Throughput Optimal Algorithm for Stochastic Quantum Networks

A Project Report

submitted by

VISHNU B

EE16B044

Under the guidance of

Dr. Abhishek Sinha

in partial fulfilment of the requirements

for the award of the degree of

Bachelor of Technology

in

Electrical Engineering

DEPARTMENT OF ELECTRICAL ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY MADRAS.

June 2020

Department of Electrical Engineering

INDIAN INSTITUTE OF TECHNOLOGY MADRAS

Certificate

This is to certify that this is a bonafide record of the project titled **Throughput Optimal Algorithm for Stochastic Quantum Networks**, submitted by **Vishnu B (EE16B044)**, to the Indian Institute of Technology Madras, in partial fulfillment of the requirements of the degree of Bachelor of Technology in Electrical Engineering. The contents of this project report, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Prof. Abhishek Sinha
Research Guide
Assistant Professor
Dept. of Electrical Engineering
IIT-Madras, 600 036

Place: Chennai

Date: 10th June 2020

ACKNOWLEDGEMENTS

I would like to sincerely thank my guide Dr. Abhishek Sinha for his valuable technical guidance as well as for the much needed encouragement and support. It has been a great opportunity working under him.

I would also like to thank my parents for their continued support during the course of this work, which covers an extended period during the pandemic lock down. Their moral support has been essential for the completion of this project during such trying times.

ABSTRACT

KEYWORDS: QKD ; Energy Harvesting Networks; Virtual Queues, Throughput Optimality

Quantum key distribution (QKD) allows remote communication parties to share identical and private keys for encryption and decryption, whose information-theoretical security is guaranteed by the fundamental principles of quantum mechanics. QKD systems generate shared, secret random numbers between two distant parties. A similar network model can be applied to Energy Harvesting Networks as well. To allow for long distance communication, we will need to use trusted quantum repeaters, and thus it is suitable to have a network model.

For efficient routing and packet forwarding in the network, we develop an algorithm that works in a generalised setting of wired topology. We will use the concept of virtual queues to come up with a throughput optimal algorithm for a general network with a 'Key' constraint, that offers a better performance in several parameters as compared to the existing methods implemented for such quantum networks.

TABLE OF CONTENTS

LIST OF FIGURES

ABBREVIATIONS

QKD	Quantum Key Distribution
RSA	Rivest, Shamir, Adleman
NIST	National Institute of Standards and Technology
OTP	One-Time Pad
XOR	Exclusive-OR operator
MDI-QKD	Measurement-Device-Independent Quantum Key Distribution
LEO	Low Earth Orbit
DARPA	Defense Advanced Research Projects Agency
BP	Back-Pressure (Algorithm)
UMW	Universal Max Weight(Algorithm)
SECOQC	Secure Communication based on Quantum Cryptography
APM	Assured Path Method
FIFO	First-In First-Out
LIFO	Last-In First-Out
ENTO	Extended Nearest To Origin

NOTATION

$\mathcal{G}$	Graph
V	Nodes of the graph
E	Edges of the graph
n	Number of nodes of the graph
m	Number of edges of the graph
c	Packet class
$\mathcal{C}$	Set of all packet classes
π	A policy
Π	Policy space
π^*	Throughput optimal policy
λ	Arrival rate vector
Λ	Capacity region
λ_i	Flow rate through i^{th} route
$s^{(c)}$	Source of class c packet
$t^{(c)}$	Destination of class c packet
$\mathcal{D}^{(c)}$	Set of destinations of class c packets
$A^{(c)}(t)$	Arrivals of class c packet at source at t
$R^{(c)}(t)$	Class c packets that reached destination till t
$T^{(c)}$	A route of class c packet
$\mathcal{T}^{(c)}$	Set of all routes of class c packets
$\boldsymbol{\gamma}$	Physical edge capacity vector
$\boldsymbol{\omega}$	Key restricted edge capacity vector
$\mathcal{F}$	Set of edges in minimum s-t cut
$\tilde{Q}$	Virtual data queue
$\tilde{S}$	Virtual quantum key queue
$K_e(t)$	Key generations at edge e at slot t
η_e	Key generation rate of edge e
γ_e	Physical capacity of edge e
$\mu_e(t)$	Service process at edge e at slot t
Θ	APM link activation threshold
θ_c	Max arrivals of class c packets at source under APM
$\hat{A}^{(c)}(t)$	Arrivals of class c packets at source at t under APM
$V^{(c)}$	Source queue for class c packets
$\Delta^{\pi}(t)$	One step Lyapunov drift under policy π at slot t

CHAPTER 1

INTRODUCTION

Secure communication for the applications across networks is gaining increasing attention from the research community. Traditional security techniques mostly focus on the encryption of communication, where security depends on the mathematical complexity. Standard methods for exchanging cryptographic keys such as RSA-1024 (once commonly used to exchange keys between browsers and web servers) are in jeopardy and is no longer regarded as safe by NIST. Other standard public-key infrastructure technologies approved by NIST haven't been broken yet but will soon will be by bigger, faster computers. Once quantum computers are mainstream, data encrypted using existing key exchange technologies will become even more vulnerable. Quantum cryptography is the only known method for transmitting a secret key over long distances that is provably secure in accordance with the laws of quantum physics.

Quantum key distribution (QKD) allows remote communication parties to share identical and private keys for encryption and decryption, whose information-theoretical security is guaranteed by the fundamental principles of quantum mechanics. QKD systems generate shared, secret random numbers between two distant parties [Van Meter (2014)]. The algorithm most commonly associated with QKD is the one-time pad(OTP), as it is provably secure when used with a secret, random key [Shannon (1949)]. In OTP we XOR the data with a key that is never reused, and is provably the only perfectly secure cryptographic mechanism. In real-world situations, it is often also used with encryption using symmetric key algorithms like the Advanced Encryption Standard algorithm [Daemen and Rijmen (1999)].

The idea of quantum cryptography stems from the phenomenon of quantum entanglement. Any measurement of a property of a particle results in an irreversible wave function collapse of that particle and will change the original quantum state. In the case of entangled particles, such a measurement will affect the entangled system as a whole [Einstein *et al.* (1935); Schrödinger (1935)]. The shared noise caused by entanglement allows for an excellent OTP and other similar cryptographic protocols [Ekert (1991)] as

any amount of eavesdropping by a third party disturbs the system and can be promptly detected by key distillation.

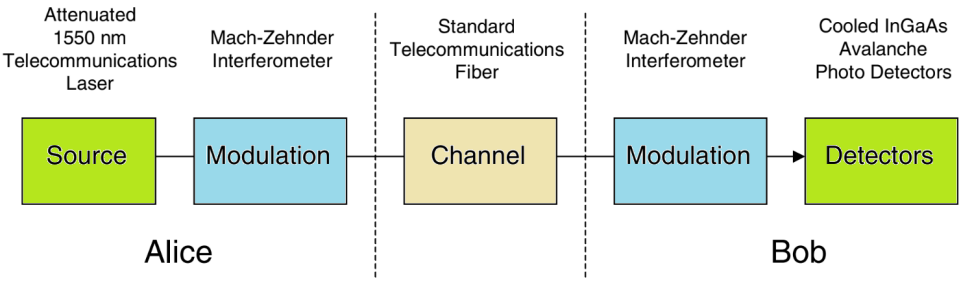

Figure 1.1: **High-level schematic of an exemplary QKD link.**
Alice and Bob are our communicating parties in this typical stand alone QKD system.

Despite having several security advantages, traditional QKD is distance limited, and can only be used across a single physical channel (e.g., free-space or telecommunications fiber) and is vulnerable to disruptions such as fiber cuts because it relies on single points of failure. However, these limitations can be mitigated or even completely removed by building QKD networks instead of the traditional stand-alone QKD links. One such network is the DARPA Quantum Network linking Harvard University, Boston University and BBN Technologies [Elliott (2004)]. Today, the topological structures of QKD networks have become more complex, such as the star structure in the measurement-device-independent quantum key distribution (MDI-QKD) [Tang *et al.* (2016*a*)] and the backbone structured 46-node Hefei Network. In recent times satellite based quantum communication has proven to be a feasible way to achieve a global-scale quantum communication network, and one such attempt was the low-Earth-orbit (LEO) satellite [Liao *et al.* (2017)].

In a quantum network with trusted nodes(repeaters) we can ensure secure transmission of data over larger distances. But as network traffic increases it is essential to have optimal routing and scheduling of packets to avoid overloading of links and improving latency, as we move closer to an era of quantum internet. In this paper, we will use be using quantum channel to share the keys and classical channel to share the message signal as shown in figures 1.2 and 1.3. We will attempt to replace the much celebrated Back-pressure algorithm [Tassiulas and Ephremides (1990)] used currently in QKD network [Zhou *et al.* (2019)], with the Universal Max Weight (UMW) algorithm [Sinha and Modiano (2017)] in view of improvisations in

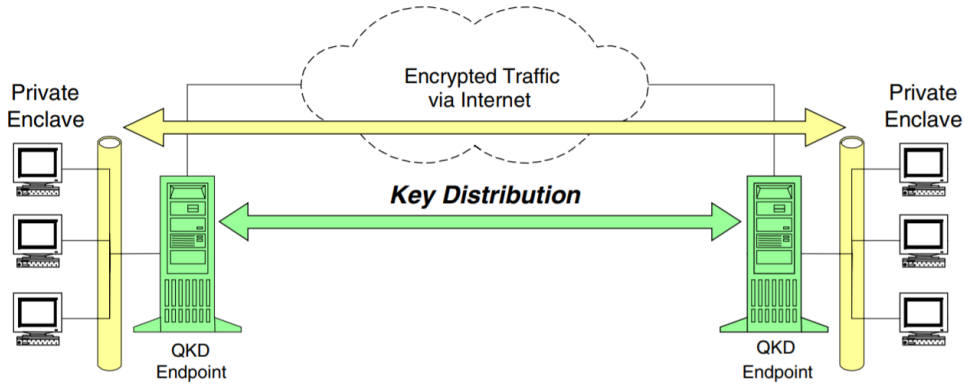

Figure 1.2: Simplified block diagram of a point-to-point QKD link in context [Elliott (2002)]

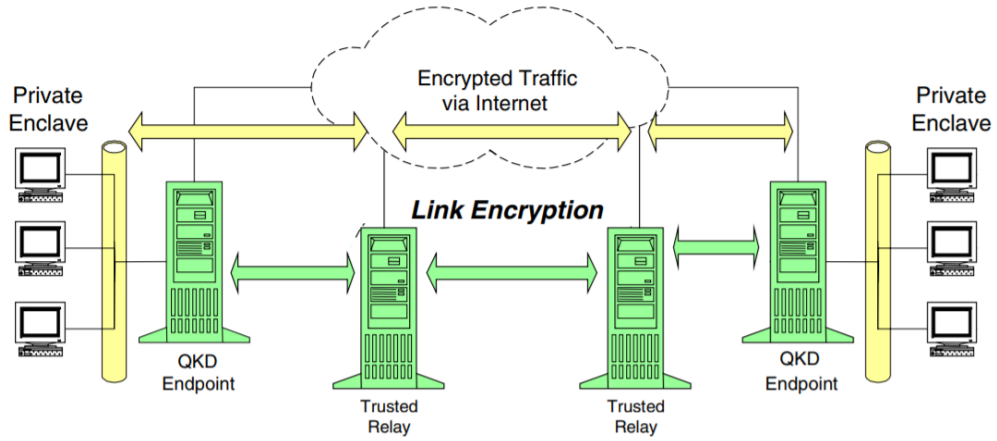

Figure 1.3: QKD network with trusted relays and link encryption [Elliott (2002)]

(i) Efficiency

(ii) Broadcasting Capability

(iii) Better Delay Properties

(iv) Reduction of State-Complexity

The objective of this project is to develop a throughput optimal policy in terms of routing in the generalised setting. We will need to formulate an appropriate capacity region of operation under the physical constraints of the given network and ensure stability of data queues. The result of this paper can be applied to Energy Harvesting Networks as well.

CHAPTER 2

RELATED WORK

2.1 Advancing Encryption with QKD

Standard encryption methodologies are becoming less reliable as the eavesdroppers and attackers are gaining powerful computing ability. Quantum cryptography is a new cryptographic technology for generating random secret keys to be used in secure communication. Quantum cryptography can provide communication security based on the laws of quantum physics (e.g., the no-cloning theorem and uncertainty principle) [Huang *et al.* (2020)]. While implementing QKD in terrestrial optical networks, distributing secret keys over a long distance (e.g., across the globe) is challenging. Single-photon signal transmitted over long-distance optical fiber suffers from high losses and depolarization. Hence, carrying the keys using optical fiber over long distances (e.g., 1000KM) is not an effective solution [Nauerth *et al.* (2013)]. Quantum key distribution (QKD) has attracted much attention on secure communications across global networks. In 2017, Mehic et. al. designed a QKD network simulation model, QKDNetSim, based on the classical Network Simulator-version 3 [Mehic *et al.* (2017)], to evaluate and validate a network solution at a low cost. Although QKDNetSim could simulate the key generation and secure communication processes, it could not accurately reflect the practical performance of a QKD network, owing to its neglect of the actual key generation capability and volatile communication demand of the QKD network [Li *et al.* (2020)]. There have also been attempts at foregoing the bipartite key distribution and developing a multipartite entanglement based key distribution into quantum networks [Takeoka *et al.* (2019)]. With the growing coverage and complexity of QKD networks, effective modeling is crucial for functional verification, quality assurance, cost control, cycle shortening, etc [Dianati *et al.* (2008)]

2.2 QKD Field Tests

There is a lot of ongoing field tests in the the realm of quantum networks at metropolitan-scale. Several test networks have been deployed that are tailored to the task of quantum key distribution either at short distances (but connecting many users), or over larger distances by relying on trusted repeaters. These networks do not yet allow for the end to end transmission of qubits or the end to end creation of entanglement between far away nodes. This includes SECOQC Vienna QKD network [Peev *et al.* (2009)], Chinese hierarchical network [Xu *et al.* (2009)], Geneva area network (SwissQuantum) [Stucki *et al.* (2011)], Tokyo QKD network [Sasaki *et al.* (2011)], Beijing-Shanghai Trunk Line and a few others.

Atmospheric attenuation in free space is less significant than in fibre. Consequently, a number of efforts are under way to raise the technology readiness level of satellite QKD. These projects range from truck-based tests of pointing and tracking mechanisms [Bourgoin *et al.* (2015)], to in-orbit testing of quantum light sources [Tang *et al.* (2016*b*)], to full QKD demonstrations using orbiting satellites[Jianwei (2014)].

2.3 Teleportation based Networks

The most powerful encryption algorithms are pretty robust, but the risk is big enough to spur some researchers to work on an alternative approach known as quantum teleportation, where entanglement between trusted nodes will provide a way of immediately transferring a qubit from one location to another without having to move a physical particle along with it. The paper [Caleffi (2017)] discusses entanglement generation rate detection algorithms and optimal quantum routing [Chakraborty *et al.* (2019)] in a network. There are many hurdles to this approach that include finding reliable ways of churning out lots of linked photons on demand, and maintaining their entanglement over very long distances. The engineering hurdles of building a purely quantum teleportation based network makes it a technology that would take years to materialize. Practically speaking the advent of a quantum internet would still mean the bulk of data transfer will occur via classical bits and most metropolitan quantum networks in development are still QKD based. Meanwhile teleportation based quantum internet can

still be used in implementing long distance QKDs and applications in domains such as distributed computing and multi-party cryptography.

2.4 Energy Harvesting Networks

Recent developments in hardware design have enabled many general wireless networks to support themselves by harvesting energy from the environment. They occur by converting mechanical vibration into energy, by using solar panels [Raghunathan *et al.* (2005)], by utilizing thermo-electric generators, or by converting ambient radio power into energy [Gorlatova *et al.* (2009)]. Such harvesting methods are also referred to as "recycling" energy. To take full advantage of the energy harvesting technology, efficient scheduling algorithms must consider the finite capacity for energy storage at each network node. The model being used in these energy harvesting networks are similar to that of the QKD network, with only difference being that in QKD the resources are Quantum Keys generated by edges whereas in the case of an Energy Harvesting Network it is the energy harvested by every node. Existing utility optimization problems used in energy harvesting networks [Huang and Neely (2012)] as well can be replaced with the efficient method we have devised in the following paper.

2.5 Optimal Control of Networks

The work we present in this paper is hugely based out of the throughput optimal policy of Universal Max Weight [Sinha and Modiano (2017)] which works in a generalized wireless setting. The policy has been modified to accommodate variants such as quantum key distribution networks and energy harvesting networks. We use UMW method based on several of its advantages over the celebrated Back Pressure policy [Tassiulas and Ephremides (1990)] which is widely used to cater to quantum key distributions networks as well[Zhou *et al.* (2019)].

CHAPTER 3

BACKGROUND AND SYSTEM SETUP

3.1 Network Model

We consider a wired quantum network of arbitrary topology represented with a graph $\mathcal{G}(V, E)$, were V are the nodes ($|V| = n$) and E are the edges(or links) ($|E| = m$). Time is slotted. Each edge e in the classical network has a physical capacity γ_e and this is the maximum number of data packets that can be transmitted via edge e per slot. The quantum network edge between nodes n_a and n_b will only be used for mutual key agreement and no actual data transfer. All links can be operated simultaneously without mutual interference, as we are considering wired network. The network topology is considered static but our proposed policy applies even in the case of a time varying network.

3.2 Traffic Model

In the Generalized Network Flow problem under observation, incoming packets at a source node are to be distributed among an arbitrary set of destination nodes, via multi-hop. Our incoming traffic is is classified and a class c traffic is given by its source node $s^{(c)} \in V$ and the set of destination nodes $D^{(c)} \subseteq V$. In the generalised framework we broadly define $D^{(c)}$ to fall in one of the following

.

- **UNICAST TRAFFIC:** All class c packets, arriving at a source node $s^{(c)}$, are required to be delivered to a single destination node $D^{(c)} = \{t^{(c)}\}$.
- **BROADCAST TRAFFIC:** All class c packets, arriving at a source node $s^{(c)}$, are required to be delivered to all nodes in the network, i.e., $D^{(c)} = V$.
- **MULTICAST TRAFFIC:** All class c packets, arriving at a source node $s^{(c)}$, are required to be delivered to a proper subset of nodes $D^{(c)} = \{t_1^{(c)}, t_2^{(c)}, ..t_k^{(c)}\} \subset V$.
- **ANYCAST TRAFFIC:** All class c packets, arriving at a source node $s^{(c)}$, are required to be delivered to any one of a given set of k nodes $D^{(c)} = t_1^{(c)} \oplus t_2^{(c)} \oplus .... \oplus t_k^{(c)}$ Thus the anycast problem is similar to the unicast problem, with all destinations forming a single super destination node.

The arrivals to the source are i.i.d at every slot. We denote $A^{(c)}(t)$ as the distinct class c arrivals at the source $s^{(c)}$ at slot t. The arrival rate vector is given by

$$\boldsymbol{\lambda} = \{\lambda^{(c)}, c \in \mathcal{C}\} \tag{3.1}$$

$$\mathbb{E}(A^{(c)}(t)) = \lambda^{(c)} \quad \forall c \in \mathcal{C} \tag{3.2}$$

We make an assumption that the total number of external packet arrivals to the entire network at a given slot is upper bounder by a finite number A_{max}.

3.3 Quantum Key Generation Process

The quantum keys generated by mutual agreement between two trusted nodes are shared in the quantum channel and is stored as a queue corresponding to each edge e. Due to the difficulties arising from entanglement generation and quantum decoherence, key generation can be considered the key limiting factor for quantum information transmission. Since keys abundance is always desirable, we do not impose an upper limit on the key queue length. Thus we will not be requiring stability in terms of this queue and a large key storage will aid better data transmission.

We define a term key fraction,

$$\zeta = \frac{\text{no of keys}}{\text{message length}}$$

This fraction ζ determines how many keys we need to encode one bit of data. We can see that ζ combined with the key-generation rate η alters the capacity region Λ as a message can only be transferred when there are enough keys required for the communication. In the case of one-time pad, $\zeta = 1$. We will be using this assumption in the rest of the paper as the policy we formulate is invariant of such a constant of proportionality. Let $K_e(t)$ be the number of keys generated at edge e in time slot t and $P_e(t)$ be the consumption of keys at edge e in time slot t. $P_e(t)$ can be alternatively interpreted as the number of data packets physically arriving edge e in time slot t, as each of these data packets will consume one key for encryption. We denote key storage by $S_e(t)$, which is an $|E| = m$ dimensional process which depends on the key generation rate η_e and our policy π. The key storage dynamics can be written as:

$$S_e(t+1) = \big(S_e(t) + K_e(t) - P_e(t)\big)^+$$
$$\text{where } \mathbb{E}(K_e(t)) = \eta_e$$

3.4 Policy Space

An admissible policy π for generalised network flow is responsible for forwarding packets over the link and possibly duplicate them (in the case of broadcast and multicast). In a wireless case the policy will take care of link activations as well by optimizing the net flow. Since we are considering the wired case we can ignore this aspect. The set of all admissible policies is denoted by Π. The set Π is unconstrained otherwise and includes policies which may use all past and future packet arrival information.

A policy $\pi \in \Pi$ is said to support an arrival rate-vector λ if, under the action of the policy π, the destination nodes of any class c receive distinct class c packets at the rate $\lambda^{(c)}$, $c \in C$. Let $R^{(c)}(t)$ denote the number of distinct class c packets, received in common by all destination nodes $i \in \mathcal{D}^{(c)}$, under the action of policy π, up to time t.

Definition 1. *[**Policy Supporting Rate-Vector $\boldsymbol{\lambda}$**] A policy $\pi \in \Pi$ is said to support an arrival rate vector $\boldsymbol{\lambda}$ if*

$$\liminf_{t\to\infty} \frac{R^{(c)}(t)}{t} = \lambda^{(c)}, \quad \forall c \in \mathcal{C}, \quad \textit{w.p.1} \tag{3.3}$$

3.5 Network Layer Capacity Region

Stability region or capacity region Λ_π, of policy π is the set of multi-class arrival rate vectors a for which the system is stable under π. A policy π_1 dominates another policy π_2 if $\Lambda_{\pi_1} \subset \Lambda_{\pi_2}$. The set Λ contains all arrival rate vectors for which there exists a policy in Π that stabilizes the system. An optimal policy, that is, one which dominates any other policy in Π, should have stability region that is a superset of the stability region of any other policy in Π; therefore, it should have stability region equal to $\boldsymbol{\Lambda}$. Such a policy is called a maximum throughput policy [Tassiulas and Ephremides (1990)]. Formally we say:

Definition 2 (**Network-layer Capacity Region**). *The network-layer capacity region $\boldsymbol{\Lambda}(\mathcal{G},\mathcal{C})$ is defined as the set of all supportable rates, i.e.,*

$$\boldsymbol{\Lambda}(\mathcal{G},\mathcal{C}) \stackrel{def}{=} \{\boldsymbol{\lambda} \in \mathbb{R}_+^{|\mathcal{C}|} : \pi \in \Pi \textit{ supporting } \boldsymbol{\lambda}\} \tag{3.4}$$

The set $\boldsymbol{\Lambda}(\mathcal{G},\mathcal{C})$ is convex.

Definition 3 (**Throughput Optimal Policy**). *A policy $\pi^* \in \Pi$, which supports any arrival rate $\boldsymbol{\lambda}$ in the interior of the capacity region $\boldsymbol{\Lambda}(\mathcal{G},\mathcal{C})$, is called a throughput-optimal policy.*

3.6 Admissible Routes

The popular throughput-optimal unicast policy Back-Pressure [Tassiulas and Ephremides (1990)], might deliver the same packet to a node multiple times, thus potentially degrading its delay performance. To combat this, our throughput-optimal policy is designed

to deliver a packet p to any node in the network at most once. This implies that the set of all admissible routes $T^{(c)}$ for packets of any class c, in general comprises of trees rooted at the corresponding source node $s^{(c)}$. Depending on the type of class c traffic, the topology of the admissible routes $T^{(c)}$ takes the following special forms:

.

- **UNICAST TRAFFIC:** $T^{(c)}$ = set of all unique $s^{(c)} - t^{(c)}$ paths in graph $\mathcal{G}$.
- **BROADCAST TRAFFIC:** $T^{(c)}$ = set of all spanning trees in graph $\mathcal{G}$ routed at $s^{(c)}$.
- **MULTICAST TRAFFIC:** $T^{(c)}$ = set of all Steiner trees in the graph $\mathcal{G}$, routed at $s^{(c)}$ and spanning the vertices $D^{(c)} = \{t_1^{(c)}, t_2^{(c)}, ..., t_k^{(c)}\}$
- **ANYCAST TRAFFIC:** $T^{(c)}$ = union of all $s^{(c)} - t_t^{(c)}$ paths in graph $\mathcal{G}$, $i = 1, 2, ...k$

3.7 Characterization of Capacity Region

Let $\mathcal{G}_{\boldsymbol{\gamma}}$ represent our graph with physical edge capacities given by a vector $\boldsymbol{\gamma}$ and capacity of each edge is γ_e.

$$\boldsymbol{\gamma} = \{\gamma_e, e \in E\}$$

We define a key-limited edge capacity vector $\boldsymbol{\omega} = \{\omega_e, e \in E\}$ given by

$$\omega_e = \min(\gamma_e, \eta_e) \qquad \forall e \in E$$

Consider any arrival rate vector $\boldsymbol{\lambda} \in \boldsymbol{\Lambda}(\mathcal{G}, \mathcal{C})$. By definition, there exists an admissible policy $\pi \in \Pi$, which supports the arrival rate $\boldsymbol{\lambda}$ by means of storing, duplicating and forwarding packets efficiently. Taking time-averages over the actions of the policy π, it is clear that there exists a *randomized* flow-decomposition and scheduling policy to route the packets such that none of the edges in the network is overloaded. We show that for every $\boldsymbol{\lambda} \in \boldsymbol{\Lambda}(\mathcal{G}, \mathcal{C})$, there exist non-negative scalars $\{\lambda_i^{(c)}\}$, indexed by the

admissible routes $T_i^{(c)} \in \mathcal{T}^{(c)}$ such that,

$$\lambda^{(c)} = \sum_{T_i^{(c)} \in \mathcal{T}^{(c)}} \lambda_i^{(c)} \tag{3.5}$$

$$\lambda_e \overset{\text{(def.)}}{=} \sum_{\substack{(i,c):e \in T_i^{(c)}, \\ T_i^{(c)} \in \mathcal{T}^{(c)}}} \lambda_i^{(c)} \leq \omega_e \quad \forall e \in E \tag{3.6}$$

Eqn. (3.5) claims the existence of a valid flow decomposition of incoming flows amongst existing routes. The flow decomposition can be non-unique. In Eqn. (3.6), λ_e stands for flow rate through edge e from all decomposed flows. The inequality in (3.6) states that no edge is overloaded, i.e. arrival rate of packets to any edge e under policy π will be at most the physical capacity γ_e of the edge or the quantum key generation rate η_e of edge e, whichever is lesser.

Definition 4. *$\overline{\mathbf{\Lambda}}$ is defined as the set of all arrival vectors $\boldsymbol{\lambda} \in \mathbb{R}_+^{|\mathcal{C}|}$ for which there exists a non-negative flow decomposition $\{\lambda_i^{(c)}\}$ such that Eqns.* (3.5) *and* (3.6) *are satisfied.*

Using Definition (4), we have the following theorem:

Theorem 1. *The network-layer capacity region $\mathbf{\Lambda}(\mathcal{G}, \mathcal{C})$ is characterized by the set $\overline{\mathbf{\Lambda}}$, up to its boundary.*

To prove this theorem we show the following:

. (i) All supportable rates belong to $\overline{\mathbf{\Lambda}}$

. (ii) All rates belonging to $\overline{\mathbf{\Lambda}}$ are supportable.

The proof of converse (i) is given in Appendix (A.1). Part (ii) is called achievability, and we prove this by developing a policy in the following paper which supports all rates in $\overline{\mathbf{\Lambda}}$.

As an example consider the case of single-commodity unicast setting. Since for single commodity $|\mathcal{C}| = 1$, capacity region can be represented by a single rate λ^* given

as:

$$\begin{aligned}\lambda^* &= \text{min-cut}(\mathcal{G}_{\boldsymbol{\omega}}) \\ &= \sum_{e \in \mathcal{F}} \omega_e\end{aligned}$$

where $\mathcal{G}_{\boldsymbol{\omega}}$ represents the graph with edge capacities given by vector $\boldsymbol{\omega}$, and $\mathcal{F}$ is the minimum s-t cut. We give a general bounding region for the capacity region boundary in Appendix (A.2).

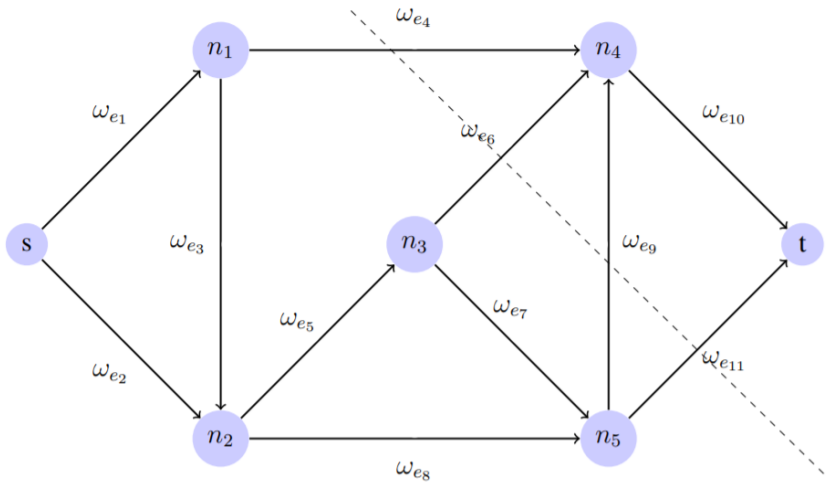

Figure 3.1: **An example 7-node QKD network**
Here if we were to have a unicast single commodity traffic then the maximum supportable rate (maximum flow) will be the minimum cut by maxflow-mincut theorem. In the figure, we consider $s-t$ cuts, i.e s and t fall on either side of the edge cut. The maximum flow will be minimum of such s-t cuts. If the dashed line represent the minimum cut then we can say that max-flow = $\omega_{e_4} + \omega_{e_6} + \omega_{e_9} + \omega_{e_{11}}$

CHAPTER 4

PROPOSED METHOD

4.1 Structure and Algorithms

4.1.1 Virtual Queues

In a multihop network, a packet p being routed along route $T = l_1 - l_2 - ...l_n$ where $l_i \in E$ will reach j^{th} link l_j only after passing through all links l_i where $1 \leq i < j$ by principle of causality. This constraint is known as the *precedence constraint* in the network scheduling literature [Lenstra and Rinnooy Kan (1978)]. In our method we will be relaxing this constraint to obtain a single-hop virtual network.

The UMW algorithm works on the principle of virtual queue process $\tilde{Q}(t)$, which is an $|\mathrm{E}| = m$ dimensional stochastic process. It is a fictitious queueing network without precedence constraints. The policy π describes a route $T^{(c)}(t) \in \mathcal{T}^{(c)}$ for packet of class c upon its arrival at the source. If we represent the links in a prescribed route $T^{(c)}(t)$ by the set $\{l_i | i = 1, 2, ...k\}$, an incoming packet determined to be routed via $T^{(c)}(t)$ by the policy π will induce a virtual arrival simultaneously at each of the virtual queues $\{\tilde{Q}_{l_i} | i = 1, 2..k\}$, right upon its arrival to the source. By foregoing the precedence constraints, any packet present in the virtual queue can be serviced.

To an edge e the arrival process is marked by $A_e^\pi(t)$ and service process by $\mu_e^\pi(t)$. The service process in our wired topology will be $\mu_e^\pi(t) = c_e$, which is the constant edge capacity. Since in wired case all links are interference free, we can activate all of them simultaneously. The arrival process to a virtual queue $\tilde{Q}_e$ at time t under the action of policy π can be written as

$$A_e^\pi(t) = \sum_{c \in \mathcal{C}} A^{(c)}(t) \mathbb{1}\left(e \in T^{(c)}(t)\right), \quad e \in E \tag{4.1}$$

The one-step evolution (Lindley recursion) of the virtual queue process can be thus

written as:

$$\tilde{Q}_e(t+1) = \left(\tilde{Q}_e(t) + A_e^{\pi}(t) - \mu_e^{\pi}(t)\right)^+ \quad \forall e \in \mathrm{E} \tag{4.2}$$

where $\mathbb{E}(A_e(t)) = \lambda_e$.

Now we define the *Virtual Key queue process* $\tilde{\boldsymbol{S}}(t) = \{\tilde{S}_e(t), e \in E\}$ which is another $|E| = m$ dimensional stochastic process. When a packet p of class c is assigned route $T^{(c)}(t) \in \mathcal{T}^{(c)}$ then the keys needed for secured transmission along the route is reserved in advanced upon arrival at source $s^{(c)}$ itself, *i.e* this incoming packet induces a virtual key usage (reservation) simultaneously from virtual key queues $\{\tilde{S}_{l_i} | i = 1, 2, ...k\}$ along links $\{l_i | i = 1, 2, ...k\}$ in the route $T^{(c)}(t)$, right upon arrival at its source. As $K_e(t)$ denotes the key generation at the edge we can define the one-step evolution (Lindley recursion) for Virtual Key queue process as:

$$\tilde{S}_e(t+1) = \left(\tilde{S}_e(t) + K_e(t) - A_e^{\pi}(t)\right)^+ \quad \forall e \in \mathrm{E} \tag{4.3}$$

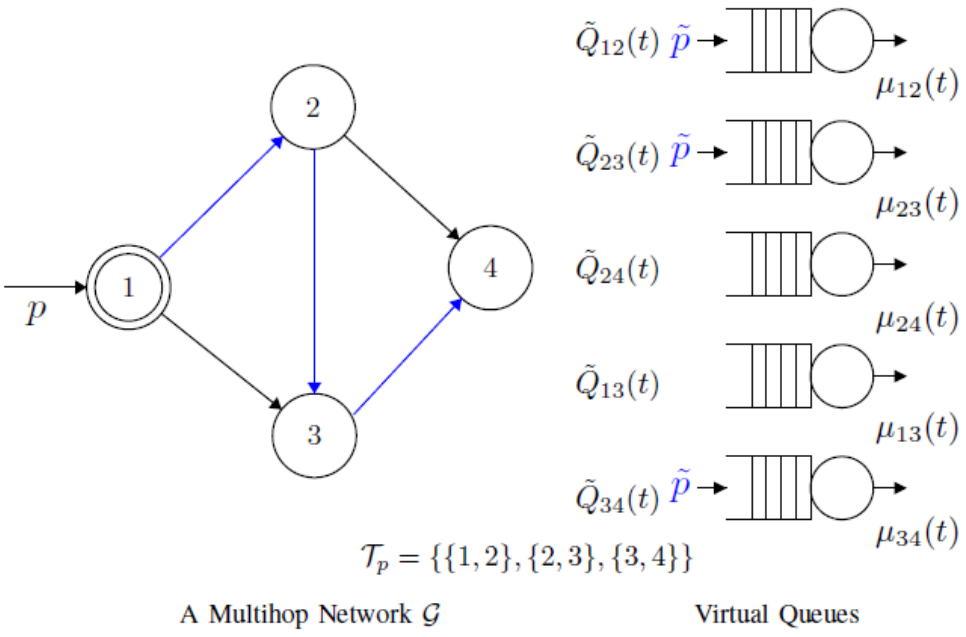

Figure 4.1: Illustration of the virtual queue system for the four-node network $\mathcal{G}$. Upon arrival, the incoming packet p, belonging to a unicast session from node 1 to 4, is prescribed a path $\mathcal{T}_p = \{\{1,2\},\{2,3\},\{3,4\}\}$. Relaxing the precedence constraints, the packet p is counted as an arrival to the virtual queues $\tilde{Q}_{12}$ and $\tilde{Q}_{23}$ and $\tilde{Q}_{34}$ simultaneously *at the same slot*. In the physical system, the packet p may take a while before reaching any edge in its path, depending on the control policy.

4.1.2 Assured Path Method

As we have earlier seen, the number of keys in each step need not be stabilised and we will thereby need to decouple it from the original path assignment problem. For this we design a queue for each class c at source $s^{(c)}$ as $V^{(c)}$, to temporarily store the incoming packets. The arrivals to this queue will be denoted by $A^{(c)}(t)$ and the service process is denoted as $\hat{A}^{(c)}(t)$. $\hat{A}^{(c)}(t)$ will be the modified number of arrivals of class c packets into the network at time t. We define a set of thresholds:

$$\theta_c \geq \lambda^{(c)}$$

$$\Theta \geq \sum_{c \in \mathcal{C}} \theta_c$$

where Θ is the minimum number of virtual quantum keys required in every link for it to be considered as an active edge. This is a mere hypothetical threshold we define for controlling the packet arrivals and not the activation condition of a quantum link. The new active edges form a graph $\mathcal{G}^{'}(V, E)$.
When the $s^{(c)} - t^{(c)}$ min-cut of $\mathcal{G}^{'}(V, E)$ evaluates to zero, i.e when the new graph is disconnected between $s^{(c)}$ and $t^{(c)}$, we store the packet in $V^{(c)}$. Every time-step when there exists atleast one $s^{(c)} - t^{(c)}$ path, we assign a route $T^{(c)}(t)$. This way any packet that enters the network after being released from the temporary storage at source will not have to worry about a lack of quantum keys for communication at each link. Every packet that enters the network gets keys assigned ahead of time for each link part of its route.

The service process from $V^{(c)}$, denoted by $\hat{A}^{(c)}(t)$ will be as follows:

$$\hat{A}^{(c)}(t) = \begin{cases} \min(V^{(c)}(t), \theta_c) & \text{when} \quad \mathcal{T}^{(c)}(t) \neq \phi \\ 0 & \text{when} \quad \mathcal{T}^{(c)}(t) = \phi \end{cases}$$

4.1.3 Stability of Source Queue

Before getting into the routing and service of packets we will look at the source queue $V^{(c)}$ under the assured path method. If $V^{(c)}$ blows up then it indicates that data packets are not reaching the network and our controlled arrival process is restraining the flow of

data. Therefore it is necessary to ensure the stability $V^{(c)}$ queue.

Let $A^{(c)}(0,t)$ represent the actual number of class-c arrivals to the network till time t and $\hat{A}^{(c)}(0,t)$ the modified class-c arrivals to the network till time t. Let $x(t)$ represent the number of slots till time t where mincut $\mathcal{C}_s$ with edge weights as $\tilde{S}_e(t)$.

$$\begin{aligned}\hat{A}^{(c)}(0,t) &\leq \theta_c * (t - x(t)) \\ \lim_{t\to\infty} \frac{\hat{A}^{(c)}(0,t)}{t} &\leq \theta_c \lim_{t\to\infty} (1 - \mathbb{P}(\text{min-cut = 0})) \\ &\leq \theta_c\end{aligned}$$

as the least probability of 0-mincut occurrence is 0. Thus we see that if $\theta_c < \lambda^{(c)}$ then the modified arrival process will have an arrival rate less than that of original rate, leading to blowing up of source queue. Thus we always choose θ_c such that $\theta_c \geq \lambda^{(c)}$.

The minimum number of virtual quantum keys required in every edge to be considered active is set as Θ. The value of Θ would not change the results as we can always assume that at $t = 0$ reference, every edge has Θ keys. The source queue dynamics will thus only depend on the key generation rate of each edge and key usage rates.

$$\hat{A}^{(c)}(0,t) \leq \sum_{e\in\mathcal{F}} \left(\Theta + \sum_{\tau=0}^{t} K_e(\tau)\right)$$

where $\mathcal{F}$ is the minimum s-t cut in network graph G with edge weights replaced as $\sum_{\tau=0}^{t} K_e(\tau)$. This inequality says that we practically cannot send more packets than that of the keys generated till time t.

A more noteworthy observation is that

$$\sum_{e\in\mathcal{F}} \left(\Theta + \sum_{\tau=0}^{t} K_e(\tau)\right) \leq \hat{A}^{(c)}(0, t + t_s) \tag{4.4}$$

where

$$t_s = \left\lfloor \frac{\sum_{e\in\mathcal{F}} \tilde{S}_e(t)}{\theta_c} \right\rfloor + 1$$

$\lfloor . \rfloor$ is the floor function. Inequation (4.4) arises from the fact that the modified arrival process will utilize all keys generated till time t after a time $t + t_s$, and t_s will be

number of time slots needed to consume the remaining unallotted keys at a rate θ_c per slot. The extra one slot indicates the case when there are less than θ_c keys remaining. The remainder from the floor function will be processed in the last slot. Dividing Eqn. (4.4) by t and taking limit to ∞ we get

$$\begin{aligned}\lim_{t\to\infty}\frac{\hat{A}^{(c)}(0,t+t_s)}{t} &\geq \lim_{t\to\infty}\frac{\sum_{e\in\mathcal{F}}\left(\Theta+\sum_{\tau=0}^{t}K_e(\tau)\right)}{t}\\ &=\sum_{e\in\mathcal{F}}\lim_{t\to\infty}\frac{\sum_{\tau=0}^{t}K_e(\tau)}{t}\\ &=\sum_{e\in\mathcal{F}}\eta_e\end{aligned}$$

In our case $\lim_{t\to\infty}\tilde{S}_e(t)$ will be finite because if $\tilde{S}_e(t)$ were to be infinite when $t\to\infty$ then packets can be sent at maximum rate θ_c in every slot and $\mathbb{P}(\text{min-cut}=0)$. Thus t_s will be finite as $t\to\infty$ and we can say:

$$\lim_{t\to\infty}\frac{\hat{A}^{(c)}(0,t+t_s)}{t}=\lim_{t\to\infty}\frac{\hat{A}^{(c)}(0,t)}{t} \tag{4.5}$$

Since we operate within the capacity region:

$$\lambda^{(c)}\leq\sum_{e\in\mathcal{F}}\min(\eta_e,\gamma_e)\leq\sum_{e\in\mathcal{F}}\eta_e \tag{4.6}$$

By using Eqns. (4.5) and (4.6) we get:

$$\lim_{t\to\infty}\frac{\hat{A}^{(c)}(0,t)}{t}\geq\lambda^{(c)} \tag{4.7}$$

We can write the source queue length at time t as:

$$\begin{aligned}V^{(c)}(t)&=A^{(c)}(0,t)-\hat{A}^{(c)}(0,t)\\ \lim_{t\to\infty}\frac{V^{(c)}(t)}{t}&=\lim_{t\to\infty}\frac{A^{(c)}(0,t)}{t}-\lim_{t\to\infty}\frac{\hat{A}^{(c)}(0,t)}{t}\\ &\leq\lambda^{(c)}-\lambda^{(c)}=0\end{aligned}$$

by using Eqn. (4.7). Since $V^{(c)}(t)$ is non negative we can say:

$$\lim_{t\to\infty}\frac{V^{(c)}(t)}{t}=0$$

Thus $V^{(c)}(t)$ is rate stable.

4.2 Dynamic Control and Stability of Virtual Queues

To derive a stabilizing policy for the virtual network, consider a quadratic Lyapunov function $L(\tilde{Q}(t))$ defined in terms of the virtual queue lengths:

$$L(\tilde{Q}(t)) = \sum_{e \in E} \tilde{Q}_e^2(t)$$

From one-step dynamics we have:

$$\begin{aligned}
\tilde{Q}_e^2(t+1) &\leq (\tilde{Q}_e(t) - \mu_e^\pi(t) + A_e^\pi(t))^2 \\
&= \tilde{Q}_e^2(t) + (A_e^\pi(t))^2 + \left(\mu_e^\pi(t)\right)^2 + 2\tilde{Q}_e(t)A_e^\pi(t) \\
&\quad - 2\tilde{Q}_e(t)\mu_e^\pi(t) - 2A_e^\pi(t)\mu_e^\pi(t)
\end{aligned}$$

The one-step Lyapunov drift $\Delta^\pi(t)$ conditioned on current virtual queue lengths $\tilde{Q}(t)$, under the operation of any admissible Markovian policy $\pi \in \Pi$ is upper bounded by

$$\begin{aligned}
\Delta^\pi(t) &= \mathbb{E}(L(\tilde{Q}(t+1)) - L(\tilde{Q}(t))) \\
&\leq B + 2\sum_{e \in E} \tilde{Q}_e(t)\mathbb{E}\Big(A_e^\pi(t)|\tilde{\boldsymbol{Q}}_{\boldsymbol{e}}(\boldsymbol{t})\Big) \\
&\quad - 2\sum_{e \in E} \tilde{Q}_e(t)\mathbb{E}\Big(\mu_e^\pi(t)|\tilde{\boldsymbol{Q}}_{\boldsymbol{e}}(\boldsymbol{t})\Big) \qquad (4.8)
\end{aligned}$$

4.2.1 Dynamic Routing Policy:

We minimize the routing cost which is the term in the middle.

$$\text{RoutingCost}^\pi = \sum_{e \in E} \tilde{Q}_e(t)A_e^\pi(t)$$

Using Eqn. (4.1) we can write this as:

$$\text{RoutingCost}^\pi = \sum_{c \in C} A^{(c)}(t) \sum_{e \in E} \tilde{Q}_e(t)\mathbb{1}(e \in T^{(c)}(t))$$

where $T^{(c)}(t) \in \mathcal{T}^{(c)}(t)$ and $\mathcal{T}^{(c)}(t)$ is the set of all routes such that every link e in the route satisfies $\tilde{S}_e(t) \geq \Theta$ at time slot t.

Using separability of the above equation it decomposes as:

$$T^{(c)}_{\text{opt}}(t) \in \underset{T^{(c)} \in \mathcal{T}^{(c)}(t)}{\arg\min} \sum_{e \in E} \tilde{Q}_e(t) \mathbb{1}(e \in T^{(c)}) \tag{4.9}$$

such that $\forall e \in T^{(c)} \quad \tilde{S}_e(t) \geq \Theta$

For different traffic types, the optimal route will be one of the following combinatorial problems on the graph $\mathcal{G}$, with edge weights as virtual queue lengths $\tilde{Q}$:

.

- **UNICAST TRAFFIC:** $T^{(c)}_{\text{opt}}(t)$ = The shortest $s^{(c)} - t^{(c)}$ path in the weighted-graph $\mathcal{G}$.
- **BROADCAST TRAFFIC:** $T^{(c)}_{\text{opt}}(t)$ = The minimum weight spanning tree rooted at the source $s^{(c)}$, in the weighted-graph $\mathcal{G}$.
- **MULTICAST TRAFFIC:** $T^{(c)}_{\text{opt}}(t)$ = The minimum weight Steiner tree rooted at the source $s^{(c)}$ and spanning the destinations $\mathcal{D}^{(c)} = \{t^{(c)}_1, t_{(c),\ldots} t^{(c)}_k\}$, in the weighted-graph $\mathcal{G}$.
- **ANYCAST TRAFFIC:** $T^{(c)}_{\text{opt}}(t)$ = The shortest of the k shortest $s^{(c)} - t^{(c)}_i$ paths, $i = 1, 2, \ldots k$ in the weighted-graph $\mathcal{G}$.

The routes are selected according to a dynamic source routing policy [Johnson and Maltz (1996)]. For Steiner tree problem in multicast traffic we may use an efficient approximation algorithm [Byrka *et al.* (2010)] as it is NP-hard. For all the other routing problems on the weighted virtual we can use standard algorithms[Cormen *et al.* (2009)] to solve.

4.2.2 Strong Stability of Virtual Queues

Under the given dynamic routing and link scheduling policy, the virtual queue process $\{\tilde{Q}(t)\}_{t \geq 0}$ is strongly stable for any arrival rate vector $\boldsymbol{\lambda} \in \text{int}(\overline{\boldsymbol{\Lambda}})$

$$\boxed{\limsup_{T \to \infty} \sum_{t=0}^{T-1} \sum_{e \in \text{E}} \mathbb{E}(\tilde{Q}_e(t)) < \infty}$$

Proof. We take an arrival rate vector $\boldsymbol{\lambda} \in \text{int}(\overline{\boldsymbol{\Lambda}})$. Thus we know from Eqns. (3.5) and (3.6) that there exists a scalar $\epsilon > 0$ such that we can decompose the total arrival for each class $c \in \mathcal{C}$ into a finite number of routes, such that

$$\lambda_e = \sum_{\substack{(i,c):e \in T_i^{(c)}, \\ T_i^{(c)} \in \mathcal{T}^{(c)}}} \lambda_i^{(c)} \leq \omega_e - \epsilon, \quad \forall e \in E \tag{4.10}$$

We now define an auxiliary stationary randomized routing policy $\pi_{\textbf{RAND}} \in \Pi$ for the virtual queue system $\{\tilde{Q}_e(t)\}$, such that $\pi_{\textbf{RAND}}$ routes the incoming packet of class c along the route $T_i^{(c)} \in \mathcal{T}^{(c)}(t)$ with probability $\frac{\lambda_i^{(c)}}{\lambda^{(c)}}, \forall i, c$. When $\mathcal{T}^{(c)}(t) = \phi$, we store it in the source queue $V^{(c)}$ and consider it's routing problem in the time slot $t+1$. We see that:

$$\mathbb{E}(A_e^{\pi_{\textbf{RAND}}}(t)) = \lambda_e = \sum_{\substack{(i,c):e \in T_i^{(c)}, \\ T_i^{(c)} \in \mathcal{T}^{(c)}}} \lambda_i^{(c)}, \quad \forall e \in E \tag{4.11}$$

As we are considering the wired topology here, we can say that in every slot all links are activated and thus under the policy each edge will have:

$$\mathbb{E}(\mu_e^{\pi_{\textbf{RAND}}}(t)) = \gamma_e \tag{4.12}$$

Since our policy APM minimizes the drift expression in Eqn. (4.8) from the set of all feasible policies Π, we can write:

$$\begin{aligned} \Delta^{\pi_{\textbf{APM}}}(t) \leq B + 2 \sum_{e \in E} \tilde{Q}_e(t) \mathbb{E}\Big(A_e^{\pi_{\textbf{RAND}}}(t) | \tilde{\boldsymbol{Q}}_{\boldsymbol{e}}(\boldsymbol{t}) \Big) \\ - 2 \sum_{e \in E} \tilde{Q}_e(t) \mathbb{E}\Big(\mu_e^{\pi_{\textbf{RAND}}}(t)) | \tilde{\boldsymbol{Q}}_{\boldsymbol{e}}(\boldsymbol{t}) \Big) \end{aligned}$$

Using the fact that Randomized policy is memoryless and hence independent of the

virtual queue lengths $\tilde{\boldsymbol{Q}}_e(t)$, the upper bound of the drift (RHS) will be:

$$
\begin{aligned}
&= B + 2\sum_{e\in E}\tilde{Q}_e(t)\Big(\mathbb{E}\big(A_e^{\pi_{\mathbf{RAND}}}(t)\big) - \mathbb{E}\big(\mu_e^{\pi_{\mathbf{RAND}}}(t)\big)\Big)\\
&= B + 2\sum_{e\in E}\tilde{Q}_e(t)(\lambda_e - \gamma_e)\\
&\leq B + 2\sum_{e\in E}\tilde{Q}_e(t)(\lambda_e - \min(\gamma_e, \eta_e))\\
&= B + 2\sum_{e\in E}\tilde{Q}_e(t)(\lambda_e - \omega_e)\\
&\leq B - 2\epsilon\sum_{e\in E}\tilde{Q}_e(t)
\end{aligned}
$$

using inequality in Eqn. (4.10). Taking expectation of both sides w.r.t. the virtual queue lengths $\tilde{Q}(t)$, we bound the expected drift at slot t as:

$$\mathbb{E}L(\tilde{\boldsymbol{Q}}(t+1)) - \mathbb{E}L(\tilde{\boldsymbol{Q}}(t)) \leq B - 2\epsilon\sum_{e\in E}\mathbb{E}(\tilde{Q}_e(t)) \tag{4.13}$$

Summing Eqn. (4.13) from $t = 0$ to $T-1$ and remembering that $L(\tilde{\boldsymbol{Q}}(T)) \geq 0$ and $L(\tilde{\boldsymbol{Q}}(0)) = 0$, we conclude that

$$\sum_{t=0}^{T-1}\sum_{e\in \mathrm{E}}\mathbb{E}(\tilde{Q}_e(t)) \leq \frac{B}{2\epsilon} \tag{4.14}$$

Taking lim-sup on both sides proves our claim.

4.2.3 Rate Stability of Virtual Queues

Using the condition of strong stability of virtual queues as proved in earlier section and *Lemma* 1 in [Sinha and Modiano (2017)], we can say that under the action of above policy, for any $\lambda \in \text{int}(\overline{\boldsymbol{\Lambda}})$

$$\boxed{\lim_{t\to\infty}\frac{\tilde{Q}_e(t)}{t} = 0, \quad \forall e \in E, \qquad \text{w.p.1}}$$

In other words, we say that the virtual queue process $\tilde{\boldsymbol{Q}}$ is rate stable [Neely (2010)].

4.2.4 Rate Stability of Physical Queues

There exist many possibilities for the packet scheduler, which efficiently resolves contention when multiple packets attempt to cross an edge e at the same time-slot t. Popular choices for the packet scheduler include FIFO, LIFO etc. In this paper we will use the Extended Nearest To Origin (**ENTO**) method [Sinha and Modiano (2017)] which is variation of the Nearest to Origin(NTO) method for the generalised setting. NTO is a scheduling policy which has its origin in the context of *adversarial queueing theory*.

Definition 5. *(Extended NTO). If multiple packets attempt to cross an active edge e at the same time slot t, the Extended Nearest To Origin (ENTO) policy gives priority to the packet which has traversed the least number of hops along its path from its origin up to the edge* e.

Using the condition of rate stability of virtual queues as proved in earlier section and Theorem 3 in [Sinha and Modiano (2017)], we can say that under **ENTO** packet scheduling,the physical queues are rate stable for any $\lambda \in \text{int}(\overline{\mathbf{\Lambda}})$

$$\lim_{t\to\infty} \frac{\sum_{e\in E} Q_e(t)}{t} = 0, \qquad \text{w.p.1}$$

We will use this condition to prove our throughput optimality.

4.3 Throughput-Optimality of APM

For any class $c \in \mathcal{C}$, the number of packets received $R^{(c)}(t)$ by all nodes in $\mathcal{D}^{(c)}$ is bounded as follows:

$$A^{(c)}(0,t) - V^{(c)}(t) - \sum_{e\in E} Q_e(t) \overset{(a)}{\leq} R^{(c)}(t) \overset{(b)}{\leq} A^{(c)}(0,t)$$

Here the first inequality (a) arises from the observation that if a packet p of class c has not reached all destination nodes $\mathcal{D}^{(c)}$, then at least one copy of it must be present in either one of the physical queues or in the source queue. Inequality (b) represents that the number of packets received till time t be less than the packets that have arrived at the

source till time t since packets are not generated anywhere else apart from the source. We then divide both sides of (a) and (b) with t and take limit to ∞

$$\lim_{t\to\infty} \frac{\left(A^{(c)}(0,t) - V^{(c)}(t) - \sum_{e\in E} Q_e(t)\right)}{t} \leq \lim_{t\to\infty} \frac{R^{(c)}(t)}{t}$$

and

$$\lim_{t\to\infty} \frac{R^{(c)}(t)}{t} \leq \lim_{t\to\infty} \frac{A^{(c)}(0,t)}{t}$$

As we know that

$$\lim_{t\to\infty} \frac{V^{(c)}(t)}{t} = 0 \quad \text{and} \quad \lim_{t\to\infty} \frac{\sum_{e\in E} Q_e(t)}{t} = 0$$

because of their rate stability, we can say that:

$$\lim_{t\to\infty} \frac{R^{(c)}(t)}{t} = \lim_{t\to\infty} \frac{A^{(c)}(0,t)}{t} = \lambda^{(c)}$$

Thus the algorithm is throughput optimal.

4.4 APM Algorithm

Algorithm 1 Assured Path Method at slot t for the Generalized Flow Problem in a Wired Network

Require: Graph $\mathcal{G}(V, E)$, Virtual Queue lengths $\{\tilde{Q}_e(t), e \in E\}$ at the slot t, Virtual Key Queue lengths $\{\tilde{S}_e(t), e \in E\}$ at the slot t

1: **[Edge Activation]** Activate only edges e with $\tilde{S}_e(t) \geq \Theta$ keys available to be reserved at the slot t.

2: **[Active Routes]** To the empty set of routes $\mathcal{T}^{(c)}(t)$, add all possible routes $T^{(c)}$ such that every $e \in T^{(c)}$ are active at the slot t.

3: **[Storage at Source]** Add all incoming packets of class c to the source queue $V^{(c)}$

4: **[Edge-Weight Assignment]** Assign each edge of the graph $e \in E$ a weight $W_e(t)$ equal to $\tilde{Q}_e(t)$, i.e

$$\boldsymbol{W}(t) \leftarrow \tilde{\boldsymbol{Q}}(t)$$

5: **if** $\mathcal{T}^{(c)}(t) \neq \phi$ **then**:

5.1: **[Route Assignment]** Compute a Minimum Weight Route $T^{(c)}(t) \in \mathcal{T}^{(c)}(t)$ for a class c incoming packet in the weighted graph $\mathcal{G}(V, E)$, according to Eqn.(4.9).

5.2: Assign $\hat{A}^{(c)}(t) = \min(\theta_c, V^{(c)}(t))$.

5.3: Input the packets $\hat{A}^{(c)}(t)$ into the network at the first link in their assigned route

else:

Assign $\hat{A}^{(c)}(t) = 0$

6: **[Packet Forwarding]** Forward physical packets from the physical queues over the all links(irrespective of threshold activation) according to a desired scheduling policy (ENTO, FIFO etc).

7: **[Queue Counter Updation]** Update the virtual key queues and virtual data queues assuming a precedence-relaxed system, i.e.,

$$\tilde{S}_e(t+1) \leftarrow \left(\tilde{S}_e(t) + K_e(t) - A_e^{\pi}(t)\right)^+, \quad \forall e \in \mathrm{E}$$

$$\tilde{Q}_e(t+1) \leftarrow \left(\tilde{Q}_e(t) + A_e^{\pi}(t) - \mu_e^{\pi}(t)\right)^+, \quad \forall e \in \mathrm{E}$$

CHAPTER 5

RESULTS AND DISCUSSION

5.1 Simulation Details

We have simulated the following arbitrary unicast network with a standard FIFO scheduling for physical queues in the single-commodity wired setting. The wired setting implies that no two edges interfere with each other and we can simultaneously operate them all. The edges values in the graph indicate the key generation rates η_e. All links have a unit capacity. We have used Ford Fulkerson algorithm [Cormen *et al.* (2009)] to compute the minimum cut of the graph to be used as bound of capacity region (as single commodity). In the given graph maximum supportable unicast rate $\lambda^* = 0.9$, and we will load the network at 95% capacity. We assume poisson arrivals at the source and poisson key generation process. Shortest paths are computed using Bellman-Ford Algorithm [Bellman (1958)].

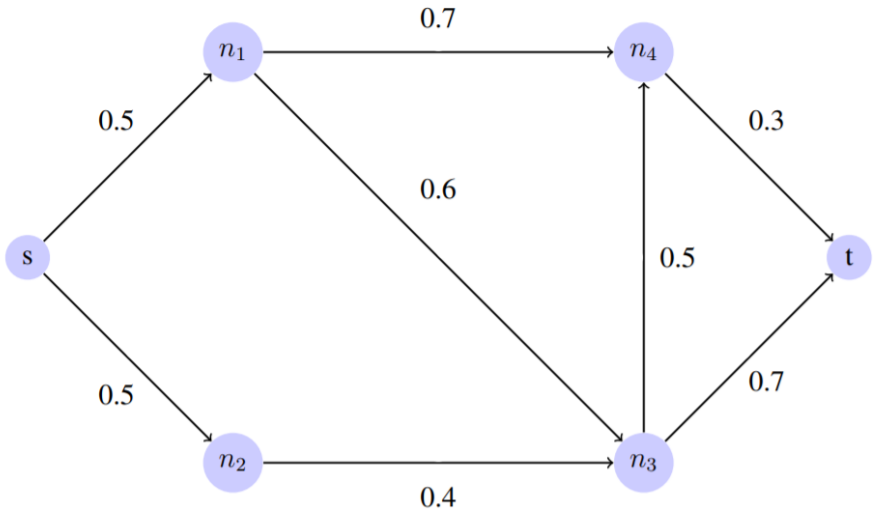

Figure 5.1: Graph used for Unicast Simulation. Note that $\lambda^* = 0.9$ by the minimum cut $\{(s - n_1), (n_2, n_3)\}$

To show the relevance of *Assured Path Method*, we first simulate an obvious heuristic that involves minimization of virtual queue lengths and maximization of virtual key

lengths along a path. This leads to the following minimization problem:

$$\arg\min_{T^{(c)} \in \mathcal{T}^{(c)}} \sum_{e \in E} (\tilde{Q}_e(t) - k\tilde{S}_e(t)) \mathbb{1}(e \in T^{(c)}) \tag{5.1}$$

k is an arbitrary constant of proportionality and tweaking its value does not affect the results significantly. In fig 5.2 we observe that the randomized policy stabilizes the virtual queues whereas the proposed heuristic does not. An inference based reason is that certain paths may have an edge with a significantly high key generation rate along with edges with low key generation rate. Even while the route attracts more traffic due to its high rate edge, the flow gets choked at the low rate edge. This leads to blowing up of those queues. We have compared the evolution of queue lengths in this method with that of randomized policy $\pi_{\mathbf{RAND}}$.

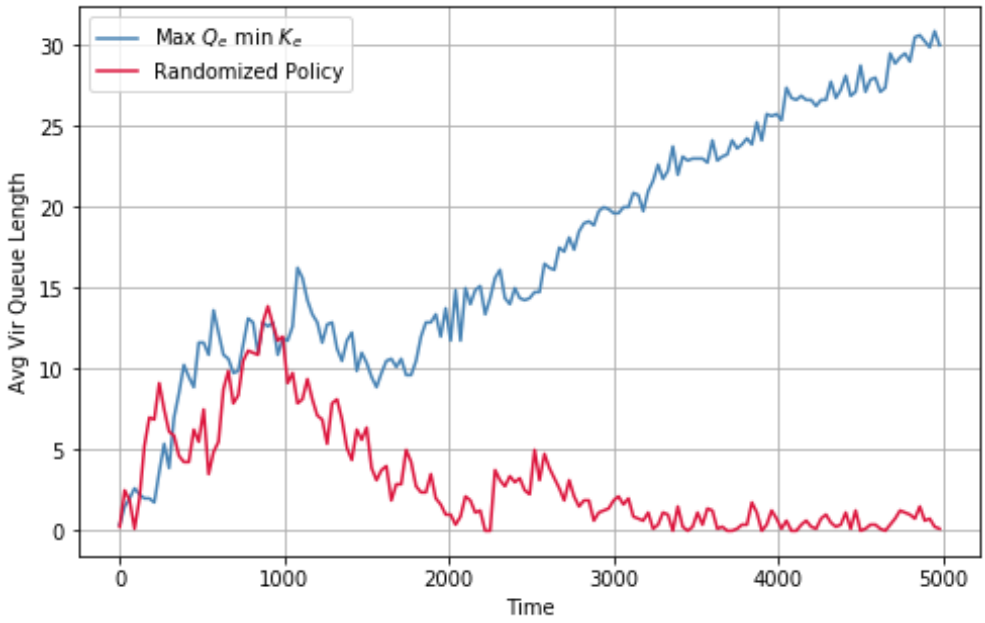

Figure 5.2: Max-Virtual Data Queue Min Virtual Key Queue in comparison with Randomized policy $\pi_{\mathbf{RAND}}$

5.2 Simulations for Assured Path Method

The Assured Path Method unlike previous heuristic is proven to be throughput optimal and we can observe the stability of the virtual queues in Fig 5.3. We see that even with higher Θ the queue is stabilized under *Assured Path* Algorithm, and only time required

for stabilization changes. Since we are looking at stability in the long run any $\Theta > \lambda$ will give us similar outcomes.

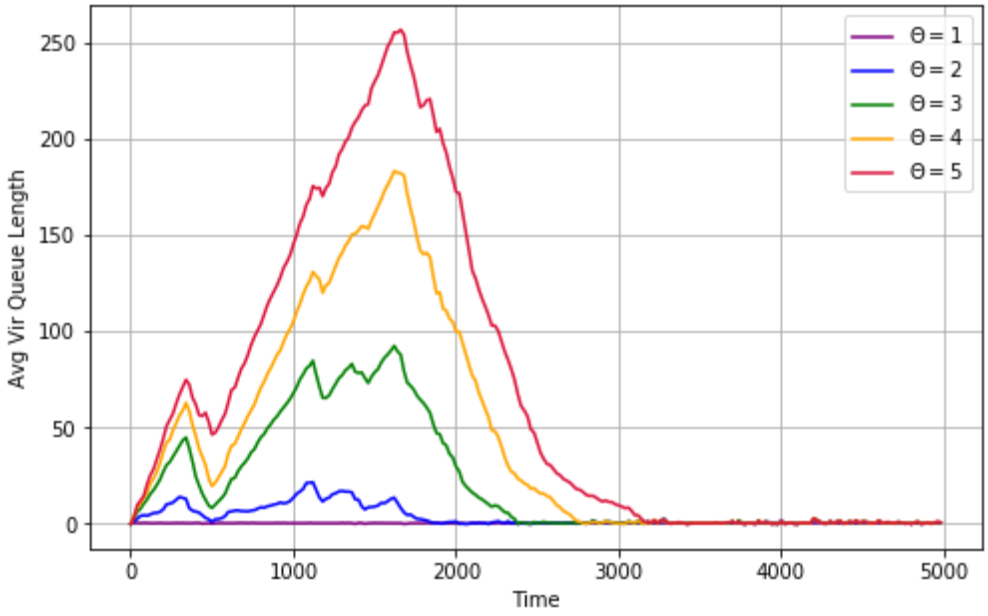

Figure 5.3: Evolution of Average Queue lengths with time under APM for different values of Θ. Note that we do not take the case $\Theta = 0$, as we need $\Theta > \lambda$, and $= 0.855$ under 95% capacity

Comparing the source queue length $V^{(c)}(t)$ evolution with time for different values of Θ as shown in Fig 5.4, we see that they are more or less the same. Any $\Theta < \lambda$, in this case 0, will result in the blowing up of source queue. This is why in all practical implementations we take $\Theta > \lambda$.

Next we compare the performance of *Assured Path* policy with that of the randomized policy $\pi_{\mathbf{RAND}}$. We fixed $\Theta = 2$ and varied a load factor $\rho; 0 \leq \rho \leq 1$ which controls at what fraction of maximum capacity we are operating at. In fig 5.7, we see that for higher traffic there is difference in average queue lengths between APM and randomized policy $\pi_{\mathbf{RAND}}$, APM fares better. This will translate to better delay performance for APM over the randomized policy.

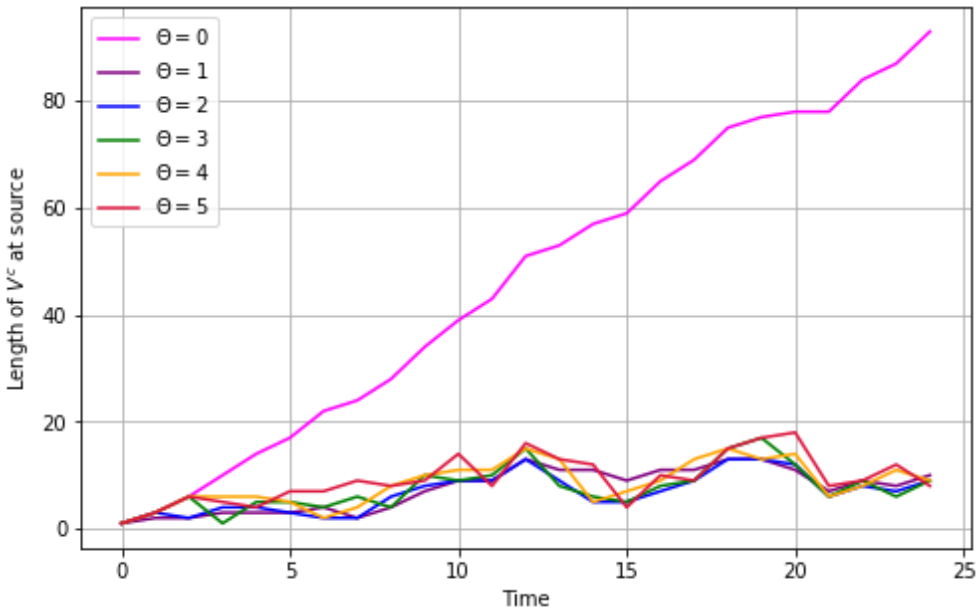

Figure 5.4: Evolution of Source lengths with time under APM for different values of Θ. Note that for the case $\Theta = 0$, we observe an unstable queue.

5.3 APM in Broadcast Setting

Broadcast capacity λ^* is the maximum rate at which source recieve packets so that every node receives the packets at the same rate λ^*. In wired networks, the broadcast capacity can be achieved by routing packets using maximal spanning trees, which can be efficiently computed using Edmonds' algorithm [Edmonds (1967)] for minimum spanning arborescence [Chu (1965)] in the case of directed acyclic graphs. We consider the following 9-node grid for our simulation. Each edge has a key generation rate $\eta_e = 0.5$ and edge capacity $\gamma_e = 1$. Thus we get $\lambda^* = 0.5$ in wired case [Sinha and Modiano (2019)]. The simulation that follows illustrates the variation of physical queue lengths as a function of load factor ρ. For higher Θ the average queue lengths are also larger. We have used Edmonds's algorithm to find the minimum spanning arborescence.

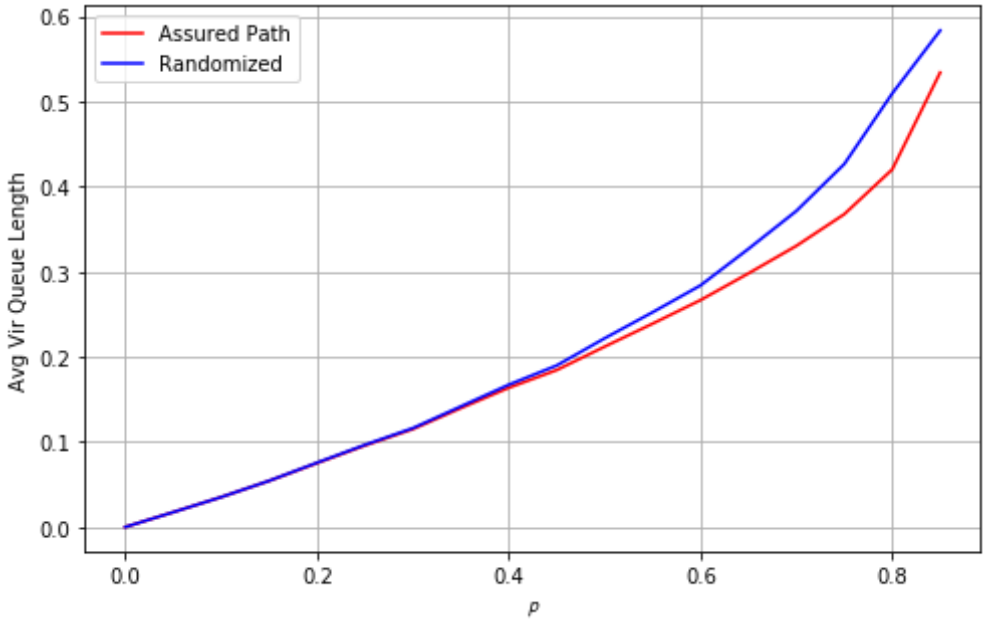

Figure 5.5: **APM and RAND in wired unicast setting.** Average queue lengths varying with load factor ρ at $\Theta = 2$. For low load the performance is nearly same and for higher traffic APM has lesser average queue length than randomized policy $\pi_{\mathbf{RAND}}$

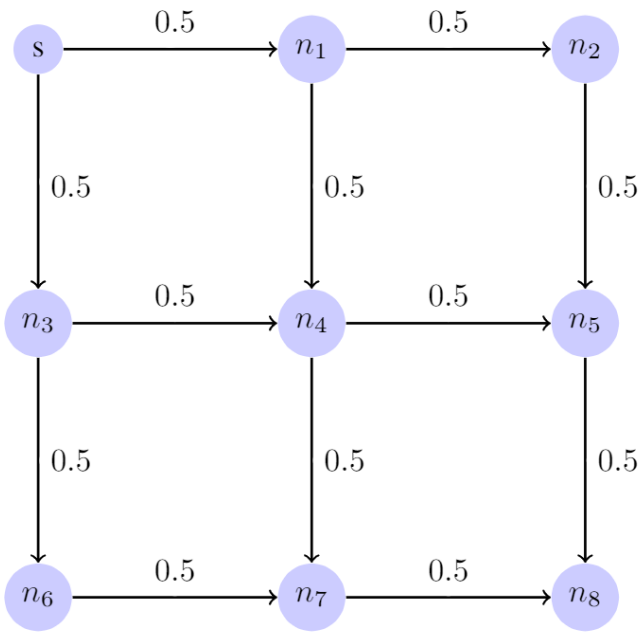

Figure 5.6: 9-node grid used for broadcast simulation. Under wired topology, $\eta_e = 0.5, \forall e \in E$ and $\gamma_e = 1, \forall e \in E$. Broadcast capacity λ^* = 0.5

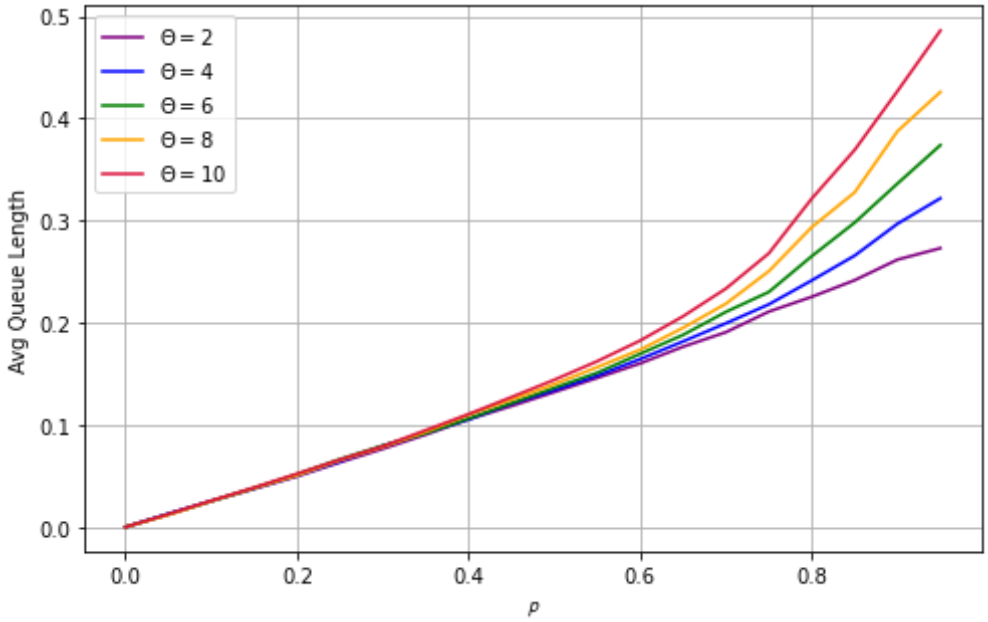

Figure 5.7: **Assured Path Method- Wired Broadcast setting.**
Average queue lengths varying with load factor ρ for different values of Θ. For low load the performance is nearly same.

CHAPTER 6

CONCLUSION

We see that *Assured Path Method*(APM) allows for a throughput optimal routing scheme in a QKD Network. We define an appropriate Network Layer Capacity region that considers the limitation put forward by QKD's low key generation rate as compared with the data rate and prove the achievability of this capacity region by APM. Number of keys determine the eligibility of an edge to be considered active, but once above the defined threshold, neither the number of unused keys nor the key generation rates of an edge play a role in determining the route. The routing problem is solved by Lyapunov Drift minimization which leads to packet routing similar to that of Universal Max-Weight(UMW) scheme [Sinha and Modiano (2017)]. The optimization is purely based on stabilising the virtual queues and not the key queues, as we do not put a cap on the number of keys that can be generated in each edge. UMW has advantages over the popular Back-Pressure policy in terms of delay properties, state-complexity reduction, efficient implementation and capability to be extended to the generalised setting. The policy is better off in delay properties when compared to a randomised policy based on flow decomposition as per our simulations. The research on the topic can be further extended by designing it appropriately in the wireless topology as well so that it can be implemented in satellite based QKD networks.

APPENDIX A

APPENDIX

A.1 Proof of Converse of Theorem 1

Consider any admissible arrival rate vector $\boldsymbol{\lambda} \in \boldsymbol{\Lambda}(\mathcal{G}, \mathcal{C})$. By definition, there exists an admissible policy $\pi \in \Pi$ which supports the arrival vector $\boldsymbol{\lambda}$. Without any loss of generality, we may assume the policy π to be stationary and the associated DTMC to be ergodic. Let $A_i^{(c)}(0,t)$ represent the number of class-c packets that have arrived at their destination along the route $T_i^{(c)} \in \mathcal{T}^{(c)}$ up to time t. Each packet is routed along one admissible route only. Thus we can say:

$$\sum_{T_i^{(c)} \in \mathcal{T}^{(c)}} A_i^{(c)}(0,t) = R^{(c)}(t) \tag{A.1}$$

where $R^{(c)}(t)$ represents the number of distinct class-c packets received by all destination nodes under the action of policy π, up to time t.

We also know that if $A^{(c)}(0,t)$ represents the total number of class-c packet arrivals to the source $s^{(c)}$ up to time t, then:

$$A^{(c)}(0,t) \geq \sum_{T_i^{(c)} \in \mathcal{T}^{(c)}} A_i^{(c)}(0,t) \tag{A.2}$$

as any packet p which has finished its routing along some route $T_i^{(c)} \in \mathcal{T}^{(c)}$ by the time t, must have arrived at the source before that.

By dividing the inequality (A.2) by t and taking limit $t \to \infty$, we have:

$$\begin{aligned}
\lim_{t\to\infty} \frac{A^c(0,t)}{t} &\geq \liminf_{t\to\infty} \frac{1}{t} \sum_{T_i^{(c)} \in \mathcal{T}^{(c)}} A_i^{(c)}(0,t) \\
&\overset{(a)}{=} \liminf_{t\to\infty} \frac{R^{(c)}(t)}{t} \\
&\overset{(b)}{=} \lambda^{(c)}
\end{aligned}$$

The equality (a) holds from Eqn. (A.1) and equality (b) holds from Definition (1) and the fact that our policy $\pi \in \Pi$ is said to support the arrival rate. By using SLLN we can say that

$$\lambda^{(c)} = \lim_{t\to\infty} \frac{A^c(0,t)}{t}$$

From this we can conclude that w.p. 1

$$\lim_{t\to\infty} \frac{1}{t} \sum_{T_i^{(c)} \in \mathcal{T}^{(c)}} A_i^{(c)}(0,t) = \lambda^{(c)}, \quad \forall c \in \mathcal{C} \tag{A.3}$$

Now we use the fact that the policy π is stationary and the associated DTMC is ergodic. Thus the time-average limits exist and they are constant a.s.. For all $T_i^{(c)} \in \mathcal{T}^{(c)}$ and $c \in \mathcal{C}$, define

$$\lambda_i^{(c)} \overset{\text{(def)}}{=} \lim_{t\to\infty} \frac{A_i^{(c)}(0,t)}{t} \tag{A.4}$$

Using Eqns. (A.3) and (A.4) we get

$$\lambda^{(c)} = \sum_{T_i^{(c)} \in \mathcal{T}^{(c)}} \lambda_i^{(c)} \tag{A.5}$$

This proves Eqn. (3.5), that there exists a non-negative flow decomposition of incoming packets amongst the existing routes.

For the second part of proof, we consider an edge $e \in E$ in the graph $\mathcal{G}$. We have that:

$$\sum_{\substack{(i,c):e\in T_i^{(c)},\\ T_i^{(c)}\in\mathcal{T}^{(c)}}} A_i^{(c)}(0,t) \;\le\; A_e(0,t) \overset{(a)}{\le} \sum_{\tau=0}^{t} K_e(\tau) \tag{A.6}$$

and

$$\sum_{\substack{(i,c):e\in T_i^{(c)},\\ T_i^{(c)}\in\mathcal{T}^{(c)}}} A_i^{(c)}(0,t) \;\le\; A_e(0,t) \overset{(b)}{\le} \sum_{\tau=0}^{t} \gamma_e(\tau) \tag{A.7}$$

where left-most sides of both inequations (A.6) and (A.7) represent the number of packets which have finished routing till time t which has passed through edge e. This will be a lower-bound on $A_e(0,t)$, which is the number of packets that have crossed edge e till

time slot t by the policy π. The inequality (a) in Eqn. (A.6) arises from the fact that total number of quantum keys generated in the edge e up to time t will be an upper bound for the number of packets that can cross the edge till time t. Similarly The inequality (b) in Eqn. (A.7) arises from the fact that the number of packets that have crossed edge e till time t cannot be greater than the maximum physical capacity of the link in each time slot, summed up till time t.

Combining the inequations (A.6) and (A.7) with the conjunctive operation and using the fact that edge capacity γ_e is a constant invariant with time, we get a tighter bound:

$$\sum_{\substack{(i,c):e\in T_i^{(c)},\\ T_i^{(c)}\in\mathcal{T}^{(c)}}} A_i^{(c)}(0,t) \ \leq \min\left(\sum_{\tau=0}^{t} K_e(\tau), \sum_{\tau=0}^{t}\gamma_e\right) \tag{A.8}$$

Dividing both sides by t and taking limit to ∞ we get

$$\lim_{t\to\infty}\sum_{\substack{(i,c):e\in T_i^{(c)},\\ T_i^{(c)}\in\mathcal{T}^{(c)}}} \frac{A_i^{(c)}(0,t)}{t} \leq \min\left(\lim_{t\to\infty}\frac{\sum_{\tau=0}^{t}K_e(\tau)}{t}, \lim_{t\to\infty}\frac{\gamma_e t}{t}\right) \tag{A.9}$$

Using Eqn. (A.4) on LHS and SLLN on first term in the RHS of Eqn. (A.9), we get

$$\sum_{\substack{(i,c):e\in T_i^{(c)},\\ T_i^{(c)}\in\mathcal{T}^{(c)}}} \lambda_i^{(c)} \leq \min(\eta_e,\gamma_e) = \omega_e \tag{A.10}$$

By the definition in Eqn. (3.6) we see that the condition that no edge shall be overloaded will translate to

$$\lambda_e \leq \omega_e \tag{A.11}$$

Thus we have proven the converse that any admissible rate will fall inside $\overline{\mathbf{\Lambda}}$.

A.2 Bounds for Multicommodity Unicast Capacity Region Boundary

No policy can support a rate higher than the maximum physical flow capacity of the network. From multicommodity max-flow min-cut theorem [Leighton and Rao (1999)] we can say that the maximum flow is upper-bounded by the minimum cut of the graph, such that:

$$(s^{(c)} \in U \wedge t^{(c)} \in \tilde{U}) \vee (s^{(c)} \in \tilde{U} \wedge t^{(c)} \in U), \quad \forall c \in \mathcal{C} \tag{A.12}$$

where U is any subset of V and $\tilde{U} = V - U$, such that any edge $< U, \tilde{U} >$ will be a cut of the network. The above condition states that the cut in consideration should not have the source and its corresponding destination(s) in the same subset.

Using this theorem we get the following;

$$\sum_{c \in \mathcal{C}} R^{(c)}(t) \leq \min_{\mathcal{C}^*} \sum_{e \in \mathcal{C}^*} \left(\sum_{\substack{(i,c): e \in T_i^{(c)}, \\ T_i^{(c)} \in \mathcal{T}^{(c)}}} A_i^{(c)}(0,t) \right) \tag{A.13}$$

The left-hand side of the inequation corresponds to the total flow represented as the sum of all packets received at their corresponding destination nodes up to time t, and the right-hand side is the minimum cut of the graph $\mathcal{G}$ that follows condition (A.12). $\mathcal{C}^*$ is any cut-set $< U, \tilde{U} >$ of $\mathcal{G}$ that follows condition (A.12).

From the flow decomposition in (A.1) and using inequality (A.8), we can modify (A.13) as follows:

$$\begin{aligned} \sum_{c \in \mathcal{C}} R^{(c)}(t) &= \sum_{c \in \mathcal{C}} \sum_{T_i^{(c)} \in \mathcal{T}^{(c)}} A_i^{(c)}(0,t) \\ &\leq \min_{\mathcal{C}^*} \sum_{e \in \mathcal{C}^*} \left(\min \left(\sum_{\tau=0}^{t} K_e(\tau), \sum_{\tau=0}^{t} \gamma_e(\tau) \right) \right) \end{aligned}$$

Dividing either sides with t and taking limit to ∞ we get the LHS to be:

$$\lim_{t\to\infty}\frac{1}{t}\sum_{c\in\mathcal{C}}\sum_{T_i^{(c)}\in\mathcal{T}^{(c)}}A_i^{(c)}(t)=\sum_{c\in\mathcal{C}}\lim_{t\to\infty}\frac{1}{t}\sum_{T_i^{(c)}\in\mathcal{T}^{(c)}}A_i^{(c)}(t)$$
$$\overset{(b)}{=}\sum_{c\in\mathcal{C}}\lambda^{(c)}$$

and the RHS to be:

$$\lim_{t\to\infty}\frac{1}{t}\min_{\mathcal{C}^*}\sum_{e\in\mathcal{C}^*}\left(\min\Big(\sum_{\tau=0}^{t}K_e(\tau),\sum_{\tau=0}^{t}\gamma_e(\tau)\Big)\right)$$
$$=\min_{\mathcal{C}^*}\sum_{e\in\mathcal{C}^*}\left(\min\Big(\lim_{t\to\infty}\frac{1}{t}\sum_{\tau=0}^{t}K_e(\tau),\lim_{t\to\infty}\frac{1}{t}\sum_{\tau=0}^{t}\gamma_e(\tau)\Big)\right)$$
$$\overset{(c)}{=}\min_{\mathcal{C}^*}\sum_{e\in\mathcal{C}^*}\left(\min\Big(\eta_e,\gamma_e\Big)\right)$$

Equation (b) comes from (A.3), and equation (c) from the fact that $gamma_e$ which represents the physical capacity of an edge is a constant and $\gamma_e(t)=\gamma_e$ for all t. Also, using SLLN we have taken

$$\lim_{t\to\infty}\frac{1}{t}\sum_{\tau=0}^{t}K_e(\tau)=\eta_e$$

Now using this we can re-write the inequation as:

$$\sum_{c\in\mathcal{C}}\lambda^{(c)}\leq\min_{\mathcal{C}^*}\sum_{e\in\mathcal{C}^*}\Big(\min(\eta_e,\gamma_e)\Big)$$

Thus if we had a graph $\mathcal{G}_\omega$ with edge capacities given by

$$\omega_e=\min(\eta_e,\gamma_e)\quad\forall e\in E$$

then a tight upper bound for network layer capacity region will be given by the Linear Programming solution of the following inequalities

$$\sum_{c\in\mathcal{C}}\lambda^{(c)}\leq\text{min-cut}(\mathcal{G}_\omega)$$
$$\lambda^{(c)}\geq 0\qquad\forall c\in\mathcal{C}$$

The using the main proof in Leighton and Rao (1999), the boundary of the capacity

region will fall in the region:

$$\mathcal{O}\Big(\frac{\text{min-cut}(\mathcal{G}_\omega)}{\log(n)}\Big) \leq \sum_{c\in\mathcal{C}} \lambda^{*(c)} \leq \text{min-cut}(\mathcal{G}_\omega) \tag{A.14}$$

Where λ^* will indicate the boundary of capacity region. In the case of single commodity unicast the maxflow will be exactly equal to the mincut of $\mathcal{G}_\omega$ and we get the result:

$$\begin{aligned}\lambda^* &= \text{min-cut}(\mathcal{G}_{\boldsymbol{\omega}}) \\ &= \sum_{e\in\mathcal{F}} \omega_e\end{aligned}$$

where $\mathcal{F}$ is the minimum edge cut of $\mathcal{G}_\omega$

REFERENCES

1. **Bellman, R.** (1958). On a routing problem. *Quarterly of applied mathematics*, **16**(1), 87–90.

2. **Bourgoin, J.-P.**, **B. L. Higgins**, **N. Gigov**, **C. Holloway**, **C. J. Pugh**, **S. Kaiser**, **M. Cranmer**, and **T. Jennewein** (2015). Free-space quantum key distribution to a moving receiver. *Optics express*, **23**(26), 33437–33447.

3. **Byrka, J.**, **F. Grandoni**, **T. Rothvoß**, and **L. Sanità**, An improved lp-based approximation for steiner tree. *In Proceedings of the forty-second ACM symposium on Theory of computing*. 2010.

4. **Caleffi, M.** (2017). Optimal routing for quantum networks. *IEEE Access*, **5**, 22299–22312.

5. **Chakraborty, K.**, **F. Rozpedek**, **A. Dahlberg**, and **S. Wehner** (2019). Distributed routing in a quantum internet. *arXiv preprint arXiv:1907.11630*.

6. **Chu, Y.-J.** (1965). On the shortest arborescence of a directed graph. *Scientia Sinica*, **14**, 1396–1400.

7. **Cormen, T. H.**, **C. E. Leiserson**, **R. L. Rivest**, and **C. Stein**, *Introduction to algorithms*. MIT press, 2009.

8. **Daemen, J.** and **V. Rijmen** (1999). Aes proposal: Rijndael.

9. **Dianati, M.**, **R. Alléaume**, **M. Gagnaire**, and **X. Shen** (2008). Architecture and protocols of the future european quantum key distribution network. *Security and Communication Networks*, **1**(1), 57–74.

10. **Edmonds, J.** (1967). Optimum branchings. *Journal of Research of the national Bureau of Standards B*, **71**(4), 233–240.

11. **Einstein, A.**, **B. Podolsky**, and **N. Rosen** (1935). Can quantum-mechanical description of physical reality be considered complete? *Physical review*, **47**(10), 777.

12. **Ekert, A. K.** (1991). Quantum cryptography based on bell's theorem. *Physical review letters*, **67**(6), 661.

13. **Elliott, C.** (2002). Building the quantum network. *New Journal of Physics*, **4**(1), 46.

14. **Elliott, C.** (2004). The darpa quantum network. *arXiv preprint quant-ph/0412029*.

15. **Gorlatova, M.**, **P. Kinget**, **I. Kymissis**, **D. Rubenstein**, **X. Wang**, and **G. Zussman**, Challenge: ultra-low-power energy-harvesting active networked tags (enhants). *In Proceedings of the 15th annual international conference on Mobile computing and networking*. 2009.

16. **Huang, D.**, **Y. Zhao**, **T. Yang**, **S. Rahman**, **X. Yu**, **X. He**, and **J. Zhang** (2020). Quantum key distribution over double-layer quantum satellite networks. *IEEE Access*, **8**, 16087–16098.

17. **Huang, L.** and **M. J. Neely** (2012). Utility optimal scheduling in energy-harvesting networks. *IEEE/ACM Transactions on Networking*, **21**(4), 1117–1130.

18. **Jianwei, P.** (2014). Quantum science satellite. , **34**(5), 547–549.

19. **Johnson, D. B.** and **D. A. Maltz**, Dynamic source routing in ad hoc wireless networks. *In Mobile computing*. Springer, 1996, 153–181.

20. **Leighton, T.** and **S. Rao** (1999). Multicommodity max-flow min-cut theorems and their use in designing approximation algorithms. *Journal of the ACM (JACM)*, **46**(6), 787–832.

21. **Lenstra, J. K.** and **A. Rinnooy Kan** (1978). Complexity of scheduling under precedence constraints. *Operations Research*, **26**(1), 22–35.

22. **Li, Q.**, **Y. Wang**, **H. Mao**, **J. Yao**, and **Q. Han** (2020). Mathematical model and topology evaluation of quantum key distribution network. *Optics Express*, **28**(7), 9419–9434.

23. **Liao, S.-K.**, **H.-L. Yong**, **C. Liu**, **G.-L. Shentu**, **D.-D. Li**, **J. Lin**, **H. Dai**, **S.-Q. Zhao**, **B. Li**, **J.-Y. Guan**, *et al.* (2017). Long-distance free-space quantum key distribution in daylight towards inter-satellite communication. *Nature Photonics*, **11**(8), 509–513.

24. **Mehic, M.**, **O. Maurhart**, **S. Rass**, and **M. Voznak** (2017). Implementation of quantum key distribution network simulation module in the network simulator ns-3. *Quantum Information Processing*, **16**(10), 253.

25. **Nauerth, S.**, **F. Moll**, **M. Rau**, **C. Fuchs**, **J. Horwath**, **S. Frick**, and **H. Weinfurter** (2013). Air-to-ground quantum communication. *Nature Photonics*, **7**(5), 382–386.

26. **Neely, M. J.** (2010). Stochastic network optimization with application to communication and queueing systems. *Synthesis Lectures on Communication Networks*, **3**(1), 1–211.

27. **Peev, M.**, **C. Pacher**, **R. Alléaume**, **C. Barreiro**, **J. Bouda**, **W. Boxleitner**, **T. Debuisschert**, **E. Diamanti**, **M. Dianati**, **J. Dynes**, *et al.* (2009). The secoqc quantum key distribution network in vienna. *New Journal of Physics*, **11**(7), 075001.

28. **Raghunathan, V.**, **A. Kansal**, **J. Hsu**, **J. Friedman**, and **M. Srivastava**, Design considerations for solar energy harvesting wireless embedded systems. *In IPSN 2005. Fourth International Symposium on Information Processing in Sensor Networks, 2005.*. IEEE, 2005.

29. **Sasaki, M.**, **M. Fujiwara**, **H. Ishizuka**, **W. Klaus**, **K. Wakui**, **M. Takeoka**, **S. Miki**, **T. Yamashita**, **Z. Wang**, **A. Tanaka**, *et al.* (2011). Field test of quantum key distribution in the tokyo qkd network. *Optics express*, **19**(11), 10387–10409.

30. **Schrödinger, E.**, Discussion of probability relations between separated systems. *In Mathematical Proceedings of the Cambridge Philosophical Society*, volume 31. Cambridge University Press, 1935.

31. **Shannon, C. E.** (1949). Communication theory of secrecy systems. *The Bell system technical journal*, **28**(4), 656–715.

32. **Sinha, A.** and **E. Modiano** (2017). Optimal control for generalized network-flow problems. *IEEE/ACM Transactions on Networking*, **26**(1), 506–519.

33. **Sinha, A.** and **E. Modiano** (2019). Throughput-optimal broadcast in wireless networks with point-to-multipoint transmissions. *IEEE Transactions on Mobile Computing*.

34. **Stucki, D.**, **M. Legre**, **F. Buntschu**, **B. Clausen**, **N. Felber**, **N. Gisin**, **L. Henzen**, **P. Junod**, **G. Litzistorf**, **P. Monbaron**, *et al.* (2011). Long-term performance of the

swissquantum quantum key distribution network in a field environment. *New Journal of Physics*, **13**(12), 123001.

35. **Takeoka, M.**, **E. Kaur**, **W. Roga**, and **M. M. Wilde** (2019). Multipartite entanglement and secret key distribution in quantum networks. *arXiv preprint arXiv:1912.10658*.

36. **Tang, Y.-L.**, **H.-L. Yin**, **Q. Zhao**, **H. Liu**, **X.-X. Sun**, **M.-Q. Huang**, **W.-J. Zhang**, **S.-J. Chen**, **L. Zhang**, **L.-X. You**, *et al.* (2016*a*). Measurement-device-independent quantum key distribution over untrustful metropolitan network. *Physical Review X*, **6**(1), 011024.

37. **Tang, Z.**, **R. Chandrasekara**, **Y. C. Tan**, **C. Cheng**, **L. Sha**, **G. C. Hiang**, **D. K. Oi**, and **A. Ling** (2016*b*). Generation and analysis of correlated pairs of photons aboard a nanosatellite. *Physical Review Applied*, **5**(5), 054022.

38. **Tassiulas, L.** and **A. Ephremides**, Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks. *In 29th IEEE Conference on Decision and Control*. IEEE, 1990.

39. **Van Meter, R.**, *Quantum networking*. John Wiley & Sons, 2014.

40. **Xu, F.**, **W. Chen**, **S. Wang**, **Z. Yin**, **Y. Zhang**, **Y. Liu**, **Z. Zhou**, **Y. Zhao**, **H. Li**, **D. Liu**, *et al.* (2009). Field experiment on a robust hierarchical metropolitan quantum cryptography network. *Chinese Science Bulletin*, **54**(17), 2991–2997.

41. **Zhou, H.**, **K. Lv**, **L. Huang**, and **X. Ma** (2019). Security assessment and key management in a quantum network. *arXiv preprint arXiv:1907.08963*.