 2019.

[9] M. Dong, M. Zimmermann, D. Heim, H. Choi, G. Clark, A. J. Leenheer, K. J. Palm, A. Witte, D. Dominguez, G. Gilbert *et al.*, "Programmable photonic integrated meshes for modular generation of optical entanglement links," *npj Quantum Information*, vol. 9, no. 1, p. 42, 2023.

[10] A. Barenco, C. H. Bennett, R. Cleve, D. P. DiVincenzo, N. Margolus, P. Shor, T. Sleator, J. A. Smolin, and H. Weinfurter, "Elementary gates for quantum computation," *Physical review A*, vol. 52, no. 5, p. 3457, 1995.

[11] T. Harty, D. Allcock, C. J. Ballance, L. Guidoni, H. Janacek, N. Linke, D. Stacey, and D. Lucas, "High-fidelity preparation, gates, memory, and readout of a trapped-ion quantum bit," *Physical review letters*, vol. 113, no. 22, p. 220501, 2014.

[12] J. Hannegan, J. D. Siverns, J. Cassell, and Q. Quraishi, "Improving entanglement generation rates in trapped-ion quantum networks using nondestructive photon measurement and storage," *Physical Review A*, vol. 103, no. 5, p. 052433, 2021.

[13] Z. Chen, K. J. Satzinger, and *et al.*, "Exponential suppression of bit or phase errors with cyclic error correction," *Nature*, vol. 595, no. 7867, p. 383–387, Jul. 2021.

[14] L. Burgholzer, "Mqt qmap - a tool for quantum circuit compilation," Available at: https://github.com/cda-tum/mqt-qmap, 07 2023, accessed: 2023-07-30.

[15] P. Drmota, D. Main, D. Nadlinger, B. Nichol, M. Weber, E. Ainley, A. Agrawal, R. Srinivas, G. Araneda, C. Ballance *et al.*, "Robust quantum memory in a trapped-ion quantum network node," *Physical Review Letters*, vol. 130, no. 9, p. 090803, 2023.

[16] P. Gerbert and F. Rueß, "The next decade in quantum computing and how to play," *Boston Consulting Group*, p. 5, 2018.

[17] A. VanDevender, Y. Colombe, J. Amini, D. Leibfried, and D. J. Wineland, "Efficient fiber optic detection of trapped ion fluorescence," *Physical review letters*, vol. 105, no. 2, p. 023001, 2010.

[18] F. Ruf, L. Nielsen, M. Balauroiu, N. Volet, and M. J. Heck, "Low-loss all-optical ns-switching for single-photon routing in scalable integrated quantum photonics," in *Integrated Photonics Platforms II*, vol. 12148. SPIE, 2022, pp. 23–29.

[19] X. Wang and S. Mookherjea, "Voltage-controlled fast and low-loss single-photon cross-bar switch using integrated photonics," *Journal of Lightwave Technology*, vol. 41, no. 3, pp. 855–864, 2023.

[20] W. Dai, A. Rinaldi, and D. Towsley, "Entanglement swapping in quantum switches: Protocol design and stability analysis," *arXiv preprint arXiv:2110.04116*, 2021.