Application Aware Routing in Multi-Hop Wireless Network using OpenFlow

By
Muhammad Adil Iqbal
2010-NUST-MS-PHD-IT-11

Supervisor
Dr Adeel Baig
NUST-SEECS

Committee Members
Dr Saad Bin Qaisar
Dr Adnan Khalid Kiyani
Dr Kashif Sharif

School of Electrical Engineering and Computer Science,
National University of Sciences and Technology (NUST),
Islamabad, Pakistan.

(September, 2014)

Approval

It is certified that the contents and form of the thesis entitled "**Application Aware Routing in Multi-Hop Wireless Network using OpenFlow**" submitted by **Muhammad Adil Iqbal** have been found satisfactory for the requirement of the degree.

Advisor: **Dr Adeel Baig**

Signature: _____

Date: _____

Committee Member 1: **Dr Saad Bin Qaisar**

Signature: _____

Date: _____

Committee Member 2: **Dr Adnan Khalid Kiyani**

Signature: _____

Date: _____

Committee Member 3: **Dr Kashif Sharif**

Signature: _____

Date: _____

Dedication

I would like to dedicate my thesis to my beloved son TAHA ADIL RAO

Certificate of Originality

I hereby declare that this submission is my own work and to the best of my knowledge it contains no materials previously published or written by another person, nor material which to a substantial extent has been accepted for the award of any degree or diploma at National University of Sciences & Technology (NUST) School of Electrical Engineering & Computer Science (SEECS) or at any other educational institute, except where due acknowledgement has been made in the thesis. Any contribution made to the research by others, with whom I have worked at NUST SEECS or elsewhere, is explicitly acknowledged in the thesis.

I also declare that the intellectual content of this thesis is the product of my own work, except for the assistance from others in the project's design and conception or in style, presentation and linguistics which has been acknowledged.

Author Name: **Muhammad Adil Iqbal**

Signature: _____

Acknowledgment

First of all I would like to thank Almighty Allah for giving me strength and courage to complete my degree.It is not possible to do anything with out Allah will and blessing.

I am thankful and owe my deepest gratitude to my thesis supervisor Dr. Adeel Baig for his guidance,support and help not only in academics but also related to other issues of life which I faced throughout every stage of my thesis. I am also thankful to my committee members Dr. Adnan Kiani, Dr Kashif Sharif and Dr.Saad Bin Qaisar in helping me and sparing time to review me work. Special thanks to Miss Bushra Sadia for her support in thesis and overall support in my MS degree and Mr. Amer Shahzad for his motivation he has provided to me.

I would like to thank my mother for her support throughout my educational career, specially in my Master degree. She has always supported and encouraged me to do my best in all matters of life. I would also like to thank my brothers and sisters for their love and prayers for the successful completion of this work.

Muhammad Adil Iqbal

Abstract

Conventional routing algorithms in multi-hop wireless networks do not consider application characteristics in terms of bandwidth and latency while making routing decisions. There is prolific increase in real time applications over the wireless networks. Some of these applications may require a certain bandwidth, for exchange of data for a specific amount of time.Some applications may also require a minimum end-to-end latency. For such applications, the use of IEEE 802.11 access technology application aware routing can be managed efficiently by using software defined network technology.

The biggest challenge this research addressed was how to use software defined networking with existing network devices. This thesis focuses on the application ware routing in the wireless networks using OpenFlow. Most of the attention has been given on the proposed routing algorithm that forwards traffic on the path most suitable to the applications. Routing metric (cost) is being calculated on the basis of dynamic capacity of each link on all paths and hop-count from source to destination. Capacity of each link is based on the traffic being forwarded, which changes with the addition and removal of flow. Our algorithm selects the appropriate path based on the analysis of flow. Controller from a given flow identifies a particular application and then decides the specific bandwidth and delay requirements of that applications. As the path selection is based on the capacity available on each link and the

capacity is dynamic based on the current traffic. Finding the appropriate route and subsequently rerouting the traffic was a major challenge. Our algorithm calculates all possible paths between source and destination along-with their available capacity. Then finds the most appropriate path from the calculated paths which satisfies application requirements.

Contents

List of Figures

Chapter 1

Introduction

In wireless networks, it is very critical to allocate bandwidth for real time applications as it affects all participating nodes in a given network. Strict requirement of guaranteed bandwidth allocation can be a reason of unavailability of network resources to other nodes. The problem is how to efficiently allocate a specified bandwidth to a certain node so that other nodes can use all the remaining bandwidth for their operation.To have better throughput in a network, the routers need to have a global view of the current state of entire network, which is not possible without a centrally coordinated network.Reactive or adaptive routing mechanism is more suitable for such networks. Software Defined Networking can be used as solution to implement a network management and routing protocol. As compare to SDN, flow based routing implemented using traditional routing methods is difficult. On the other hand, OpenFlow, an emerging technology, makes it easy to complete this task. OpenFlow can be used to manage high throughput flows, hence achieving objective of bandwidth sensitive application to perform better. Centrally placed OpenFlow controller can solve the issue of reliability of service being provided and guaranteed bandwidth requirement.

1.1 Wireless Network

Wireless networks are types of computer networks that use air as medium for data communication to make connections with other network nodes. Different types of wireless networks are defined as under:

1.1.1 Ad hoc Wireless Network

A special type of wireless network called Mobile Ad-hoc NETwork (MANET) is characterized by infrastructure-less, independent devices, with out a central control. Each node acts as a router for its own traffic as well as for other nodes. The biggest challenge in designing and building a MANET is to maintain routing information with limited power consumption.

1.1.2 Wireless Sensor Network

Wireless networks which are categorized as Wireless Sensor Networks (WSN) are established to gather sensor data, like weather temperature, fire detection, machine health monitoring etc. These are low-cost, low-power multi functional devices. When large number of sensors are deployed, they automatically form ad hoc multi-hop network to communicate with each other to reach at a sink node.

1.1.3 Wireless Mesh Network

A category of wireless networks refers to as Wireless Mesh Networks(WMN) are formed by the mesh topology. The nodes in mesh network can be Laptop, PDA or any other small wireless devices. Nodes are called mesh clients and some nodes act as mesh router or gateway. Wireless mesh network is considered reliable network, consisting of backups and redundancy of links. Mesh

network can be formed using 802.11, 802.15 or 802.16. For the management purpose mesh network can be with no central server or it can be managed by a central server. Application of mesh network can be military in field operations, electric meter reading with-out human interaction. Routing in wireless mesh network is similar in many aspects to routing in wired network [2]. This research being carried out is more close to wireless mesh networks.

1.1.4 Stand-Alone Wireless Network

Stand-alone wireless network refer to WLAN where very few wireless devices communicate via access point. Normally these types of networks are setup where a network is shared from main office or building to sub-office or a temporary location.

1.1.5 Centrally Controlled Wireless Network

Centrally coordinated WLAN refer to network of multiple of small wireless coverage areas. Centrally coordinated network benefits from the centralized management and high availability. Coordinated AP can provide better benefits in enhancement of quality of service and traffic-sensitive application.

1.2 Routing Mechanism in Wireless Networks

Most of the wireless routing protocols available are based on adapting routing where router has complexity of load balancing but enjoys the better throughput and low latency. There are many types of routing protocols available for wireless networks, each suitable for the different wireless topologies. Methods of routing include:

- Reactive routing
- Proactive routing
- Hybrid routing

Different protocols are used in wireless mesh network infrastructure like the Ad-hoc On demand Distance Vector (AODV), Optimized Link State vector Routing protocol (OLSR),Open shortest path first (OSPF), and also protocols like DOS [3]. This routing protocol is used for the instance in a mesh network approach. Like the DOS, the SLRP [4] also maintains the multi loop free paths by using the abstract node label. There are three main multi path strategies:

- Unipath Routing (UNI)
- Link Quality Minimum Distance Weighted (LQMDW)
- Link Quality Distance Weighted (LQDW)

Unipath Routing (UNI) uses a single minimum hop count path, Link Quality Minimum Distance Weighted (LQMDW) uses minimum distance paths and distributes the traffic load over different minimum-hop path. The LQDW uses all path distances but distributes the traffic using a joint distance and link quality function over each path. The multi path protocols first discovers the loop free paths and reports by using the RREQ/RREP relaying rules [5]. This rule is also same as [3], except that here the middle node sends the RREQs and the base accepts the RREQ on the loop free order. Certain number of multi paths is maintained so that if the link is broken, the source node will find new a path. The above multi-path protocols do not consider application awareness as an integral part of their algorithms. In this research

algorithm bo proposed is not only multi-path, but also provides application awareness.

1.3 Application Aware Routing

i In Traditional Routing, there are no mechanisms for the centralized request router to communicate with the L3 router to dynamically increase the bandwidth available for a certain application. In this case the network is always provisioned for peak bandwidth usage, resulting in unused bandwidth or certain application run under low bandwidth availability. The latency and bandwidth requirements of the applications are not taken into consideration by L3 routers when routing the requests. Network equipment are agnostic to application being served by the network. Traditional network forward packet on minimum hop count.

While in application-aware routing, network can be configured dynamically to provide service differentiation. Intelligence is built in cross layers to gain the improved user experience. Application aware routing can find high throughput route or low latency path dependent on application. Application aware routing can be provided by both adaptive and oblivious routing. It provides QoS, security and network visualization.

1.4 Software Defined Networks

In traditional networking environments, computers connect through switches and routers across the globe. Networking switch device only perform the function of packet switching where as routers perform both switching and routing function. Figure 1.1 shows typical network device.

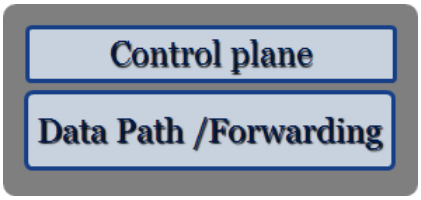

Figure 1.1: Typical Network Device

Switching means physically peering the ports (communication links) and making junctions of and establishing a communication path. Whereas routing means learning the new path and making forwarding decision. Software defined networking separated these two function, namely control plane and data forwarding plane. Network intelligence become centralized in software defined networking. Due to these features SDN enable enterprises to control their networks as per their requirements through programmable networks. It facilitates the deployment of new applications and services across the network with minimal effort. In March 2011 Open Networking Foundation was established for the purpose to standardize the software defined networking. Interest in software defined networking is increased due to better management and smarter networks [6]. Figure 1.2 shows network device with SDN.

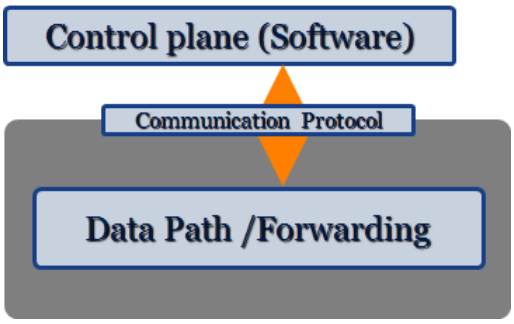

Figure 1.2: Network Device with SDN

1.5 OpenFlow

OpenFlow is designed to run experiments in the networks. It is the first software defined network technology that networking vendors agreed to implement. Most network switches and routers contain flow tables. A switch can only be an OpenFlow switch or hybrid switch if it provides functionality of both traditional as well SDN network. OpenFlow researcher exploited this feature of devices and built flow tables as basic building block of OpenFlow as shown in [7].

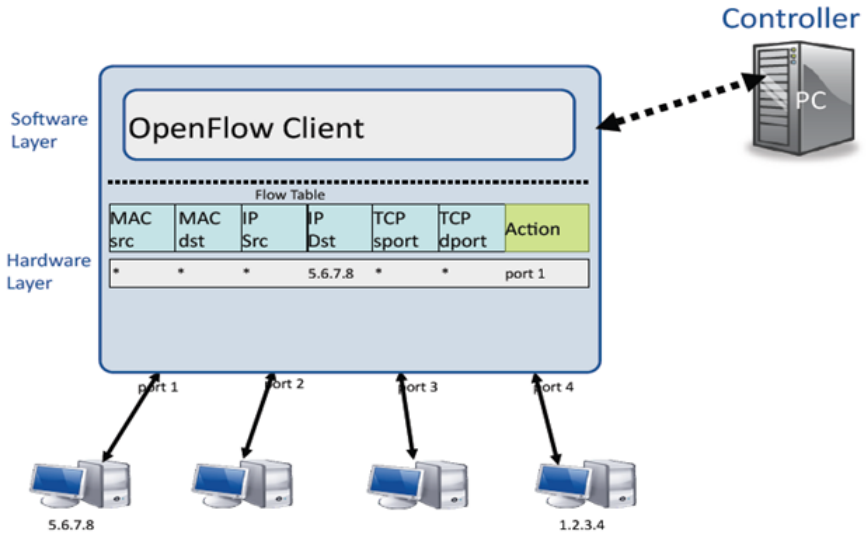

Figure 1.3: Basic Building Block of OpenFlow [1]

OpenFlow protocol determines the structure of communication between the data plane and control plane of supported devices. OpenFlow specification Version 1.3.0 has been released on 25th June 2012. The OpenFlow version which is mostly used is OpenFlow 1.0, which was introduced in December 2009. This version defines the core of OpenFlow into three parts.

- OpenFlow controller

- OpenFlow secure communication channel
- OpenFlow forwarding database called Flow Tables.

1.6 OpenFlow Development Tools

Main objective of OpenFlow was to provide a method to researcher to test their algorithms and protocols. With the passage of time new development tools are being introduces in academia and industry. Authors of [8] surveyed the past and present SDN tools and SDN architecture. They have also highlighted the potential future applications for SDN. Some of popular tools for SDN are discussed below:

1.6.1 Emulation and Simulation

Mininet

Mininet is light weight emulation tool to test entire OpenFlow network. It is open source tool and learning curve to this tools is easy.

Estinet

Estinet is both emulation and simulation tools for OpenFlow but it required licensing before to use.It can use real world OpenFlow controller like Nox, and Pox.

NS-3

NS-3 is widely used simulation tools in networks. OpenFlow protocol is also supported by NS-3 version 0.89 which is quite old.

1.6.2 OpenFlow Controller

POX

POX is general, open-source SDN controller written in Python.

Beacon

Beacon is an OpenFlow controller developed for cross-platform. It is modular and has been developed in Java language. Beacon controller supports event and threaded operations.

Floodlight

Floodlight controller is also developed in Java like Beacon. Floodlight is based on the Beacon implementation. Advantage of Floodlight includes, but not limited to, that it works with both physical and virtual OpenFlow switches. Floodlight is supported and enhanced by open community of developers with major number of software engineers from big switch networks. OpenFlow specifies set of rules called protocols that can modify the flow table of OpenFlow switch devices using standard OpeFlow API which are provided by ONF. These API set flow tables by remote controller. Floodlight is designed in such a way that it can work with large number of OpenFlow switches, access points, routers and virtual switches that can support open standards. Working process of floodlight is shown in Figure 1.4. Certain advantages of floodlight include [9]:

- It is very easy to install and configure with minimal dependencies.
- It offers a module loading system that is used to make it simple to extend and enhance.

- It supports different model and version of virtual as well as physical Openflow switches.
- It can handle hybrid OpenFlow switches which are designed for both conventional and open Flow based switching and OpenFlow networks.
- It is designed to give high performance and Big Switch commercial product use floodLight controller as core or as base for their project.

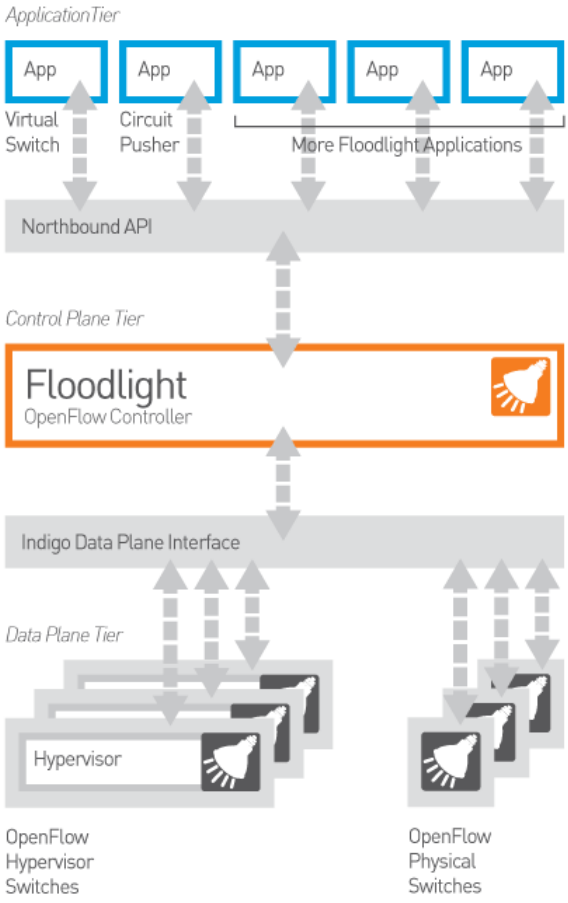

Figure 1.4: Working Principle of Floodlight

1.7 Thesis Contributions

This research work is aimed to build an application aware routing protocol for multi-hop wireless network using OpenFlow. The goal of this research was to provide a solution that can be used in multi hop wireless network with minimum change in the devices network stack. Some assumptions are makde t which are not affecting the algorithm development.

Assumption 1: Application would be identified by a third party algorithms, and their requirements would be know in advance to our routing algorithms.

Assumption 2: Link capacity (throughput) will be knows by third party algorithm e.g ETX.

Interference in network will yield low capacity on certain links but as taking assumption 2 we are dependent on capacity of link for third party algorithm we do not need to worry about it. Third party algorithm needs to take care of this. Output may yield a significant improvement in bandwidth reservation and hence providing more throughput to application. With the use of the software defined networking (OpenFlow), additional cost to implement and maintain the network is negligible.

1.8 Thesis Organization

In this thesis there are six chapters. The organization of chapters are as under. This is thesis chapter one and in chapter two existing application aware routing protocols are discussed. Different tools for SDN are also discussed in Chapter 2. In Chapter 3, a work model is explained in detail and introduction to the techniques used to build the algorithm is explained. In Chapter 4, design and implementation of Route Selection based on Application Characteristics (RSAC) algorithm is discussed in details. In Chapter 5,

Simulation results are given and explained. Chapter 6 includes conclusion and proposes some possible extensions to this work. In the end, references to related work are given.

Chapter 2

Literature Review

Different service providers are exploring different avenues for optimizing CAPEX/OPEX and differentiating themselves with the help of improving user experience. SDN provides an opportunity to these service providers for a programmable interface to control their networks based on the underlying applications. Based on an application's characteristics and requirements it is possible to control network switches dynamically with the help of SDN. It leads towards better overall user experience and reduction in bandwidth wastage.

This chapter presents review on existing application awareness in traditional networks, wireless network QoS, use of Software Defined Networks (OpenFlow) in wireless network. At the end this chapter highlights the motivation for this research by distinguishing the work which has already been done and the work done in this research.

2.1 Bandwidth Reservation Protocol for Quality of Service Routing in a Multi-hop Wireless Network

In [10], bandwidth reservation problem in a mobile ad hoc network (MANET) is explained in order to support QoS (Quality of Service) routing. It focuses on the bandwidth reservation problem by assuming that a common channel is shared by all hosts under a Time Division Multiple Access(TDMA) channel model. In literature, this problems is addressed by taking an assumption of a stronger multi-antenna model. In this model, bandwidth of a link is not dependent on transmitting / receiving activities of its neighboring link. The authors proposed a new protocol that can reserve routes by addressing hidden-node terminal as well as exposed-terminal problems using TDMA based bandwidth reservation protocol for QoS in MANET. As it is considering both the hidden-node terminal as well as exposed-node terminal, therefore more accurate route bandwidth can be calculated and previous wireless bandwidth can be utilized in a better way.

In [11], authors presented the model for bandwidth reservation problem in wireless networks. Author highlighted that multimedia flows have certain bandwidth requirements which need to be guaranteed as QoS.

Another paper [12] discusses bandwidth reservation by lowering end-to-end delay QoS routing and admission control. It is based on AODV with no central command available.

In [13], a survey of QoS based routing solutions for mobile and ad-hoc network has been carried out. The major challenge discussed in this survey paper is the lack of centralized control.

The Expected Transmission count metric (ETX), given in [14], is used

to determine high throughput paths using multi-hop wireless networks. The expected total number of packet transmissions is minimized by ETX that is required to successfully deliver a packet to the ultimate destination. Different ETX metrics are incorporated by the effects of link loss ratios, asymmetry in the loss ratios between the two directions of each link and interface among the successive links of path. Whereas minimum hop count metric chooses different paths of same length arbitrarily without bothering about the large differences in throughput among those paths. It also ignores the possibility of offering higher throughput on a longer path. Implementation as well as design of ETX as a metric has been explained by the author for DSDV and DSR routing protocols. It also explained the modification of DSDV and DSR routing protocol which allows them to use ETX. Values taken from 29 nodes using 802.11 test-beds demonstrated very poor performance in case of minimum hop count. ETX improves performance of the nodes.

2.2 OpenFlow for Wireless Networks

The use of SDN in wireless networks is highlighted in [2]. For wireless mesh network (WMN), several protocols are proposed including AODV, OLSR etc. However flow based routing is hard to implement in such networks where flow can adopt different path. Authors proposed an architecture that integrates OpenFlow with WMNs and provide flow based routing and having forwarding capabilities.

In [15], author explained their experiences and challenges they faced while deploying OpenFlow in WMN. Authors of this paper claimed that extending the WMNs with OpenFlow offers many benefits as high level service like mobility. The main challenge they highlighted is the placement of centralized

control of OpenFlow and distributed architecture of WMNs.

Two types of nodes were introduced in wireless mesh network client nodes in [16], mobile and the mesh nodes that are static. Author pointed out that multimedia and real time traffic application require bandwidth reservations. The network under consideration is 802.11s. In [17], author tried to use multi path routing along with admission control in wireless mesh network for the improvement of reliability of services and hence providing the QoS [18].

2.3 Application Aware Routing

The authors in [19] proposed a solution that integrates the SDN and WMN using OpenFlow switches. They used a centralized controller and setup arbitrary path having fine grain control over the network traffic of wireless network. Authors used NS3 to simulate their wmSDN toolkit.

The paper [20] is mainly written to advocate the advantages of SDN in wireless network. The author used the argument such as low cost and reduced complexity, simplifying the network management functions. In the further work authors stated that they will build and simulate wmSDN using NS3. How the data is send in wmSDN network and how control function are implemented in the centralized SDN?

Authors in [21] state that conventional routing algorithms do not consider the bandwidth and latency requirement of application. These algorithms are not aware of the demand of a specific application. The Result of such ignorance is an under-utilization of network resources. Authors claim that they have developed a method for statically and efficiently allocating virtual channel to flows or packet. The developed framework can be used to produce application-aware routes that target the minimization of latency and number

of flows through a link or a combination of both.

2.4 Use of Software Defined Network (SDN)for Application Aware Routing

Several application of software defined networks have emerged during the past few years which started with the introduction of the idea of OpenFlow in [22]. Few year later after the publication of [22] one of the authors of this article published an another article [23]. In this paper the author highlighted the use of OpenFlow for application-awareness. The authors are of the belove that they can use OpenFlow for providing low latency path for VOIP traffic and can create high throughput path for video traffic where low latency and jitter are not important. Authors of [21] made a claim that for best performance in routing through application adaptivity router should have knowledge of current network state. OpenFlow provide central control and it is also applicable in wireless network as we have discussed in 2.2
In this research we have surveyed the research papers which are raising the requirement of application aware routing, bandwidth reservation and admission control. Most of papers refer to Ad-hoc network whereas some papers also refer to sensor networks and wireless mesh network. The main difference between the literature and our area of research is centrally coordinated network which is being provided by SDN. Existing work lacks central control in wireless networks for application awareness.

Chapter 3

Methodology

This chapter briefly explains the aims and objectives of the research being carried out in this thesis. Problem statement is discussed and solution is proposed.

3.1 Aim and Objective of Research

The aim of this research is:

- Diagnose the role of non-traditional application aware routing.
- Understand the role of OpenFlow and central controller in multi-hop wireless networks.
- Propose a mechanisms for application aware routing in multi-hop wireless network depending on user application requirements.

3.2 Problem Formulation

The guaranteed bandwidth is an important factor in modern era of communication. Some applications require guaranteed bandwidth and low end-to-end latency to work in a desired manner. Provision of network resources according to application requirements is a key factor in quality of service. Provision of quality of service in multi-hop wireless networks is more challenging due to mobile nature of networks [24]. Several techniques have been proposed by researcher but none is suitable for all kind of wireless scenarios. QoS technique for wired network is not suitable to use in wireless networks due to added complexity of wireless networks [24].

Various researcher are working on specific custom-built scenarios for QoS requirements in computer networks. In wireless networks, improving the QoS is an endless process due to a large number of scenario and different types of problem for each scenario. In quality of service of wireless network the bandwidth guaranty is major challenges along-with other challengers like mobile network and multi hoping. In this study efforts are made to improve the QoS in centrally coordinated wireless network using OpenFlow.

3.2.1 Traditional Decentralized Mobile Adhoc Network

Figure 3.1 illustrates the traditional decentralized mobile adhoc network. Nodes are connected to each other via any wireless technology. Each node in this network forwards traffic of users connected to this node along-with traffic of other nodes acting as relay node for others. The decision which node should act as relay node for others depends on many factors including but not limited to interference level.

The delays of different paths from a given node to the gateway node

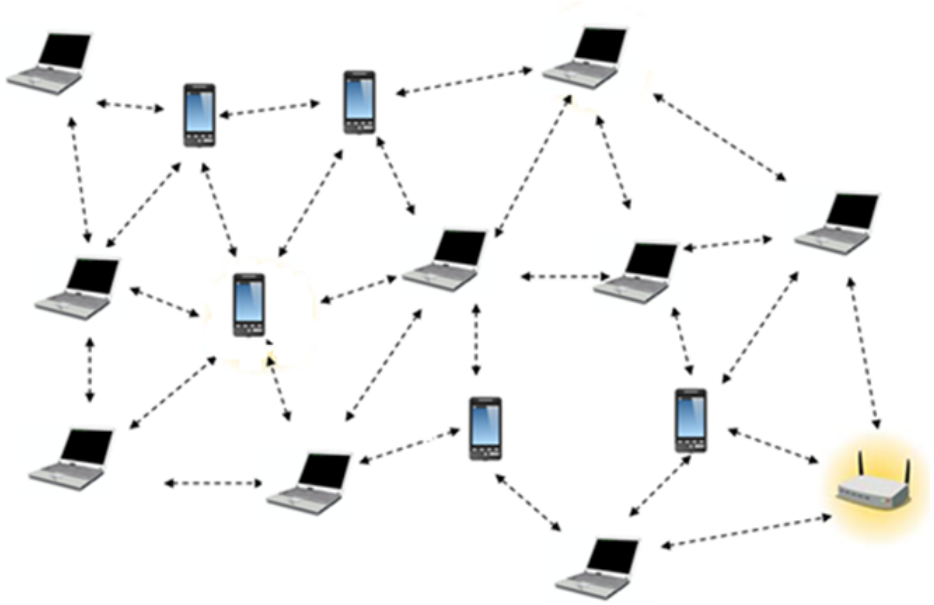

Figure 3.1: Traditional Decentralized Mobile Ad hoc Network

are different. Each node in such a network has information of inward and outward traffic through this node. The next hop is decided based on the amount of traffic through the current node. The data received at a given node has to be fully transmitted to the next node in the the path.

Figure 3.2 constitutes two types of nodes, mobil nodes denoted by symbol of a mobile phone, and laptop nodes denoted by symbol of a laptop. In addition there is a gateway denoted by a AP-router image. Lets assume, mobile node can communicate at maximum bandwidth of 11 M bits whereas a laptop nodes can communicate at maximum bandwidth of 54 M bits.

Lets take a example of communication between client (C) and server (S). In traditional networks, when the path discovery is initiated in it selects the path with minimum hop count as shown in Figure 3.2 with red line (with bars). This path include two mobile nodes. This path is selected regardless of the application type which has requested this path. Underlaying network lacks the capability to diagnose if running application requires a high

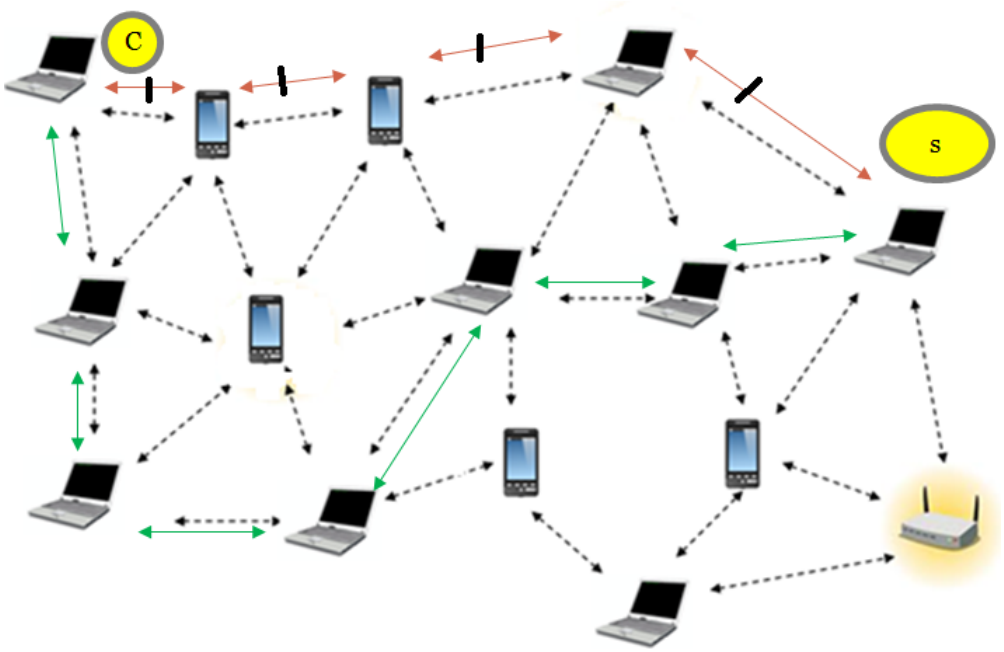

Figure 3.2: High Throughput Path Selection

throughput path as shown in figure 3.2 in green line (with out bar) or shortest path for non bandwidth hungry application.

3.2.2 OpenFlow and Application Aware Routing

In application aware routing, network can be configured dynamically to provide service differentiation. OpenFlow provides centralized control, so it is of great importance in wireless network which is decentralized control. OpenFlow can dynamically provide the network resources, based on application characteristics as shown in Figure 3.3.

3.2.3 Research motivation

A lot of research has been done on the topic of "High Throughput Path". Hundreds of research papers are available in this area, but nearly all of them

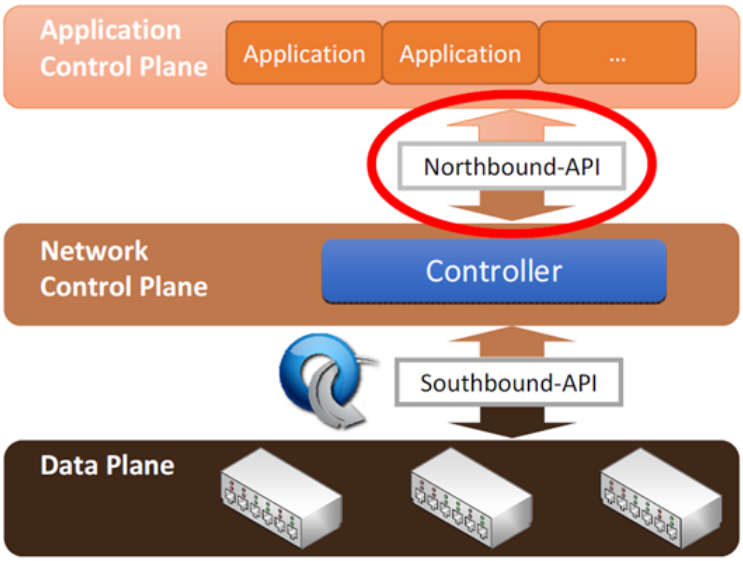

Figure 3.3: OpenFlow and Application Aware Routing

focus on high throughput, QoS, admission control and resource reservation for wireless network without considering application characteristics being run on these networks. The solution provided by these papers are application agnostic. Combining multiple high throughput links may form a high throughput path but at the cost of increased number of hops as well as reduced throughput of the entire network. The high throughput path may not be always desired. There may be some applications which do not need high throughput but require low end-to-end delay in communication.

With the introduction of OpenFlow and increasing number of multi-hop wireless network there is need to address this problem the current research proposes the solution for this problem. The primary objective of this research is to select high throughput path only when it is required based on the application characteristics. Application-aware routing using SDN is ideal choice to address this issues.

3.2.4 Problem Statement

Problem statement for this research thesis is: "What will be the benefit of OpenFlow in non traditional 'application-aware' routing in multi-hop wireless network", where the scope of 'application-awareness' is limited to bandwidth requirement and delay tolerance.

3.2.5 Research Questions

Our study should answer the following research questions which are derived from our problem statement

- How wireless nodes will contact the OpenFlow controller?
- Does the nodes need to change their network stack to support guaranteed bandwidth and for multi-hop forwarding?
- How OpenFlow controller will determine the available network bandwidth?
- How much network overhead will be generated by control messages of our algorithm?
- How OpenFlow controller ensures that other nodes will not go into starvation, if high bandwidth is allocated to one node?

3.3 Proposed Solution

The research word was divided in to three phases. In first phase, a thorough literature survey of existing wireless technologies and application aware routing was conducted along with working of OpenFlow in wireless. Emphasis was given to those modern techniques which are using sensor network or

wireless mesh network.In second phase, a new routing component was implemented in OpenFlow for the proof of concept. In third phase, an experimental virtual network was emulated for acquiring real network statistical data for analysis and conclusion.

For the purpose of quick testing only few parameters, namely protocol, destination port numbers, were used for the identification of applications. A table with protocol, port number, flow duration and required bandwidth was filled and used by our proposed algorithms. A better approach for identification of an application could be deep packet inspection and applications state.

OpenFlow provides API to get network topology, flow statistics, network statistics, and network elements states. We combined these features in a logical manner to get the desire result, e.g in our case finding the network congestion for flow optimization.

This proposed solution is applicable on fixed wireless networks. The mobility variable was not considered in this research, as this would have increased the complexity of solution. Our proposed solution is applicable for networks which are built as last hop for internet provision where no mobility is required and customer use very diversified applications. These applications are characterized by different bandwidth requirements and delay tolerance. Other possible application can be, environment monitoring, video surveillance, etc.

Chapter 4

RSAC Design and Implementation

4.1 RSAC Implementation

Route Selection based on Application Characteristic (RSAC) design is very modular. It is divided in 3 sub algorithms and explained in later sections:

- Capacity Calculation and Assignment explained in Algorithm 4.1
- Multi-path Calculation Algorithm explained in Algorithm 4.2
- Path Optimization, Route Discovery Algorithm explained in Algorithm 4.3

4.2 RSAC Block Diagram

For the identification of an application from the "packet in" message of OpenFlow, static tables are used as shown in Table 4.1.

Application ID	Protocol	Destination port	Average flow duration (In Sec)	Required bandwidth in KB	Delay tolerance
0	Any	Any	15	0 (Best Effort)	Yes
1	TCP	9999	90	1024	Yes
2	TCP	8888	60	2048	Yes
3	UDP	7000	Any	4096	Yes
4	TCP	6000	Any	0	No

Table 4.1: Application Requirement

Applications were identified using Applications Requirement Table 4.1. This table is shown as application database in the RSAC block diagram shown in figure 4.1.

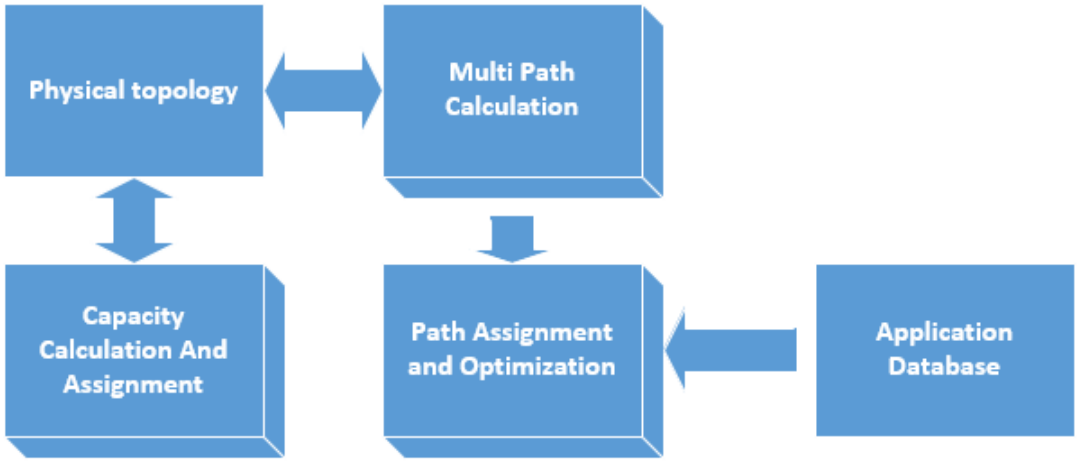

Figure 4.1: Logical View of Link State and Application Requirements

Physical topology of a network is required for all the algorithms. However this physical network topology is dynamic and it changes as the node link changes. It is very important to know the complete network including:

number of OpenFlow switches, their interconnection, nodes connected to switches,link established between the nodes and their link quality in term of bandwidth. OpenFlow enable us to know all these required elements through its various API.

Our selection for OpenFlow controller is Floodlight [9]. It is open source software defined network controller. This controller can work with both virtual and physical OpenFlow switches explained in 1.6.2.

In our implementation for building class of network topology, we used the Floodlight controller services *IFloodlightProviderService*, *ILinkDiscoveryService* and *IDeviceService*. Using these services, we built the full network topology. In our code we used term "Vertex" for the OpenFlow enabled switches and term Edges for the links between vertex.

Our Vertex class is defined as:

```
class Vertex implements Comparable < Vertex >
{
private final IOFSwitch sw;
private Map< Vertex, Edge > adjacencies = new HashMap< Vertex, Edge >
();
public double minDistance = Double.POSITIVE_INFINITY;
public long maxThrougput = Long.MIN_VALUE;
public List< path > pathList = new ArrayList< path > ();
Vertex previous;
}
```

Each Vertex (OpenFlow switch) contains the list of path leading to destination Vertex and this list is updated during algorithms execution mentioned above. We have defined Edge class which contains, source and destination Vertex along with the physical ports on which this link is incident.

4.2.1 Application Identification

Static Table 4.1 is used for the purpose of application identification. This table is built for the application characteristics. Our Algorithm relates to one of the *ApplicationID* from Column 1 of this table for each incoming flow. On arrival of flow, table is searched for the parameters from protocol and destination port. Flow is bounded to one of application ID. JAVA class is used to build this sample table to test our algorithm.

Application identification is not part of this research study. There are several other ways for identification of application such as deep packet inspection, direct input from the application as discussed by authors in [25]. We have used protocol and port no for our application identification in table. Literature survey of [26] also confirm our understanding.

Fourth column of the Table 4.1 was used for average flow duration. This was added to test the route optimization algorithm. When route optimization algorithm is running, it move only those flow which have their run time larger then the average flow duration.

Fifth column was used for bandwidth requirement of particular application. This column is used for finding the path which satisfy this basic requirement.

Last column was used for such application which does not allow delay in communication. A ***No*** written in this column means that application has been routed on shortest path where delay is minimized. This type of application does not bother the bandwidth.

4.2.2 Link Bandwidth Status

Logical Link Bandwidth Status as shown in Table 4.2 is a logical representation of the link status at any given time. Each link in network topology has following attributes which are calculated recursively in the solution.
This table is populated at run time by our algorithm along with capacity calculation and assignment. Our algorithm first identifies all OpenFlow enabled nodes in network. All edges (Links) between these identified nodes are discovered. Network states from nodes provide us with the nodes link max bandwidth, which is then assigned to link maximum bandwidth. Our path assignment algorithm subtracts assigned bandwidth from the link and updates its corresponding attributes.

Vertex-A	**Vertex-B**	**Maximum Available Bandwidth**	**Assigned Bandwidth**	**Remaining Bandwidth**
7	3	56000	36000	20000
7	8	56000	10000	46000
7	9	36000	4000	32000

Table 4.2: Logical Link Bandwidth Status Table

First two column of this table jointly identify the link (Edge) between two OpenFlow switches (Vertex). Third column of the table specifis maximum available bandwidth on this link. It will be the connection speed in case of wireless A, B, G or N.

Forth column, “Assigned Bandwidth” is updated every time a flow is added or deleted on this link. Combined load of this link is added in this column. Last column gives us the real status of the available bandwidth on specific link.

4.3 RSAC Flow Chart

Route Selection based on Application Characteristics (RSAC) flow is summarized in the flow chart given in figure 4.2.

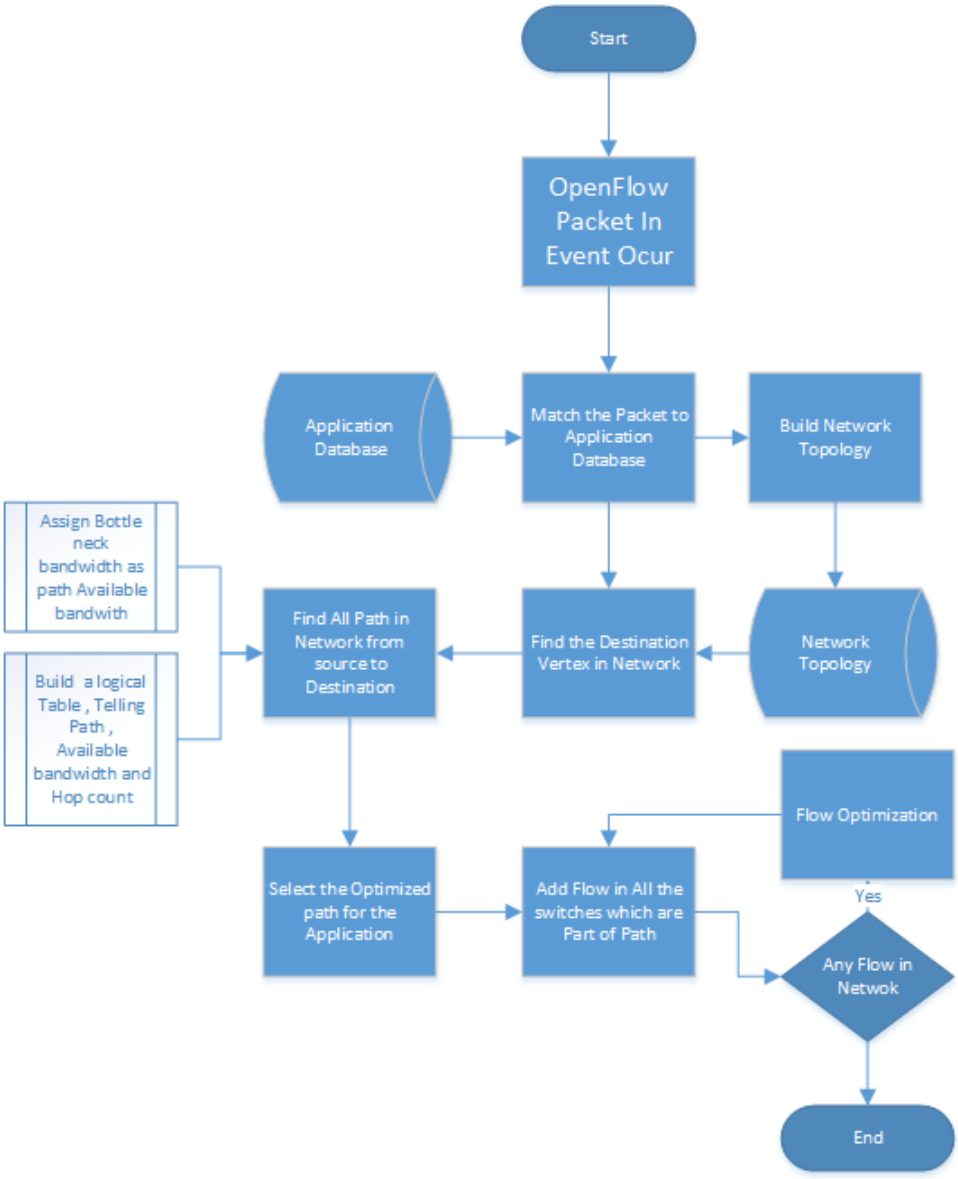

Figure 4.2: Solution Flow Chart

RSAC processing starts with the reception of any incoming packet at the

OpenFlow switch. This switch searches its flow tables for action associated with this flow. If finds any associated flow in the tables then corresponding actions are taken, which could be, generally, to forward packet on particular port. In case of non matching of flow, packet is forwarded to OpenFlow controller where it will be handled by our algorithm. RSAC matches packet to application database sample table given in subsection 4.2.1.

Network topology is updated with a function call ***"public Collection¡Vertex¿ getFullTopology()"***. This function calls sub-functions to get all switches and all links in the network. After building network topology, a destination vertex is identified. This destination vertex is OpenFlow's switch to whom the destination node is connected with.

From here onward our algorithm enters into a stage where it finds all possible paths from source vertex to destination vertex. This process involves two of our proposed algorithms; Capacity Calculation and Assignment Algorithm 4.1 and second Multi-path Calculation Algorithm 4.2. Detail of these algorithms are provided in the next sections. Once all of the paths are discovered, the optimal path for the particular application is selected to forward the flow. Path includes all vertex (OpenFlow switches) from source vertex to destination vertex. After identification of path, flow is added to all of the switches which fall within this path.

Beside this normal flow addition, we have proposed a Path Optimization algorithm which continuously monitor the network states and whenever the traffic on any link exceeds a specified limit, this algorithm optimizes the network and find new path in network. More detail on this algorithm is given in section Path Optimization, Route Discovery algorithm 4.3.

4.4 RSAC Algorithms

4.4.1 Capacity Calculation and Assignment Algorithm

Capacity calculation and assignment algorithm mentioned in Algorithm 4.1 is the core of our solution.

Algorithm 4.1: Capacity Calculation and Assignment

Input: Physical Network Topology
Output: Capacity Assignment on each link
1 **Capacity:** Calculate Capacity for each node to all its outgoing link Capacity(u,v)
2 **Calculation:** Maximum Capacity of each link is equal to its theoretical maximum speed of connectivity (e.g. 11Mbs, 36Mbs and 54Mbs)
3 Flow added on the link reduce its capacity with the amount of equal to requirement of capacity of application whose flow is being added.
4 For each link, remaining capacity as Total Capacity - Consumed Capacity
5 each Capacity remains valid for until flow addition or deletion
6 **Assumption:** Interference not considered

It takes the whole network topology as input and then searches load on each link and assigns the load on those links. Link bandwidth status table 4.2 is built as a result of this algorithm. Current load on link is saved in a circular buffer. This buffer saves the last 5 second data which include the data received and transmitted. This algorithm recursively calculates bandwidth of each out going link and stores as link weight. Later, in Multipath Calculation Algorithm, this weight will be used as a metric for path selection.

Code snippet of update link load function is given Appendix A.1.

4.4.2 Multipath Calculation Algorithm

Multipath Calculation Algorithm 4.2 build multipath from the network topology.

Algorithm 4.2: Multipath Calculation Algorithm

Input: Physical Network Topology, Source Node, Destination Node
Output: Lists of paths with maximum bandwidth available on each path (Path is equal to list of vertex(Nodes))

1 **MultiPath:** Calculate all paths from source and destination using DFS, Multipath (u,v)
2 **Steps:**
3 Put Source Node (Vertex) in a stack
4 Iterate on all elements (Vertex) in stack until it is empty
5 Pop the Element (Vertex) from stack
6 List All edges of selected Vertex, List_of_Edges
7 Find the List_of_Vertex on other end from List_of _Edges
8 Check the Visited status of all vertexes and Mark all un-visited Vertex as visited and push in stack.
9 Update Path List of Visited Vertex.
10 Set the Capacity of Path as = Minimum of capacity of all link in path
11 Reached Destination Vertex, Reverse the Path List to get all Path.
12 Finish the Iteration if no un-visited child in Stack

This algorithm can be used when delay is acceptable for an application. If the application is critical to delay then the Dijkstra shortest path algorithm is called to find the shortest path and flow is added on this path. This Algorithm was a tricky part in our solution. As multipath calculation is NP hard problem we have to have certain condition to converge to our solution.

We marked after visiting each vertex as 'visited'. This prevents us from indefinite loop in the network. In this algorithm was used depth first approach for finding all the paths. Stack data structure was used for storing the vertex and and edges of all vertices discovered. Code snippet is given in Appendix A.2.

4.4.3 Path Optimization, Route Discovery Algorithm

Path Optimization, Route Discovery algorithm 4.3 is developed specially to remove any congestion in the network.

Algorithm 4.3: Path Optimization, route Discovery

Input: Physical Network Topology, Flows in Networks
Output: Optimized Network
1 **Optimization:** Monitor the all the flows with their life time statistics and remaining link capacity the path of that flows, if the remaining capacity is less than the 80% of the theatrical max capacity Mark flow as optimized, remove the flow and find new path for that flow.
2 **Path Selection**
3 Select the best path from the Multipath available for a flow depending on the application requirement. Select the path with minimum hop count from the subset of multiple path which can fill application bandwidth requirement.

This algorithm runs continuously in the background checking network states and remaining capacity of each link. It compares the remaining capacity of link with its theoretical peak bandwidth. If it finds that remaining capacity is less than twenty percent of its peak capacity then this link need optimization. Code snippet of this algorithm is given in Appendix A.3.

Chapter 5

Results and Discussions

5.1 Network Topology Model

The proposed model in this thesis takes as input the entire topology of wireless part of the network with traffic load updated with the addition and deletion of flow. Main purpose is to design efficient routing algorithm that forwards packet on path which is most suitable to application flow. Model generates output with multiple paths and the one which is optimized is selected to forward the flow.

Some Mathematically terminologies used are:-

Nodes **N{u,v,w,x}**

OpenFlow Controler **C{c}**

OpenFlow Switches **S{q,r,s,t}**

Standard Switch S(Prime) **{q,r,st}** (All primes)

$\forall n \in N$ at least there are two wireless interfaces which are configured with two different IP subnets.

$\forall n \in N$ Nodes are connected to OpenFlow switches.

$\forall s \in S$, Switches are connected to OpenFlow Controller **'c'**.

$\exists Edge \in E$, {e}

$\exists Controller \in$C, {c}

5.1.1 Selection of Path

5.1.1.1 Network Topology Scenario 1

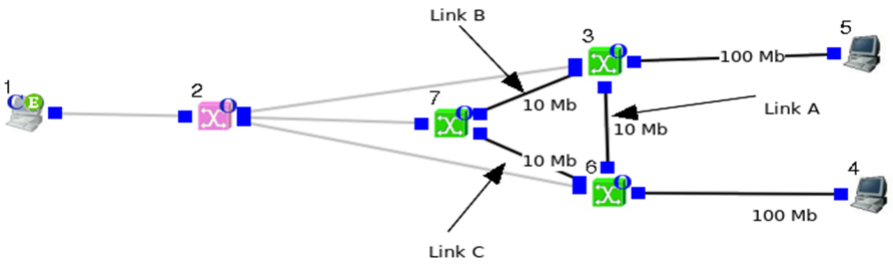

Figure 5.1: Network Topology Example

When any **'u'** belongs to 'N' send packet to 'v' via 's' belongs to 'S', 's' Start algorithm Capacity Calculation 4.1, on all 'e' which are incident on 's'. This algorithm runs on all 's' belong to 'S', result will be updated the capacity of each 'e'. 's' primes does not participate in this algorithms.

At the end of Algorithm 4.1, a 2nd Algorithm Multipath Calculation Algorithm 4.2 is started from 'S{q,r}', where 'q' is source switch, which has received the packet form 'n', and 'r' is the destination switch where the node 'v' is connected.

Multi-path from 'u' to 'v' are calculated and then it entered into 3rd Algorithm Path optimization Algorithm 4.3. This algorithm selects a path for 'u' to 'v' on the basis of minimum cost which is calculated as: "C = Min(Cost uv)". This function compares cost of each path from 'u' to 'v', and selects the path with the minimum cost path. The input to Min function is

subset of paths which satisfy the application requirements.

In the example Figure 5.1, 'Node 5' start two flows of 6Mb each. In traditional routing, both flows takes the link 'A' Using our algorithm (RSAC) first flow takes link 'A' 2nd Flow takes link 'B' and link 'C'.

Reason for this is, 2nd 6Mb flow is not adjustable on Link 'A', it require higher bandwidth than the remaining bandwidth on link 'A' where as link 'B' and link 'C' provide with required bandwith.

5.1.2 Network Topology Scenario 2

In this scenario, shown in Figure 5.2, we have used 4 OpenFlow switches, all connected with each other. They form a fully connected mesh network. Two nodes were connected each with different OpenFlow switches. In this scenario the central links are low bandwidth links while all the other link have higher bandwidth and all have same theoretical bandwidth limit. Central links are removes to test a variant of this scenario.

5.1.3 Network Topology Scenario 3

In this scenario, shown in figure 5.3, hosts are shown with set Host 4, 5,12,13,14, OpenFlow enabled switches are show as set OFSwitches 3, 6,7,8,9,10,11, standard ethernet switch is labeled as 2 and OpenFlow Controller is labled as 1. This scenario was build specially to test our optimization algorithm. In this scenario we over-burden certain link and optimization algorithm runs. The network congestion was reduced by our algorithm.

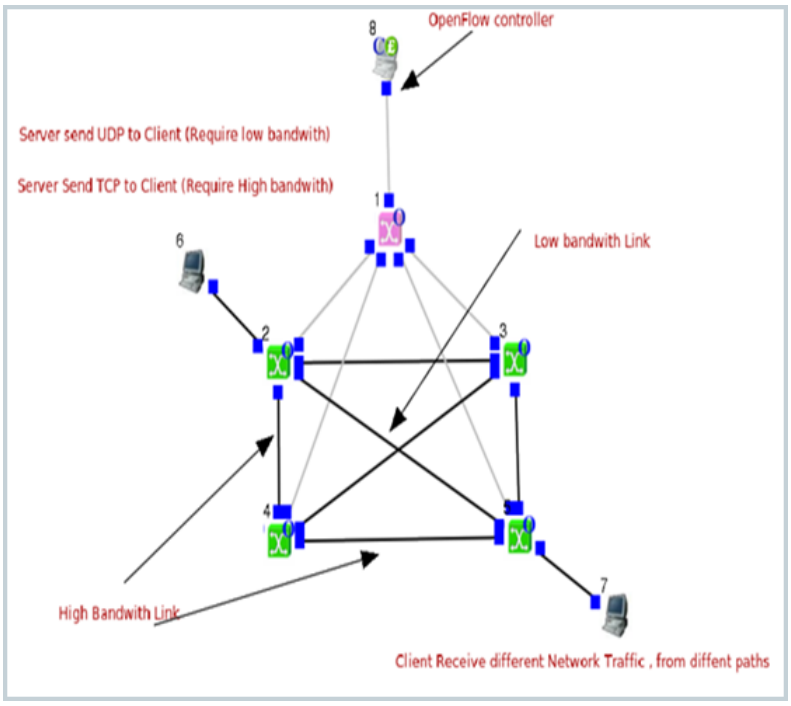

Figure 5.2: Example Scenario 1 Network Topology for Testing of algorithm

5.2 Smulation Test Bed Setup

The Table 5.1 summarizes all the tools and development environment we have used in our thesis. We have used two different operating system on virtual machines. We were bound to use Fedora 15 for EstiNet simulation tool. NUST SEECS purchased the EstiNet educational license for this research.

Our simulation setup were build on VM. Guest operating system was windows 8 with Oracle virtual box installed. One VM was build Ubuntu 12.04, this virtual machine was our development machine. Floodlight controller was also installed on this machine. Mininet was also configured on this virtual machine for quick tests. It was found Mininet is very easy to operate if you do not have any customization in network. OpenVswitch was also configured on this virtual machine for testing of solution concept on Mininet plus Open-

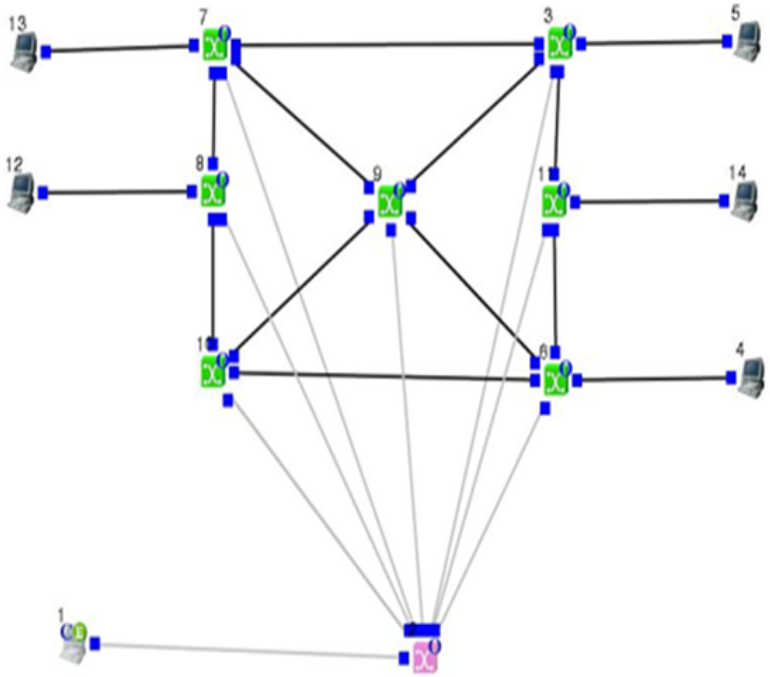

Figure 5.3: Example Scenario 2 Network Topology for Testing of algorithm

Sr No.	Purpose to Use	Tool/Method
1	Operating System	Linux Ubuntu 12.04 LTS and Fedora 15
2	OpernFlow Controller	Floodlight
3	Development Language	Java
4	Development Environment	eclipse
5	Simulation Software	Estinet
6	OpenFlow Version	OpenFlow 1.0
7	Simulation Time	90 Seconds and 320 seconds
8	Simulation Scenario	Three
9	Simulation Run	More than 100 run
10	Machine Used in simulation	Two, One for Estinet, Other for OF controller
11	Simulation or Emulation	Simulation and Emulation both

Table 5.1: Simulation Tool and setup

Vswitch. Our second virtual machine was Fedora 15. It was required due to Estinet simulator software. It was hard part to configure machie for Estinet software. There were some customization options each time we have to set

before running a simulation. These setting were customization of machine routing table, IP address access. We built a batch file for setting up options each time we boot Estinet simulator software. The shapshot of batch file is given below:

```
echo 1 >/proc/sys/net/ipv4/ip_forward
echo 1 >/sys/net/ipv4/conf/p7p1/proxy_arp
echo 0 >/proc/sys/net/ipv4/conf/p7p1/rp_filter
route del -net 192.168.1.0/24 dev p7p1
route add 192.168.1.1 dev p7p1
route add 192.168.1.3 dev lo
iptables –F
service iptable stop
```

This batch file helped us to forward Estinet internal network traffic to external OpenFlow controller. External controller integration with simulated switches made our test bed a combination of simulation and emulation. Due to this kind of setup we were very close to real network environment.

5.3 Simulation Results

We have shown the performance gain for elephant flows of our proposed RSAC algorithm via simulation. Different test beds were used for our simulation. For simulation of our solution, EstiNet OpenFlow Network Simulator and Emulator is used. For the purpose of SDN simulation and emulation, EstiNet is one of best option available these days. It supports both OpenFlow version 1 and 1.3. EstiNet has three main component.

A) Estinet dispatcher B) Estinet Coordinator C) Estinet GUI.

For designing our solution, we have used Estinet GUI in D (Design Mode). We designed our simulation to near realities. A real open source network controller was used on different machine. EstiNet simulator was deployed on one virtual machine and Floodlight OpenFlow controller was installed on other machine.

A module was build from our source code which was added in Floodlight controller to act on the flow request it receives. A summary of our simulation setup and tools used in setup are given in Table 5.1.

5.3.1 Simulation Scenarios

Three simulation scenario were build for test our proposed algorithm.

1. Network Topology Scenario 1 5.1.1.1
2. Network Topology Scenario 2 5.1.2
3. Network Topology Scenario 3 5.1.3

On these simulation scenario different runs of simulation were done to obtain the results.

5.3.2 Throughput

In the throughput graph, shown in Figure 5.4, we have shown that with the addition of flows, network throughput increases but after a certain point network throughput becomes constant. With the use of RSAC, this constant level comes later as compared to non-RSAC algorithms. We achieved better throughput using RSAC.

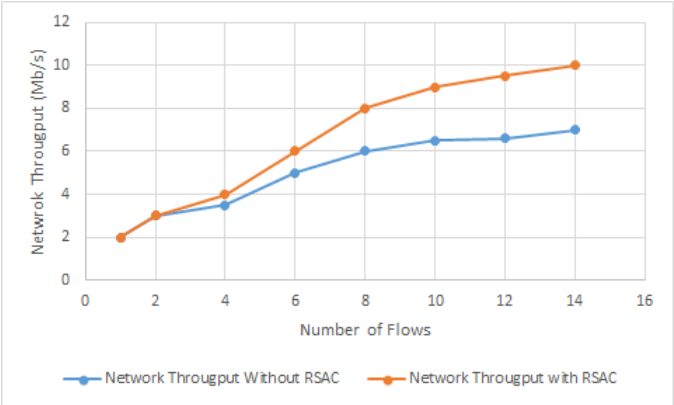

Figure 5.4: Throughput with and without RSAC

5.3.3 Delay Comparison

Delay occurs due to addition and deletion of flows and it also occurs when network statistics change or network gathers updates. Delay comparison is shown in Figure 5.5.

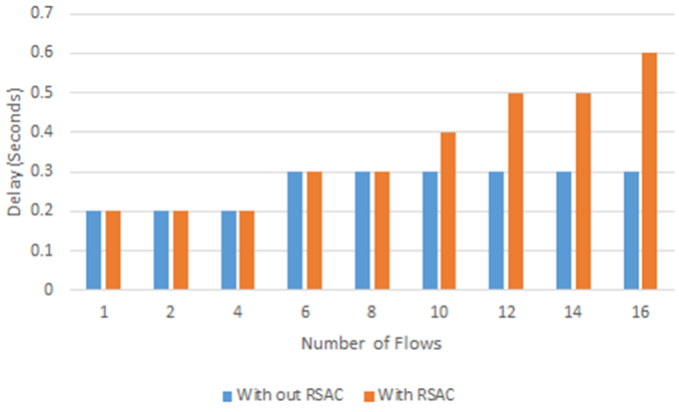

Figure 5.5: Delay Comparison with and without RSAC

5.3.4 Hop Count

In the Flow hop count comparison graph, Figure 5.6, we have shown that with the increase in flows number of hops also increases. This increase is

because of bandwidth requirement of application.

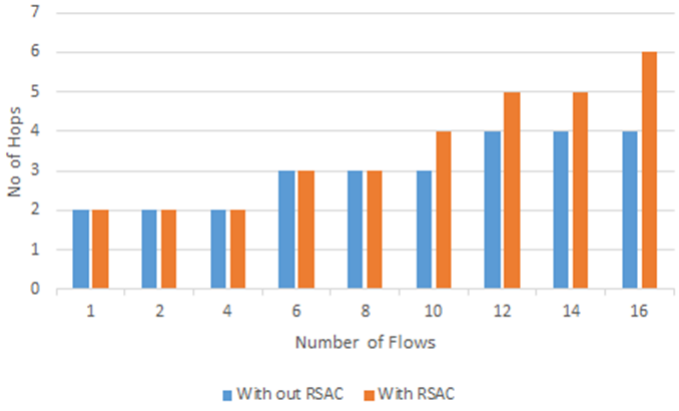

Figure 5.6: Hop Count Comparison with and without RSAC

5.3.5 Optimal Path Selection Time

In optimal path selection time graph, Figure 5.7, we have shown that with the increase of OpenFlow switches, conversion time of path selection is also increased. This increase is nonlinear. With the increase in OpenFlow switches, time taken to select a path increased more than twice.

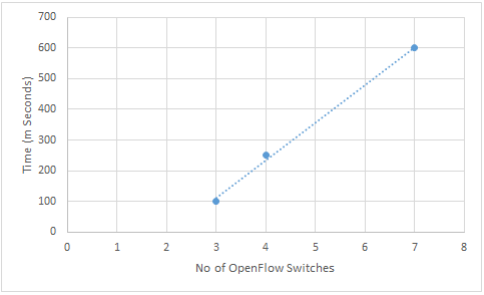

Figure 5.7: Optimal Path Selection Time

5.4 Discussions

Route Selection based on Application Characteristics (RSAC) aims at finding the use of software defined networking in general and OpenFlow in particular in multi-hop wireless network. Some very basic question were raised in our problem definition which we tried to answer in this thesis.

Basic computer science technique are used in this research project. A classical model of software development was followed. Problem has been analyzed, solution has been designed and then implemented. Later on tested the solution using simulation and emulation. Different state of art solutions are discussed, that use OpenFlow for wireless network. In literature, people are trying to use OpenFlow in wireless network. Open networking foundation issued a white paper which highlights the use of OpenFlow in wireless networks. It is found in literature that traditional routing techniques being used are application agnostics. Underling routing algorithms do not consider the application characteristics while taking routing decisions.

In this thesis, It is stated that OpenFlow can be used in wireless network with the use of multiple channel or within one channel using different classes of network IP to communicate with the OpenFlow controller. Due to this design, participating nodes which do not forward network traffic of other nodes do not need to change their network stack. Nodes which forward traffic of others called intermediate nodes need to support OpenFlow standard either in software defined switches or via change in their network stack which would make them OpenFlow enabled.

In RSAC, application awareness is built-in the centralized network. This central controller is responsible to get states of network and store this information for upcoming flow requests. The process which involves network state gathering is started from OpenFlow controller.

In implementation chapter, various algorithms are explained that are designed and developed. For the proper functioning of these algorithm central controller needs to gather certain information from the participating nodes which support OpenFlow protocol and form other OpenFlow switches which are playing intermediate role in flow forwarding. These informations includes, list of connected hosts, list of ports, send and receive bytes of the ports and the connection edges between hosts. This thesis lacks quantifying the overhead generated by these messages. During the simulation phase no significance of these overhead were noted and hence no analysis were done on this. In a similar research using OpenFlow for wireless mesh network in [2] OpenFlow traffic overhead has been shown liner increase with the increase of rule installation rat.

Problem statement of this research was aimed to find the advantage of OpenFlow in application awareness. With the throughput graph in figure 5.4 we can safely draw the conclusion that with the increase in number of flows in network with application aware algorithm running at controller end, we can get better network throughput.

The result without RSAC and with RSAC are shown in graphs. In comparison, throughput is achieved on the cost of some delay added in the network. This delay is introduced in network due to OpenFlow controller path selection, and rerouting of already routed path. Quantifying the delay is done in figure 5.5. Delay for the first eight flows was equal, but latter it was introduced when number of flows on certain link began to chock its throughput. An Optimization algorithm was run to over-come the over-burdened link, which cause this delay.

Optimization algorithm added to minimize the network saturation on a single point. This algorithm meets the requirements but it also yields a delay

in the network. It can be stated that proposed algorithm maximizes network throughput for specific applications. Examples include but not limited to HD multimedia, real-time capturing and transmission. It also minimizes end-to-end delay for specific applications. For example for machine control instructions, control messages for applications.

Network scenario no 3 was built to test our optimization algorithm. One of advantages we are taking from this algorithm is, it prevent starvation. Participating nodes in network can go in starvation if they are not able to send or received their required data. In our test environment it would only happens if a node has only one path to send its traffic and that path is being used by other nodes. Our optimization algorithm runs in the background. It continuously checks if any link is being used more than 80 percent of its capacity. If any such link is found then optimization algorithm finds alternate routes for such flows whose runtime less than their average runtime.

In our testing and simulation we have used three network scenarios and added different counts of network flows in each scenario. Results prepared in this thesis are based on 4 flows added in 1st scenario, 10 flows in 2nd network scenario, and 16 flows in 3rd network scenario. With these information we presented a graph in figure 5.6 of number of hop for all these flows.

One interesting and optimists result was presented in figure 5.7. As number of OpenFlow switches increase the time taken to search desired path increases but this relationship is nonlinear.

Chapter 6

Conclusion & Future Work

6.1 Conclusion

Software Defined Networks have opened an opportunity window for the network professionals. Using SDN, application aware routing can easily be integrated in network infrastructure. Route selection in multi-hop wireless networks required an intelligent algorithm through which nodes can select an appropriate route to send their data to the destination. An algorithm, "Route Selection based on Application Characteristics (RSAC)", is proposed in a multi-hop wireless network. At first step, capacity at each edge (Link) is identified in whole network. After that, all the available paths to the destination are searched. Based on path capacity and hop count, cost is calculated and a path is selected to transmit data over. This research has shown improvement in throughput at the cost of a small delay. One can use RSAC in QoS based networks.

6.2 Future Work

In the extension of this work, one can model entire algorithm mathematically with objective function to minimize cost. Throughput analysis can to also be done. We used reactive approach using OpenFlow which has its own inherent disadvantages, especially in mobile networks. This approach can be altered in such a way that it becomes hybrid in nature and some of inherent disadvantages can be minimized. Our approach can be improved to be used for future networks, even beyond path selection.

Bibliography

[1] *OpenFlow for Campuses A Tutorial at GEC10*, http://www.openflow.org/downloads/OpenFlowTutorial_GEC10.ppt.

[2] P. Dely, A. Kassler, and N. Bayer, "Openflow for wireless mesh networks," in *Computer Communications and Networks (ICCCN), 2011 Proceedings of 20th International Conference on.* IEEE, 2011, pp. 1–6.

[3] N. S. Nandiraju, D. S. Nandiraju, and D. P. Agrawal, "Multipath routing in wireless mesh networks," in *Mobile adhoc and sensor systems (MASS), 2006 IEEE international conference on.* IEEE, 2006, pp. 741–746.

[4] M. Mosko and J. Garcia-Luna-Aceves, "Loop-free routing using a dense label set in wireless networks," in *Distributed Computing Systems, 2004. Proceedings. 24th International Conference on.* IEEE, 2004, pp. 380–389.

[5] S.-J. Lee and M. Gerla, "Split multipath routing with maximally disjoint paths in ad hoc networks," in *Communications, 2001. ICC 2001. IEEE International Conference on*, vol. 10. IEEE, 2001, pp. 3201–3205.

[6] S. J. Vaughan-Nichols, "Openflow: The next generation of the network?" *Computer*, vol. 44, no. 8, pp. 13–15, 2011.

[7] G. Goth, "Software-defined networking could shake up more than packets," *Internet Computing, IEEE*, vol. 15, no. 4, pp. 6–9, 2011.

[8] B. N. Astuto, M. Mendonça, X. N. Nguyen, K. Obraczka, and T. Turletti, "A Survey of Software-Defined Networking: Past, Present, and Future of Programmable Networks," Oct. 2013, in Submission In Submission. [Online]. Available: http://hal.inria.fr/hal-00825087

[9] *FloodLight OpenFlow Controller*, http://www.projectfloodlight.org/floodlight/.

[10] W.-H. Liao, Y.-C. Tseng, and K.-P. Shih, "A tdma-based bandwidth reservation protocol for qos routing in a wireless mobile ad hoc network," in *Communications, 2002. ICC 2002. IEEE International Conference on*, vol. 5. IEEE, 2002, pp. 3186–3190.

[11] L. Georgiadis, P. Jacquet, B. Mans *et al.*, "Bandwidth reservation in multihop wireless networks: Complexity and mechanisms," 2003.

[12] H. Zhu and I. Chlamtac, "Admission control and bandwidth reservation in multi-hop ad hoc networks," *Computer Networks*, vol. 50, no. 11, pp. 1653–1674, 2006.

[13] L. Hanzo and R. Tafazolli, "A survey of qos routing solutions for mobile ad hoc networks," *IEEE Communications Surveys & Tutorials*, vol. 9, no. 2 2nd, pp. 50–70, 2007.

[14] D. S. De Couto, D. Aguayo, J. Bicket, and R. Morris, "A high-throughput path metric for multi-hop wireless routing," *Wireless Networks*, vol. 11, no. 4, pp. 419–434, 2005.

[15] J. Chung, G. Gonzalez, I. Armuelles, T. Robles, R. Alcarria, and A. Morales, "Experiences and challenges in deploying openflow over real wireless mesh networks," *Latin America Transactions, IEEE (Revista IEEE America Latina)*, vol. 11, no. 3, pp. 955–961, 2013.

[16] S. Ould Cheikh and A. Gueroui, "Multi-hop bandwidth reservation in wmn-based ieee 802.11 s (mbrwmn)," in *Communications and Information Technology (ICCIT), 2012 International Conference on.* IEEE, 2012, pp. 211–215.

[17] P. Zhao, X. Yang, J. Wang, B. Liu, and J. Wang, "Admission control on multipath routing in 802.11-based wireless mesh networks," *Ad Hoc Networks*, vol. 11, no. 8, pp. 2235–2251, 2013.

[18] L. Khoukhi, H. Badis, L. Merghem-Boulahia, and M. Esseghir, "Admission control in wireless ad hoc networks: a survey," *EURASIP Journal on Wireless Communications and Networking*, vol. 2013, no. 1, pp. 1–13, 2013.

[19] A. Detti, C. Pisa, S. Salsano, and N. Blefari-Melazzi, "Wireless mesh software defined networks (wmsdn)," in *Wireless and Mobile Computing, Networking and Communications (WiMob), 2013 IEEE 9th International Conference on.* IEEE, 2013, pp. 89–95.

[20] K. Venkatraman and G. Ilakkia, "Wireless network managed through sdn," 2013.

[21] M. A. Kinsy, M. H. Cho, K. S. Shim, M. Lis, G. E. Suh, and S. Devadas, "Optimal and heuristic application-aware oblivious routing," *Computers, IEEE Transactions on*, vol. 62, no. 1, pp. 59–73, 2013.

[22] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "Openflow: enabling innovation in campus networks," *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 2, pp. 69–74, 2008.

[23] S. Das, Y. Yiakoumis, G. Parulkar, N. McKeown, P. Singh, D. Getachew, and P. D. Desai, "Application-aware aggregation and traffic engineering in a converged packet-circuit network," in *National Fiber Optic Engineers Conference*. Optical Society of America, 2011, p. NThD3.

[24] A. Petcher, "Qos in wireless data networks," *Washington University in St. Louis. Department of Computer Science & Engineering. Esitelmä (27.2. 2006 & 1.3. 2006) kurssilla CSE574S: Advanced Topics in Networking: Wireless and Mobile Networking (Spring 2006)*, 2006.

[25] M. Jarschel, F. Wamser, T. Hohn, T. Zinner, and P. Tran-Gia, "Sdn-based application-aware networking on the example of youtube video streaming," in *Software Defined Networks (EWSDN), 2013 Second European Workshop on*. IEEE, 2013, pp. 87–92.

[26] S. Zander, T. Nguyen, and G. Armitage, "Automated traffic classification and application identification using machine learning," in *Local Computer Networks, 2005. 30th Anniversary. The IEEE Conference on*. IEEE, 2005, pp. 250–257.

Appendix A

RSAC Code

A.1 Capacity Calculation and Assignment Algorithm

```
private void updateLinkLoads(){
        for(Vertex myVertex: fullTopo){
                Collection<Edge> allEdges =
                 myVertex.getAdjacencies();
                for(Edge myEdge:allEdges){
                long availableBandwith=(states.getCurrentFeatures
                (myVertex.getSwitch().getStringId(),
                Short.toString(myEdge.getSrcSwitchPort())) -
                states.getLoadAverage(myVertex.getSwitch().getStringId(),
                Short.toString(myEdge.getSrcSwitchPort()))/1000);
                debugFileWrite(myVertex.getSwitch().getStringId() +" "+
                myEdge.getSrcSwitchPort() +" Bandwith "
                + availableBandwith);
                if(availableBandwith<1)
                {
                availableBandwith=1;
```

```
                }
                        myEdge.setWeight(availableBandwith);
                }
        }
}
```

A.2 Multipath Calculation Algorithm

```
source.maxThrougput = 100000;
//source.maxThrougput = 0;
Stack<Vertex> vertexStack = new Stack<Vertex>();
vertexStack.add(source);

while (!vertexStack.isEmpty()) {
Vertex u = vertexStack.pop();

// Visit each edge exiting u
for (Edge e : u.getAdjacencies())
{
Vertex v = e.getDstVertex();

//check v if already in path
boolean isVertexInpath=false;
if(v.pathList!=null)
for(path inPath: v.pathList){
if (inPath.hasVertex(u))
isVertexInpath=true;
}
if(isVertexInpath)
continue;

double weight = e.getWeight();
```

```
long distanceThroughU = (long) java.lang.Math.min(u.maxThrougput, weight);
//long distanceThroughU = (long) (weight+u.maxThrougput);
if (distanceThroughU >= v.maxThrougput) {
vertexStack.remove(v);

v.maxThrougput = distanceThroughU ;
path p =u.getBestPath(v);
path newPath = new path();
if(p!=null)
for(Vertex v1:p.getPath()){
newPath.addInPath(v1);
}
newPath.addInPath(u);
newPath.setWeight(distanceThroughU);
//System.out.println("path"+ newPath.getPath() +
"␣Weight␣"+ newPath.getWeight() +
"␣in␣Vertex" + v.getSwitch().getStringId());
v.pathList.add(newPath);

vertexStack.add(v);
}
//v.previous = u;

}
}
}
```

A.3 Path Optimization, Route Discovery Algorithm

```
if((availableBandwith/(Long.valueOf(p.getCurrentFeatures()))*100) <20)
{
```

```
//Here is link saturated we need to move flow from this link to other link.
for(Flow f: sw4FLP.getFlows()){
if(Long.valueOf(f.getDurationSeconds())>10){
debugFileWrite("Deleting flow");
installer.delFlowAtTransPort(netTopo,
(Short.valueOf(f.getMatch().getDataLayerDestination())));
}
}
}
```