Calculating the Capacity Region of a Quantum Switch

Ian Tillman,[1,*] Thirupathaiah Vasantam,[2] Don Towsley,[3] and Kaushik P. Seshadreesan,[4]

[1]*College of Optical Sciences*, *University of Arizona*, Tucson, USA
[2]*Department of Computer Science*, *Durham University*, Durham, UK
[3]*Manning College of Information & Computer Sciences*, *University of Massachusetts Amherst*, Amherst, USA
[4] *School of Computing & Information*, *University of Pittsburgh*, Pittsburgh, USA
[*]ijtillman@arizona.edu

2024 IEEE International Conference on Quantum Computing and Engineering (QCE) | 979-8-3315-4137-8/24/$31.00 ©2024 IEEE | DOI: 10.1109/QCE60285.2024.00216

***Abstract*—Quantum repeaters are necessary to fully realize the capabilities of the emerging quantum internet, especially applications involving distributing entanglement across long distances. A more general notion of this can be called a quantum switch, which connects to many users and can act as a repeater to create end-to-end entanglement between different subsets of these users. Here we present a method of calculating the capacity region of both discrete- and continuous-variable quantum switches that in general support mixed-partite entanglement generation. The method uses tools from convex analysis to generate the boundaries of the capacity region. We show example calculations with illustrative topologies and perform simulations to support the analytical results.**

***Index Terms*—quantum switch, quantum repeater, entanglement distribution, maximum weight scheduling, quantum continuous variables, quantum discrete variables**

I. Introduction

A major step in the world of classical computing was the idea to design devices whose sole purpose was connecting many computing devices to each other [1]. This naturally progressed to entire infrastructures of such networking devices that now, thanks to research done on fiber optics, routers, and switching algorithms, allow for near instantaneous communication between computing devices thousands of kilometers apart [2], [3]. The internet is the backbone of many aspects of our day-to-day life, including secure online banking, social media, and remote education. Quantum 2.0 technologies such as quantum computers, quantum sensor networks, and quantum simulators are in early stages of design and development, but in order to fully harness their capabilities we will need to dedicate ourselves to similar goals of interconnecting these devices and creating a quantum internet [4]–[6]. The fundamental resource that is created and shared in a quantum network is entanglement. Some applications, quantum sensor networks in particular, will rely on many memories being able to generate entanglement between various subsets of themselves [7], [8]. This is analogous to a switch in classical computing, where one hub node connects to many end users and facilitates requests to send information between any specified set of users. A quantum switch will rely on the users entangling themselves with the hub node first, followed up by the hub performing a 'swap' operation that consumes the user-switch entanglements and creates the requested end-to-end entangled state between users [9], [10]. Much like a classical switch, a quantum switch will be impacted by both the entanglement request rate from the users connected to it as well as the scheduling policy it adopts in order to service these requests. However quantum switches will also be impacted by the often probabilistic nature of entanglement generation and manipulation, something that is effectively not present in classical networking. One condition commonly imposed on switches, both classical and quantum, is the concept of *stability* – a term with many closely related definitions that all aim to give a quantitative way to test whether a switch is able to keep up with the incoming requests [10], [11].

In concept, a quantum network can be just as general as a classical network—they are envisioned to facilitate the exchange of quantum information over local, metropolitan, or global scales [12]–[15]. Quantum switches together with quantum repeaters are sufficient to enable arbitrary topologies for quantum networks. The bulk of what these quantum networks will be useful for is distributing entanglement between quantum memories, as most quantum-enhanced activities are based on utilizing the non-classical correlations that come from quantum entanglement. Devices and protocols have been theorized and tested for many different platforms of entanglement, many of which are designed with just one particular use case in mind. However there are still many fundamental, platform and application independent questions to be answered in this field, and so there are many groups that study quantum networks from as general of a perspective as they can [16]. Many of these questions deal with bounds on the *capacity* of a quantum channel, a term which normally means the maximum rates at which a network can create entanglement. For networks which can create many different entanglements this is extended to a *capacity region*, where we consider what possible combinations of rates are possible. In these scenarios it is often unclear how to define the 'optimal' rate vector, so in general it is more useful to characterize the entire region and then let the application decide what vector within the region best suites its needs. Some work has been done to

IT thanks NSF grant 1941583 and ORNL grant 4000178321 and KPS thanks NSF grant 2204985 and Cisco Systems.

979-8-3315-4137-8/24/$31.00 ©2024 IEEE
DOI 10.1109/QCE60285.2024.00216

characterize the capacity region of quantum switches [17]–[21], however these characterizations are often not conducive to certain calculations. For example, we cannot easily verify whether a given rate vector is in the capacity region or not. Motivated by this, in the present work we showcase a method to analytically calculate the capacity region of a single quantum switch that is allowed to use general swap operations to create entanglement between any subset of the many users connected to it. We do this by algorithmically generating the smallest set of linear inequalities that is necessary and sufficient for a request rate vector to be in the capacity region. We use elementary convex analysis to derive our algorithm and then present example calculations alongside simulation data that supports our analytic results.

The manuscript is organized as follows: We start with some background on quantum repeaters, quantum switches, and convex analysis in Section II. In Section III we describe our model for a quantum switch, discuss simulating the stability of a quantum switch, and explain the complications that arise when using continuous variable encodings. In Section IV we describe the algorithm to generate the capacity region of a quantum switch. Section V contains example calculations and discussions on how to interpret our results. Finally, in Section VI, we bring up a particular problem that can be solved faster than expected and then discuss the complexity of our algorithm

II. Background

A. Related Work

Much focus of quantum networking research today is spent on quantum repeaters, devices that are used to extend the range of quantum entanglement distribution. In classical optical networking one can place a repeater node at any point along a long link to read in a lossy, noisy signal and send a restored copy. This alleviates the user on the other end of having to deal with noise, and in some cases the signal would be unreadable at the other end without the repeater. When a repeater can successfully restore the signal, multiple repeaters can be chained to extend a channel's usable distance even more. This is, for example, how we are able to communicate reliably with fiber optic cables under the oceans even though we expect roughly $0.2\ \mathrm{dB/km} \times 5500\ \mathrm{km} = 1100\ \mathrm{dB}$ of loss in optical power between New York City and London. Unfortunately a quantum analog of the classical optical repeater is impossible for quantum communication channels because of the No-Cloning theorem [22]. The theorem states that it is impossible to make a copy of an arbitrary quantum state, which is exactly what a repeater would need to do. Because of this, researchers have looked for ways to effectively amplify quantum signals that get around this.

It has been proven [23] that there is a bound on the generation rate, R, of quantum entanglement that can be shared between two users separated by a channel with transmissivity η that does not contain a repeater:

$$R \leq C(\eta) = -\log_2(1-\eta) \ \ \text{ebits/mode} \tag{1}$$

where C is the channel capacity and 'ebit' stands for entangled bit. Note that as $\eta \to 0$ we have $C(\eta) \sim \eta$. However, some repeater designs have been shown to support capacities that scale as $\sqrt{\eta}$, the half-channel loss, when the repeater is placed between Alice and Bob. Often these repeater protocols perform poorly when $\eta \approx 1$ due to inherently probabilistic operations that can be avoided with direct transmission. However these protocols are designed to exhibit a slower drop in capacity as transmissivity decreases, so there will be a critical transmissivity (i.e. channel distance) where the repeater protocol is able to generate more ebits per mode than a direct transmission protocol.

Many different approaches have been taken to implement repeaters [24]–[27], however outside of one specific architecture that requires special care in our model we will not discuss how repeaters have been modelled or implemented. The one repeater design we will briefly look at uses the fact that the No-Cloning theorem only disallows deterministic amplification and says nothing about probabilistic protocols. One class of probabilistic methods is known as noiseless linear amplification (NLA) which seeks to probabilistically amplify an incoming quantum-limited state without amplifying its noise [28], [29]. Unfortunately it has been proven that ideal NLA must have a probability of success equal to zero, so the best one can hope to do is approximate the action of NLA [30]. One example of this is the Quantum Scissor (QS) [31]–[33] which approximates NLA by performing projection measurements. The QS is the basis of the continuous variable repeater model used in this paper, which we describe in detail in Section III-C.

B. Quantum Switches

A natural extension to the idea of a quantum repeater is a quantum switch, which performs the same operation as a repeater but there are many users connected to it instead of just two. This is analogous to a classical switch or router, where many users connect to one device that can route information between any pair of them. In the quantum scenario, one of the models of a quantum switch considers end users making requests to generate end-to-end entanglement rather than simply routing a packet of data between users as in a classical scenario. Because the switch is responsible for generating entanglement between two end users, the physical model or implementation of a specific quantum switch will differ for different entanglement platforms. For example a request for a Bell state can be serviced with atomic ensembles [34] while a request for a continuous variable entanglement may require devices like two-mode squeezed vacuum (TMSV) generators, homodyne detection, and distillation schemes [35], [36].

Regardless of the underlying model, we can reduce our analysis to a set of users and a set of end-to-end states that subsets of users can request. We call the entanglements between the users and the switch *elementary entanglements*, and we call the operations that consume elementary entanglements to create end-to-end states *swap operations*. We refer to the switch creating an end-to-end entanglement as the switch

servicing an entanglement flow, with each possible end-to-end state being its own flow. We then use two sets of parameters: the probabilities that a given user generates an elementary entanglement with the switch, and the probability that the switch successfully creates a given end-to-end entanglement provided that all necessary elementary entanglements exist. With these parameters we can analyze the performance of a quantum switch, such as the end-to-end state generation rates it can support, while being agnostic to its underlying physical platform. Some work has been done to characterize these models, mostly concerning the scheduling policies one should use to handle competing requests [10], [17]. These papers also characterize the set of request rates that are supportable by the switch, a set called the *capacity region*. Specifically, it was shown that any request rate vector in the capacity region can be decomposed into a certain form which we also use here. However, none of them provided a direct method for calculating the boundaries of the region. Knowing the boundaries would, for example, make it easy to determine whether or not a specific request rate vector will be supportable by the switch. Two current methods for solving this problem involve either partitioning [17] or optimization [37], neither of which can be done efficiently.

C. Extreme Point Analysis

Our method of calculating the capacity region of a quantum switch uses ideas from convex analysis, so we introduce the necessary tools here. A set S is said to be *convex* if $x_1, x_2 \in S \Rightarrow tx_1 + (1-t)x_2 \in S$ for all $0 < t < 1$. A convex combination of points $x_1, \cdots, x_n$ is expressed as $\sum_{k=1}^{n} \alpha_k x_k$ where $\alpha_k > 0$ and $\sum_{k=1}^{n} \alpha_k = 1$. We can further define $\mathrm{cch}(S)$ to be the set of all convex combinations of elements of S, known as the closed convex hull of S. A point $e \in S$ is said to be an *extreme point* of S if it cannot be expressed as a convex combination of two or more unique points in S, i.e. $e = \sum_{k=1}^{n} \alpha_k x_k \Rightarrow x_1 = x_2 = \cdots = x_n$. Define $\mathcal{E}(S)$ to be the set of extreme points of S. The Krein-Milman Theorem says that a compact convex set can be fully recreated by the closed convex hull of its extreme points [38], $S = \mathrm{cch}\left(\mathcal{E}(S)\right)$ for any compact convex set S. This implies that any point in S can be expressed as a convex combination of extreme points. The following theorem will be useful in later sections:

Theorem 1. *Let $L : S \to S'$ be a linear transformation acting on a compact convex S, then $\mathcal{E}\left(L(S)\right) \subseteq L\left(\mathcal{E}(S)\right)$*

In other words, if we know all extreme points of S then we have a way of generating a set containing all extreme points of $L(S)$. For completeness, we include a proof of this fact here.

Proof. Start with an extreme point $e' \in \mathcal{E}\left(L(S)\right)$ and write it as a convex combination of transformed extreme points of S. Say that $L(x_0) = e'$ and $x_0 = \sum_{k=1}^{n} \alpha_k e_k$, where $e_k \in \mathcal{E}(S)$ for $1 \leq k \leq n$ and $\sum_{k=1}^{n} \alpha_k = 1$.

$$e' = L\left(\sum_{k=1}^{n} \alpha_k e_k\right) = \sum_{k=1}^{n} \alpha_k L(e_k) \tag{2}$$

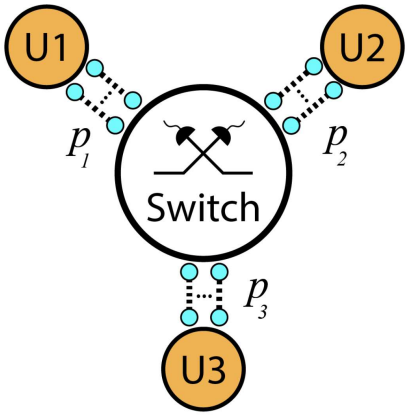


Fig. 1: An example of the underlying Physical topology we consider. On top of this is a virtual topology containing the M different flows the switch is capable of generating.

Because $L(e_k) \in L(S)$, we have written an extreme point of $L(S)$ as a convex combination of points in $L(S)$. This means that $L(e_1) = L(e_2) = \cdots = L(e_n)$, implying that we can replace $L(e_k)$ with $L(e_1)$ for $2 \leq k \leq n$:

$$e' = \sum_{k=1}^{n} \alpha_k L(e_1) = L(e_1) \tag{3}$$

proving that $e' \in L\left(\mathcal{E}(S)\right)$. □

If $L(S)$ is a subset of $\mathbb{R}^n$ and has finitely many extreme points we can use an algorithm like QuickHull [39] to reduce $L\left(\mathcal{E}(S)\right)$ to $\mathcal{E}\left(L(S)\right)$, stripping away all non-extreme points and allowing us to easily define the planar/hyperplanar boundaries of $L(S)$ using an algorithm like Delaunay Triangulation [40].

III. Model

A. Quantum Switch Model

Our quantum switch model is based on a hub-and-spoke topology, with the switch acting as the central hub node and each end users' one or many memories being the spokes. Let K be the number of end users, D_i be the number of memories at user i, $\mathbf{D}$ be a $K \times 1$ vector whose ith component is D_i, and M be the number of entanglement flows the switch supports. D_i can be thought of as the degree of spatial multiplexing that user i implements. Associated with each user memory there is a memory at the switch. Fig. 1 shows an example of this physical topology we consider. Let $\mathbf{M} = [\mathbf{M}_{i,j}]$, where $\mathbf{M}_{i,j}$ is the number of user i elementary links needed to service flow j. During each timestep the switch and users perform the following actions:

1) Each unoccupied user memory attempts to generate entanglement with its associated switch memory
2) The switch uses the information of which elementary links currently exist to make a scheduling decision
3) The switch services the flow(s) it chose to serve by attempting entanglement swap operations

Notation	Definition
λ_j	Request rate for flow j
λ_j^*	Servicing rate for flow j
$\boldsymbol{\lambda}$	Vector of request rates
$\boldsymbol{\lambda}^*$	Vector of servicing rates
Λ	Capacity region
Λ^*	Servicing region
M	Number of flows the swtich can service
q_j	Swap success probability for flow j
$\mathbf{Q}$	Diagonal matrix of swap success probabilities
K	Number of users connected to the switch
p_i	Success probability for user i's elementary links
D_i	Number of elementary links owned by user i
$\mathbf{D}$	Vector of D_i values
$\boldsymbol{\pi}$	Matching vector
$\boldsymbol{\pi}^*$	Matrix of matching vectors
$\mathcal{M}$	Set of all valid matchings
$\mathbf{M}_{i,j}$	Number of user i's elementary links necessary to service flow j
$\mathbf{M}$	Matrix of $\mathbf{M}_{i,j}$ values
$\mathbf{a}$	Arrangement of elementary links
$\mathcal{A}$	Set of all possible elementary link arrangements
$\mathbf{P}$	Vector of arrangement probabilities
$c_{i,j}$	Servicing probability for flow i given arrangement j
$\mathbf{c}$	Matrix of all servicing probabilities, $c_{i,j}$
C	Set of all possible servicing matrices
$Q_i(n)$	Length of request queue for flow i at timestep n
$\mathcal{E}(S)$	Set of all extreme points of a convex set S

TABLE I: Notation for relevant parameters

4) The switch discards all elementary entanglements that are older than their maximum allowed lifetime.

User i's memories each have a probability p_i of successfully entangling with the switch, and the swap success probability for flow j is q_j. User i's elementary links can be stored for τ_i timesteps, where the timestep it is created counts as the first timestep (i.e. $\tau = 1$ means elementary links are either used immediately or discarded). This paper only considers *bufferless* switches, defined as switches where all memories have $\tau = 1$. We do not take into account any measure of quality of the end-to-end entanglements generated.

Define an *arrangement*, $\mathbf{a}(n)$, $n \in \{1, 2, \cdots\}$ to be a $K \times 1$ vector whose ith component is the number of elementary links between user i and the switch available in timestep n. Here $\mathbf{a}(n) \in \mathcal{A}$, where $\mathcal{A}$ is the set of all possible elementary link arrangements. Give $\mathcal{A}$ an ordering and call the kth arrangement $\mathbf{a}^{(k)}$. It is important to note that we do not assume any particular model for the memories, swap operations, or entangled states in our model. To apply our analysis to specific physical models one would need to calculate the probability parameters required for our methods and then follow the steps outlined in the rest of the paper.

We allow the switch to effectively serve multiple swaps simultaneously, given that there are enough existing elementary links to service all of them. To generalize the idea of the switch choosing a flow to service, we define a matching $\boldsymbol{\pi}$ to be an $M \times 1$ vector where the jth element is the number of requests of flow j that the switch services in a particular timestep. Therefore $\mathbf{M}\boldsymbol{\pi}$ is a vector that contains the number of elementary links required from each user to service the matching $\boldsymbol{\pi}$. A matching is *valid* if there are enough memories available to support all the requests it asks for, i.e. $\mathbf{M}\boldsymbol{\pi} \leq \mathbf{a}$. Define $\mathcal{M}$ to be the set of all matchings that would be valid if every elementary link existed simultaneously, $\mathcal{M} = \{\boldsymbol{\pi} \mid \mathbf{M}\boldsymbol{\pi} \leq \mathbf{D}\}$. Give this set an ordering and call the kth matching $\boldsymbol{\pi}^{(k)}$. Let $\boldsymbol{\lambda}^* = (\lambda_1^*, \cdots, \lambda_M^*)^T$ denote a servicing rate vector, where λ_j^* is the average number of end-to-end entangled states of flow j generated per timestep. Similarly, let $\boldsymbol{\lambda} = (\lambda_1, \cdots, \lambda_M)^T$ denote a request rate vector, where λ_j is the average number of end-to-end entangled states of flow j requested by the users per timestep. We say the switch is stable if it can guarantee that the average number of timesteps it takes for the switch to service a specific request is always finite [17]. Clearly if the switch is stable then $\boldsymbol{\lambda}^* = \boldsymbol{\lambda}$, though the converse is not necessarily true.

A switch scheduling policy is defined as the set of probabilities that matching $\boldsymbol{\pi}$ is selected given that elementary link arrangement $\mathbf{a}$ exists,

$$\mathbb{P}\left(\boldsymbol{\pi} \text{ chosen} \mid \mathbf{a}\right) \equiv c_{\boldsymbol{\pi},\mathbf{a}}, \quad \boldsymbol{\pi} \in \mathcal{M},\ \mathbf{a} \in \mathcal{A}.$$

If $\mathbf{M}\boldsymbol{\pi} \nleq \mathbf{a}$ then we set $c_{\boldsymbol{\pi},\mathbf{a}} = 0$ since that means matching $\boldsymbol{\pi}$ cannot be serviced when $\mathbf{a}$ is the elementary link arrangement. We store these probabilities in a matrix in the following way:

$$\mathbf{c} = \begin{bmatrix} c_{\boldsymbol{\pi}^{(1)},\mathbf{a}^{(1)}} & c_{\boldsymbol{\pi}^{(1)},\mathbf{a}^{(2)}} & \cdots \\ c_{\boldsymbol{\pi}^{(2)},\mathbf{a}^{(1)}} & c_{\boldsymbol{\pi}^{(2)},\mathbf{a}^{(2)}} & \cdots \\ \vdots & \vdots & \ddots \end{bmatrix} \triangleq \begin{bmatrix} c_{1,1} & c_{1,2} & \cdots \\ c_{2,1} & c_{2,2} & \cdots \\ \vdots & \vdots & \ddots \end{bmatrix}.$$

$\mathbf{c}$ is an $|\mathcal{M}| \times |\mathcal{A}|$ matrix. Because the terms of $\mathbf{c}$ are probabilities and the switch can only choose one matching at a time, we include the following two constraints:

$$\begin{aligned} c_{i,j} &\geq 0 \quad \forall i,j \\ \sum_i c_{i,j} &= 1 \quad \forall j \end{aligned} \tag{4}$$

where the last constraint is equality because we allow the matching $\boldsymbol{\pi} = [0, \cdots, 0]^T$, therefore even the switch doing nothing is considered a scheduling decision.

Having defined a scheduling policy in terms of matching probabilities conditioned on elementary link arrangements, to get the average number of end-to-end entangled states generated per timestep we need to determine the distribution of those arrangements. Because we only consider the case $\tau = 1$, arrangements are independently and identically distributed across timesteps. Denote the elementary link arrangement distribution $\mathbf{P} = \left[\mathbb{P}(\mathbf{a}^{(1)}), \mathbb{P}(\mathbf{a}^{(2)}), \cdots, \mathbb{P}(\mathbf{a}^{(|\mathcal{A}|)})\right]^T$ that can be used for every timestep. Let $\boldsymbol{\pi}^*$ denote a matrix whose jth

column is $\boldsymbol{\pi}^{(j)}$ and $\mathbf{Q} = \text{diag}\,(q_1, \cdots, q_M)$. Theorem 4.1 in [17] says that any servicing rate vector in Λ^* can be expressed as

$$\boldsymbol{\lambda}^* = \sum_{\mathbf{a}\in\mathcal{A}} \mathbb{P}(\mathbf{a}) \sum_{\boldsymbol{\pi}\in\mathcal{M}} c_{\boldsymbol{\pi},\mathbf{a}} \mathbf{Q}\boldsymbol{\pi} \tag{5}$$

which is equivalent to

$$\boldsymbol{\lambda}^* = \mathbf{Q}\boldsymbol{\pi}^*\mathbf{c}\mathbf{P}. \tag{6}$$

This can be understood as a weighted average of matching vectors that is then scaled anisotropically by each flows' associated swap probability of success.

The main goal of this paper is to calculate the set of possible servicing rate vectors, which is a set over all valid scheduling policies:

$$\Lambda^* \equiv \{\boldsymbol{\lambda}^* \mid \mathbf{c} \text{ valid}\}.$$

Although this is a full description of the servicing region, it does not provide a way of calculating it. In Section IV we present a method that uses (6) to provide a better characterization of Λ^*.

One distinction still needs to be made; we define the *capacity region* of a switch to be the set Λ of request rate vectors $\boldsymbol{\lambda}$ that the switch can service while remaining stable. This corresponds to Λ^* once the boundary of Λ^* is removed, i.e. $\Lambda = \Lambda^* \setminus \partial\Lambda^*$ [17]. In other words, if it is possible for the switch to create end-to-end entangled states with rate vector $\boldsymbol{\lambda}^*$ then there exists a scheduling policy that can support a request rate vector $\boldsymbol{\lambda} < \boldsymbol{\lambda}^*$. Because this distinction is trivial to take into account and has no effect on physical implementations, we ignore it and focus solely on the easier to manage servicing region for the remainder of the paper.

B. Simulating a Quantum Switch

As mentioned in Section II-B some prior work has studied how to efficiently allocate the switch's resources when different flows compete with each other. For our model, the most useful result is a proven throughput optimal scheduling policy [17] that a switch can adopt to stably service any incoming request rate vector $\boldsymbol{\lambda}$ that is in the capacity region Λ (i.e. the interior of the servicing region Λ^*). This policy is based on the MaxWeight framework, and it selects the following matching in timestep n:

$$\arg\max_{\mathbf{M}\boldsymbol{\pi}\leq\mathbf{a}(n)} \sum_{i=1}^{M} Q_i(n) q_i \pi_i \tag{7}$$

where $Q_i(n)$ is the number of requests of flow i in the queue at timestep n, π_i is the ith element of $\boldsymbol{\pi}$, and we maximize over all matchings $\boldsymbol{\pi}$ that satisfy $\mathbf{M}\boldsymbol{\pi} \leq \mathbf{a}(n)$. Because this scheduling policy is throughput optimal for the scenario we are studying, we can use it to simulate a switch's performance after calculating its capacity region. We will use these Monte Carlo simulations to corroborate our analytical calculations of servicing regions. Details on this are at the beginning of Section V.

C. Continuous Variables

Later we will address how the use of continuous variable (CV) protocols will affect the quantum switch model, but first we describe the particular repeater architecture that we base it on. The basis for our CV repeater model is previous work [33], [35], [36], [41] that studied a quantum scissor (QS) based repeater, which uses two-mode squeezed vacuum (TMSV) states as sources. This repeater provided an inherent directionality, i.e. *parity*, to each half-channel and the final end-to-end state can only be created from two half-channels of opposite parity. We assume that every elementary link is capable of being either parity and for simplification we give each memory a $\frac{1}{2}$ probability of being either. This may seem inappropriate because the protocol to create the links requires that the parity be agreed upon beforehand, but this repeater design heavily relies on multiplexing so alternating attempts is very natural. This complicates our model from the last section slightly, as we now have to keep track of which parity each elementary link is. We will address this issue by splitting one elementary link into two virtual links, one for each parity, and altering the probabilities $\mathbb{P}(\mathbf{a}^{(j)})$ to enforce the fact that both parities cannot exist in the same elementary link simultaneously. This in turn creates new virtual flows, since we can create entanglement between the same two users with two different parity combinations of virtual elementary links. This model imposes the restriction that our switch can only create bipartite end-to-end states, as the QS-based repeater does not trivially generalize to multipartite entanglement generation. Therefore if we are given a switch topology that creates M different bipartite CV end-to-end states, we can use a non-CV model with $2M$ flows to calculate the capacity region. If one wants to treat the two end-to-end parities the same then they can simply reduce all derived bounds by combining the two equivalent virtual flows associated with each real flow; however in practice it may be useful to keep track of the different parities.

IV. Results

Though we can input any distribution of elementary link arrangements, if we treat the generation of all link-level entanglements as independent Bernoulli trials then in the DV case we have

$$\mathbb{P}\left(\mathbf{a}(n) = \mathbf{a}\right) = \mathbb{P}(\mathbf{a}) = \prod_{i=1}^{K} \binom{D_i}{a_i} p_i^{a_i} (1-p_i)^{(D_i - a_i)}, \ \mathbf{a} \in \mathcal{A}$$

Observing that the set of valid $\mathbf{c}$ matrices, C, is convex and that the relationship between a scheduling matrix $\mathbf{c}$ and the servicing vector $\boldsymbol{\lambda}$ shown in (6) is a linear transformation, if we can find all extreme points of C then we have a way to calculate all extreme points of Λ^*. In order for this computation to be feasible we need to find a systematic way to generate all extreme points of C and show that it will finish in a finite amount of time.

We first characterize the extreme points of C. In short, every column of an extreme point of C must contain a single 1

and the other entries must be 0. To see why this must be true, we will prove that any matrix that does not have this property cannot be an extreme point of C. Let $\mathbf{c}_0 \in C$ be a matrix whose jth column has exactly two nonzero elements, $c_{i_1,j} = \alpha_1$ and $c_{i_2,j} = \alpha_2$, meaning $\alpha_1 + \alpha_2 = 1$. Let $\mathbf{c}_1$ and $\mathbf{c}_2$ be matrices with entries identical to $\mathbf{c}_0$ except in $\mathbf{c}_1$ we have $c_{i_1,j} = 1$ and $c_{i_2,j} = 0$, while in $\mathbf{c}_2$ we have $c_{i_1,j} = 0$ and $c_{i_2,j} = 1$. Clearly if $\mathbf{c}_0 \in C$ then $\mathbf{c}_1, \mathbf{c}_2 \in C$ as well. Because $\mathbf{c}_0$ can be written as a convex combination of these two:

$$\mathbf{c}_0 = \alpha_1 \mathbf{c}_1 + \alpha_2 \mathbf{c}_2$$

$\mathbf{c}_0$ cannot be an extreme point of C. This argument can be trivially extended to cases where more than two elements in a column are nonzero, showing that all extreme points of C cannot have more than one nonzero entry per column. These matrices being column stochastic implies that this single nonzero value must be 1 in every column. Because these matrices are finite in size there will be a finite number of matrices with this property and thus a finite number of extreme points of C, meaning there will be a finite number of extreme points of the servicing region $\Lambda^* = \mathbf{Q}\boldsymbol{\pi}^* C \mathbf{P} \subseteq \mathbb{R}^M$ and we have a method of generating a subset of Λ^* that contains them all.

Once we filter the set $\mathbf{Q}\boldsymbol{\pi}^* [\mathcal{E}(C)] \mathbf{P}$ down to just $\mathcal{E}(\Lambda^*)$ using an algorithm like QuickHull, we can characterize the set much easier as $\Lambda^* = \text{cch}(\mathcal{E}(\Lambda^*))$. There also exist algorithms like Delaunay Triangulation [40] to convert the extreme points of Λ^* to the hyperplanar boundaries of Λ^*. We know that the boundaries of Λ^* will be hyperplanar because it is a convex compact subset of $\mathbb{R}^M$ with a finite number of extreme points, meaning the region must be a polytope [38]. Thus all its bounds will be of the form

$$\sum_{j=1}^{M} \alpha_j \lambda_j^* \leq \gamma_{\boldsymbol{\alpha}}$$

where $\boldsymbol{\alpha}$ is an $M \times 1$ vector of weights and $\gamma_{\boldsymbol{\alpha}} \in \mathbb{R}$ is the upper bound of the weighted sum of rates. These can be used to quickly test whether a given rate vector $\boldsymbol{\lambda}^*$ is an element of the servicing region or not. Without these boundaries one would need to find a valid scheduling matrix, $\mathbf{c}$, that gives $\boldsymbol{\lambda}^*$ using (6), which is possible but very challenging. It is often more convenient to write these bounds in the form:

$$\sum_{j=1}^{M} \alpha_j \frac{\lambda_j^*}{q_j} \leq \beta_{\boldsymbol{\alpha}}$$

where the ratio λ_j^*/q_j can be thought of as an effective rate because $1/q_j$ is the average number of attempts it takes for the swap operation of flow j to succeed and service one request.

V. Example Calculations

As mentioned in Section III-B, we use simulation data to support our analytical results. We mark a request rate vector that was simulated to be stable by placing a blue dot at its corresponding point in space, where we consider a vector to be stable if we simulate the system with that request arrival rate for 10^6 timesteps and a linear regression on the function $\frac{1}{M}\sum_{i=1}^{M} Q_i(n)$ returns a slope of less than 10^{-4} requests/timestep. This threshold slope was chosen somewhat arbitrarily, but the transition from stable to unstable is very sharp and this value seems to consistently cull clearly unstable vectors while correctly labelling clearly stable vectors. We see in the figures below that there are many points being labelled as stable that are outside the planar boundaries that we predict bound the capacity region. This is to be expected since these data are numerical in nature, and points just outside of the capacity region are expected to result in very small slopes anyway. In fact, a plane having an inconsistent pattern of these points peeking through can be used as evidence that the bound is correct. A plane with too low of a bound will underestimate the capacity region (i.e. the bound will be too tight) and have stable points consistently appearing outside the boundary it forms. Similarly, planes with too high of a bound will overestimate the capacity region (i.e. the bound will be loose) and have no points coming through. Furthermore, a plane at an incorrect angle will exhibit a different density of these incorrectly labelled points at different parts of the plane, so a homogeneous pattern of these can tell us that the plane is at the correct height and angle.

A. Triangular Topology

We start with a triangular topology with probabilities $p = q = \frac{1}{2}$:

Flows:	{(1,2), (2,3), (1,3)}
p	$\{\frac{1}{2}, \frac{1}{2}, \frac{1}{2}\}$
q	$\{\frac{1}{2}, \frac{1}{2}, \frac{1}{2}\}$
D	$\{1, 1, 1\}$

thus we have $\mathbf{P} = [\frac{1}{8}, \cdots, \frac{1}{8}]^T$. The $\mathbf{M}$ matrix is easily generated from this information:

$$\mathbf{M} = \begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix}.$$

No two flows can be serviced simultaneously, so $\boldsymbol{\pi}^*$ also has a very basic structure:

$$\boldsymbol{\pi}^* = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

Defining $\mathbf{a}^*$ to be a matrix whose ith column is $\mathbf{a}^{(i)}$, we can also define our ordering of elementary link arrangements as:

$$\mathbf{a}^* = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix}.$$

From these orderings we can fill in the $\mathbf{c}$ matrix with its zero and non-zero terms:

$$\mathbf{c} = \begin{bmatrix} c_{1,1} & c_{1,2} & c_{1,3} & c_{1,4} & c_{1,5} & c_{1,6} & c_{1,7} & c_{1,8} \\ 0 & 0 & 0 & 0 & 0 & 0 & c_{2,7} & c_{2,8} \\ 0 & 0 & 0 & c_{3,4} & 0 & 0 & 0 & c_{3,8} \\ 0 & 0 & 0 & 0 & 0 & c_{4,6} & 0 & c_{4,8} \end{bmatrix}.$$

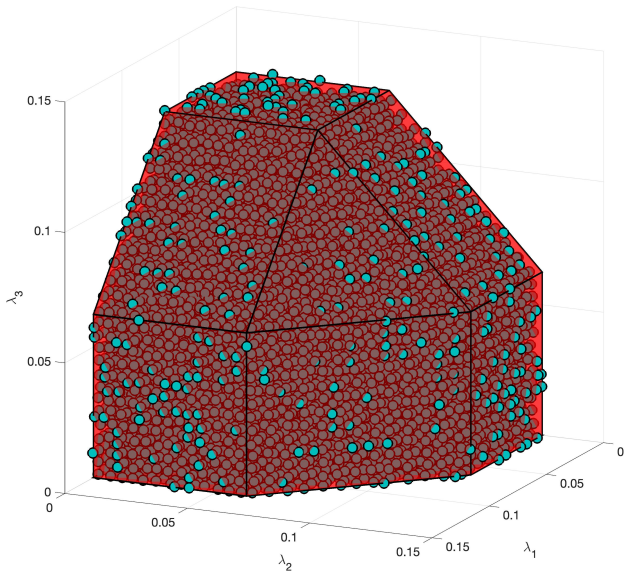


Fig. 2: Servicing region Λ^* for the triangular topology described in Section V-A.

For example, $c_{4,7}$ must be zero because the 7th column of $\mathbf{a}^*$ is $[1,1,0]^T$ which cannot support the matching in the 4th column of $\boldsymbol{\pi}^*$, $[0,0,1]^T$.

Recalling that all extreme points of C must have exactly one nonzero value per column and that all elements of C must be column stochastic, we recognize that we have $32 = 1\times1\times1\times2\times1\times2\times2\times4$ extreme points of the convex set of all $\mathbf{c}$ matrices. We then perform the linear transformation $L(A) = \mathbf{Q}\boldsymbol{\pi}^* A\mathbf{P}$ on all of these points and remove the non-extreme points with an algorithm like QuickHull. This leaves us with 13 unique extreme points of Λ^*, which can be further processed to give us 10 boundary planes of Λ^*. Three of these planes simply enforce that $\lambda_i \geq 0$ for all i, so we say that there are 7 non-trivial planes forming the boundary of Λ^*. The region in question is shown in Fig. 2 and analytically it can be expressed with the following planes (assume $i \neq j$ and $1 \leq i,j \leq 3$):

$$\begin{aligned} \frac{\lambda_i^*}{q_i} &\leq \frac{1}{4} \\ \frac{\lambda_i^*}{q_i} + \frac{\lambda_j^*}{q_j} &\leq \frac{3}{8} \\ \frac{\lambda_1^*}{q_1} + \frac{\lambda_2^*}{q_2} + \frac{\lambda_3^*}{q_3} &\leq \frac{1}{2} \end{aligned} \tag{8}$$

where we have generalized the swap probabilities to highlight specifically how they impact the geometry of the capacity region's boundaries. This result can be understood with elementary combinatorial arguments that give necessary conditions on the capacity region [21], though this method of analysis does not readily scale up.

Now we go through the same calculation but for a CV switch. This means we now have a 6-flow model with $3^3 = 27$ possible elementary link arrangements, since each user's elementary link can be in three possible states (not existing, parity 1, and parity 2) and there are two ways to service each flow. In the last section we defined the $\mathbf{a}^*$ matrix by having its rows count from 0 to $2^K - 1$ in binary, and here we create a matrix that counts from 0 to $3^K - 1$ in ternary. Our $\boldsymbol{\pi}^*$ matrix simply expands to be a 6×6 identity matrix with a column of all zeros appended on the left. We also alter the $\mathbf{M}$ matrix by doubling the number of columns to account for there being twice as many flows, and doubling the number of rows to account for each user now having two different link parities:

$$\mathbf{M} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 \end{bmatrix}.$$

The first two rows correspond to user 1, the next two correspond to user 2, and the last two correspond to user 3. For example the first two columns correspond to flow $(1,2)$, the first column being when user 1 has a parity 1 elementary link while user 2's is parity 2 and the second column corresponding to the opposite scenario. All that is left to do is compute $\mathbb{P}(\mathbf{a}^{(j)})$ for every scenario $\mathbf{a}^{(j)}$, which is similar to the normal Bernoulli trial method except we need to enforce that one user cannot have both parities at the same time and we need to account for the fact that a successfully created elementary link has probability $1/2$ of being either parity.

Now that we redefined every necessary matrix, we can recompute $\mathcal{E}(C)$ and transform the set to get 450 unique extreme points in $\mathbb{R}^6$. If we want we can further reduce this to 16 extreme points of $\mathbb{R}^3$ after combining virtual flows that correspond to the same real flow. This 3-dimensional region is shown in Fig. 3. Note that any specific flow has an upper bound of exactly $1/2$ of the upper bounds in Fig. 2, which makes sense because if two of the flows are almost entirely ignored then the only limiting factor on the third flow is the probability of success for both elementary links multiplied by the probability $1/2$ that the two links have opposite parity.

We analyze this triangular example one last time, now returning to DV and adding an extra memory to user 1's bank. In other words, we set $D = \{2,1,1\}$. The new capacity region is shown in Fig. 4 and Fig. 5. Clearly we have increased its size significantly, but it is interesting to note that the only new matching we are allowed to use is $\boldsymbol{\pi} = [1,0,1]^T$, so there is only a change in the region when considering flows 1 and 3.

B. Square Topology

Flows:	{(1,2), (2,3), (3,4), (1,4)}
p	$\{\frac{1}{2},\frac{1}{2},\frac{1}{2},\frac{1}{2}\}$
q	$\{\frac{1}{2},\frac{1}{2},\frac{1}{2},\frac{1}{2}\}$
D	$\{1,1,1,1\}$

Just as in the last section we start with a polygon, place users with one memory at the vertices, and consider only

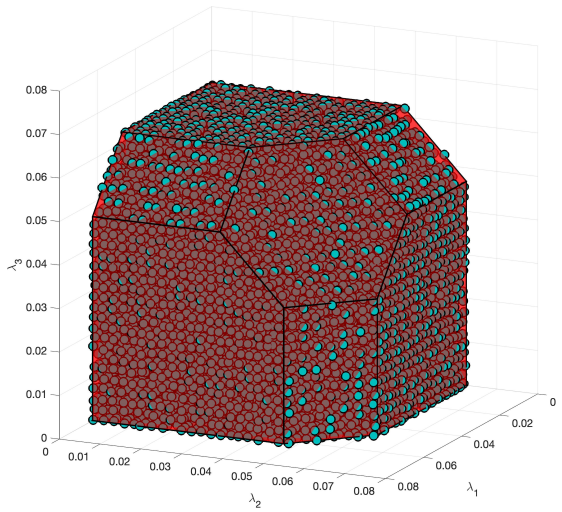


Fig. 3: Servicing region Λ^* for the triangular topology but with a CV protocol being used. Note that the maximum rate for any given flow is exactly half of the max rates shown in Fig. 2.

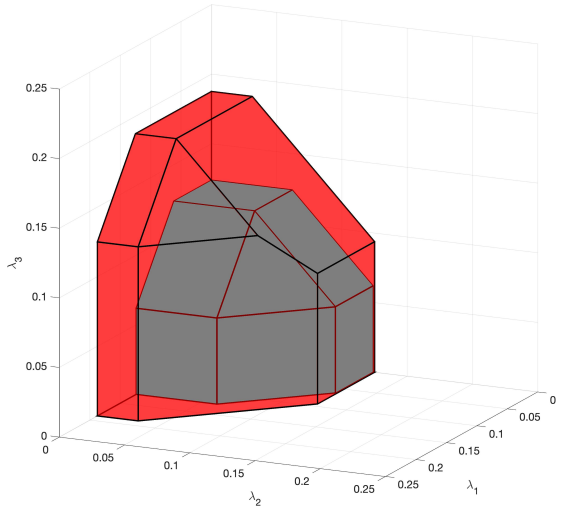


Fig. 4: Servicing region Λ^* for the triangular topology but where user 1 is given two quantum memories instead of only one (red, larger). We include the capacity region of Fig. 2 for reference (blue, smaller).

the bipartite flows formed by the edges of the polygon. An important change from the last example is that there are now scenarios where the switch can service multiple flows simultaneously (namely 1 and 3 or 2 and 4). Because we are considering a four-flow model, we are no longer able to view the entire capacity region at once, so we show projections of Λ^* for different values of λ_4. These projections are shown in Fig. 6. The full region can again be expressed with the following planes (assume $i \neq j$, $i \neq k$, $j \neq k$, and $1 \leq i, j, k \leq 4$):

$$\begin{aligned} \frac{\lambda_i^*}{q_i} &\leq \frac{1}{4} \\ \frac{\lambda_i^*}{q_i} + \frac{\lambda_j^*}{q_j} &\leq \frac{3}{8} \\ \frac{\lambda_i^*}{q_i} + \frac{\lambda_j^*}{q_j} + \frac{\lambda_k^*}{q_k} &\leq \frac{9}{16} \\ \frac{\lambda_1^*}{q_1} + \frac{\lambda_2^*}{q_2} + \frac{\lambda_3^*}{q_3} + \frac{\lambda_4^*}{q_4} &\leq \frac{5}{8} \end{aligned} \tag{9}$$

We unfortunately cannot compute the capacity region for the CV case of the square model because moving to an 8-flow model increases $|\mathcal{E}(C)|$ to almost 10^{19}, an infeasible size for brute force methods.

C. Pentagon with mixed-partite flows

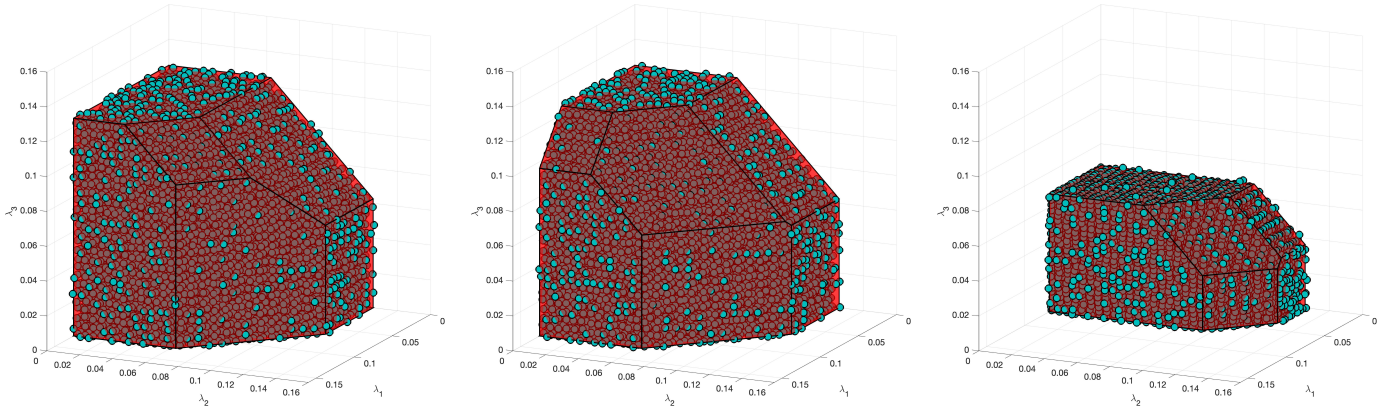
Looking at (8) and (9) one might conjecture that we only need bounds of the form

$$\sum_{k \in W} \frac{\lambda_k^*}{q_k} \leq \beta_W \tag{10}$$

for various subsets $W \subseteq \{1, \cdots, M\}$, but here we show an example where we need to weight the flows differently. Consider a more complicated example with both bipartite and tripartite flows:

Flows:	{(1,2,3), (2,3), (3,4,5), (4,5), (5,1,2)}
p	$\{\frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}\}$
q	$\{\frac{1}{4}, \frac{1}{2}, \frac{1}{4}, \frac{1}{2}, \frac{1}{4}\}$
D	$\{1, 1, 1, 1, 1\}$

We consider this model both to increase the complexity of the competition between flows and to avoid the infeasible calculation of the pentagon of users with bipartite flows.



(a) $\lambda_4 = 0$ (b) $\lambda_4 = 0.06$ (c) $\lambda_4 = 0.12$

Fig. 6: Projections of the capacity region for a switch with four users connected to it, allowing any two adjacent users to request the generation of a bipartite end-to-end entanglement. We can see that these regions are symmetric about the plane $\lambda_1 = \lambda_3$, while their extent on the λ_2 axis remains unchanged as we vary λ_4 because those flows do not directly compete.

Note that we have reduced the swap probabilities for the tripartite flows. As a reminder, the specific probabilities that we use are not based on any physical model – they can be replaced with any probabilities that come from specific models of entanglement generation and entanglement swapping. Our $\boldsymbol{\pi}^*$ matrix has a slightly more complicated form due to the irregular overlaps of the different flows:

$$\boldsymbol{\pi}^* = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}.$$

The $\mathbf{c}$ matrix will now be 8×32 because we have 8 flows and $2^5 = 32$ possible arrangements of the elementary links. After constructing this matrix by setting $c_{i,j} \to 0$ if $M\boldsymbol{\pi}^{(i)} \not\leq \mathbf{a}^{(j)}$ and doing some arithmetic, we see that $|\mathcal{E}(C)| = 4976640$ which is a reasonable number of extreme points to brute force transform one-by-one. After performing $Q\boldsymbol{\pi}^*\mathcal{E}(C)\mathbf{P}$ we get a set with 4448 unique points, 105 of which are extreme points forming 22 boundary planes (17 non-trivial) of the servicing region Λ^*. We show projections of this region in Fig. 7. Along with inequalities of the form shown in (10), which we will omit for brevity, we also need the following two inequalities to describe the boundary of the capacity region:

$$\frac{\lambda_1^*}{q_1} + 2\frac{\lambda_3^*}{q_3} + \frac{\lambda_4^*}{q_4} + \frac{\lambda_5^*}{q_5} \leq \frac{1}{2}$$
$$\frac{\lambda_1^*}{q_1} + \frac{\lambda_2^*}{q_2} + 2\frac{\lambda_3^*}{q_3} + \frac{\lambda_4^*}{q_4} + \frac{\lambda_5^*}{q_5} \leq \frac{19}{32}$$

which require non-unity coefficients on the rates.

We choose not to analyze the CV version of this network because, using our model for CV state generation, the optimal end-to-end state generation protocol is only known for bipartite links [35]. It is an open question to find the optimal arrangement of elementary link parities when creating a multipartite entangled state with a swap operation acting on NLA- and TMSV-based links.

VI. Discussion

A. Shortcut for a Particular Use Case

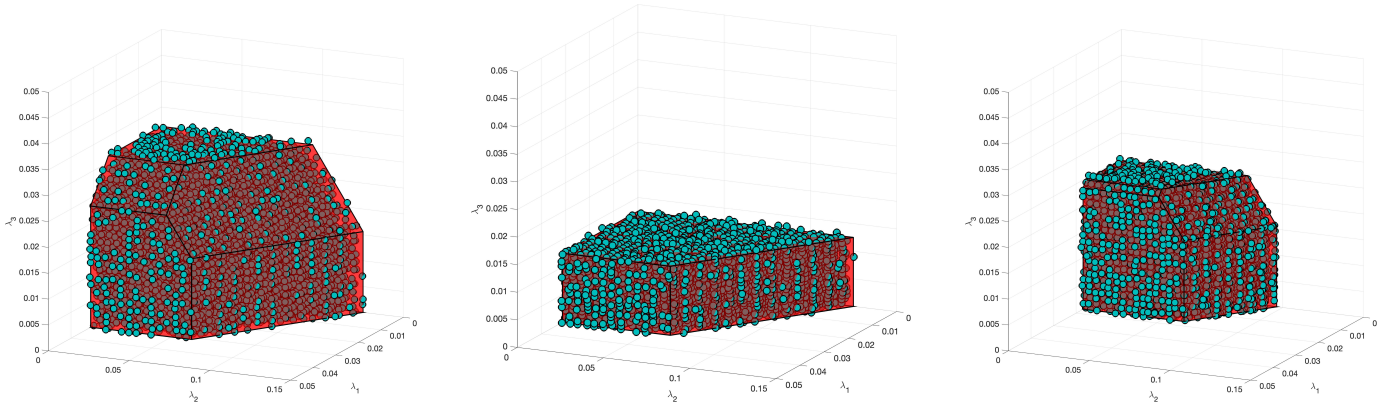
A problem one might be interested in solving is maximizing a weighted sum of servicing rates over the servicing region, i.e. calculating

$$\max_{\boldsymbol{\lambda}^* \in \Lambda^*} \sum_j \alpha_j \lambda_j^*. \tag{11}$$

One option to solve this is to use Section IV to calculate the capacity region and then maximize over its boundaries, but it turns out this specific problem can be solved much more easily with a different method. This can be done by applying elementary linear programming directly to (6) instead of calculating the entire capacity region, which is the computationally expensive part behind all results previously shown. This shortcut could be very useful to a network administrator that, instead of taking in requests for entanglement, just generates whatever end-to-end entangled state they choose in a given timestep. They will then most likely have a metric that they are trying to optimize, and many reasonable metrics will take the form of (11).

B. Computational Complexity

Calculating $\mathcal{E}(C)$ for even modest topologies can be infeasible. For example if we look at a topology where users are vertices on a regular K-sided polygon and we only consider the bipartite entanglement flows between adjacent users, i.e. the edges of the polygon, then $|\mathcal{E}(C_K)|$ grows faster than 2^{2^K} where we define C_K as the set of valid scheduling matrices for the K-sided polygon topology. Expanding upon Section V-C, analyzing the pentagonal case ($K = 5$) requires transforming almost 10^{10} extreme points of C if the proposed algorithm is followed strictly. Future work can focus on reducing this computational complexity, either with numerical methods or



(a) $\lambda_4 = 0$, $\lambda_5 = 0$ (b) $\lambda_4 = 0.1$, $\lambda_5 = 0$ (c) $\lambda_4 = 0$, $\lambda_5 = 0.03$

Fig. 7: Projections of the capacity region for a switch with five users connected to it, where we only consider the flows named in Section V-C. By increasing the servicing rate for either flow 4 or flow 5 we can decrease the possible servicing rates for the flows they compete with.

by proving a less expensive algorithm can also give analytical results.

VII. Conclusion

In summary, we presented an algorithm based on tools from convex analysis for calculating the boundary of the capacity region for a bufferless quantum switch. While previous work had successfully characterized the capacity region, there was no immediately obvious way of obtaining its boundaries from their results. Afterwards we presented several example calculations using our algorithm, some that can also be calculated by simple combinatorial methods and some that cannot.

Some obvious ways to extend this work include allowing elementary links to survive for multiple timesteps (i.e. $\tau > 1$), switching to a continuous time model rather than using slotted time, and including a purification protocol after many end-to-end states are generated. One could also work on improving the efficiency of the algorithm or finding shortcuts, such as proving that all boundary planes will be of a certain form and then using Section VI-A to jump straight to the boundaries of Λ^*. We will also use this section to reiterate that the capacity region, Λ, is simply the servicing region minus its boundary, so although all calculations shown have been used to calculate the servicing region, Λ^*, one can trivially get the capacity region from the results shown.

References

[1] Leonard Kleinrock. An early history of the internet [history of communications]. *IEEE Communications Magazine*, 48(8):26–36, 2010.

[2] Rongqing Hui. *Introduction to fiber-optic communications*. Academic Press, 2019.

[3] Dimitri Bertsekas and Robert Gallager. *Data networks*. Athena Scientific, 2021.

[4] Stephanie Wehner, David Elkouss, and Ronald Hanson. Quantum internet: A vision for the road ahead. *Science*, 362(6412):eaam9288, 2018.

[5] Jonathan P Dowling and Gerard J Milburn. Quantum technology: the second quantum revolution. *Philosophical Transactions of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, 361(1809):1655–1674, 2003.

[6] H Jeff Kimble. The quantum internet. *Nature*, 453(7198):1023–1030, 2008.

[7] Anthony J Brady, Christina Gao, Roni Harnik, Zhen Liu, Zheshen Zhang, and Quntao Zhuang. Entangled sensor-networks for dark-matter searches. *PRX Quantum*, 3(3):030333, 2022.

[8] Stefano Pirandola, B Roy Bardhan, Tobias Gehring, Christian Weedbrook, and Seth Lloyd. Advances in photonic quantum sensing. *Nature Photonics*, 12(12):724–733, 2018.

[9] Stephen DiAdamo, Bing Qi, Glen Miller, Ramana Kompella, and Alireza Shabani. Packet switching in quantum networks: A path to the quantum internet. *Physical Review Research*, 4(4):043064, 2022.

[10] Wenhan Dai, Anthony Rinaldi, and Don Towsley. Entanglement swapping in quantum switches: Protocol design and stability analysis. *arXiv preprint arXiv:2110.04116*, 2021.

[11] Thirupathaiah Vasantam and Don Towsley. Stability analysis of a quantum network with max-weight scheduling. *arXiv preprint arXiv:2106.00831*, 2021.

[12] Joaquin Chung, Ely M Eastman, Gregory S Kanter, Keshav Kapoor, Nikolai Lauk, Cristian H Pena, Robert K Plunkett, Neil Sinclair, Jordan M Thomas, Raju Valivarthi, et al. Design and implementation of the illinois express quantum metropolitan area network. *IEEE Transactions on Quantum Engineering*, 3:1–20, 2022.

[13] Marcus J Clark, Obada Alia, Rui Wang, Sima Bahrani, M Peranić, Djeylan Aktas, George T Kanellos, Martin Loncaric, Ž Samec, Anton Radman, et al. Entanglement distribution quantum networking within deployed telecommunications fibre-optic infrastructure. In *Quantum Technology: Driving Commercialisation of an Enabling Science III*, volume 12335, pages 96–103. SPIE, 2023.

[14] Cillian Harney and Stefano Pirandola. Analytical methods for high-rate global quantum networks. *PRX Quantum*, 3(1):010349, 2022.

[15] Kevin C Chen, Prajit Dhara, Mikkel Heuck, Yuan Lee, Wenhan Dai, Saikat Guha, and Dirk Englund. Zero-added-loss entangled-photon multiplexing for ground-and space-based quantum networks. *Physical Review Applied*, 19(5):054029, 2023.

[16] Claudio Cicconetti, Marco Conti, and Andrea Passarella. Resource allocation in quantum networks for distributed quantum computing. In *2022 IEEE International Conference on Smart Computing (SMARTCOMP)*, pages 124–132. IEEE, 2022.

[17] Thirupathaiah Vasantam and Don Towsley. A throughput optimal scheduling policy for a quantum switch. In Philip R. Hemmer and Alan L. Migdall, editors, *Quantum Computing, Communication, and Simulation II*, volume 12015, pages 14 – 23. International Society for Optics and Photonics, SPIE, 2022.

[18] Gayane Vardoyan, Saikat Guha, Philippe Nain, and Don Towsley. On the stochastic analysis of a quantum entanglement distribution switch. *IEEE Transactions on Quantum Engineering*, 2:1–16, 2021.

[19] Gayane Vardoyan, Philippe Nain, Saikat Guha, and Don Towsley. On the capacity region of bipartite and tripartite entanglement switching. *ACM*

Transactions on Modeling and Performance Evaluation of Computing Systems, 8(1-2):1–18, 2023.

[20] Panagiotis Promponas, Víctor Valls, and Leandros Tassiulas. Full exploitation of limited memory in quantum entanglement switching. *arXiv preprint arXiv:2304.10602*, 2023.

[21] Ian Tillman, Thirupathaiah Vasantam, and Kaushik P Seshadreesan. A continuous variable quantum switch. In *2022 IEEE International Conference on Quantum Computing and Engineering (QCE)*, pages 365–371. IEEE, 2022.

[22] William K Wootters and Wojciech H Zurek. A single quantum cannot be cloned. *Nature*, 299(5886):802–803, 1982.

[23] Stefano Pirandola, Riccardo Laurenza, Carlo Ottaviani, and Leonardo Banchi. Fundamental limits of repeaterless quantum communications. *Nature communications*, 8(1):1–15, 2017.

[24] William J Munro, Koji Azuma, Kiyoshi Tamaki, and Kae Nemoto. Inside quantum repeaters. *IEEE Journal of Selected Topics in Quantum Electronics*, 21(3):78–90, 2015.

[25] Sreraman Muralidharan, Linshu Li, Jungsang Kim, Norbert Lütkenhaus, Mikhail D Lukin, and Liang Jiang. Optimal architectures for long distance quantum communication. *Scientific reports*, 6(1):1–10, 2016.

[26] Pei-Shun Yan, Lan Zhou, Wei Zhong, and Yu-Bo Sheng. A survey on advances of quantum repeater. *Europhysics Letters*, 136(1):14001, 2021.

[27] Koji Azuma, Sophia E Economou, David Elkouss, Paul Hilaire, Liang Jiang, Hoi-Kwong Lo, and Ilan Tzitrin. Quantum repeaters: From quantum networks to the quantum internet. *arXiv preprint arXiv:2212.10820*, 2022.

[28] Timothy C Ralph and AP Lund. Nondeterministic noiseless linear amplification of quantum systems. In *AIP Conference Proceedings*, pages 155–160. American Institute of Physics, 2009.

[29] Mingjian He, Robert Malaney, and Benjamin A Burnett. Noiseless linear amplifiers for multimode states. *Physical Review A*, 103(1):012414, 2021.

[30] Shashank Pandey, Zhang Jiang, Joshua Combes, and Carlton M. Caves. Quantum limits on probabilistic amplifiers. *Phys. Rev. A*, 88:033852, Sep 2013.

[31] David T Pegg, Lee S Phillips, and Stephen M Barnett. Optical state truncation by projection synthesis. *Physical review letters*, 81(8):1604, 1998.

[32] Josephine Dias and Timothy C Ralph. Quantum error correction of continuous-variable states with realistic resources. *Physical Review A*, 97(3):032335, 2018.

[33] Kaushik P Seshadreesan, Hari Krovi, and Saikat Guha. Continuous-variable quantum repeater based on quantum scissors and mode multiplexing. *Physical Review Research*, 2(1):013310, 2020.

[34] L-M Duan, Mikhail D Lukin, J Ignacio Cirac, and Peter Zoller. Long-distance quantum communication with atomic ensembles and linear optics. *Nature*, 414(6862):413–418, 2001.

[35] Ian J Tillman, Allison Rubenok, Saikat Guha, and Kaushik P Seshadreesan. Supporting multiple entanglement flows through a continuous-variable quantum repeater. *Physical Review A*, 106(6):062611, 2022.

[36] Josephine Dias, Matthew S Winnel, Nedasadat Hosseinidehaj, and Timothy C Ralph. Quantum repeater for continuous-variable entanglement distribution. *Physical Review A*, 102(5):052425, 2020.

[37] Nitish K Panigrahy, Thirupathaiah Vasantam, Don Towsley, and Leandros Tassiulas. On the capacity region of a quantum switch with entanglement purification. In *IEEE INFOCOM 2023-IEEE Conference on Computer Communications*, pages 1–10. IEEE, 2023.

[38] Barry Simon. *Convexity: an analytic viewpoint*, volume 187. Cambridge University Press, 2011.

[39] C Bradford Barber, David P Dobkin, and Hannu Huhdanpaa. The quickhull algorithm for convex hulls. *ACM Transactions on Mathematical Software (TOMS)*, 22(4):469–483, 1996.

[40] Franco P Preparata and Michael I Shamos. *Computational geometry: an introduction*. Springer Science & Business Media, 2012.

[41] Fabian Furrer and William J Munro. Repeaters for continuous-variable quantum communication. *Physical Review A*, 98(3):032335, 2018.