Design and Analysis of an Intelligent Wireless Ad Hoc Routing Protocol and Controller for UAV Networks

By

Abhinandan Ramaprasath

A thesis submitted to the Faculty of Graduate and Postdoctoral Affairs in partial fulfillment of the requirements for the degree of

Master of Applied Science

In

Electrical and Computer Engineering

Carleton University
Ottawa, Ontario

Abstract

This thesis proposes a UAV to UAV communication approach that is based on Software Defined Networking (SDN). The proposed approach uses a controller as a central source of information to assign routes that maximize throughput, distribute traffic evenly, reduce network delay and utilize all network elements. An SDN based WiFi approach is used in order to provide seamless WiFi connectivity to users as they move around the network. Simulation results show that the proposed approach can improve throughput by over 300%. Simulation results also show a reduction in network delay for delay sensitive packets to nearly 25% and increase in packet delivery ratio (PDR) by 26 times for packets with higher priority. Simulation results were compared to two common ad hoc routing protocols AODV and OLSR.

Acknowledgements

I would like to express my deepest gratitude to Dr. Chung-Horng Lung and Dr. Marc St-Hilaire for their guidance and support for the thesis. I would also like to thank the department of Systems and Computer and Engineering, Carleton University, NSERC and EION Inc. for supporting the research.

I would like to thank my parents and my family for supporting me throughout my education. Without their help, I could not have accomplished my work.

List of Abbreviations

ABR	Associativity-based routing
AODV	Ad hoc On-Demand Distance Vector Routing
AP	Access Point
BPSK	Binary Phase Shift Keying
CAPWAP	Control and Provisioning of Wireless Access Points
CBRP	Cluster Based Routing Protocol
CGSR	Clustered Gateway Switch Routing Algorithm
CPU	Central Processing Unit
CSMA	Carrier Sense Multiple Access
CTS	Clear to Send
DSDV	Destination-Sequenced Distance Vector routing
FCS	Frame Check Sequence
GPS	Global Positioning System
IEEE	The Institute of Electrical and Electronics Engineers
IP	Internet Protocol
IR	InfraRed
LTE	Long-Term Evolution
MAC	Medium Access Control
NFV	Network Function Virtualization
OLSR	Optimized Link State Routing
QAM	Quadrature Amplitude Modulation
QoS	Quality of Service

QPSK	Quadrature Phase Shift Keying
RA	Receiver Address
RREQ	Route REQuest
RTS	Request to Send
Rx	Receive
SDN	Software Defined Network
SSID	Service Set IDentifier
SSR	Scalable Source Routing
TA	Transmitter Address
TCP	Transmission Control Protocol
TORA	Temporally Ordered Routing Algorithm
TX	Transmit
UAV	Unmanned Aerial Vehicle
UAANET	Unmanned Aerial Ad hoc Networks
VM	Virtual Machine
WLAN	Wireless Local Area Network
WRP	Wireless Routing Protocol

List of Figures

List of Tables

Table of Contents

Chapter 1

Introduction

This chapter introduces the concept of creating ad hoc networks with Unmanned Aerial Vehicles (UAVs), also referred to as Unmanned Aerial Ad hoc Networks (UAANETs). Then, the motivation and the problem statement are described followed by the main contributions made by the thesis. Finally, a general overview of the thesis is outlined.

1.1 Motivation and problem statement

Wired, wireless and satellite based communication technologies have seen immense advancements recently and are being employed for various scenarios. Wired networks require an extensive infrastructural setup whereas in wireless or satellite communication networks, a minimal infrastructural setup is sufficient. Once these networks are set up, they are generally very hard to change and in some cases, reconfiguration of the entire network may be required. There are situations where wired, wireless and satellite technologies may not be readily usable. For example, in a scenario such as a forest fire or disaster recovery, it is not possible to do ground level communications as remote areas do not have infrastructure on the ground for communications.

In situations like the ones mentioned above, technologies using UAVs can be useful. A UAV can be easily deployed from a fire truck, police vehicle or disaster recovery agency. UAVs are becoming more powerful, cheaper and faster. It is important to note that in a large territory, e.g., a 1000 square kilometers area, it is impossible for the data from one

UAV to reach the destination directly. Hence, the data must be relayed by other UAVs in order to reach the destination. Therefore, a network of UAVs is necessary to monitor a disaster recovery scenario. A network of UAVs can span areas of many square kilometers and are resilient to changes at the ground level. Since a UAV can be moved around the area on demand, a network using UAVs is flexible and scalable. Additional UAVs can also be deployed on demand to expand the area of connectivity or replace dying UAVs. The density of UAVs can be increased in areas where there is a higher demand for network resources. Moreover, this network must be able to provide Internet or network connectivity through VPN to users and be able to support transmitting priority packets that need to reach the destination before other packets. For example, in case of a disaster management scenario, data transmitted from a fire truck would be classified as higher priority and must reach the destination quickly. Thus, a network of UAVs is more effective than traditional communication technologies for real-time and non-real-time packet transfers where time and flexibility of network deployment is important.

For the purposes of this research, we define a UAV as a quad copter or a similar lightweight device carrying two sets of wireless antennas for communications. These UAVs can fly typically 100-500 meters off the ground. UAVs are battery powered and can stay above the ground as long as the battery/fuel can last. For example, Jump 20, a UAV made by the company Arcturus UAV can last from 9 – 16 hours depending on the payload [Jump20]. UAVs circulate within an area to provide wireless connectivity to users in that area. In order to utilize network resources and to maximize throughput, we need a central repository of all the information or a “controller”. A controller monitors

the network providing data to network administrators and assigning routes within the UAV network for data and control packet transmissions. The controller is the main entity in Software Defined Networking (SDN) and therefore, applying SDN concepts can provide advantages in the target domain. Figure 1.1 shows a basic architecture of the proposed solution. Smartphones, laptops, tablets, or fire trucks connect to the UAV. This network of UAV is monitored by the controller and provides internet access to the users.

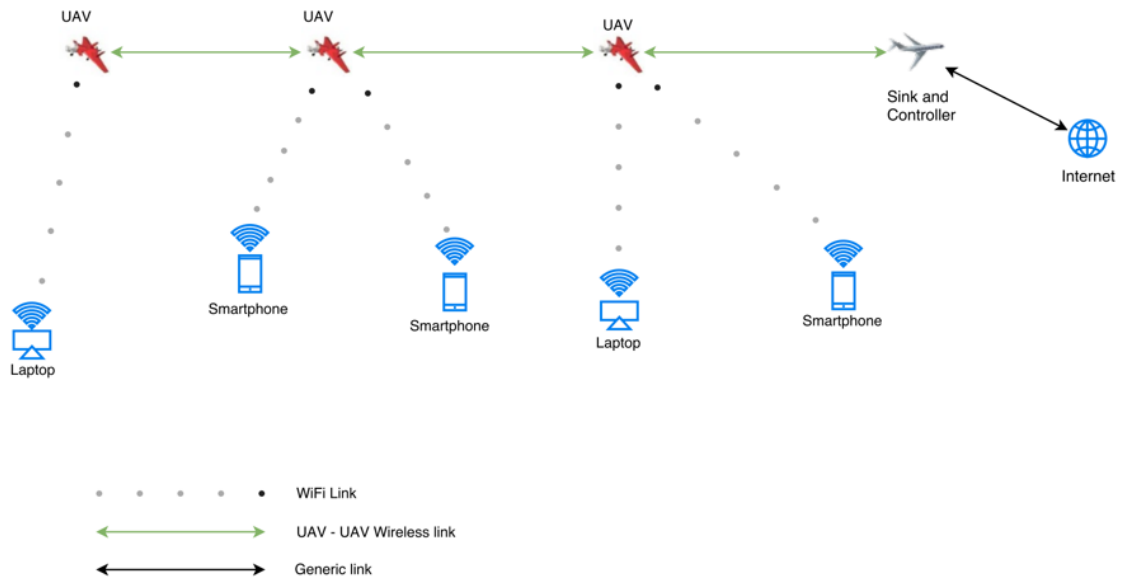

Figure 1.1: UAV-UAV and UAV-User communications

The problem further compounds when the UAVs are trying to provide Internet or data transfer access for the fire trucks or users for reasons like surveillance, reporting, GPS based maps etc. Traditionally, UAVs would carry an 802.11b/g/n/ac module on board to create an infrastructure based WiFi network that the fire truck would then connect to. As shown in Figure 1.1, UAVs are forming the backbone infrastructure to allow users to

have Internet connectivity. However, traditional WiFi networks do not allow the user to change Access Points (APs) readily, which means the user is connected to the same AP until connectivity is lost. If the fire truck is connected to an AP, it will continue to be connected to that particular AP until it has completely lost connection, even though another AP has a better signal strength. Once the connection is lost, the fire truck has to reconnect to the network again, which leads to a temporary loss in connection. It would be far more beneficial for the fire truck to be able to maintain the optimal connectivity with APs, since the fire truck needs to communicate critical information about the environment. Ad Hoc UAV-UAV networks solve this issue by making the connectivity flexible and agnostic.

As can be seen from the problem description above, providing Internet connectivity through a UAV based ad hoc networks creates several problems. On the UAV-UAV communication side, the main issues are:

- Finding optimal routes to reach the destination
- Use multiple disjoint routes simultaneously in order to improve throughput
- Prioritize packets and send them through the network on the shortest path
- Integrate support for priority packets in lower network layers

On the UAV-User communication side, the problems are:

- Roaming between APs without losing connectivity
- Load balancing traffic as evenly as possible across the network

Based on the problems mentioned above, the next section will state our thesis research objectives.

1.2 Research objectives

The main objective of this thesis is to create a new SDN based UAV-UAV communications approach to provide users with Internet connectivity. To tackle this complex problem, we break it down into two smaller objectives that are inter-related:

- Create a new approach for UAV to UAV communications. The proposed approach should address all the important features that are encountered in disaster recovery situations such as redundancy, packet prioritization and optimal throughput.
- Improve the UAV to user connectivity by providing mechanisms to allow users to roam across access points without losing connectivity. Other features such as load balancing will also enhance the quality of service for the end users.

1.3 Contributions

Based on the objectives listed in the previous section, the main contributions of this thesis include:

1. A novel SDN based UAV-UAV communications approach that has a support for four levels of priority packets. The proposed approach uses a controller in order to determine the best route to achieve higher throughput, lower network delay and uniform distribution of traffic.
2. A new medium access control protocol (MAC) to support priority packets at the lower layers.

3. Modifications to OpenSDWN [Schulz15a] [Schulz15b] that enable users to roam seamlessly within the WiFi network created by the UAVs. These modifications also explain how users are handed over in case of overcrowding or UAV failure.
4. Performance analysis to validate the approach proposed. The simulation results demonstrate that the proposed approach can improve throughput, reduce network delay, increase Packet Delivery Ratio (PDR) for packets with higher priority and switch to an alternate route immediately in case of failure.

1.4 Thesis outline

The rest of the thesis is structured into four chapters. Chapter 2 outlines the related works and different approaches taken in the past to solve the problem. Chapter 3 explains the proposed approach for UAV-UAV and UAV-User communications. Chapter 4 demonstrates simulation and analyzes the results. Chapter 5 presents the conclusions and future work.

Chapter 2

Background and Related Work

This chapter explores papers related to traditional WiFi roaming, SDN based WiFi roaming and various routing protocols that have been used in wired networks, infrastructure based wireless networks and wireless ad hoc networks. This chapter also describes the limitations of those approaches for that problem.

2.1 WiFi Roaming

Communication via WiFi networks has dominated the market in recent years. Explosion of WiFi capable devices has led to the deployment on WiFi networks in every place imaginable, even in public places in the city of Ottawa [FreeWiFi16]. Users are free to roam within the WiFi network and enjoy connectivity wirelessly. WiFi however has significant drawbacks. The lack of built in roaming capabilities within a large network has led to many third party solutions implemented on the client and server side. WiFi roaming is when a user is moving out of range from a certain AP and must re-associate with another AP in order to retain connectivity. In order for this handover to be seamless the association with the new access point must take place before the disconnection from the first AP. Creating seamless WiFi have not been a major concern in the far past but is growing to be one of the biggest problems in WiFi today.

In our problem, UAVs are going to be providing Internet access over a city or for a disaster area, there is a need for the UAVs to be able to provide access to full-speed WiFi. Moreover, it is required that clients that are connected to a UAV are optimized for

performance and bandwidth, while taking into account the network characteristics to maintain low latency and conserve battery power of UAV. In most traditional routing protocols (explained in detail in Section 2.2) most traffic is routed through the same UAV causing the UAV to drain power while some UAVs are underutilized. Since UAVs battery power are crucial, it is important to conserve battery power by distributing traffic evenly throughout the network. This would also result in an overall increase in network efficiency. Wireless communications using the radio is one of the most power hungry operations. Therefore, handing over users to neighboring UAVs to conserve battery is also of utmost importance or even critical for some scenarios.

We present some routing problem solutions explored in the past and include papers that present interesting ideas on how to scale the networks further as the UAV density increases. Furthermore, this section also investigates different handover mechanisms proposed in the past including handover within heterogeneous networks.

The authors in [Monin14] show the benefits of having Wireless Local Area Networks or WLANs on top of the SDN/OpenFlow infrastructure. A SDN controller can manage the APs and in turn the way they behave. By switching the routes and the traffic flow pattern beforehand, the authors demonstrate that a SDN-based WLAN can reduce the switching time from 2.934 seconds to 0.85 seconds. However, the problem with this approach is that the switch from one AP to the next is made by the client. This means that the network has no control over which user device is connected to which AP. We need to be

able to load balance our network at any point to make sure that the users are distributed evenly and the traffic among the UAVs is evenly distributed.

Moreover, this does not solve the roaming problem either. As the paper describes: “CAPWAP controller tells the SDN/OpenFlow controller about the client migration and necessity to push new flows to the switches” [Monin14], this means that for the client migration to take place, the client has to disconnect with the old AP and reconnect with the new one. This requires a 4-way handshake which means that for the 0.85 seconds needed for the handoff procedure, the client is disconnected from the network.

The approach proposed in [Schulz14] explains how nearsighted controllers can be used to create a scalable architecture using WiFi SDN networks. A similar architecture could be useful for the algorithm proposed in the thesis as the UAV network scales. A nearsighted controller only controls its immediate environment and does not take decisions based on the global network. As the network scales, taking decisions based on all the UAVs may be time consuming due to the fact that UAVs would need to periodically update the controller about their current state. However, hierarchical structure of controllers several nearsighted controllers can be polled by a global controller to obtain information from these nearsighted controllers. The proposed approach [Schulz14], however, does not solve any of the problems mentioned in Section 1.1. It does not solve the problem of seamless roaming of users within a network or provide any energy management techniques.

Connectivity management for eneRgy Optimized Wireless Dense networks (CROWD) [Ahmed13] solves the problem of optimizing connectivity in a dense, heterogeneous network popularly known as “densenet”. The paper uses SDN to solve this problem in a way similar to [Schulz14]. The paper introduces a hierarchy of controllers, each controlling one aspect of the network they are associated to. When a user device switches between networks, the controllers associated with the networks make sure that the user device is associated with the destination network before the handover takes place. Then transfer of the user device over to the new network starts, which provides seamless mobility across multiple networks.

Although, the approach in [Ahmed13] does not solve our problem, it provides a seamless experience for the user. The proposed approach enables deployment of UAV based WiFi networks in places where there is no existing WiFi infrastructure. It also enables the transfer the user device over to a different WiFi network when connectivity is available to reduce the load on the UAV network.

Apple recently introduced an application based on a similar technology that would seamlessly transfer phone calls from WiFi to LTE networks [WiFiCalling16], making a call possible as long as some connectivity is available. However, the approach proposed by Apple does not solve the WiFi handover issue.

An earlier work to such an idea was the patent published in 1999 by Nortel Networks entitled “Method and system for seamless roaming between wireless communication

networks with a mobile terminal" [Jawanda01]. The patent presents architecture to support seamless roaming between networks using a SDN-like approach between different networks well before the inception of OpenFlow. This patent fails to explain how seamless mobility can be achieved within a WiFi network. Nevertheless, the approach gives the basis on how mobility can occur between heterogeneous networks.

Moving closer to our goal is a technology developed by Intel in the early 2000s [Intel15a]. Intel's client side roaming made roaming possible by switching over the network on the client side as long as the Service Set IDentifier (SSID) remains the same and used the same authentication methods and keys. When a client's adapter detects that a network is going out of range, it immediately listens to beacons from APs with the same SSID. When a client side detects that the new AP is favorable, it initiates authentication procedures with this new AP to save the time required to authenticate and deauthenticate. The decision is based on a parameter called WiFi roaming aggressiveness [Intel15b].

The approach based on the client side roaming enables seamless roaming within a wireless network if the network is configured properly. Some AP may have a subnet of its own, which means that TCP connections maintained by the client are no longer valid because there is a change in IP and there is no way to notify the server of this change in IP. In approaches similar to client side roaming, all the APs have to use the same frequency for such a technique to work unless the clients can scan multiple frequencies at

once. This technique is dependent on the capabilities on the client which might not always be present.

One other major drawback of the client side roaming mechanism is that the switching is initiated by the client. This causes many problems including inefficient load balancing. Client reauthenticating itself with a new AP is beneficial for the client but may not be for the overall network. This method of reauthentication may crowd a few APs, while other APs do not see any traffic at all. A crowded AP is not useful to any client as the bandwidth is shared by all the clients associated to that AP which leads to low throughput. Also, for analytical purposes, it is important for a network administrator to be able to control and see the traffic experienced by the APs. A solution to this problem is for the network to intelligently hand over users to neighboring APs in order to increase the efficiency of the overall network. With the client side roaming model such moderation is not possible.

On the other hand, specification changes in 802.11k [80211IEEE2008] and 802.11r [80211IEEE2012] enabled roaming between APs. 802.11k was the first standard that enables a user to scan for other APs, while the user was already connected to one. This means that the user can now scan for APs while the device is roaming to preauthorize a connection prior to disconnection in a way similar to how the Intel's client side roaming worked [Intel15a].

802.11r [80211IEEE2012] was mostly focused on the performance of the handoff. The handoff times were drastically reduced in this specification. The speedup was made possible by enabling the negotiation and request for wireless resources to occur in parallel. Unfortunately, this technology was introduced very late and did not gain widespread adoption.

Walking through the years we still see that there is no consistent server side solution to this problem. Handing off on the client side works seamlessly for a client and also works well on a network that is less dense. When the network becomes denser, this handover must go through a server side/network side entity to ensure that the network is prepared and has the time to allocate sufficient resources to the users. The server side entity will also use network statistics to determine if the switch is favorable on the network side in terms of maintaining latency and providing consistent bandwidth.

However, there is one solution that solves most of the problems we are trying to address. OpenSDWN [Schulz15a] [Schulz15b] provides programmatic access over WiFi using SDN and Network Function Virtualization (NFV). The approach in [Schulz15a] provides seamless mobility over a WiFi network using a “Virtual Middlebox”. Virtual middleboxes are spread throughout the network to monitor and control user characteristics, while the SDN controller handles network characteristics such as throughput and handovers.

The advantages of this method are that each user now has seamless handover to neighboring APs and steady throughput throughout the network without the fear of losing connectivity. Moreover, network characteristics are controlled to reduce interference between APs and enhance connectivity. When a user is detected to be at the edge of a certain AP, the controller hands over the user to an AP with better connectivity to prevent dropping the connection.

OpenSDWN does not take into account factors such as traffic, user overcrowding or lack of power in a UAV might also lead to user handovers. We may need to perform user handovers between APs even when the signal strength is very strong for various reasons such as accommodating more users, accommodating priority packets, reducing network latency and reducing the number of users in an AP. OpenSDWN handovers are performed depending solely on the signal strength of the user device to prevent any drop in connection.

We will base our solution for seamless WiFi connectivity on OpenSDWN [Schulz15a] [Schulz15b] with modifications to suit our needs and unique requirements for a UAV-based environment.

2.2 Routing protocols

Routing protocols have been around since the birth of networking. Protocols were necessary to define a universal set of rules that all nodes could follow. These standards

eventually evolved for different use cases on a variety of scenarios. This section describes those protocols under three broad categories.

- Wired networking standards
- Infrastructure-based wireless networking standards
- Ad hoc based wireless routing protocols

Each of the category is described in more detail in the following sub-sections. We will list the protocols present in each category with a brief description of why the algorithm or protocol is not suitable for the problem described above (Section 1.1).

A distributed protocol (i.e., protocols where the node decides the route) does not guarantee efficient usage of overall network resources due to the distributed nature of network information. We intent to provide wireless Internet access via these UAV-formed networks and it is critical to have a central repository of the information in order to efficiently use network resources. The routing protocol must also support multiple routes and intelligently distribute traffic amongst these routes to better utilize network resources and prevent idling. This would also improve other results such as transfer speeds and packet latency. Since these are UAVs, the battery plays a major role as it is a very limited resource. A UAV with lower remaining battery must be sparsely used and all of its users must be taken care of before the UAV is in a critical state. For a routing protocol to see the overall network picture, it generally would be effective to make the decision centrally. In traditional routing protocols, the routes are decided by the nodes within the network. Nodes, however, do not know the overall network status. Hence it

would be more effective to make these decisions using a global controller. More importantly, we must provide quick alternative routes in case of failure so that other nodes will still be able to reach their destinations. This typically happens when a node fails due to battery reasons or physical failures.

2.2.1 Wired networking standards

This section explores the most popular wired networking standards, the IEEE 802.3 and the token ring network describing the benefits and disadvantages for each.

We will start with the most famous wired networking standard, the IEEE 802.3. The "Ethernet" works using the Carrier Sense Multiple Access/ Collision Detection (CSMA/CD) protocol [8023IEEE2000]. CSMA/CD first detects if a carrier is free and then sends the packet directly to the destination. However, this work on the base assumption that the router is directly connected. If this was extended to our wireless scenario, this would not work because the routes need to be fixed beforehand and these routes cannot change. Hence, if a link breaks, there is no rerouting and that route will permanently fail. Moreover, CSMA/CD does not take into account any of our other considerations, such as limiting the data transferred via a node depending on its remaining battery levels. These points should well illustrate why Ethernet or any form of Ethernet-based approach would not be applicable to this type of algorithm.

Similarly, we see that any routing protocol that is based on the wired scheme, such as Token ring [Bux89] or Token bus [TokenBus16] make similar assumptions about the

nodes they are connected to. Any link failure is treated as a network failure and this cannot be recovered unless the node reconnects again.

2.2.2 Infrastructure-based wireless networking standards

Infrastructure-based wireless routing protocols come a lot closer to meet our goals. 802.11, popularly known as “WiFi”, has revolutionized the way devices communicate and has given birth to a new age of devices like the smartphones and laptops. WiFi is an infrastructure based networking standard that also has the ability to work ad hoc.

Using 802.11 protocols to solve the described problem would mean that there needs to be a centralized AP where all the nodes can communicate with. This is not possible when there is an emergency deployment. Moreover, the area covered by the UAVs could span many square kilometers. 802.11 also does not support priority levels nor any QoS guarantees for packets. The Medium Access Control or MAC protocol in 802.11 does not allow packets to be treated preferentially. This implies that all packets transmitted through an 802.11 network are treated the same. This would not be ideal as we have packets that need to be transmitted in urgency and some that could be delayed in order to provide network resources for packets in need.

2.2.3 Ad hoc wireless routing protocols

Ad hoc wireless routing protocols gained popularity in the early 1990s with the explosion of tiny sensors capable of communications. This meant that these nodes could communicate without the presence of a bulky CPU attached to them. These protocols

could be applied in various parts, including temperature monitoring and weather reporting, unmanned vehicular missions over space, video surveillance and many other applications.

We can broadly classify Ad hoc routing protocols into four categories [Royer99a]:

1. Table-driven or proactive routing protocols: This type of protocols maintains a fresh list of destinations and their routes by periodically distributing routing tables throughout the network.
2. On-demand or reactive routing protocols: This type of protocols finds a route when needed (on demand) by flooding the network with RREQ or Route REQuest packets.
3. Hybrid (proactive and reactive) routing protocols: This type of protocols combines the advantages of proactive and reactive routing protocols. The routing is initially established with some proactively prospected routes and then serves the demand from additionally activated nodes through reactive flooding. The choice of one or the other method requires predetermination for typical cases.
4. Hierarchical routing protocols: With this type of protocols the choice of proactive and of reactive routing depends on the hierarchic level in which a node resides. The routing is initially established with some proactively prospected routes and then serves the demand from additionally activated nodes through reactive flooding on the lower levels. The choice for one or the other method requires proper attribution for respective levels.

Optimized Link State Routing (OLSR) [Clausen03] [Jacquet01] is one of the most popular table-driven routing protocols. It is widely used for networks that have a tight delay constraint. OLSR works by initially creating a list of its neighbors and obtaining the list of neighboring nodes from each of its immediate neighbors. It then filters out the minimum set of nodes so that it can reach all the nodes and stores the next hop in a neighboring table. So when a packet needs to be transmitted to another node, the node looks up on the table to see which node would be the next hop to send the packet over. "HELLO" messages are transmitted periodically in order to determine which nodes are still in range and the tables are updated appropriately.

The advantages of OLSR are that in a low mobility scenario the nodes do not have to update as quickly, meaning that reliable connectivity could be established with little overhead throughout time [Clausen01]. If a node goes down over the course of time, OLSR will eventually adapt itself to work around that node. Alternate paths will be established and communication will be eventually re-established.

OLSR does not measure the state of a link in the network. OLSR only measures if a certain link exists/still exists between the nodes, but it does not optimize the route based on the amount of bandwidth required. In other words, OLSR will select a next node that offers connectivity to more nodes over a route where the network resources are underutilized. OLSR was defined for the purpose of transferring control packets (as in the case of sensor networks) and not efficiently using the resources in the network. OLSR does not offer a multi-route solution. If a large amount of data is to be transferred via the

network, it becomes critical that any protocol selected for this purpose offers a multi-route solution. When a node goes down, an alternate path is established after a temporary blackout in order to regain connectivity. This is a major drawback as data is not transmitted during this period. The data is stored in a node that could lead to buffer overflow or dropping of packets and high delays in packet transmission. Therefore, OLSR is not suitable for UAV to UAV communication as required by the problem described in the thesis.

Optimized Link State Routing version 2 (OLSRv2 or olsrd2) [Barz13] is the second revision of OLSR that offers significant performance improvements and other benefits over its predecessor. OLSRv2 is known to show significant improvements in route discovery times, much better performance in terms of bandwidth and data transfer volume, offers support for discovery of the shortest link to a given node and lower power consumption per node [Clausen01] [Vara15]. The new features may overcome some of the disadvantages that were part of OLSR, but they still do not solve the critical downsides of its predecessor. OLSRv2 still does not guarantee that the most optimal route in terms of both bandwidth and latency will be selected. In addition, multi-route packet transfer is still not possible and there is no provision for priority packet transfer.

Ad hoc On-Demand Distance Vector (AODV) Routing is a popular routing protocol for reactive routing [Perkins03]. AODV was designed for mobile nodes when the network is constantly changing. AODV uses RREQ packets in order to request for routes and uses these routes until the routes are no longer reachable. In order to efficiently use network

resources, we may need to switch routes during operation and therefore fixed routes until they expire would be disadvantageous. AODV does not support priority levels or multipath routing. Many other enhancements to AODV fail to address these concerns [Royer99b] [Zapata02] [Marina01] [Narasimhan13].

AODV however is useful when the network is not stable and all assigned routes by the controller are no longer available. We will use AODV in order to discover a route to the controller as it is a reactive protocol. Once a route is discovered a node will await further instructions from the controller in order to re-establish connectivity with the network. The use of AODV is discussed in detail in chapter 3.

Destination-Sequenced Distance Vector routing (DSDV) [Perkins94] is a table-driven routing protocol that is not used much these days. DSDV is based on the Bellman-Ford algorithm [Bellman58] [Ford56]. DSDV was the predecessor of many other widely used routing protocols like AODV. DSDV, however, still offers some features. DSDV was originally created to avoid the routing loop problem, a problem that plagued many protocols before it. The routing loop is when packets are routed along a loop amongst the nodes thus creating no network throughput.

DSDV works by using a simple table that records how each node can be reached. It stores the next hop and the number of hops required to reach the destination node. This table is primarily static although some improvements have been made to this over the years [Lu11] [Liu07] [Rahman09]. DSDV offers no quality of service (QoS). It does not make

any guarantees on the latency or availability of the nodes in between. It does not offer support for multiple paths or for priority path for priority packets.

The Wireless Routing Protocol (WRP) [Murthy96] is a proactive unicast routing protocol for mobile ad hoc network. Since it is a proactive routing protocol, it maintains a table of all the destination nodes in the network and periodically updates this table to keep up to date with any network changes. This is similar to how DSDV works and has very similar advantages and disadvantages to DSDV. There is no centralized monitor for the network and hence data communicated is not always through the best route or the route that maximizes the network throughput.

A paper containing a review of routing protocols showed that all the protocols had disadvantages such as lack of multi routes, priority levels and central decision server [Royer99a]. Routing protocols such as Temporally Ordered Routing Algorithm (TORA) [TORA16], Associativity-based routing (ABR) [Toh97], Signal Stability Routing (SSR) [Dube97] and Scalable Source Routing [Fuhrmann06] lack support for priority levels within packets and multiple paths for packet distribution throughout the network. An enhanced version of ABR includes support for multipath routes but lacks the support for priority levels [Carthy05].

Clustered Gateway Switch Routing (CGSR) Algorithm is a table driven routing protocol based on the DSDV protocol [Chiang97]. Mobile nodes are aggregated into clusters and for each cluster a cluster head is elected. These cluster heads act as the gateways for

packets to go through. Gateway-based approach enables CGSR to provide priority tokens to internal nodes to transmit information ahead of other packets. However, since all the traffic flows through the cluster heads, the cluster heads potentially become the bottleneck while other nodes are underutilized.

A protocol that works similar to CGSR is Cluster Based Routing Protocol (CBRP) [Jiang99]. CBPR divides the network into clusters but unlike CGSR, CBPR is a reactive routing protocol. When a source has to send data through the network, the source floods the neighboring cluster heads with route request packets. The cluster heads allocate routes and the source uses the assigned route to communicate. Similar to CGSR, CBPR lacks a central repository of information and hence the ability to efficiently use all resources of the network. CBPR also lacks the ability transmit packets in different routes based on the priority level.

In conclusion, we see that ad hoc routing protocols satisfy one or few of our requirements but not all at the same time. We see that most routing protocols lack the support for priority packets and the ability to transmit packets in multiple routes. Most routing protocols and the network standards were designed for different scenarios and hence do not meet all of our requirements.

Chapter 3

Proposed Approach

This chapter describes the methodology and the high level architecture of the controller based routing approach and modifications to OpenSDWN. This chapter also explains the proposed MAC protocol in order to accommodate priority packets.

To solve the problems listed in Section 1.2, we designed a new approach to UAV-UAV communications for ad hoc networks inspired by SDN. We have established in previous chapters that a UAV level of knowledge is not sufficient to efficiently use all available network resources and a central repository of information is required in order to effectively handle the traffic flow in the network.

To simplify the problem, we will split the problem into two sub-problems:

1. UAV-UAV communication algorithm: This algorithm is used to set up the UAV backbone and allow communication
2. UAV-User communication: Providing seamless WiFi roaming over the UAV-UAV backbone.

An overall network structure is illustrated in Figure 3.1. This helps separate two loosely coupled concerns: UAV-UAV communication as shown in Figure 3.2 and UAV-User communication as shown in Figure 3.3. This separation is possible because the users can communicate only with the UAVs. UAVs communicate with the controller in order to

provide connectivity to the users and other UAVs. The UAV-UAV communication is responsible for transferring packets generated by the users and UAVs through the network and UAV-User communication takes care of managing user traffic within the network. The controller coordinates with the two layers in order to make decisions, in turn keeping the network stable and functional.

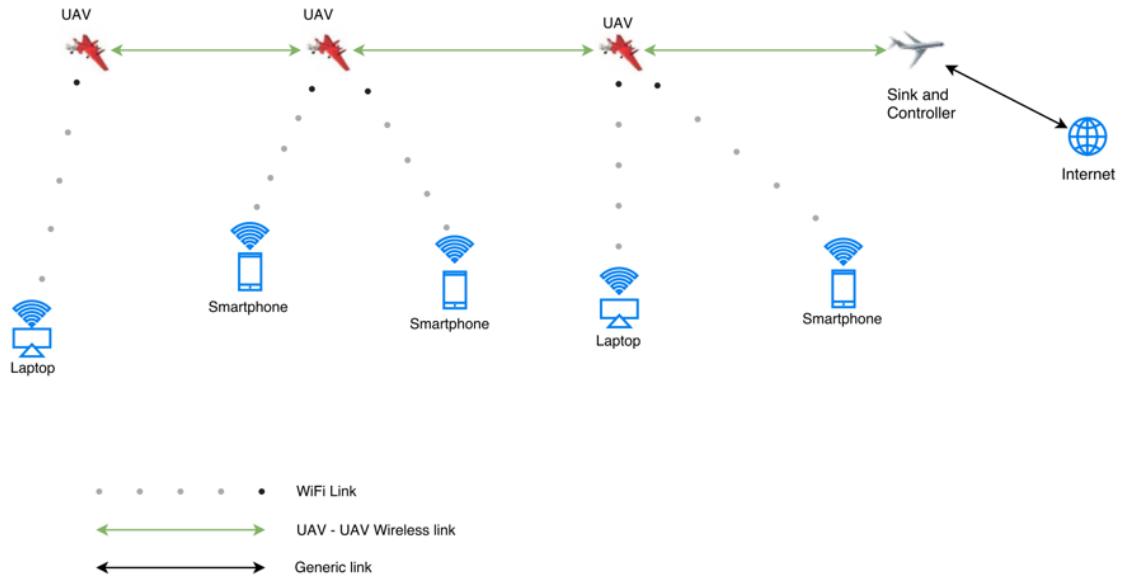

Figure 3.1: Overall network architecture

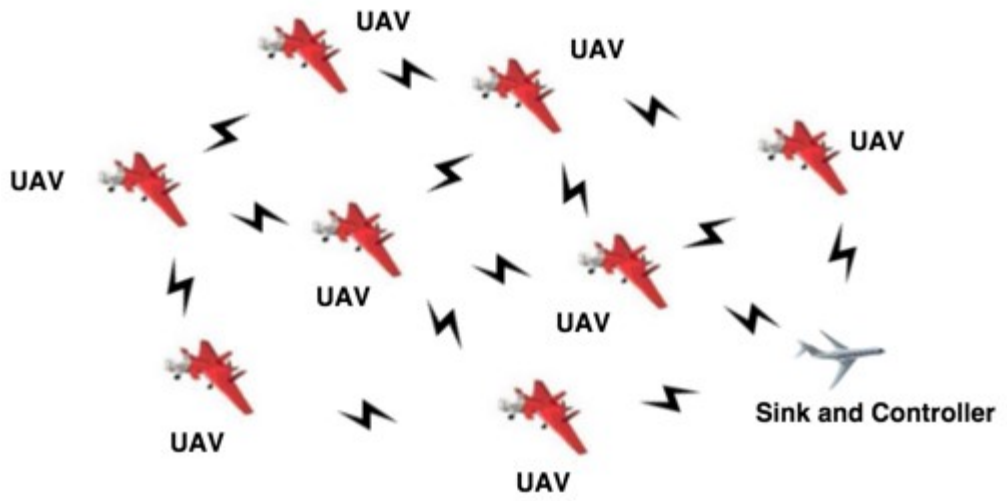

Figure 3.2: UAV-UAV communication backbone

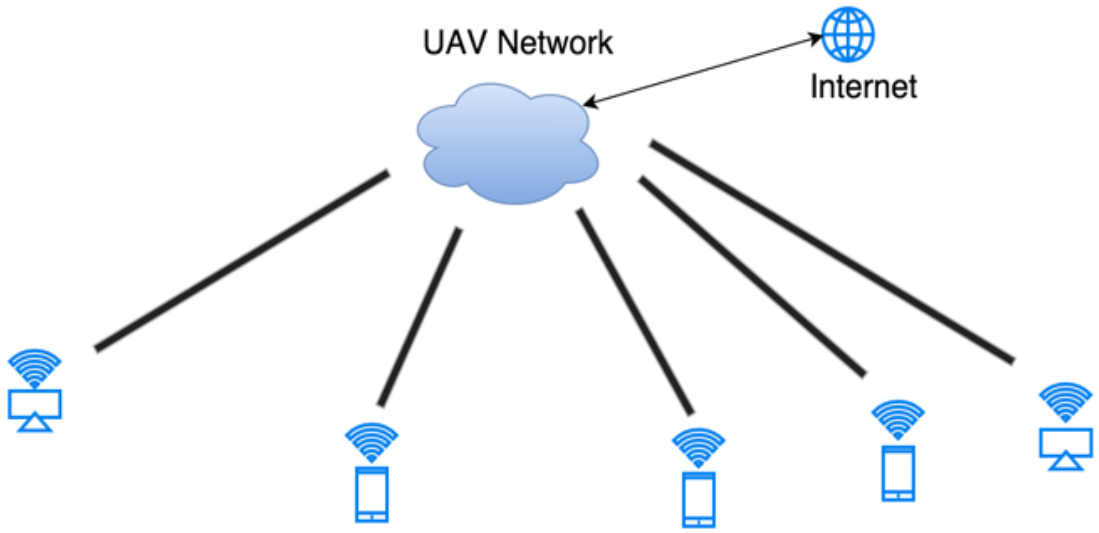

Figure 3.3: UAV-User communication diagram

The rest of this chapter is split into four parts:

1. The UAV-UAV communications algorithm with the controller.
2. Modifications made to OpenSDWN in order to make the WiFi work seamlessly within our UAV-UAV backbone.
3. Handling low power and redirecting routes to minimize packet loss.
4. Transferring users when an AP is overcrowded or when a UAV is in a low power state.

3.1 Algorithm for UAV-UAV communication

Figure 3.4 shows a simplified ad hoc network. We have a source that needs to transmit information to the destination (Internet) and the packet has to go through a series of hops (as directed by the controller) to reach the sink. The sink then relays the packet to the destination completing the packet transfer.

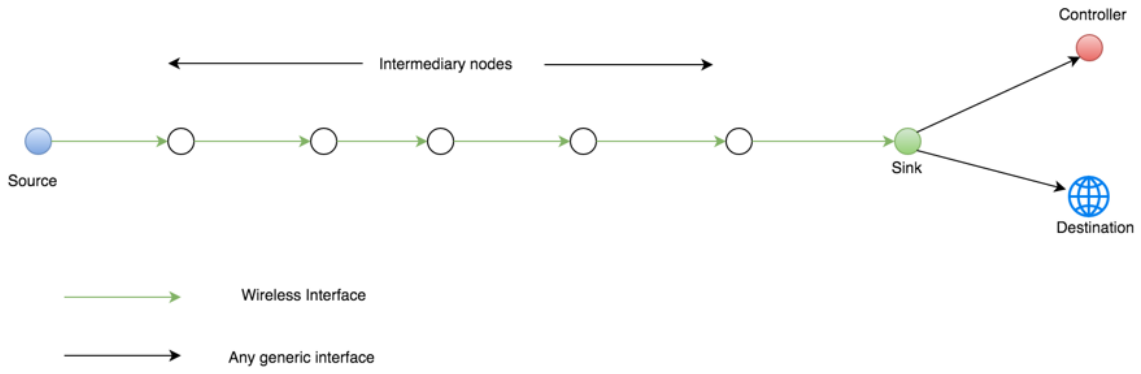

Figure 3.4: Base network architecture

3.1.1 Network assumptions

Before describing the proposed new algorithm, the following assumptions are made about the network:

1. Each UAV is fitted with a GPS module, Gyroscope, accelerometer, Proximity sensor and Altimeter.
2. Each UAV has a minimum of 2 radio modules supplied with them, one for UAV-UAV communication and one for UAV-User communication.
3. The radio module used to for the UAV-UAV communication has the ability to transmit and receive information at a frequency range of 950 - 1050 MHz. This frequency was chosen because it offers high bandwidth and attenuates lesser than higher frequencies. This enables long range, high speed communication between the UAVs. This value can be changed to suit the needs of the implementation. However, the methodology described can be adapted to other frequencies as well.
4. The radio module used for UAV-User communication can communicate at 2.4GHz and 5GHz as this is typically the case with the latest WiFi standards.
5. The initial positions of the UAVs are set up beforehand and each UAV is set up so that there is at least one other UAV in its range (typically done through a pre-planning phase). In other words, no UAV is outside the network and they can all reach the sink (either directly or through a sequence of hops).
6. All UAVs are launched at the same time.
7. For the purpose of the thesis, the controller is built into the sink.
8. UAVs are assigned ID before they are deployed in the network. These IDs are hardcoded and cannot be changed by the controller or the UAV.

9. The controller and the sink have a fixed ID represented as “CID”.
10. The controller is a more powerful UAV. It has higher processing capabilities and higher memory. This thesis assumes that the controller is reliable and always “up”.

3.1.2 The controller

The controller is a UAV that monitors the network using control packets communicated from other UAVs to the controller and back. The controller, however, is not part of the data network, i.e., it does not get involved in the transfer of data throughout the network. The controller gathers information from the UAVs “reporting” to it and responds with control packets that contain the routes for packets in each priority level and information for the continued functioning of the network. Since these control packets communicate the state of the UAV, we will refer to them as a “report”. Each UAV on the data network periodically sends reports to the controller about its current state and the state of its immediate environment. These reports are stored by the controller for both monitoring and analysis purposes. This model provides the network administrators with all the information needed to monitor the network.

3.1.3 UAV structure

Each UAV is used to provide connectivity for a certain area. UAVs are equipped with a set of sensors and radio units in order to stay in position. This thesis assumes that each UAV is equipped with a sufficient battery to last one flight time, GPS (or equivalent) and gyroscope modules to read the location information and keep the UAV in balance,

respectively. This thesis also assumes that multiple sensors such as the proximity sensor along with IR sensors, as illustrated in Figure 3.5, are placed in order to avoid collision. All these modules are managed by the "Flight control" module that takes care of movement and altitude adjustments.

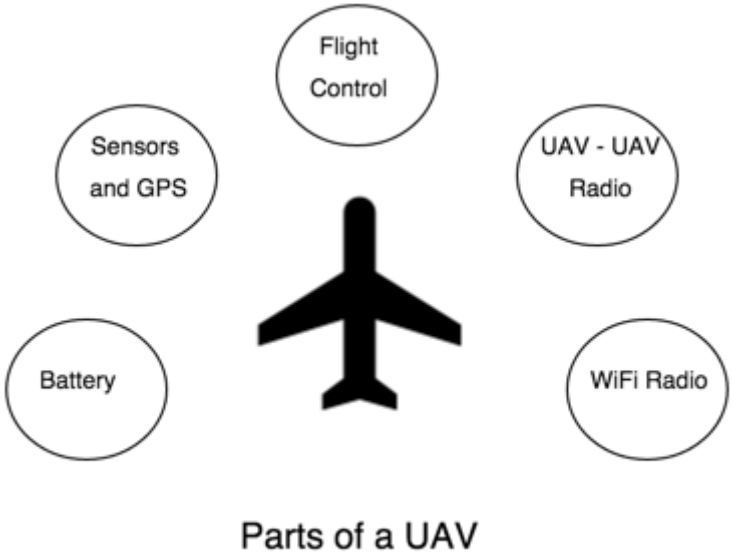

Figure 3.5: Parts of a UAV

Each UAV is also equipped with two sets of radio modules, one for UAV-UAV communications and one for the UAV-User communication as illustrated in Figure 3.5. The first radio module used for UAV-UAV communication transmits and receives at frequencies from 950 MHz to 1050 MHz. To provide users with Internet connectivity, the WiFi (802.11ac) protocol is used. We use the second radio in order to create the WiFi network that the users can connect to. This WiFi network is coordinated by the controller (explained later) in order to provide seamless Internet connectivity. Figure 3.6 shows a chart of all the algorithms in the thesis.

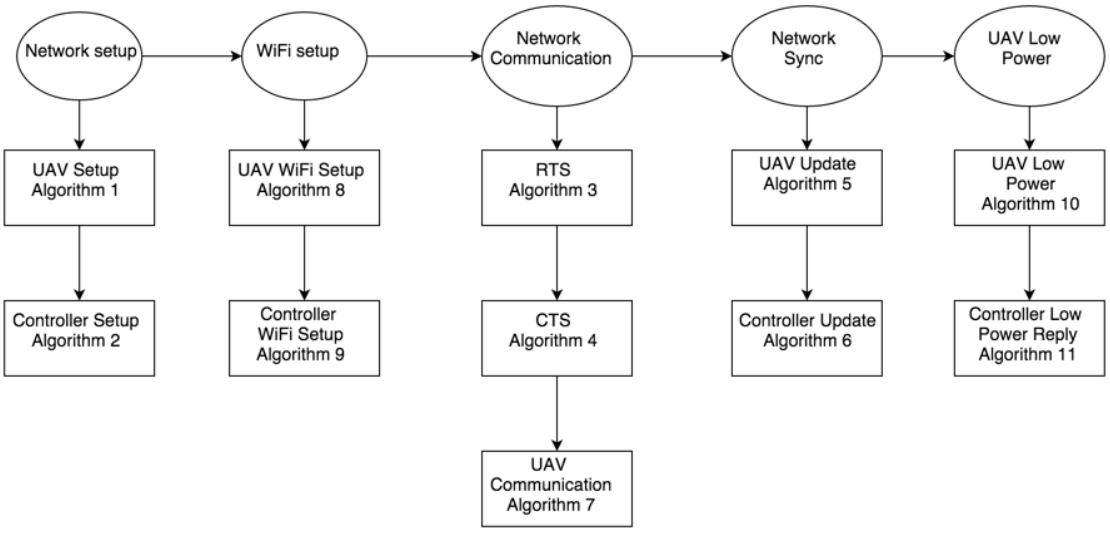

Figure 3.6: Chart of all algorithms

3.1.4 Priority levels

In order to maintain the stability of the network, packets have to be prioritized. For example, a report packet has the highest priority level and needs to be communicated to the controller urgently. Since this protocol differs from typical ad hoc networks, where the decision is made by the UAV, we need to make sure that these decisions are made and communicated quickly without delay. To facilitate this, packets transmitted through the network are classified into one of the four categories described below:

1. Priority control packets: These are packets with the highest priority and require immediate attention. The key difference between the traditional networks and SDNs is that the controller is the source of all information. This places a big responsibility on the controller to communicate responses as quickly as possible to the UAVs that require attention. For example, assume there is a strong influx of users that has led to the severe drain of the UAVs power level. In this case, the

UAV generates a report to the controller about this issue so that the network can respond to counteract this influx of users by reducing the bandwidth of individual users or increasing the density of UAVs in that area or by handing over few users to neighboring UAVs.

2. Non-priority control packets: These are control packets that are sent to the controller at regular intervals such as the information packets that are sent every 30 seconds. These packets are important for the continued maintenance of the network and hence have a priority higher than data packets. These packets generally update the information on the controller with new information. These packets are also of interest to the network administrators to monitor and maintain the network.
3. Priority data packets: Data packets are categorized into two levels: priority and non-priority. These levels are determined by the UAVs by monitoring the data sent over the network. For example, when video information is transmitted over the network, it can be recognized by detecting the video headers within the data packets. We opted to snoop information because this approach introduces no changes on the client device. Usually, video information encoded in h.264 [H264] is divided into I, B and P frames. I frame contain the base information of the frame and P and B frames are predictive frames that overlay on top of the I frame to create the next frame. If P frames are lost or delayed, the user would experience only a drop of a single frame. Whereas if an I frame is dropped, all frames from that I frame are rendered incorrectly causing inconvenience to the user. Hence, I frames would fall under priority data packets category.

4. Non-priority data packet: All data packets that do not fall under the priority data packets fall under the non-priority category. For example, requests to access a web page or streaming music from the Internet.

3.1.5 Network setup and discovery

For the purpose of this thesis, we assume that all UAVs are dispatched from a base station. During dispatch, each UAV receives an initial location to begin setup. The controller is placed close to the base station in order to maintain connectivity and provide Internet access to other UAVs. This location is then passed on to the flight control module present in the UAV that guides the UAV to the specified location.

Once all UAVs have reached their destination, each UAV turns on its UAV-UAV communication radio and starts listening to HELLO messages. UAVs start transmitting HELLO messages, one message per second as illustrated by Figure 3.7. The UAVs listen to HELLO messages from other nodes during this 30 second time period. This enables the UAVs to discover its neighbors. It is recommended that each UAV transmits HELLO messages 30 times in order to average out errors due to environmental factors like wind, objects in line of sight or even interference due to other radio sources.

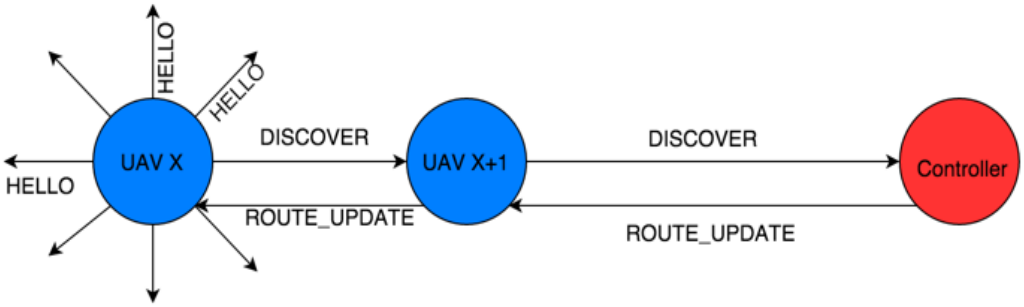

Figure 3.7: Network setup and discovery

Once a UAV has discovered its neighbors, it transmits a DISCOVER packet to the controller. Each DISCOVER packet contains the following fields:

1. UAV ID
2. Number of HELLO messages received
3. Average Signal Strength (dBm)
4. Variance of Signal Strength (dBm)
5. Highest received signal strength (dBm)
6. Lowest received signal strength (dBm)

Once a DISCOVER packet is constructed, all UAVs transmit this packet to the controller using AODV, as highlighted in Chapter 2. AODV is used because the routes have not been established by the controller yet.

The algorithm for network setup for UAV is summarized below.

Algorithm 1: Network Setup - UAV

1. Wait for a random backoff time
2. Listen for HELLO messages
3. Transmit HELLO messages 30 times
4. **If** there was a collision
5. Backoff for random time
6. **End If**
7. **For each** HELLO message received
8. Record the UAV_id and signal strength from the HELLO message
9. **End for each**
10. **If** the UAV_id already has an entry in the table
11. Increase the readings column and replace signal strength with the average value
12. **End If**
13. Transmit DISCOVER message to the controller with all the neighboring UAV information
14. Wait for ROUTE_UPDATE packet from the controller
15. Update the routing tables.

The controller waits for DISCOVER packets from all the UAVs in the network until a timeout has elapsed. Any UAVs that failed to communicate are marked as lost so the network administrators can take an appropriate action. Using the information within the DISCOVER packets, the controller constructs an adjacency table. Using the average signal strength values, the average modulation scheme for the communication is calculated (explained below). Since the frequency of communication is fixed, the maximum throughput for the given measurements is calculated using the Shannon-Hartley theorem [Hartley28]. This way, the controller knows how much information can be transmitted with each link.

For the purposes of this thesis, the calculation for the routes will use Dijkstra's algorithm [Dijkstra16] and the Ford-Fulkerson Algorithm [FordFulkerson16] in order to calculate next hops for control and data priority levels, respectively. Since control packets are required to reach the destination at its earliest, we need a route with the smallest number of hops to the controller. The simplest way to accomplish we use Dijkstra's algorithm to determine the shortest path through the UAVs. This method is used to calculate the main route. The alternative route is calculated by removing the first link from the main route from the graph and using Dijkstra's algorithm on the resulting graph. This is based on the assumption that there are multiple connections available between the UAVs. If a UAV has no other connections, the main and alternative hops point to the same UAV in the network. Similarly, the network flow algorithm is used to calculate the main and alternative routes for data packet priority levels. To calculate the main route, from the residual graph from Ford-Fulkerson algorithm, we will iterate through all routes in order to find one path that is allows for maximum throughput. This is the main route. We will remove the first edge from the source and redo the Ford-Fulkerson algorithm in order to obtain the alternate path. After the calculations, SYNC_TIMEOUT is set by the network administrator. SYNC_TIMEOUT specifies how often a UAV sends reports to the controller under stable conditions. If the network is not stable, reports are generated immediately. All of the above information is bundled into a ROUTE_UPDATE packet that contains the following information:

1. UAV ID
2. SYNC_TIMEOUT (in seconds or ms)
3. Optimal route for priority control packets (UAV ID)

4. Alternate route for priority control packets (UAV ID)
5. Optimal route for non-priority control packets (UAV ID)
6. Alternate route for non-priority control packets (UAV ID)
7. Optimal route for priority data packets (UAV ID)
8. Alternate route for priority data packets (UAV ID)
9. Optimal route for non-priority data packets (UAV ID)
10. Alternate route for non-priority data packets (UAV ID)

The algorithm for Controller network setup is summarized below:

Algorithm 2: Network Setup - Controller

1. Wait for the DISCOVER packet from all the UAVs within a TIMEOUT
2. Update adjacency tables with the information from the DISCOVER packets
3. Calculate the average throughput and update tables
4. Compute the routes for each UAV
5. Specify SYNC_TIMEOUT for each of the UAVs in the packet
6. Send the ROUTE_UPDATE packet to the UAVs

Let us consider N0 to be the UAV of concern as shown in Figure 3.8. N0 is the source that produces constant data and control packets to be transmitted through the network. Assume that the next hops assigned for control packets is N1 and data packets is N3. The route assignment packet sent out by the controller consists of the following information:

1. UAV ID - N0
2. Optimal route for priority control packets - N1
3. Alternate route for priority control packets - N2
4. Optimal route for non-priority control packets - N1

5. Alternate route for non-priority control packets - N2
6. Optimal route for priority data packets - N3
7. Alternate route for priority data packets - N2
8. Optimal route for non-priority data packets - N3
9. Alternate route for non-priority data packets - N2

Figure 3.8 illustrates a complete topology of all the paths taken by N0. This way the data is distributed throughout the network and high performance can be achieved.

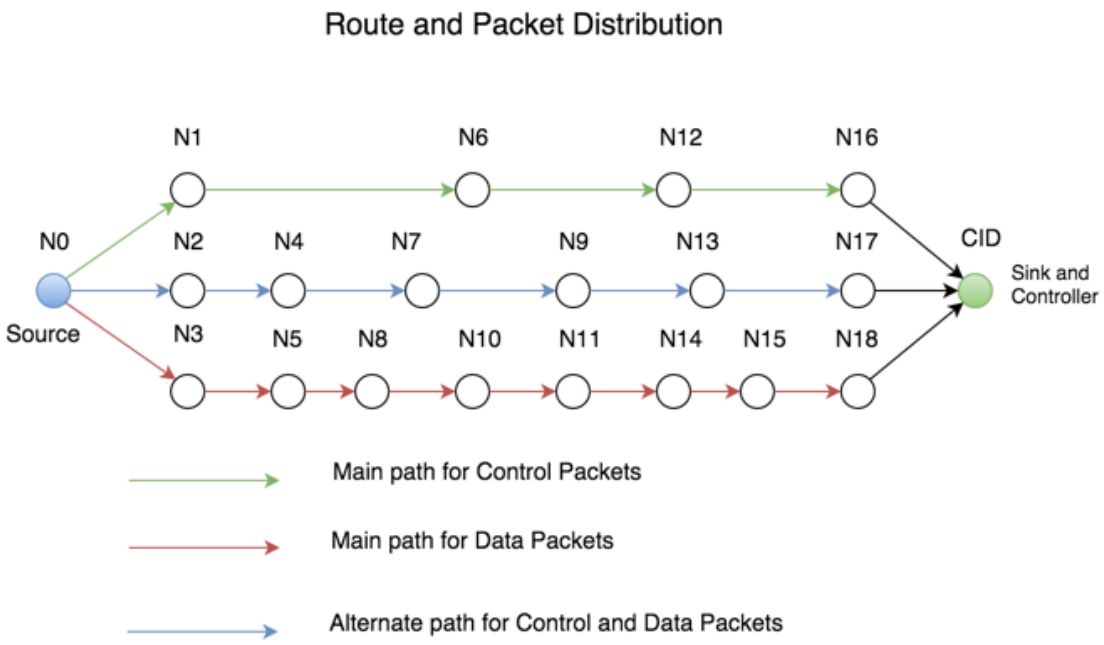

Figure 3.8: Paths for different priority levels assigned by the controller

3.1.6 Responsibilities of the Controller

The controller is responsible for collecting information from the UAVs about their surrounding and internal state. This information is used by the controller to make decisions that are beneficial to the network. Therefore, packets containing information or reports must be prioritized in order to keep the network functioning at the highest efficiency.

Each report packet collected by the controller contains the following information:

1. UAV ID
2. Battery level (in %)
3. Number of users connected to the UAV
4. Last known signal strength of each of its neighboring UAVs and the time at which this information was recorded (see last known power table below)
5. User traffic characteristics (see User characteristics table below)
6. Data originated from this UAV since last control packet
7. Total amount of data transmitted
8. Total amount of data received
9. Calculated position (X, Y and Z)
10. X Position error (X1, X2 - X range)
11. Y Position error (Y1, Y2 - Y range)
12. Z Position error (Z1, Z2 - Z range)
13. Number of packets dropped

The last known power level table contains the following information:

1. UAV ID (UAV ID of the neighboring UAVs)
2. Last known signal strength (in dB)
3. Bandwidth for communication (is a function of power level in a fixed frequency range)

The user characteristics table consists of the following information:

1. User ID
2. Total data transmitted (to user)
3. Total data received (from user)
4. Bandwidth limit

The UAV generates a report periodically to the controller or when there is an alarming change in traffic characteristics, user count or a change in neighboring UAVs. The controller then stores this information in a database and responds with a packet containing the following information similar to the example presented in Figure 3.8:

1. Optimal route for priority control packets (UAV ID)
2. Alternate route for priority control packets (UAV ID)
3. Optimal route for non-priority control packets (UAV ID)
4. Alternate route for non-priority control packets (UAV ID)
5. Optimal route for priority data packets (UAV ID)
6. Alternate route for priority data packets (UAV ID)
7. Optimal route for non-priority data packets (UAV ID)

8. Alternate route for non-priority data packets (UAV ID)

3.1.7 Physical layer

The physical layer used in this thesis can be any generic wireless layer operating at a certain frequency. We will represent this frequency to be *f*. This frequency combined with the modulation scheme of the communication used determines the bandwidth available between any two UAVs of the network. A modulation scheme is chosen depending on the signal strength (BPSK, QPSK, 8-QAM, 16-QAM, 64-QAM, 256-QAM) between a pair of UAVs. The decision to choose a specific modulation scheme is done using a technique called adaptive modulation. When the Signal to Noise Ratio (SNR) is very high, an efficient modulation scheme such as 64-QAM or 256-QAM is chosen for communication. When the SNR is low, a defensive modulation scheme such as BPSK or QPSK is chosen to reduce the Bit Error Rate (BER) during communication.

Depending on the modulation scheme, it is easy to predict the maximum capacity of the channel by using the Shannon - Hartley theorem [Hartley28] as shown below:

$$C = B \log_2 \left(1 + \frac{S}{N}\right)$$

Where:

C is the channel capacity

B is the bandwidth of the channel (twice the frequency *f*)

S is the signal strength

N is the strength of noise

Since the frequency of a channel is fixed to *f* and the ratio of S/N is obtained by the UAVs, it is possible to determine the maximum capacity of a given channel.

3.1.8 Medium access control

The MAC layer is quite different from the MAC in traditional ad hoc networks. A receiver UAV must be notified of the types of packets that are to be sent by each UAV in order to prioritize transmission. Since there is a division of packet priorities, we need to define a new MAC protocol. This new MAC is similar to the existing MAC layer from the 802.11ac standard with a few modifications to accommodate priority levels. The MAC works by communicating Request To Send (RTS) and Clear To Send (CTS) packets. RTS packets are sent when the UAV, say UAV *A*, that wants to transmit information to another UAV, listens for any communication in that channel and then transmits the RTS packet to the receiver UAV, say UAV *B*. UAV *B* makes sure that there are no hidden terminals for the transmitter UAV and sends a CTS packet if it is clear for communication to UAV *A*. UAV *A* is now clear to send information to UAV *B*.

This exchange of RTS and CTS is crucial to avoid any collisions during the communication. This exchange of packets ensures that no UAV communicates out of turn. However, the traditional RTS/CTS packets used do not communicate the types of packets that are being sent over the network. For example, UAV *A* and UAV *B* try to communicate to the sink via another UAV, say UAV *C*. UAV *A* has only non-priority data packets, whereas UAV *B* has a crucial control packet to transmit over this link.

Ideally, UAV *B* should be allowed to go first, but the traditional RTS packet does not accommodate to the requirement. RTS packets only indicate that a UAV wants to transmit data over the link, but gives no information on the type or amount of data to be transmitted. It is also important for the network to be "fair". Fairness can be defined in many ways, but for the purposes of this thesis, we will define fairness as sharing the total data bandwidth among all the UAVs equally.

A typical RTS (or CTS) packet contains the following information. The RTS frame contains five fields, which are:

1. Frame Control
2. Duration
3. Receiver Address (RA)
4. Transmitter Address (TA)
5. Frame Check Sequence (FCS)

The CTS frame contains four fields, which are:

1. Frame Control
2. Duration
3. Receiver Address (RA)
4. Frame Check Sequence (FCS)

The RTS packet contains a Frame Control and a Frame Check Sequence that are used to ensure that there were no errors in the frame. However, the fields of interest to us are the

duration, receiver address and transmitter address. The duration, specified in time, is the estimated amount of time required to complete the transmission of all the data in the queue. Although, the important note here is the "Duration" on the CTS packet. It does not necessarily have to match with the duration requested by the sender of the RTS packet, but it could be any number smaller than that.

The traditional RTS/CTS model works well when all data packets are treated equally. However, it does not work well when there are different priority levels of packets. For example, the traditional RTS packet sent by UAV *B* in the example above would be similar to the RTS packet sent out by UAV *A*. It would not indicate the number of packets that needs to be sent in priority. Therefore, we modify this scheme to match our requirements as follows:

- The mechanism used to communicate remains the same as 802.11, but with the modified MAC layer protocol.
- Each UAV receiving an RTS packet has more information in order to decide what information and what kind of information takes more priority over other packets sent over the network.
- The RTS packet defined by this thesis is as follows. The Frame Control, Receiver and Transmitter Address remain the same, but now there are five new fields in the place of duration field.
- The CTS packet has one additional field to indicate the priority levels of packets that must be transmitted.

The fields of the RTS frame are listed as follows:

1. Frame Control
2. Receiver Address (RA)
3. Transmitter Address (TA)
4. Allowance flag
5. Priority control packets duration (0 if no packets are present)
6. Non-priority control packets duration (0 if no packets are present)
7. Priority data packets duration (0 if no packets are present)
8. Non-priority data packets duration (0 if no packets are present)
9. Frame Check Sequence (FCS)

The fields of a CTS frame include:

1. Frame Control
2. Receiver Address (RA)
3. Allowance flag
4. Duration
5. Frame Check Sequence (FCS)

The allowance flag is used to convey what priority levels of packets need to be transmitted in the case of RTS and the priority levels that are permitted to be transmitted in the case of CTS. Figure 3.9 explains in the structure of the Allowance flag. The receiving and transmitter addresses are used to identify the UAVs that the information is sent to and sent from. Once the receiver receives this information, it then identifies the

number of UAVs that requires access to the channel and sends out a CTS packet to tell the UAV with the highest priority data that it is now clear to transmit. This modified MAC protocol is used only for UAV – UAV communications.

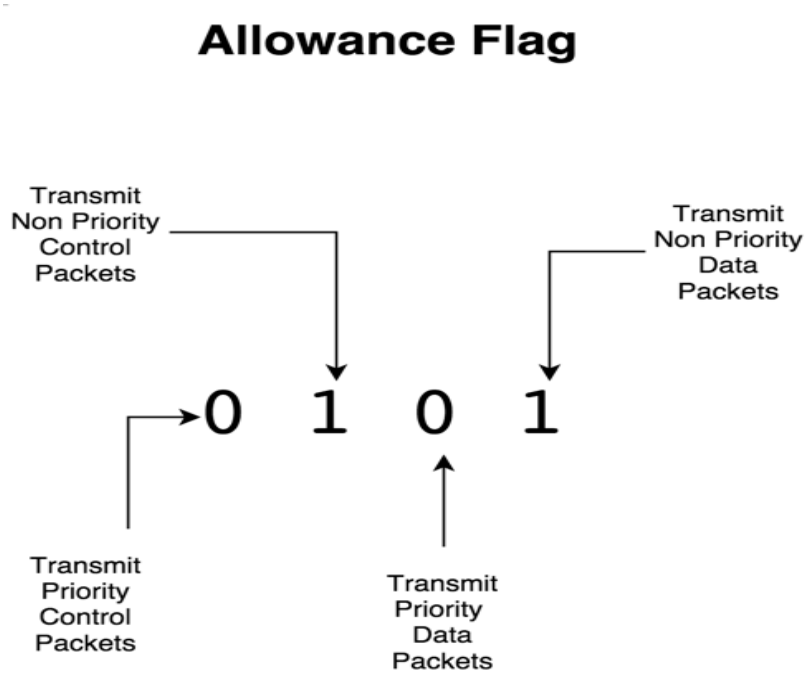

Figure 3.9: Allowance flag structure

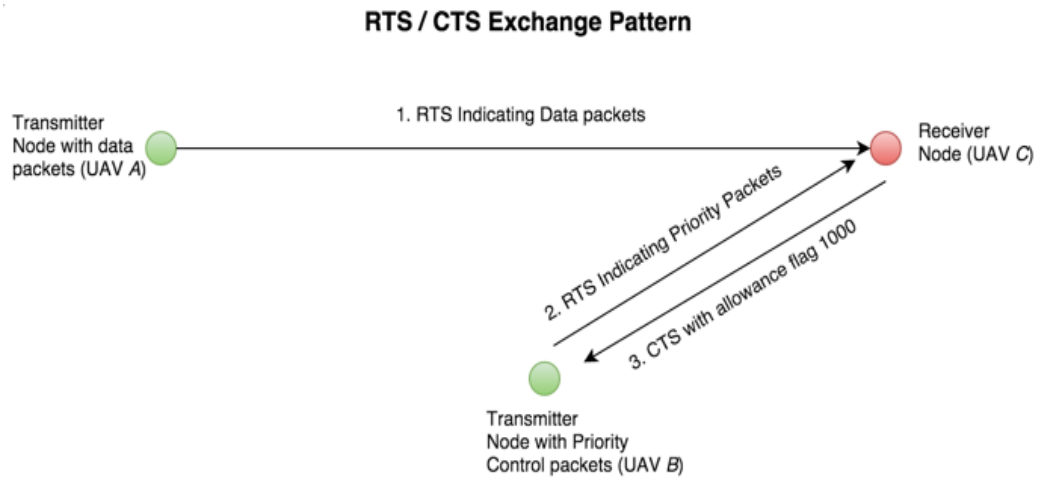

Figure 3.10: RTS / CTS exchange pattern

An example of the new proposed RTS and CTS mechanisms in illustrated in Figure 3.10. Assume two UAVs want to send information via the same channel and via the same UAV to the sink. UAV *A* has data packets to send to the receiver and UAV *B* wants to send priority control packets as shown in Figure 3.10. UAV *A* transmits an RTS packet that has a Frame Control sequence, its own ID as the transmitter address, and the receiver UAVs ID as the Receiver Address, Allocation flag set to 0011 and indicates an approximate amount of time it would require to transmit this information. Similarly, UAV *B* generates and sends an RTS packet that has the allocation flag set to 1000 and the approximate time it would take to send these packets over the channel. Now, the receiver (UAV *C*), receives the two RTS packets. UAV *C* now has all the information it requires to make the appropriate decision. In this case, UAV *C* allows UAV *B* to transmit the information first, since UAV *B* has priority control information to transmit over the network. Therefore, UAV *C* now generates a CTS packet that has the flags 1000, as illustrated in Figure 3.10, indicating that UAV *B* is now granted the permission to transmit all of its priority control information over the network. To indicate that UAV *A* is not permitted to transfer, UAV *C* also transmits a CTS with Allowance flag 0000. The purpose of the CTS with Allowance flag 0000 is to indicate that the receiver is not lost. Once UAV *B* has completed the transfer of information, the RTS window is open and UAV *A* sends RTS packet again. UAV *C* now transmits a CTS packet with the flag "0011", as shown in Figure 3.11 indicating that UAV *A* is now permitted to transmit all of its data packets over the channel. This way we can ensure that priority packets are handled faster and more efficiently than non-priority packets.

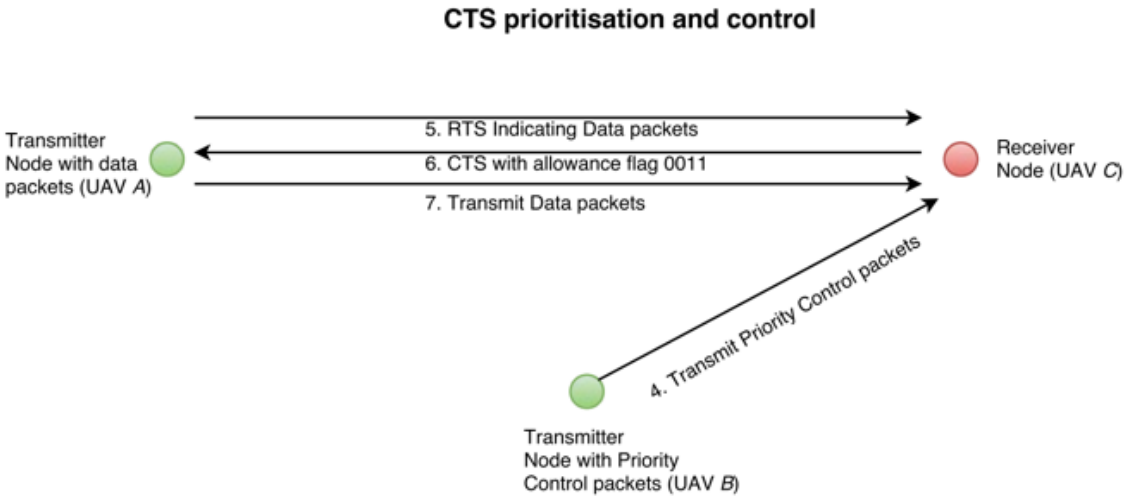

Figure 3.11: CTS prioritization and control

We have shown how the proposed MAC is more robust in delivering priority packets when compared to non-priority packets over the network. Priority control packets are necessary for the continual function of the network and maintenance, while priority data packets are bits of information that are required to provide a continual service to the users of the network. Figure 3.12 illustrates the process in a Flowchart.

Algorithm 3: RTS communication algorithm

1. **If** there are packets to transmit
2. Construct RTS packet
3. Fill in Frame Control, Receiver and Transmitter Address
4. **If** there are priority control packets in queue
5. Set the first bit of the Allowance flag to 1
6. Set duration of priority control packets
7. **Else**
8. Set the first bit of Allowance flag to 0
9. Set duration of priority control packets to 0
10. **End If**
11. **If** there are non-priority control packets in queue
12. Set the second bit of Allowance flag to 1
13. Set duration of non-priority control packets
14. **Else**
15. Set the second bit of Allowance flag to 0
16. Set duration of non-priority control packets to 0
17. **End If**
18. **If** there are priority data packets in queue
19. Set the third bit of Allowance flag to 1
20. Set duration of priority data packets
21. **Else**
22. Set the third bit of Allowance flag to 0
23. Set duration of priority data packets to 0
24. **End If**
25. **If** there are non-priority data packets in queue
26. Set the fourth bit of Allowance flag to 1
27. Set duration of non-priority data packets
28. **Else**
29. Set the fourth bit of Allowance flag to 0
30. Set duration of priority control packets to 0
31. **End If**
32. **End If**

Algorithm 4: CTS communication algorithm

1. Wait for RTS_RECEIVE_WINDOW milliseconds
2. Receive all incoming RTS packets
3. Prioritize RTS packets according to Allowance flag bits
4. Send CTS packet to the UAV that needs to transmit with highest priority level
5. Send CTS with Allowance flag 0000 to all other UAVs

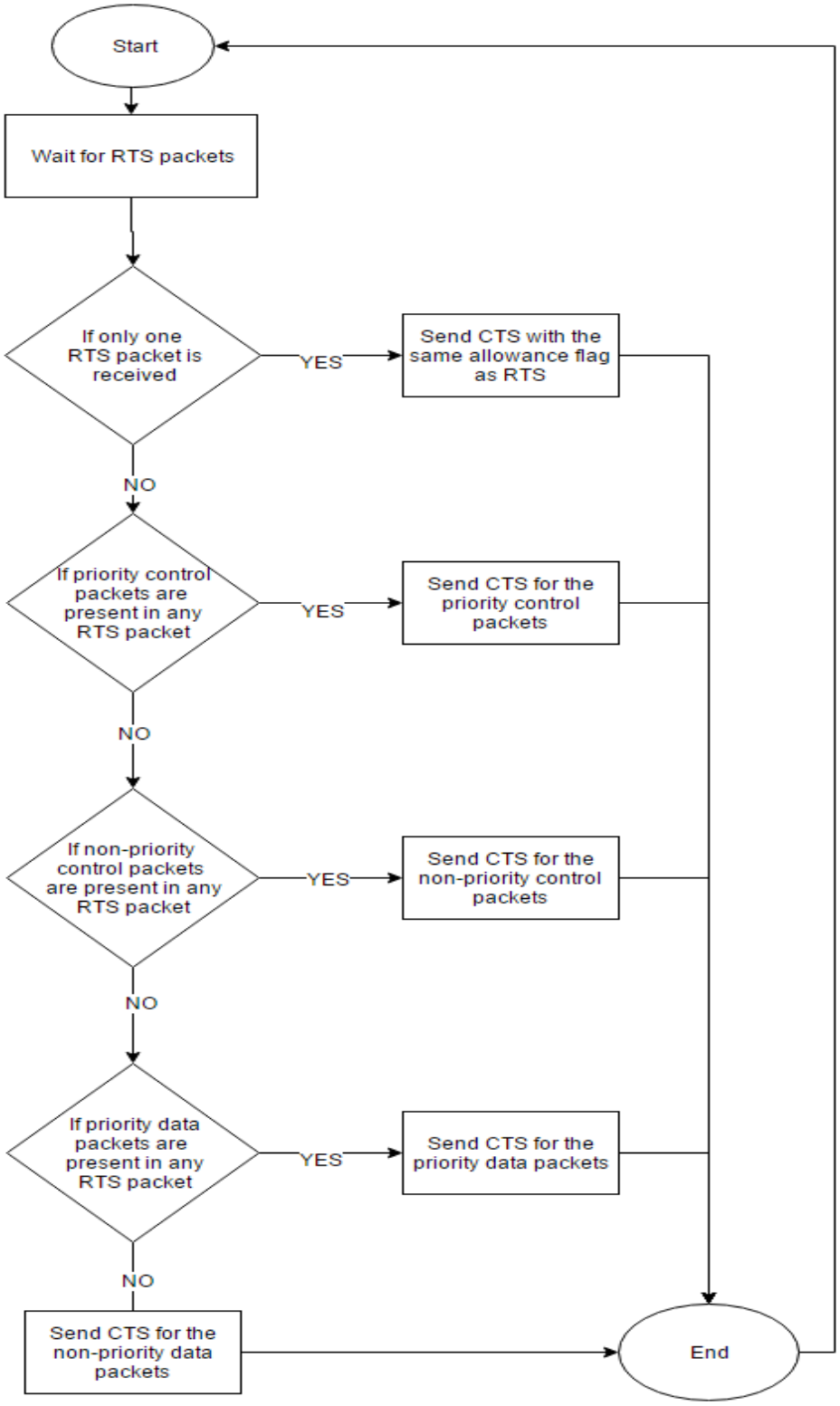

Figure 3.12: Flow chart for the RTS / CTS exchange

3.1.9 UAV reports and route updates

Once a route is established, communications can take place in the network. Data can flow from the source to the sink using the routes defined by the controller during the setup. In order to keep the network functional and for routes to be periodically updated, it is necessary that each UAV sends regular reports to the controller as shown in Figure 3.13. The algorithm for that is specified below:

Algorithm 5: UAV reporting algorithm

1. **If** SYNC_REQUIRED flag is true
2. Go to step 10
3. **Else If** SYNC_TIMEOUT is 0
4. Go to step 10
5. **Else**
6. Decrement SYNC_TIMEOUT by one second
7. Wait for 1 second
8. Go to step 1
9. **End If**
10. Prepare UPDATE packet to send to the controller
11. Mark packet as critical control packet with Allowance flag 1000.
12. Send packet over the network
13. Wait for an UPDATE_REPLY from the controller.
14. **If** REPLY_TIMEOUT is reached before UPDATE_REPLY is received
15. Send packet over alternate route.
16. **If** alternate route fails
17. Start AODV communication to reach the controller for further instructions
18. Mark packet as critical control packet
19. **End If**
20. **Else**
21. Set SYNC_REQUIRED flag to false
22. Update routing tables according to information
23. Set SYNC_TIMEOUT time to max
24. **End If**

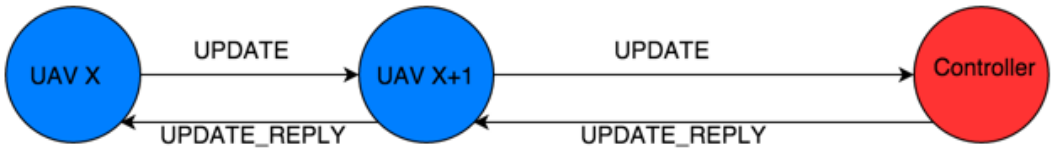

Figure 3.13: UAV UPDATE and UPDATE_REPLY

In order for the other components of the UAV (such as flight control or altimeter) to flag that a synchronization with the controller is required, we use a Boolean flag called SYNC_REQUIRED. The UAV keeps checking for this flag. If the flag is set to true, then the UAV generates an UPDATE packet that is sent to the controller. The SYNC_REQUIRED flag is used only when there is a major change in the network or UAV characteristics and the UAV urgently wants to communicate to the controller for further instructions. To synchronize the minor changes or errors that occur, SYNC_TIMEOUT is used to count down for synchronization when the countdown reaches 0. This SYNC_TIMEOUT is specified by the UPDATE_REPLY packet. The update packet (report) structure is specified in Section 3.1.6.

When the controller receives an UPDATE packet, it processes the information according to the Controller update algorithm as shown below.

Algorithm 6: Controller update algorithm

1. Wait for update packet
2. Parse update packet information
3. Update adjacency tables according to packet information
4. Recalculate average bandwidth and update table, ignore all UAVs with POWER_LOW flag turned on
5. Recompute all routes
6. **For each** route change
7. Generate a UPDATE_REPLY packet to send to that UAV
8. Insert Primary and Alternative routes for each priority level
9. Send packet to corresponding UAV
10. **End for each**

The controller recomputes all the routes after updating the adjacency table and resends any route changes to the UAVs. Each UAV is responsible for reporting any changes in its environment. This reply is sent via an UPDATE_REPLY packet.

3.1.10 UAV-UAV communication

UAV-UAV communications take place using all the components mentioned above. Once a packet is queued at a UAV, an RTS packet is generated with an Allowance flag according to the priority level of the packet. It waits for a CTS packet with a certain timeout. If the timeout is exceeded, then the sender assumes that the receiving UAV is lost and restarts the RTS/CTS communication using the alternate route. A UAV marked as lost triggers the SYNC_REQUIRED flag to be set forcing the UAV to notify the controller of this change as described in Algorithm 7.

Algorithm 7: UAV communication algorithm

1. Wait for packets in queue
2. Look for packets with highest priority
3. Lookup next UAV for that priority level
4. Prepare and send RTS packet
5. **If** CTS is received within timeout
6. Transfer all packets in that priority level according to Allowance flag on CTS
7. **Else**
8. Set Flag SYNC_REQUIRED to true
9. Mark destination UAV as unreachable
10. Lookup table for alternate route for that priority level
11. Go to step 7
12. **End If**

3.1.11 Loss of connection

Since UAVs are susceptible to be damaged from the environment, there is a risk for UAVs to go down without warning. In this case, the network must be able to adapt to the situation and reroute the traffic in order to seamlessly sustain communications. In order to accomplish this, the mechanism mentioned in Section 3.1.10 is extended by using the SYNC_REQUIRED flag to force the synchronization with the controller as soon as a connection is down. The scenario is illustrated in Figure 3.14. As we can see, if the connection to UAV $X + 2$ is lost, the SYNC_REQUIRED flag is set. This forces an UPDATE packet to be sent to the controller, notifying the controller of the change in the environment of UAV X. The controller responds with an UPDATE_REPLY packet that contains any change in routes.

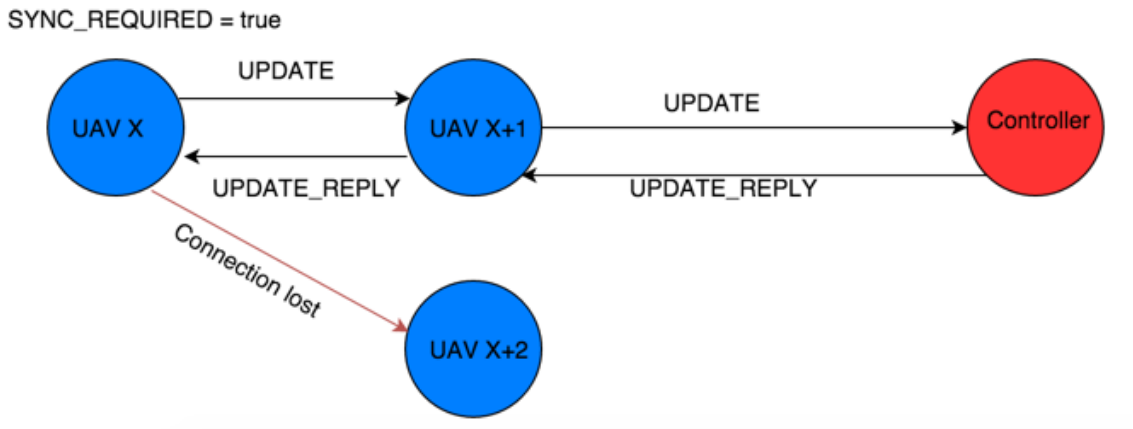

Figure 3.14: Loss of connection: an illustration

3.2 Algorithm for WiFi communications

In this subsection, we discuss the second subproblem which deals with the communication between the end user and the UAV using WiFi. This subsection is divided into 3 parts:

1. WiFi setup
2. WiFi update
3. User addition and removal

The first subsection explains the setup that is required in order to initialize the WiFi network. The second subsection explains how the controller is kept up to date with the WiFi information and statistics. The third subsection explains the procedure when a user enters or leaves a network.

3.2.1 WiFi setup

The setup for the WiFi communication continues from the DISCOVER packet, as described in Section 3.1.5. During the transmission of the HELLO messages, the flight control records the variation in location by monitoring the location and orientation of the UAV at regular intervals. This variation in location is set in the location error columns in the DISCOVER packet. Once a DISCOVER packet is sent to the controller, the WiFi module waits for the WIFI_UPDATE packet as shown in Figure 3.15. This packet contains all the necessary information required to set up the WiFi AP, such as the AP name, password and other details. This packet also specifies the maximum number of users that the UAV can support (USER_THRESHOLD).

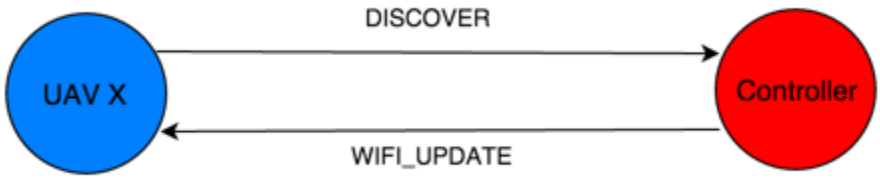

Figure 3.15: WiFi setup

Algorithm 8: WiFi setup algorithm - UAV

1. Every time a HELLO message is being broadcast, calculate the location using the GPS and the orientation using the gyroscope sensors
2. Record the location value in the table and calculate the average and error values.
3. Send this information along with the DISCOVER packet
4. Wait for WIFI_UPDATE packet
5. According to the information in the WIFI_UPDATE packet, set the signal strength and set up the WiFi network

Algorithm 9: WiFi setup algorithm - Controller

1. Wait for DISCOVER packet from all UAVs within a timeout
2. Record the location and error information for each UAV in a table
3. Using the recorded values, compute the signal strength to be used by the WiFi antenna on each UAV so there is minimal overlap and maximum coverage
4. Generate the WIFI_UPDATE packet for each UAV and transmit this packet over the network to the UAV

3.2.2 WiFi update

WiFi update takes place when the UAV update takes place. The UPDATE packet contains the user information that is used to update the controller on the total number of users and the traffic generated by the users.

3.2.3 User addition/removal

Users can enter or leave the network at will. The AP authenticates the user with the credentials supplied to it by the controller. If a user is authenticated, this user is added to the users table and the SYNC_REQUIRED flag is set to 1 in order to update the controller about this information.

A similar process takes place when the user leaves a certain UAV. The UAV updates the users table and sends this information to the controller by setting the SYNC_REQUIRED flag to true.

3.3 Low power and power management

Since UAVs are dependent on their batteries, it is critical to manage routes properly when UAVs are out of power. In this scenario, the controller has to be notified so arrangements can be made in order to seamlessly hand over all users over to neighboring APs.

Moreover, the UAV-UAV routes that are dependent on this UAV are shifted to their neighbors to make sure that the communication is uninterrupted.

When a UAV is low on power, it sends a LOW_POWER message to the controller. Once the controller receives this message, it updates the adjacency table marking the UAV low on power using the LOW_POWER flag. Once this flag has been set, the controller avoids using this UAV on all further calculations for routes. Within the next update cycle, all UAVs dependent on the low power UAV are updated to use alternative routes.

Once a UAV reaches the critical power level, the UAV sends another POWER_CRITICAL message to the controller that issues a TRANSFER_USER command for all users associated with the UAV. The controller also sends the WIFI_UPDATE command to neighboring UAVs to request them to increase their power levels in order to increase their coverage. The source UAV linearly dims its WiFi signal strength forcing all users to reassociate with neighboring APs. Figure 3.16 illustrates an example for handovers due to lower power level. Once UAV *X* detects the low power level, it sends a POWER_CRITICAL message to the controller which replies with a TRANSFER_USER message to UAV *X*. The TRANSFER_USER message contains a list of users with the UAV they need to be transferred to. The users that are currently connected to UAV *X* will then be handed over to another UAV selected by the controller. Once all users have been handed over to neighboring UAVs, the critical UAV returns to the base station.

Algorithm 10: Low power algorithm - UAV

1. Wait for DISCOVER packet from all UAVs within a timeout.
2. **If** UAV battery level is below 7%
3. Prepare a POWER_CRITICAL packet and send it to the controller with UAV_id
4. **Else If** UAV battery level is below 15%
5. Prepare a POWER_LOW packet and send it to the controller with UAV_id
6. **End If**

Algorithm 11: Low power algorithm - Controller

1. **If** POWER_LOW packet is received
2. Mark POWER_LOW flag against the UAV
3. **Else If** POWER_CRITICAL packet is received
4. **For each** user connected to POWER_CRITICAL UAV
5. Issue TRANSFER_USER command to nearest neighbor
6. **End for each**
7. Once all users have been transferred, the UAV returns to the base for a recharge
8. **End If**

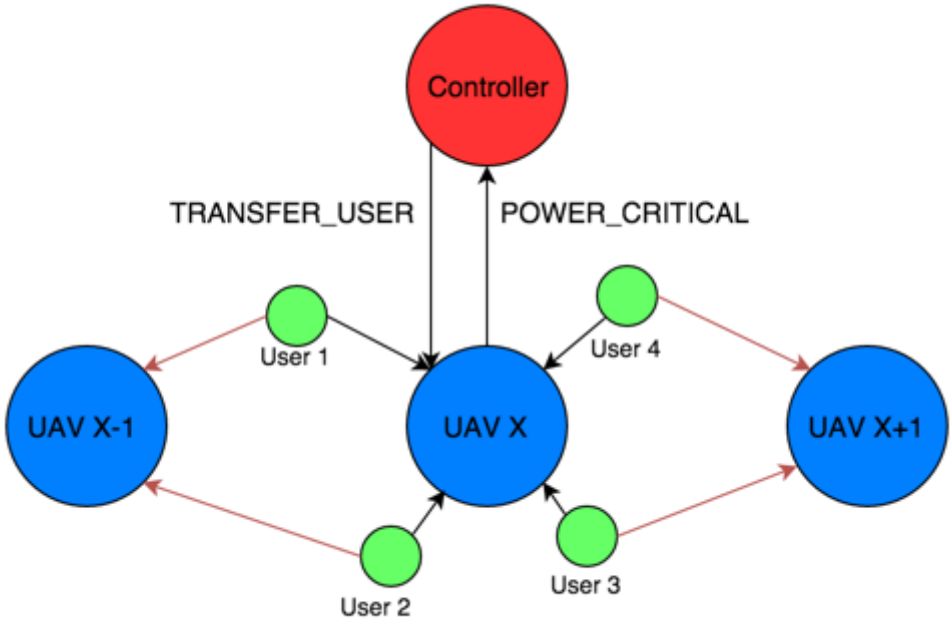

Figure 3.16: Handovers due to critical power level

As UAV *X* runs out of power it signals the controller using a POWER_CRITICAL packet. The controller responds with a TRANSFER_USER packet that forces reassociation of users 1 and 2 with neighboring UAV, UAV *X-1* and users 3 and 4 with UAV *X+1*.

3.4 User transfer due to overcrowding or traffic spikes

If the number of users connected to a certain UAV is too high or the traffic exceeds the threshold, then the controller issues a TRANSFER_USER command to transfer users to neighboring UAVs in order to balance traffic. If the number of users or the traffic exceeds a threshold, then the UAV sets the SYNC_REQUIRED flag in the UAV to true which forces an update with the controller. The controller replies with sufficient TRANSFER_USER commands to transfer users to neighboring UAVs and control the traffic flow in the network. This is illustrated in Figure 3.17. For example, in figure 3.17, node X original has four connected users, which exceeds the pre-configured number that was set at two. The UPDATE packets notify this to the controller and the controller responds with a TRANSFER_USER packet that forces users 1 and 4 to reassociate with UAVs X-1 and X+1 respectively. This reorganization helps maintain the user count and traffic through the network.

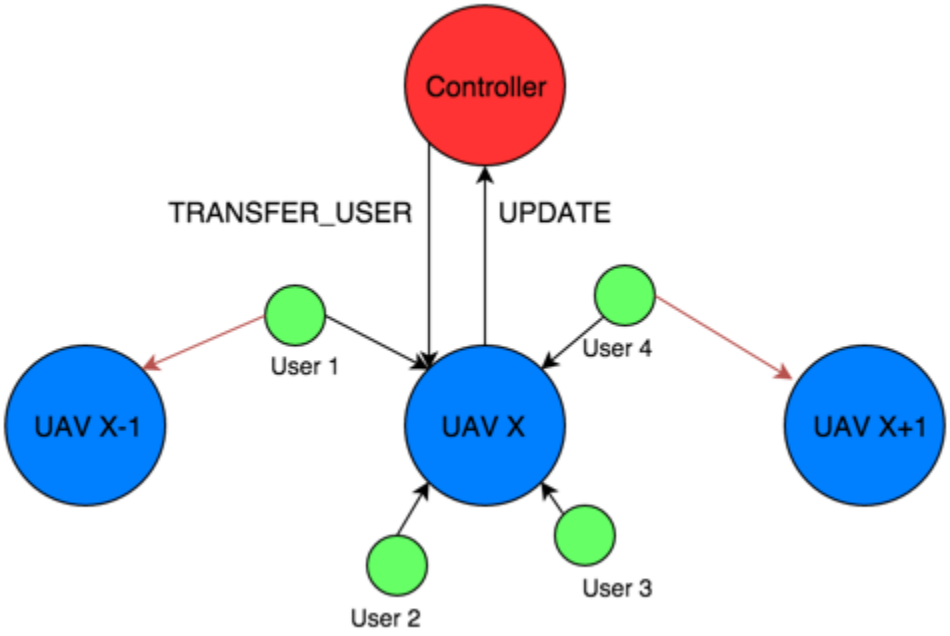

Figure 3.17: User transfer due to traffic spike

Chapter 4

Performance Analysis

In this chapter, we simulate and validate the approach against popular traditional wireless ad hoc protocols, such as OLSR and AODV, and against traditional WiFi connections.

4.1 Experimental setup and simulation

The simulation was performed in a Linux environment using NS-3 version 3.24 [NS-3.24]. The operating system used was an Ubuntu 15.04 system running on an Intel i7 processor with 16 GBs of RAM. Large amount of memory was required in order to perform simulations with a large number of UAVs.

The controller was implemented as a class called the “UAVController”. The frequency used for the UAV-UAV wireless link was 1 GHz and a bandwidth of 100 MHz. Each UAV was positioned before the simulation began and UAVs were made to form connections as soon as the simulation began. The wireless channels follow properties defined by the NS-3 framework, which are listed in Table 4.1.

Table 4.1: Network Characteristics

Parameter	Type	Value(s)
No of UAVs	Int	1, 2, 5, 10, 20, 100, 500, 1000
Frequency of WiFi communication	GHz	2.4, 5
Frequency of UAV-UAV communication	MHz	1000
Bandwidth of UAV-UAV communication	MHz	100 (950 MHz - 1050 MHz)
Full/Half Duplex	-	Full Duplex
Radio Technology	-	OFDM
Modulation Method	-	Adaptive Modulation
Supported Modulation schemes	-	BPSK, QPSK, 16 - QAM, 64 - QAM, 256 – QAM
No of Radio modules in each UAV	Int	2
Link latency	Milliseconds	2
Weight of priority control packet	-	4
Weight of non-priority control packet	-	3
Weight of priority data packet	-	2
Weight of non-priority data packet	-	1
Max communication range for UAV-UAV radio	Meters	50
Max communication range for WiFi radio	Meters	30
Threshold for 256-QAM modulation scheme	Meters	5
Threshold for 64-QAM modulation scheme	Meters	10
Threshold for 16-QAM modulation scheme	Meters	20
Threshold for 8-QAM modulation scheme	Meters	30
Threshold for QPSK modulation scheme	Meters	40

We simulated the network for a variety of scenarios with different number of UAVs with varying degrees of connectivity between the UAVs. The degree of connectivity of a UAV is defined as the number of neighboring UAVs that are within the communication range. Standard AODV and OLSR were chosen for comparison with the proposed approach, because they represent reactive and proactive protocols, respectively. AODV performs better when there are a lot of changes in the network, whereas OLSR performs better when there are very few changes in the network. By comparing the performance against AODV and OLSR, we evaluate how the proposed approach performs in each scenario.

To visualize the network during simulations, we have used a tool called NetAnim [NetAnim]. NetAnim is a Qt based visualizer tool and is part of the "ns3-allinone" package. This thesis uses NetAnim version 3.106.

Table 4.2 describes the network discovery parameters used in the proposed approach. In this thesis, we will generate 30 HELLO messages with intervals of one second between each HELLO message. If there is a collision of HELLO packets, the proposed approach backoff for a random time (any value from 5 ms to 500 ms). Since packets are transmitted at regular intervals if there are no collisions during the transmission of the first HELLO message, it is not likely that collisions will happen during subsequent transmissions.

Table 4.2: Proposed approach - discovery and setup

Parameter	Type	Value(s)
INIT TIMEOUT	Seconds	90
No of HELLO Messages	Int	30
HELLO INTERVAL	Milliseconds	1000
HELLO COLLISION INTERVAL	Milliseconds	Rand (5,500)
NETWORK REDISCOVERY	Seconds	30

Tables 4.3 and 4.4 list all the parameters for MAC and WiFi, respectively. For simulations, we will flag a UAV as low on power once the battery percentage drops below 15% and critical on power once the battery percentage drops below 7%. We also set the timeout value for CTS packets at 4 seconds and SYNC_TIMEOUT is set to 30 seconds. If an UPDATE_REPLY is not received before the SYNC_TIMEOUT expires, the UAV will try to communicate via the alternate route. If the alternate route fails, the UAV uses AODV in order to establish connectivity with the network. Each UAV will support a maximum of 64 users. This is half the limit of common WiFi APs in practice. Once the number of users crosses 64, the UAV would try to handover some of these users to neighboring UAVs to distribute users across the network in order to maintain traffic distribution.

Table 4.3: MAC parameters

Parameter	Type	Value(s)
CTS_TIMEOUT	Seconds	4
SYNC_TIMEOUT	Seconds	30
POWER_LOW	Percentage	15
POWER_CRITICAL	Percentage	7
UPDATE_REPLY_TIMEOUT	Seconds	10
RTS_WINDOW	Seconds	2
ALLOWANCE_FLAG_SIZE	Bits	4
TRAFFIC_THRESHOLD	Percentage	90
TRAFFIC_THRESHOLD_TIME	Seconds	5

Table 4.4: WiFi parameters

Parameter	Type	Value(s)
USER_THRESHOLD	Int	64
USER_TRAFFIC_FUNCTION	Mbit/s	GAMMA (5,1) k = 5 o = 1 Mean = 5 * 1 = 5 Mbit/s Variance = 5 * 1 * 1 = 5
USER_TRAFFIC_REFRESH	Seconds	120
Full/Half Duplex	-	Half Duplex
Modulation Schemes	-	BPSK, QPSK, 16 - QAM, 64 - QAM
Frequencies	GHz	2.4, 5

4.2 UAV-UAV Simulation

This section shows results of UAV-UAV protocol simulations under various degrees of connectivity and different priority packets. This section begins with the simulation of a single source, linear topology network to compare base performance against AODV and OLSR. We will gradually increase the complexity of the network by increasing the number of UAVs in a simulated network.

4.2.1 Single Source, Linear topology UAV-UAV simulation

We will start with a simple simulation to validate the proposed idea. The network was constructed with a linear topology as shown in Figure 4.1. A linear topology is the theoretical worst case scenario, because the proposed approach relies on the availability of different routes in order to distribute packets across the network to utilize all network resources. When there is a network with a linear topology (average degree of connectivity = 2), the control packets become redundant because none of the network characteristics have changed and all the routes point to the same UAV for next hop. All the additional fields in the RTS/CTS packets, UPDATE and UPDATE_REPLY packets become redundant, which is a waste of bandwidth.

4.2.1.1 Single priority test

In this setup, there is a single source that produces packets and all packets have the same priority level. None of the intermediary nodes generate traffic, they are present only to relay the packets generated by the source to the sink. The source is set to produce packets at the rate of 100 Mbps with 5 intermediate hops. We set the priority level to the lowest to compare the proposed approach against AODV and OLSR. Figure 4.1 shows the topology used for the simulation and Figure 4.2 is a graph that compared the results obtained from the proposed approach versus AODV and OLSR.

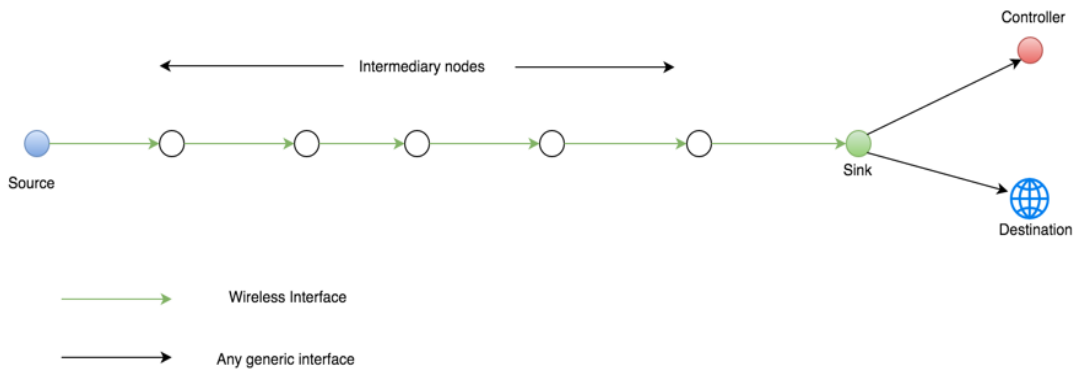

Figure 4.1: Linear topology for simulation

This simulation does not take into account the time required to set up the network. We start the timer as soon as the first packet has reached the destination. We will compare the setup time in another experiment.

Each wireless link is capped at 16 Mbps maximum throughput. The UAVs are placed sufficiently close to each other so that the modulation scheme is no longer a factor. Throughput was measured in the sink in intervals of one second. In order to average out any errors and randomness, the experiment was run 10 times and the graph shows the average throughput over the 10 tries. This experiment was rerun using 10, 20 and 50 UAVs and the results were similar to the result obtained in Figure 4.2.

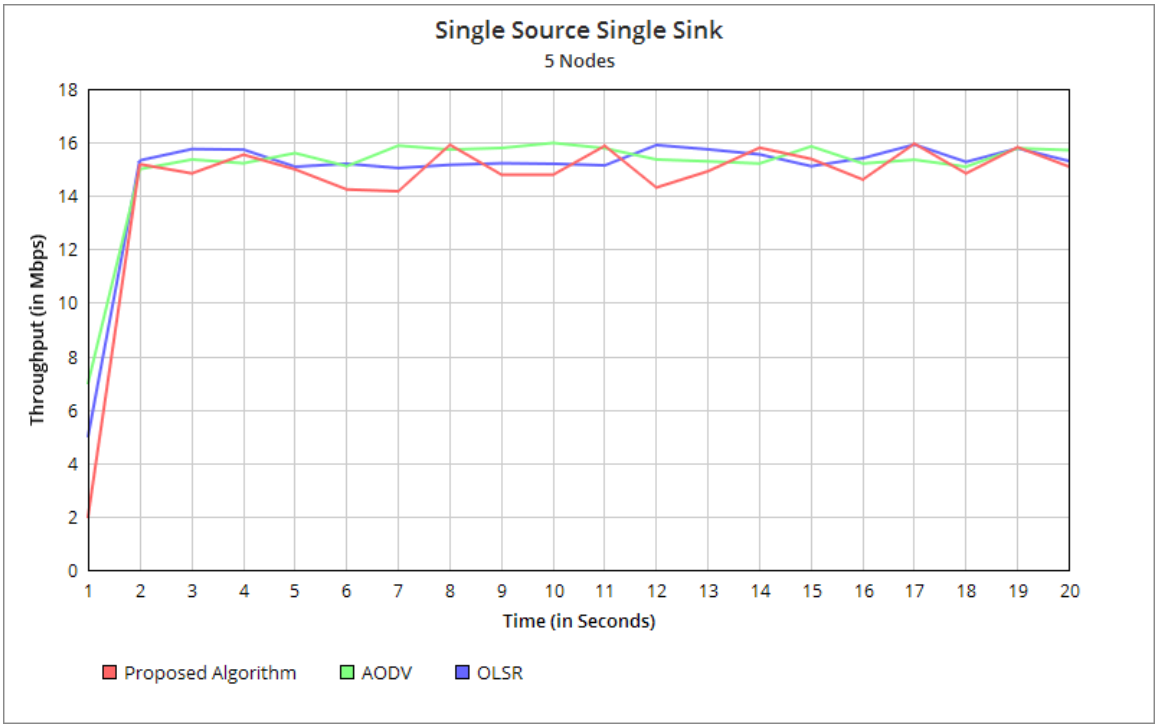

Figure 4.2: Results for a single source single sink linear topology

Figure 4.2 shows that the throughput performance of the proposed approach is comparable to AODV and OLSR. The average throughput for AODV and OLSR were 15.065 Mbps and 14.892 respectively, while the average throughput of the proposed approach was 14.452 Mbps. This is in line with the expected results for the proposed approach, i.e., the proposed approach has a slightly lower throughput in this case due to the overhead of transmitting update packets.

4.2.1.2 Setup time

For this experiment, the network setup time was calculated from the time the UAVs were deployed to the time when the first packet was received at the sink. Setup time was drastically higher for the proposed approach, since there is a 30 seconds delay before the

DISCOVER packet was sent to the controller. This delay can be changed depending on the confidence of the environment. The 30 second delay was chosen to average out any environmental factors affecting the measurement of signal strength. Changing this delay would directly affect the setup time of the algorithm. A 5 second setup would mean that a UAV only transmits 5 packets which may be less accurate than a 30 second setup.

The average setup time for the proposed approach for 5 UAVs in a linear topology was 39.3 seconds and the average setup times for AODV and OLSR were 2.2 and 4.1 seconds. As the number of UAVs increased, the setup time for AODV and OLSR also increased but the increase in setup time for the proposed approach was negligible as shown in Table 4.5. The setup time is an average over 10 runs of the simulations. As the network grows in size, the time taken for OLSR increases rapidly as OLSR floods the network with packets in order to identify the neighbor set and the 2 hop neighbor set.

Table 4.5 Average setup time

No of UAVs	AODV (Sec)	OLSR (Sec)	Proposed approach (Sec)
5	2.2	4.1	39.3
10	2.3	4.6	39.3
20	2.7	5.7	39.4
50	3.6	7.8	39.5

4.2.1.3 Multiple priority levels

We conducted the same experiment as conducted above with packets of different priority levels to test the packet drop among different priority levels (single source network with linear topology). The program was designed to accommodate 4 UDP packet generators in this scenario producing traffic at 25 Mbps each. These generators replaced the single UDP generator at the source. The experiment was carried out with 5, 10, 20, 50 UAVs and each scenario was run 10 times to average out any errors or randomness.

For AODV and OLSR, the results were almost identical to each other as shown in Figure 4.4. For AODV and OLSR, the percentage of packets dropped was almost equally distributed amongst the priority levels. The variation is due to the Random Early Detection (RED) queueing mechanism [Floyd93] which randomly drops packets as the queue is getting full. For the proposed approach, the percentage of priority packets dropped as a percentage of total packets is significantly lower than AODV and OLSR, which is demonstrated in Figure 4.3. Instead of dropping higher priority packets, the algorithm dropped the lowest priority packet more often. The number of UAVs did not make any difference to the outcome of the experiment.

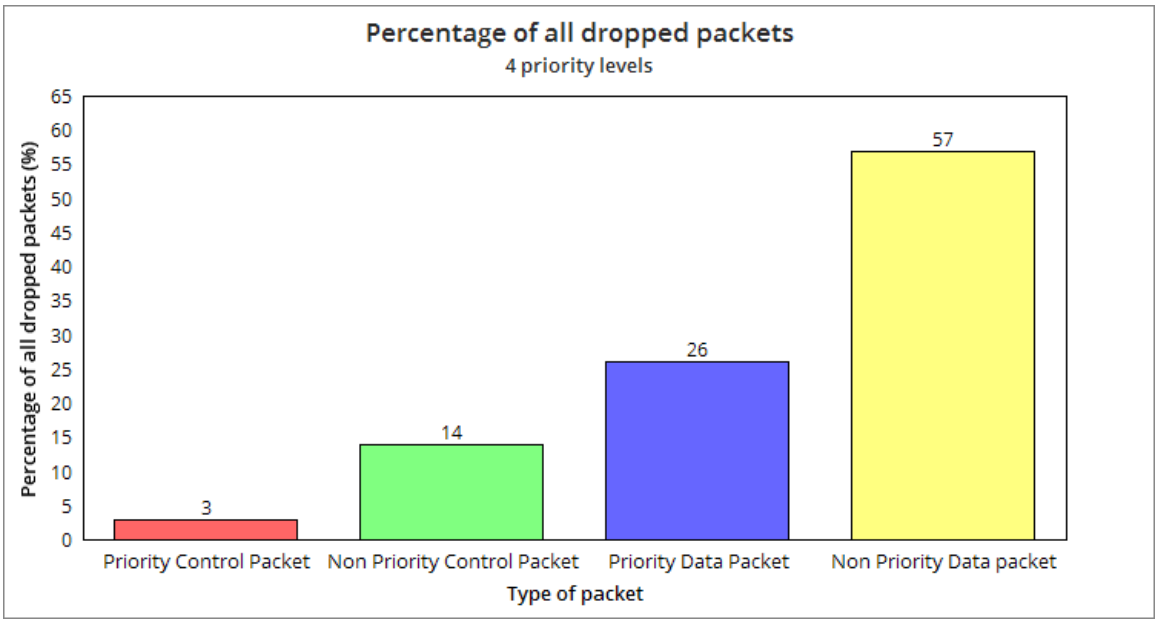

Figure 4.3: Dropped packets for each priority level - proposed approach

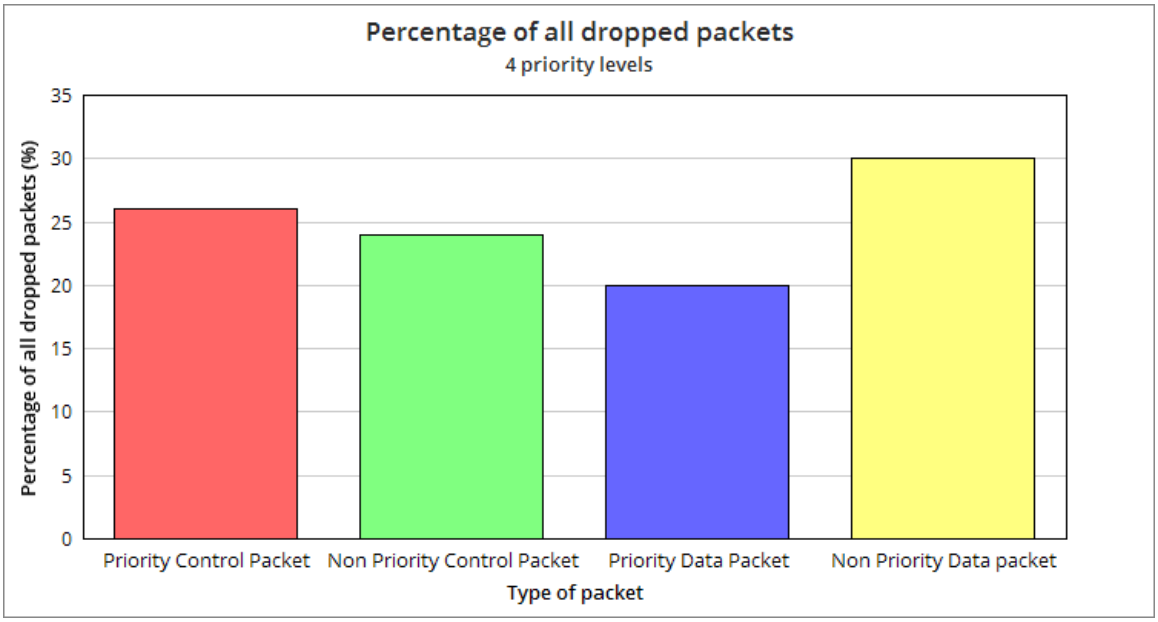

Figure 4.4: Dropped packets for each priority level - AODV and OLSR

As shown in the Figure 4.3, a packet marked as a priority control packet is 20 times less likely to be dropped than a packet marked as a non-priority data packet when the proposed approach is used. This is due to priority differentiation built into the network.

In conclusion, for the single source linear topology, the proposed approach has a comparable throughput, a longer set up time but drops less important traffic. This behavior was expected since as mentioned, there is only a single path and the proposed approach has been designed to perform well when multiple paths exist between the source and the destination. The next section will discuss this.

4.2.2 Dual source, 4-tier network

Now let us consider a 4-tier network with two-source UAVs that produce non-priority data packets at the rate of 40 Mbps, control packets at the rate of 10 Mbps and non-priority control packets at the rate of 10 Mbps. We will compare the performance of the proposed approach against the performance of AODV and OLSR. An illustration is shown in Figure 4.5. A total of 35 UAVs (including the controller) were generated and placed using a “GridAllocator” from the standard NS-3 Allocator library.

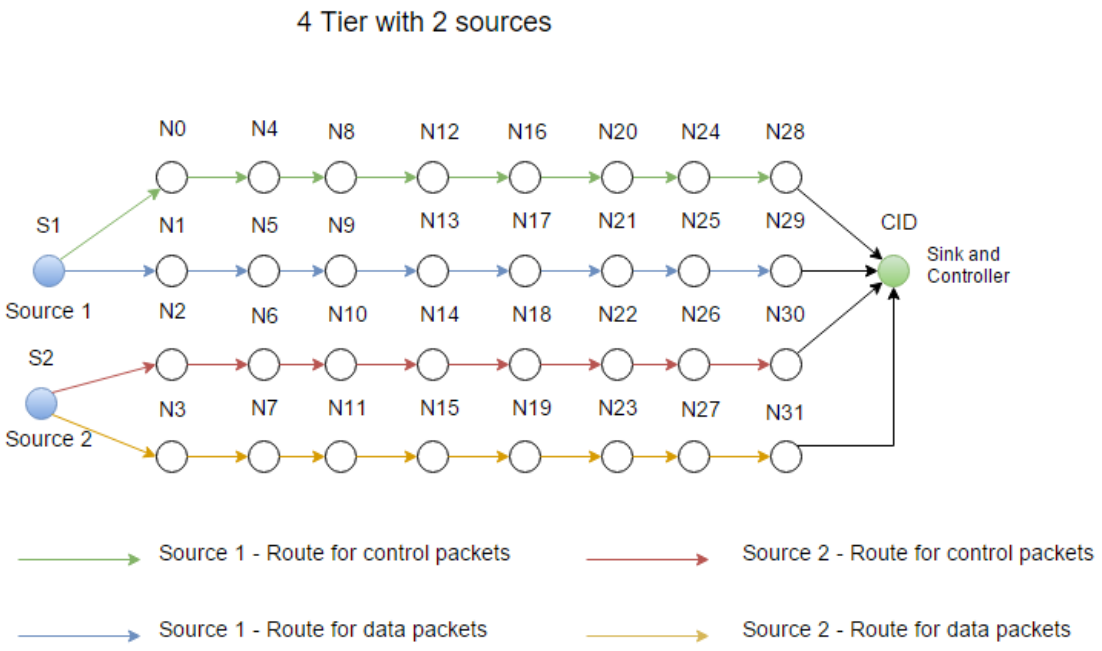

Figure 4.5: Experiment with 4 tier network consisting of 2 sources and 1 sink

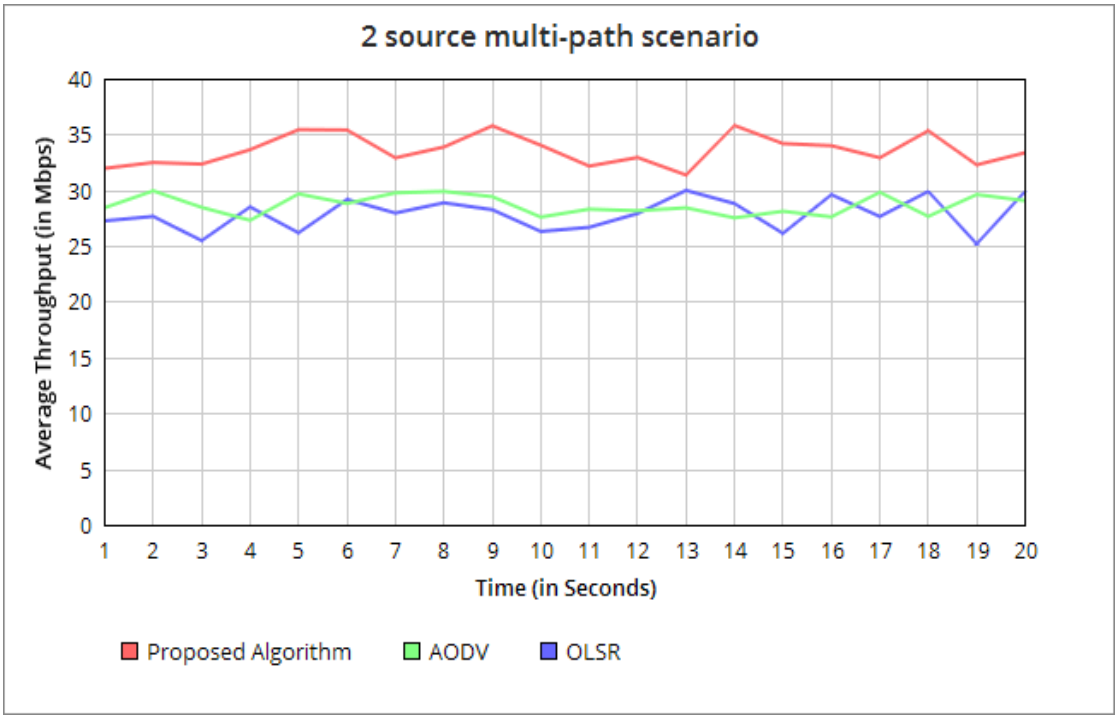

Figure 4.6: Combined throughput of sources 1 and 2 (in Mbps)

Figure 4.6 displays the average throughput per second at the sink for both sources combined. On average, the proposed approach provides approximately 15% greater throughput over AODV and OLSR. This is due to the fact that different packets have different routes. AODV and OLSR, on the other hand, have a fixed route for sending all types of packets.

The throughput of priority packets however is interesting. We define higher priority packets as any packet with a priority level greater than non-priority data packets. The proposed approach transmitted ~95% of all higher priority packets when compared to at most 55% in AODV and OLSR, as shown in Figure 4.7.

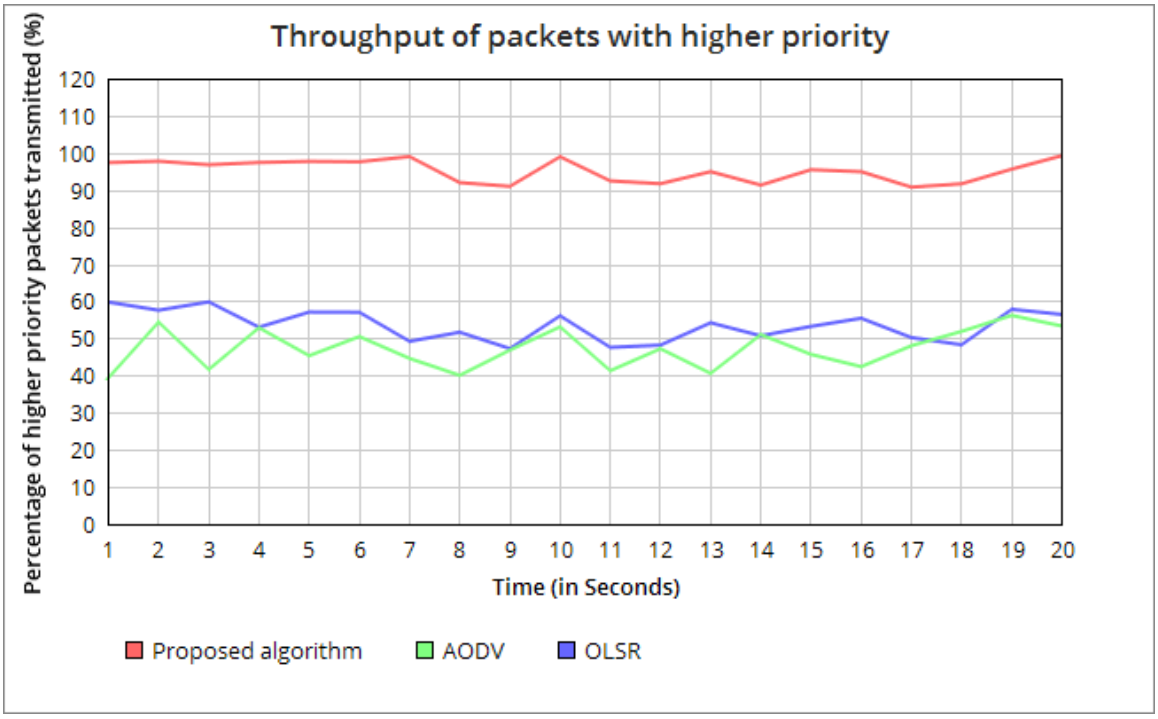

Figure 4.7: Priority packets successfully transmitted

Figure 4.8 shows the average delay of priority packets. Evident fluctuations in delay are due to the Random Early Detection (RED) queuing mechanism in a single queue for all packets when we use AODV and OLSR. However, for the proposed approach, the delay is consistently low with minimal variations. This consistency is due to the refined MAC protocol that prioritizes transmission of priority packets before other packets. A priority control packet is 4 times more likely to be transmitted than a non-priority data packet. For priority control packets, the routes are calculated using Dijkstra's algorithm which guarantees the shortest path to the controller.

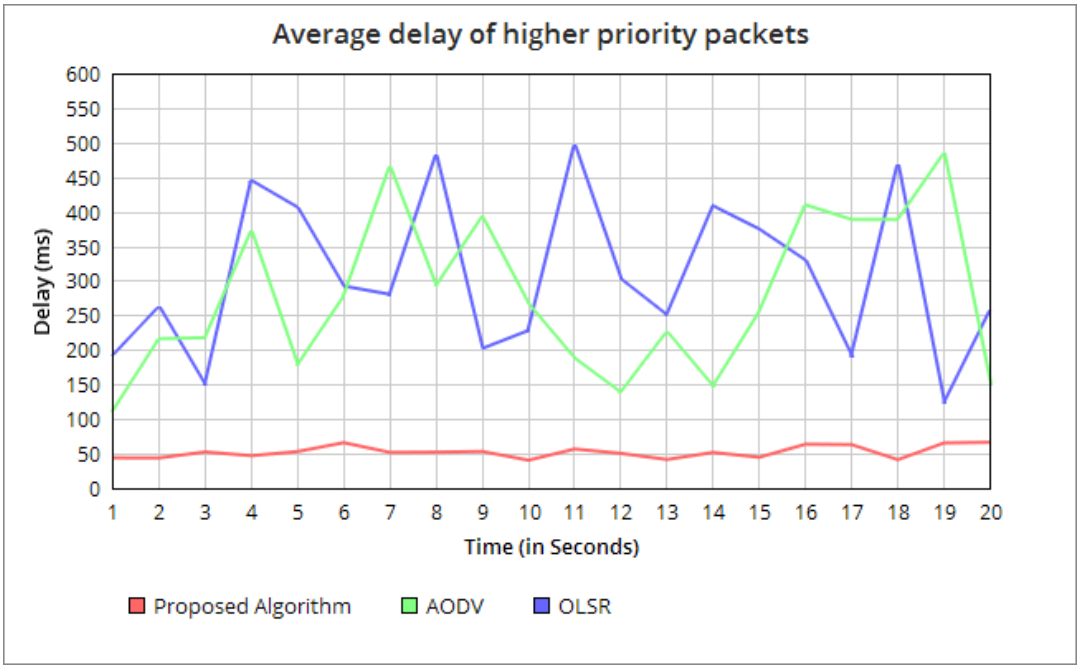

Figure 4.8: Average delay of higher priority packets

4.2.3 Multiple Source Simulations

The simulations conducted so far were either a single source or two sources. This section goes one step further by generating traffic from all UAVs. This is more realistic and much closer to the scenario we are envisioning where UAVs are acting as backbone

UAVs to relay traffic from users on the ground. UAVs act as APs providing wireless access to the Internet and users will connect to the APs. Users connected to each AP generates a non-uniform amount of traffic. Cumulatively, each UAV generates a non-uniform amount of traffic to be sent via the UAV backbone network.

4.2.3.1 Non-uniform traffic distribution with static UAVs

In this scenario, we will use a "RandomDiscPositionAllocator" class from the NS-3 library to position the UAVs centered around the controller in every which way randomly. Figures 4.9, 4.10, 4.11, 4.12 show the topologies generated for each set of UAVs. The bounds of the grid are 500 units around the controller. The controller can be in a random position within the area (0, 0) - (1000, 1000). Once the position of the controller is fixed the UAVs are generated around the controller with a 500 unit bound using the "RandomDiscPositionAllocator" built into NS-3. Simulating networks of 5 or 10 UAVs would lead to a highly disconnected distribution. Hence, in this simulation, we simulate 100, 200, 500 and 1000 UAVs.

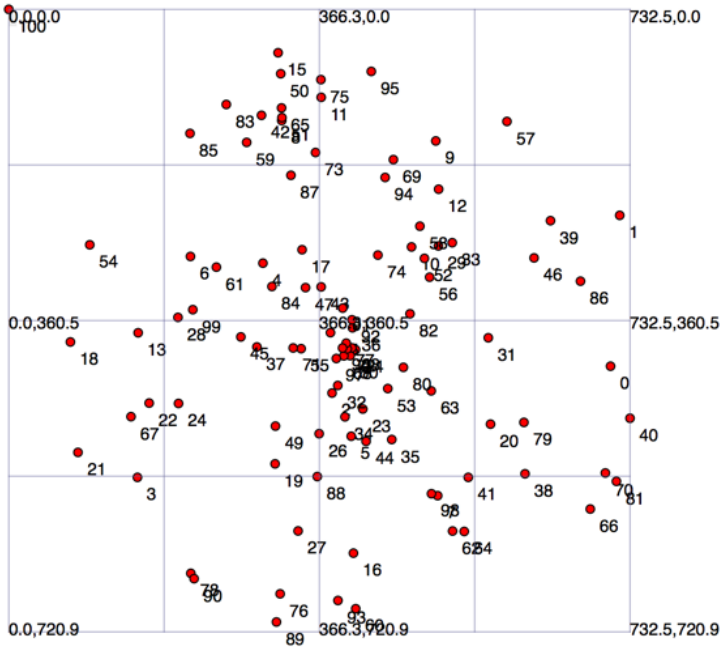

Figure 4.9: NetAnim for 100 UAVs

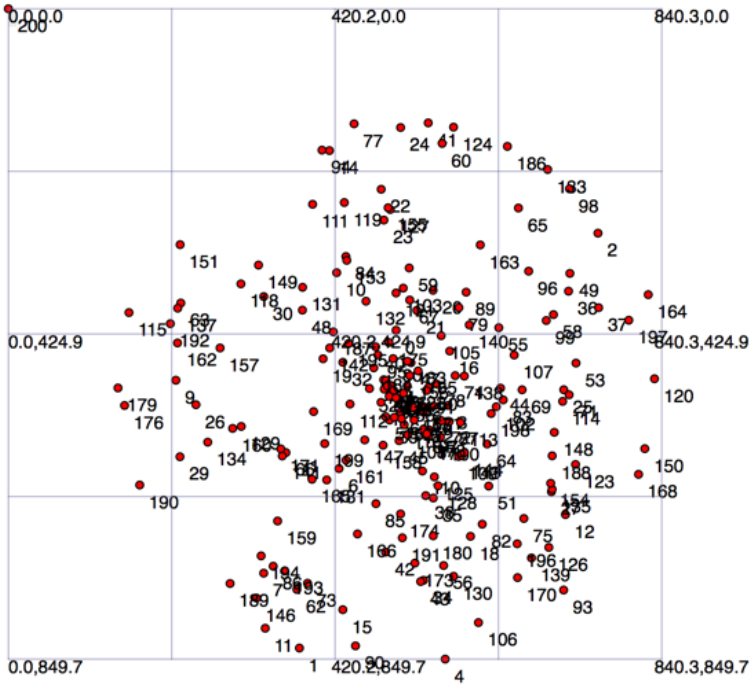

Figure 4.10: NetAnim for 200 UAVs

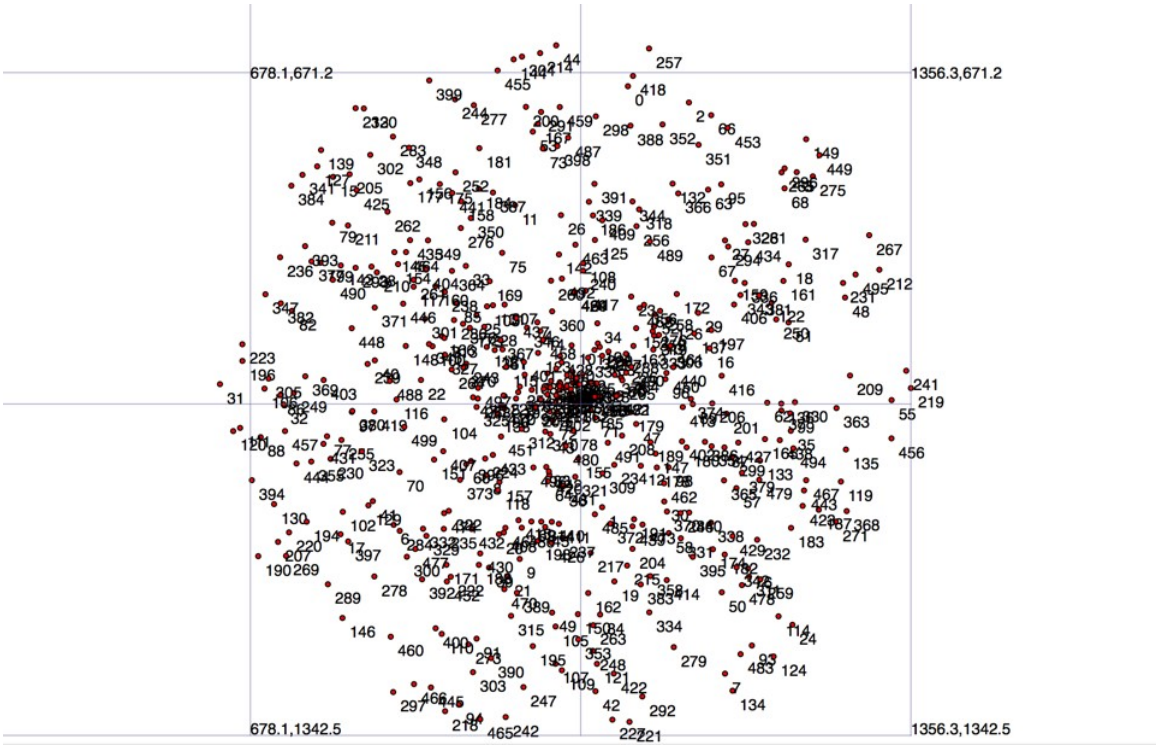

Figure 4.11: NetAnim for 500 UAVs

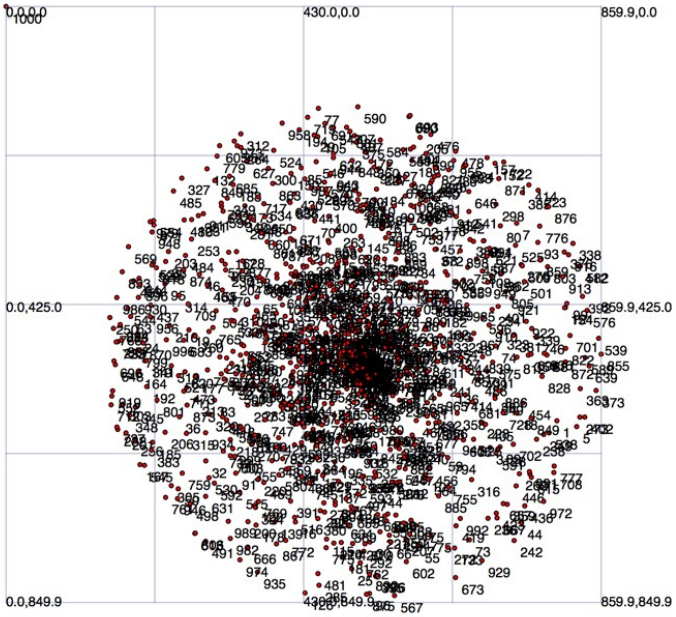

Figure 4.12: NetAnim for a 1000 UAVs

All UAVs are static with a fixed position throughout the course of the simulation. The simulation was stopped at 50 seconds. The traffic generated was using a normal distribution N (5, 5) Mbps for each priority level. This simulates the cumulative user generated traffic from a given UAV. This simulation does not consider traffic generated by every individual user but the traffic generated by a UAV as a whole. The generated random value was checked and confined within 0 Mbps and 10 Mbps. Since uneven traffic was generated, calculating the total throughput for all UAVs does not quantify the simulation. Instead, we will look at the percentage of packets transmitted successfully through the network also known as Packet Delivery Ratio (PDR). The experiment was repeated 10 times for each simulation to average out any distribution anomalies.

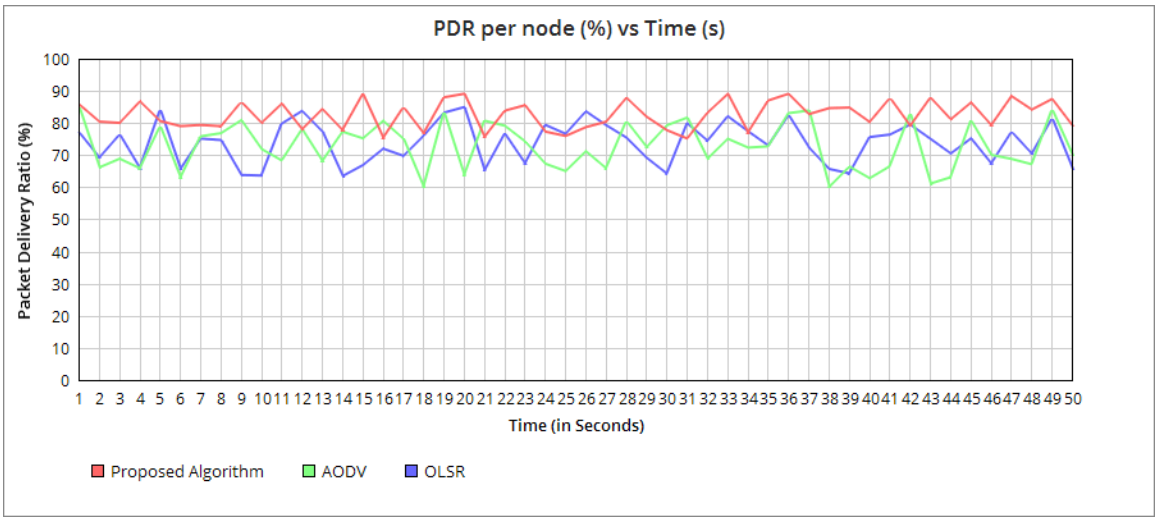

Figure 4.13: Average throughput per UAV (100 UAVs)

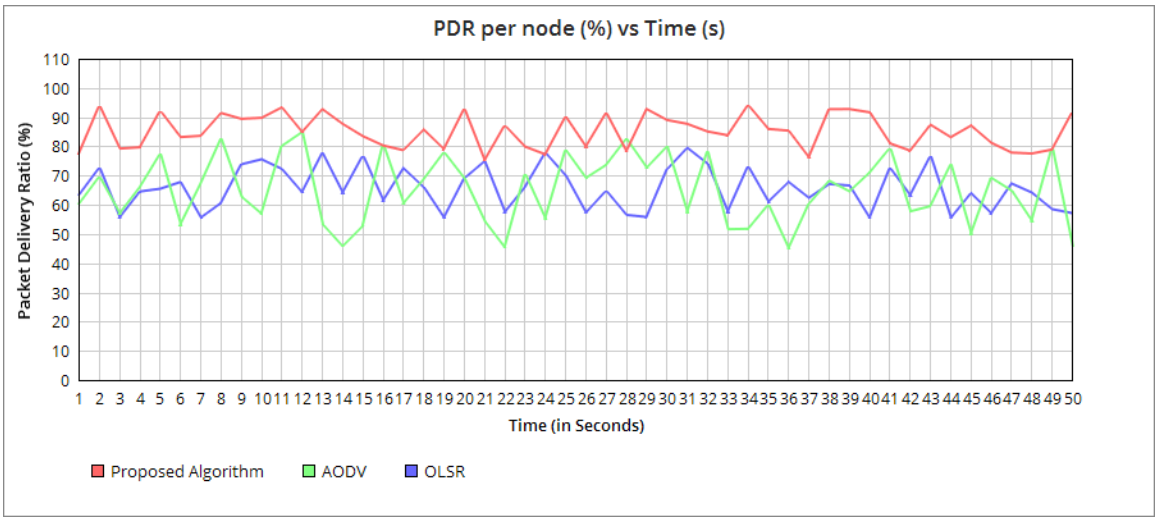

Figure 4.14: Average throughput per UAV (200 UAVs)

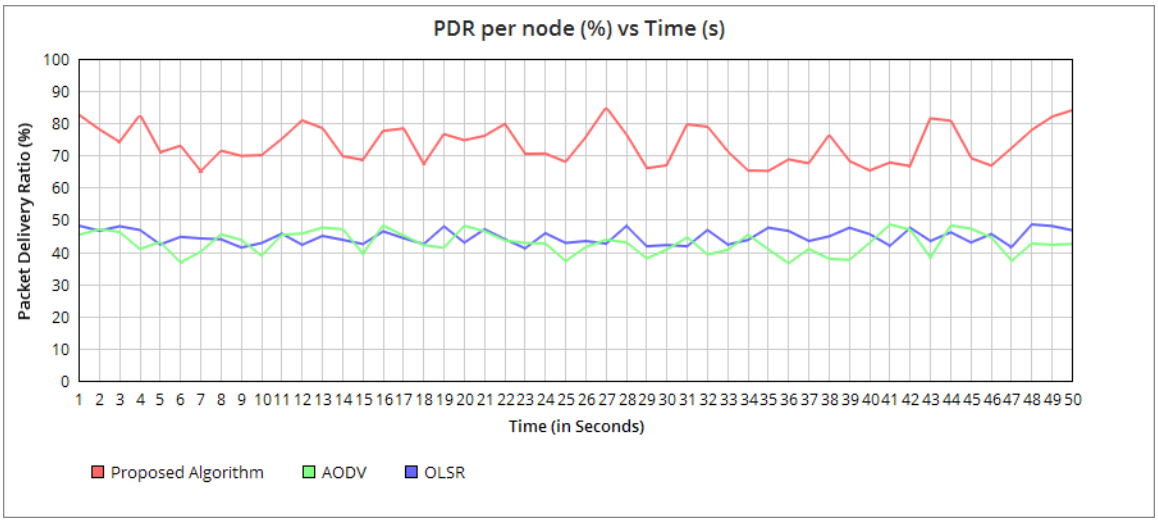

Figure 4.15: Average throughput per UAV (500 UAVs)

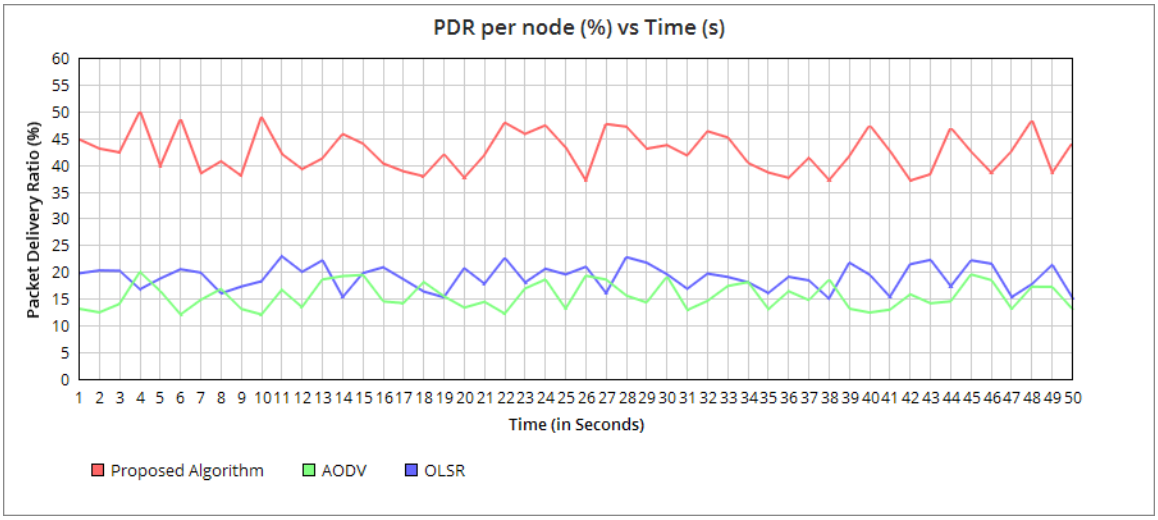

Figure 4.16: Average throughput per UAV (1000 UAVs)

From Figures 4.13, 4.14, 4.15, 4.16, we see that the proposed approach performs better in transmitting higher percentage of packets. The improvement is clearly seen in the case of 500 and 1000 UAVs. The average throughput is listed in Table 4.6 for each case.

Table 4.6 Average PDR of non-uniform traffic distribution

Number of UAVs	Average PDR (%)		
	Proposed approach	AODV	OLSR
100	83.4%	72.1%	71.5%
200	87.8%	60.4%	64.1%
500	74%	46.5%	48.4%
1000	44.3%	18.6%	20.3%

The proposed approach efficiently distributes the load around the topology of the network using all UAVs on the network, whereas in AODV and OLSR, nodes use the same UAV as next hop irrespective of the traffic flowing through that UAV. For example, during the 1000 UAV simulation, UAV 451 in Figure 4.11 dropped an average of over 80% of all packets received during the AODV simulations. However, in the simulation of the proposed approach, we see that the average usage of this UAV is little over 42%.

From the performance results depicted in Figure 4.13 to that in Figure 4.14, it is shown that the performance of the proposed approach has slightly increased. This is due to the higher degree of connectivity in the case of 200 UAVs. But as the size of the network grows, the algorithm is no longer able to maintain the efficiency due to the sheer volume as shown in Figures 4.15 and 4.16. But even in this case the algorithm performs significantly better than OLSR and AODV.

4.2.3.2 Non-uniform traffic distribution with UAV blackout

The goal of this simulation is to show that the proposed approach is able to recover in case a UAV fails during operation. This failure could be due to any unexpected condition such as a gust of wind, a lightning strike, so on. In order to simulate this scenario, a "BlackoutUAV" class was created. These blackout UAVs will stop responding after 10 seconds into the simulation. A number of UAVs were placed close to the blackout in order to force traffic through the blackout UAVs. The proposed approach recovered almost immediately after the CTS timed out. Once the CTS times out, the SYNC_REQUIRED flag is set and the UAV resumes communication using the alternate

route. The SYNC_REQUIRED flag forces an UPDATE packet to be fired which asynchronously updates the routing table replacing the main and alternate routes. AODV and OLSR responded differently to this situation. In standard AODV, only the affected UAVs (UAVs using the blackout UAV as the next hop) were forced to rediscover routes, whereas in OLSR, all UAVs in the neighborhood had to rediscover.

4.3 WiFi simulations

This section presents the solution for the second part of the problem mentioned in Section 3, UAV-User communications. This section simulates user related scenarios and verify our improvements to WiFi to enhance performance and work with our UAV-UAV backbone.

In reality, WiFi users tend to move with their devices. With devices getting more and more portable, roaming within a WiFi network becomes a greater concern. In order to verify WiFi handovers, we created a “WiFiUAVUser” class that simulates a WiFi user. This is a generic class and could represent a laptop, a mobile phone or even a fire truck. For simulation purposes, we simulated all users to “Walk” mode. “Walk” mode was set to a maximum speed of 2 m/s to simulate users walking or running. Users walk randomly in and around the simulation bounds and can disconnect and reconnect at any time. Since the following sections place more emphasis on APs rather than UAVs, we will call UAVs as APs for the following sections.

4.3.1 User handover due to user movement

To simulate user handover, 5 UAVs or APs in this case, were deployed to handle a total of 50 users spread around the network as shown in Figure 4.17. We will simulate one user walking across the network forcing a network handover.

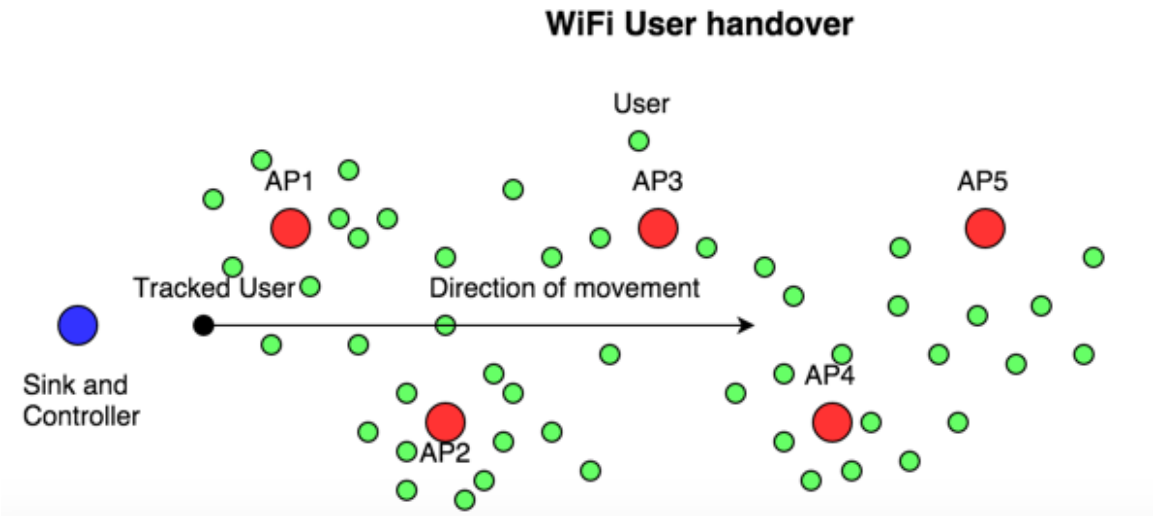

Figure 4.17: WiFi handover experiment topology

Figure 4.18 tracks the throughput of the tracked user from Figure 4.17 as a function of time. The throughput of the user is calculated after the first packet was received at the sink.

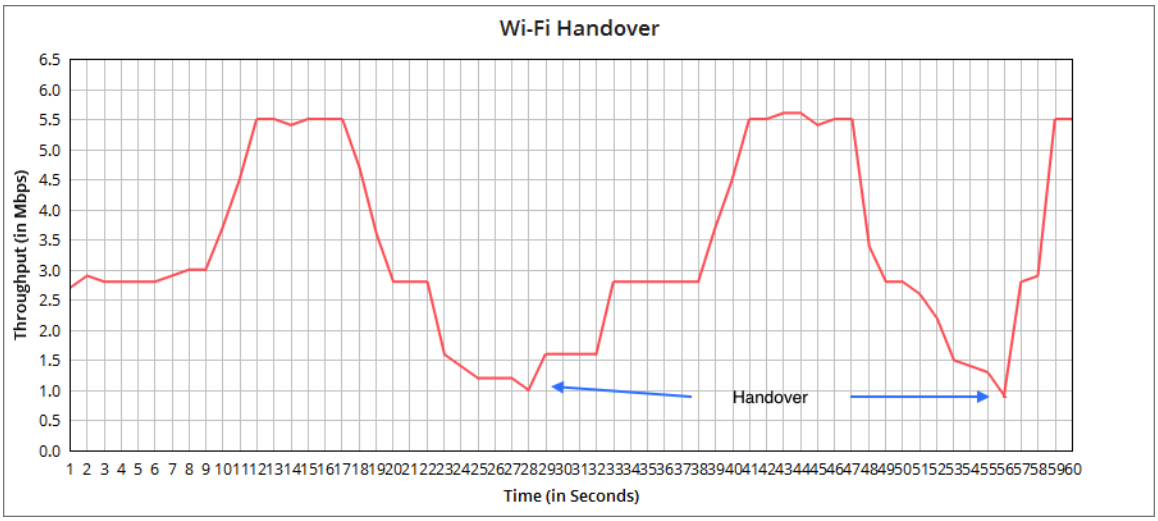

Figure 4.18: Throughput of monitored user

The user starts moving from the edge of an AP, then moves closer to the AP and eventually moving away from the AP again. This movement explains the “Bell” curve in the graph. Once the signal strength is too low a handover takes place which are marked with an arrow. The handover commands are issued as soon as the signal strength of a user begins to weaken. As described in Chapter 3, the UAVs send an UPDATE packet to the controller which replies with a TRANSFER_USER packet prompting the AP to issue a reassociate command to the user.

4.3.2 User handover due to overcrowding

Let us assume that this network was to be deployed in a carnival or big event. The user distribution in this case is highly uneven and few UAVs could be overcrowded whereas few could be underutilized. To prevent overcrowding and to even out the user distribution, the users at the edge of the coverage of a UAV are handed over to

neighboring UAVs. This helps distribute the network traffic from the WiFi perspective and UAV-UAV backbone perspective.

To make visualization easier, we reduced the USER_THRESHOLD from 64 users to 10 users temporarily so user handover can be easily simulated and user traffic can be monitored. In this simulation, we generated 3 APs. AP2 has 12 users connected to it, whereas neighboring APs 1 and 3 have only 3 users connected to each of them as shown in Figure 4.19. Each user generates 1 Mbps of UDP traffic. The throughput of UAV-UAV wireless links is limited to 10 Mbps.

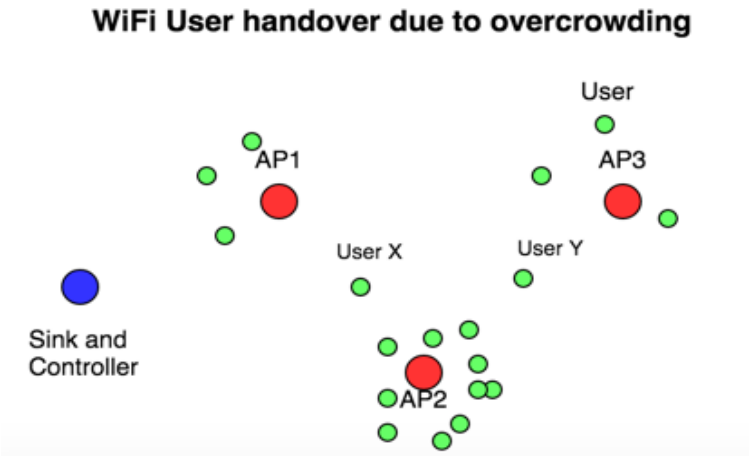

Figure 4.19: WiFi overcrowding topology

Users *X* and *Y* are associated with AP2 but are within range of APs 1 and 3, respectively. For this simulation, we will track the throughput of user *X* alone. During simulations, we notice that a UPDATE packet generated by AP2 and sent to the controller, the controller responded with a TRANSFER_USER command that contains a map from users *X* and *Y* to APs 1 and 3, respectively. Figure 4.20 shows the throughput of user *X*.

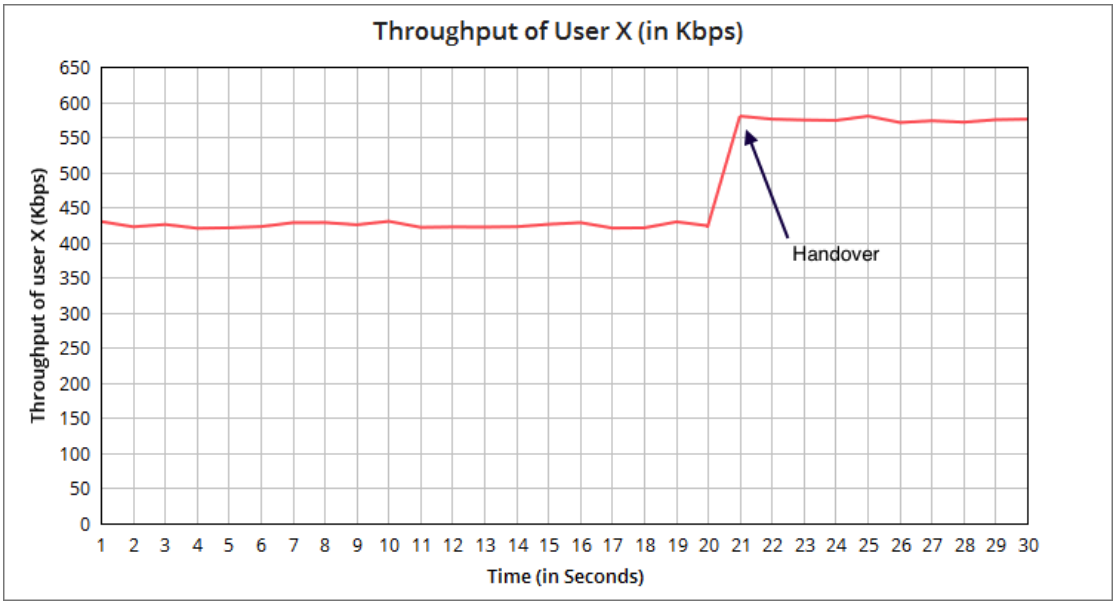

Figure 4.20: User handover due to overcrowding

Initially, the user throughput hovers around ~430 Kbps. This is due to a combination of two factors, poor signal strength and overcrowding. Even if the user were to receive a CTS for WiFi, the poor signal strength results in a bad modulation scheme that does not let the user transmit a lot of data in the duration the CTS lasted. Since the user has poor signal strength with AP1, there is nothing we can do to avoid the bad modulation scheme. However, handing over the user to AP1 leads to lesser crowding on AP2. User *X* now shares a larger portion of the bandwidth with AP1 resulting in a sudden jump in throughput as shown in Figure 4.20.

4.3.3 User handover due to low power

UAVs are battery operated. To simulate user handover due to low power, we generate 3 APs similar to the scenario above. Each AP has 3 users associated with it as shown in Figure 4.21.

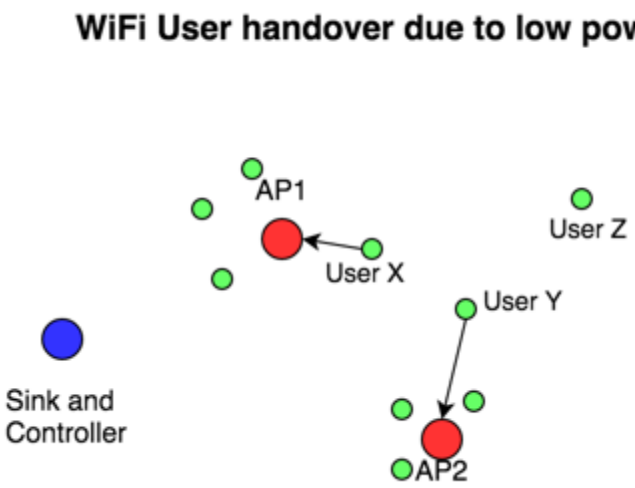

Figure 4.21: Low power UAV topology

When AP3 sends the POWER_LOW packet to the controller, the controller responds with the TRANSFER_USER packet that transfers users to the nearest access point. Users *X* and *Y* reassociate with AP 1 and 2, respectively. But, user *Z* is not within the communication range of AP1 or AP2. Unfortunately, in this case, user *Z* is lost, as shown in Figure 4.22. This limitation could be eliminated by intelligently rearranging UAVs in order to cover the lost area. This is however, not in the scope of this thesis.

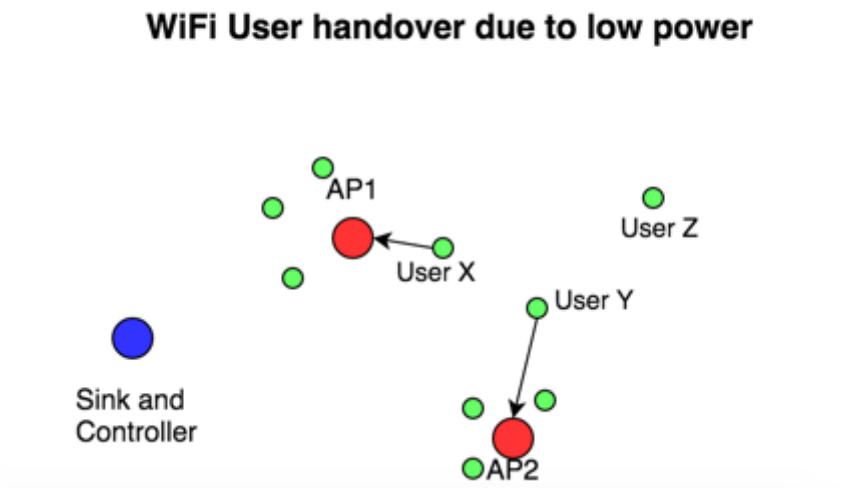

Figure 4.22: WiFi handover due to low power: User *Z* lost

4.3.4 Unexpected UAV failure

In reality, there are many factors that could cause unexpected failure of an UAV. Environment factors such as wind, lightning or weather conditions or component failures such as battery failure or memory failure could cause a UAV to blackout instantly. In Section 4.3.2, we saw how the UAV-UAV backbone reacts to such failures. In this section, we will simulate the state of the users connected to a blackout UAV.

To simulate this scenario, we reuse the "BlackoutUAV" class we used in Section 4.3.2. This blackout UAV will disappear 10 seconds into the simulation. Each user is generating UDP data packets at 1 Mbps and UAV-UAV wireless links are capped at 10 Mbps. In this simulation, we simulate 3 APs with 3 associated users each as shown in Figure 4.23.

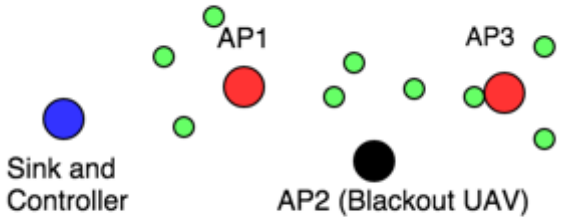

Figure 4.23: Blackout UAV topology

Ten seconds into the simulation, the blackout UAV stops transmitting and receiving. Unfortunately, there is no way for the users to be informed beforehand about this. From the user's perspective, the AP would have suddenly stopped working. In this case, it takes the user almost 3 seconds in order to recognize that it has lost connection with the AP and an additional 2.5 seconds to re-establish connection to the nearest AP. From the controller's perspective, this would be seen as the user leaving and re-entering the network. Connection is temporarily lost in this case for almost 5.5 seconds as shown in Figure 4.24.

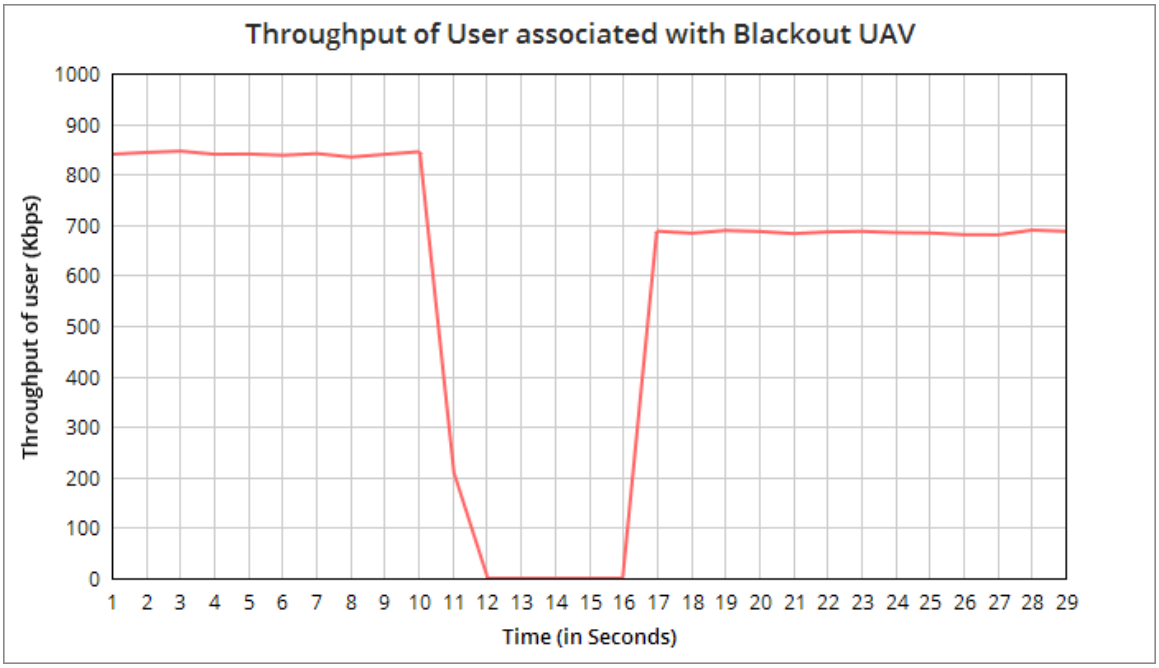

Figure 4.24: Throughput of user associated with blackout UAV

In conclusion, we see that the proposed approach was able to initiate handovers when a user was at the edge of connectivity with an AP in order to retain connectivity. This section also shows that users are handed over to neighboring UAVs in case of overcrowding or power failures in order to maintain seamless connectivity.

Chapter 5

Conclusions and Future Work

The problem addressed by this thesis was to provide Internet connectivity to users via UAVs and to design a backbone that would be scalable, provide high network efficiency in terms of bandwidth and latency, and have support for priority levels within packets. In our research, we found that there were routing protocols satisfying one or a few of the criteria but not all at once. We broke the problem into two sub-problems:

(i) Create a UAV-UAV backbone network.

(ii) Provide seamless WiFi roaming based on this backbone network.

To solve the first sub-problem, we proposed a solution that would be scalable and work towards providing the highest network efficiency at a given time and has support for priority levels. To solve the second sub-problem, we proposed a solution based on OpenSDWN, optimizing it for the proposed UAV-UAV backbone network.

5.1 Contributions, Results and Applications

The solution proposed in this thesis was to use a controller that lies within the network, which is a central hub that monitors all the control information. This hub is used for calculation of routes and to monitor information with regards to a network, which is hard to do in a typical ad hoc network. The routes communicated by the controller provide a mean to distribute traffic throughout the network evenly, hence increasing the efficiency of the global network.

The controller collects information from the UAVs, guides them to select routes that are beneficial to the network and therefore its users. The controller also ensures that the network remains functional in case of failures, both expected and unexpected ones; expected failures such as battery drains and unexpected failures such as a UAV failure. In both cases, the controller responds by switching routes that were broken by the failed UAV(s) in order to keep the network functional and thus proving that this approach is more robust than other routing mechanisms. The contributions of the thesis are as follows:

- Design a more scalable approach for UAV-UAV communication with support for packet prioritization.
- Increase overall throughput of network by evenly distributing traffic throughout the network.
- Find and transmit via faster routes for packets with low delay tolerance i.e., priority packets. Reduce latency by prioritizing transmission of packets with higher priority.
- Extend OpenSDWN to create a seamless wireless experience for the end user.
- Handover users and switch routes in case of a predicted UAV failure.
- Enhance monitoring capabilities for network administrators by creating a centralized repository of information.

The proposed algorithm was designed with specific scenarios in mind and through simulations, we have demonstrated that the proposed algorithm meets the objectives. The simulation results showed that the proposed method provided up to four times as much

throughput (Section 4.2.3.1) and reduce latency to less than 1/4 for critical packets (Section 4.2.2) compared to AODV and OLSR. High throughput is essential for delivering a jitter free experience for the user and low latency for high priority packets is crucial for maintaining the robustness and stability of the network. The improvement in throughput is made possible due to the distribution of traffic and reduction of latency due to t

5.2 Limitations and Recommendations

We do not claim this algorithm to be perfect or suitable for all cases. The proposed solution was designed with the initial problem in mind and further changes would be required to optimize the solution for different problems. This algorithm provides a platform for these future improvements. The proposed algorithms work well in densely connected network scales up in number of nodes.

There are also drawbacks to the proposed algorithm. When the degree of connectivity is low, i.e., the network is sparsely connected, the algorithm on average performs worse than the traditional networking protocols, e.g., OLSR and AODV as illustrated in Section 4.2.1. The added overhead of sending packets through the network is useful only in the case of high bandwidth connections. In the case of sensor networks where the packet size is small or available bandwidth is low, the network performs poorly due to the added overhead of sending UPDATE packets to the controller.

Another situation where the proposed algorithm would perform poorly, in theory, is when the network topology is changing rapidly. When the network is changing rapidly, connections will be lost and UPDATE packets will be forced to be sent very often.

5.3 Future Work

One research direction is to overcome the limitations of the network outlined in Section 5.2 and to make the algorithm more flexible so it can be deployed in a wider variety of scenarios. A possible solution is to simulate the network with a hierarchy of controllers by using local and global controllers to manage large networks and optimizing throughput in each of the local subgroups. The next potential research direction is to turn UAVs into controllers dynamically when the amount of controllers in a local area exceeds a threshold.

Another research direction is to construct the UAV network and test the solution in real world in order to learn practical limitations of the proposed algorithm and radio communications using this method. This would provide valuable information as we explore better solutions for communications and autonomy for traffic management and QoS support like providing higher throughput and utilizing network resources efficiently. Use of different algorithms for path finding will produce different results in latency, traffic distribution and overall throughput.

In the future, we hope to optimize network setup time by decreasing number of packets or by increasing the frequency of sending HELLO packets. This requires practical analysis of the scenario and understanding error models.

Another research direction is to be able to dynamically move around UAVs in order to cover voids left by blackout UAVs or due to overcrowding. This would give network administrators the flexibility to utilize all resources in the network to their maximum potential.

References

[Ahmed13] H. Ali-Ahmad, C. Cicconetti, A. de la Oliva, M. Draxler, R. Gupta, V. Mancuso, L. Roullet and V. Sciancalepore, "CROWD: An SDN Approach for DenseNets." In Proceedings of the 2nd European Workshop on Software Defined Networks, 2013, pp 25 – 31.

[Barz13] C. Barz, J. Niewiejska and H. Rogge. "NHDP and OLSRv2 for Community Networks." In Proceedings of IEEE 9th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob), 2013, pp. 96-102.

[Bellman58] R. Bellman. "On a Routing Problem." Quarterly of Applied Mathematics 16, 8, 1958, pp. 7–90.

[Bux89] W. Bux "Token-ring Local Area Networks and their Performance." In Proceedings of the IEEE, vol. 77, no. 2, 1989, pp. 238-256.

[Carthy05] P. M. Carthy and D. Grigoras. "Multipath Associativity based Routing." In 2nd Annual Conference on Wireless On-demand Network Systems and Services (WONS), 2005 pp. 60-69.

[Chiang97] C. Chiang, "Routing in Clustered Multihop, Mobile Wireless Networks with Fading Channel." In Proceedings of IEEE Singapore International Conference on Networks (SICON), 1997, pp.197-211.

[Clausen01] T. Clausen, G. Hansen, L. Christensen and G. Behrmann. "The Optimized Link State Routing Protocol, Evaluation through Experiments and Simulation." In Proceedings of IEEE Symposium on Wireless Personal Mobile Communications, 2001, pp 34 - 39.

[Clausen03] T. Clausen, P. Jacquet, C. Adijh, A. Laouiti, P. Minet, P. Muhlethaler, A. Qayyum and L. Viennot. “Optimized Link State Routing Protocol (OLSR).” RFC Standard 3626. 2003.

[Dijkstra16] T. H. Cormen, C. E. Leiserson, R. L. Rivest and C. Stein “Section 24.3: Dijkstra's Algorithm”. Introduction to Algorithms (2nd edition). MIT Press and McGraw–Hill, 2001, pp. 595–601.

[Dube97] R. Dube, C. D. Rais, K. Wang and S. K. Tripathi. “Signal Stability based adaptive routing for Ad Hoc Mobile Networks.” In IEEE Personal Communication, 1997, pp. 36-45.

[Floyd93] S. Floyd, V. Jacobson. “Random Early Detection Gateways for Congestion Avoidance.” In IEEE/ACM Transactions on Networking vol. 1, no. 4, 1993, pp. 397-413.

[Ford56] L. R. Ford Jr. “Network Flow Theory.” Paper P-923. Santa Monica, California. RAND Corporation. 1956.

[FordFulkerson16] L. R. Ford, D. R. Fulkerson “Maximal Flow through a Network.” In Canadian Journal of Mathematics vol. 8, 1956, pp. 399.

[FreeWiFi16] “Free public Wi-Fi access is now LIVE at City facilities | City of Ottawa”, Ottawa.ca, 2016. [Online]. Available: http://ottawa.ca/en/residents/parks-and-recreation/recreation-facilities/free-public-wi-fi-access-now-live-city. [Accessed: March 18, 2016].

[Fuhrmann06] T. Fuhrmann, P. Di, K. Kutzner and C. Cramer. “Pushing Chord into the Underlay: Scalable Routing for Hybrid MANETs.” In Fakultät für Informatik. Vol. 12. Technical Report 2006.

[H264] “ISO/IEC 14496-10:2010 - Information Technology -- Coding of Audio-Visual Objects -- Part 10: Advanced Video Coding.” Iso.org. [Online] Available: http://www.iso.org/iso/iso_catalogue/catalogue_ics/catalogue_detail_ics.htm?csnumber=56538 [Accessed March 1, 2016].

[Hartley28] R. V. L. Hartley “Transmission of Information.” In Bell System Technical Journal, 1928.

[Hermann07] S. Hermann, M. Emmelmann, O. Belaifa and A. Wolisz, “Investigation of IEEE 802.11k-based Access Point Coverage Area and Neighbor Discovery.” In Proceedings of the 32nd IEEE Conference on Local Computer Networks (LCN 2007), 2007, pp 949 – 954.

[Intel15a] “Intel WiFi Products — WiFi Client Adapter Connection and Roaming Behavior.” Intel.com, 2016. [Online]. Available: http://www.intel.com/support/wireless/wlan/sb/cs-015906.htm. [Accessed: November 11, 2015].

[Intel15b] “Intel WiFi Products — What is WiFi Roaming Aggressiveness?” Intel.com. 2015. [Online]. Available: http://www.intel.com/support/wireless/wlan/sb/CS-030101.htm. [Accessed: November 11, 2015]

[Jacquet01] P. Jacquet, P. Mühlethaler, T. Clausen et al. “Optimized link state routing Protocol for Ad Hoc Networks.” In Proceedings of IEEE International Multi Topic Conference (INMIC), 2001, pp. 62-68.

[Jawanda01] J. Jawanda. “Method and System for Seamless Roaming between Wireless Communication Networks with a Mobile Terminal.” U.S. Patent 6 243 581, June 5, 2001.

[Jiang99] M. Jiang, J. Li, Y.C. Tay, “Cluster Based Routing Protocol.” IETF Draft, 1999. [Online] Available: https://tools.ietf.org/html/draft-ietf-manet-cbrp-spec-01 [Accessed: March 18, 2016]

[Jump20] "JUMP 20 | Arcturus UAV". Arcturus-Uav.com. [Online] Available: http://arcturus-uav.com/product/jump-20 [Accessed March 18, 2016].

[Liu07] T. Liu, and K. Liu. “Improvements on DSDV in Mobile Ad Hoc Networks”. In Proceedings of the International Conference on Wireless Communications, Networking and Mobile Computing (WiCom), 2007, pp. 1637-1640.

[Lu11] J. Lu, B. Zhang, G. Han, J. Wang, and W. Dou. “A New Improvement on DSDV.” In Proceedings of the 7th International Conference on Wireless Communications, Networking and Mobile Computing (WiCOM), 2011, pp. 1-4.

[Marina01] M. K. Marina and S. R. Das. “On-demand Multipath Distance Vector Routing in Ad Hoc Networks.” In Proceedings of IEEE 9th International Conference on Network Protocols, 2001, pp. 14-23.

[Monin14] S. Monin, A. Shalimov and R. Smeliansky. “Chandelle: Smooth and Fast WiFi Roaming with SDN/OpenFlow.” A Poster Presented at the US Ignite 2014.

[Murthy96] S. Murthy and J. J. Garcia-Luna-Aceves. “An Efficient Routing Protocol for Wireless Networks.” Mobile Networks and Applications vol. 1, no. 2, 1996, pp 183-197.

[Narasimhan13] B. Narasimhan and R. Balakrishnan. “Energy Efficient Ad Hoc On-Demand Distance Vector (Ee-Aodv) Routing Protocol for Mobile Ad Hoc Networks." International Journal of Advanced Research in Computer Science vol. 4, no. 9, 2013.

[NetAnim] "Netanim – Nsnam." [Online] Available: https://www.nsnam.org/wiki/NetAnim [Accessed February 28, 2016].

[NS-3.24] "Ns-3.24 « Ns-3." Nsnam.org. [Online] Available: https://www.nsnam.org/ns-3-24 [Accessed February 28, 2016].

[Perkins03] C. E. Perkins, E. Belding-Royer and S Das. "Ad hoc On-demand Distance Vector (AODV) routing." RFC 3561. 2003.

[Perkins94] C. E. Perkins and P. Bhagwat. "Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers." In ACM SIGCOMM Computer Communication Review, vol. 24, no. 4, 1994, pp. 234-244.

[Rahman09] A. H. A. Rahman, and Z. A. Zukarnain. "Performance Comparison of AODV, DSDV and I-DSDV Routing Protocols in Mobile Ad Hoc Networks." In European Journal of Scientific Research 31, no. 4, 2009, pp 566-576.

[Royer99a] E. M. Royer and C. Toh. "A Review of Current Routing Protocols for Ad Hoc Mobile Wireless Networks." In IEEE Personal Communication, 1999, pp.46-55.

[Royer99b] E. M. Royer, and C. E. Perkins. "Multicast Operation of the Ad Hoc On-demand Distance Vector Routing Protocol." In Proceedings of the 5th Annual ACM/IEEE International Conference on Mobile Computing and Networking, 1999, pp. 207-218.

[Schulz14] J. Schulz-Zander, N. Sarrar and S. Schmid. "Towards a scalable and near-sighted control plane architecture for WiFi SDNs." In Proceedings of the 3rd Workshop on Hot Topics in Software Defined Networking (HotSDN), 2014, pp. 217-218.

[Schulz15a] J. Schulz-Zander, C. Mayer, B. Ciobotaru, S. Schmid, and A. Feldmann. "OpenSDWN: Programmatic Control over Home and Enterprise WiFi." In Proceedings of the 1st ACM SIGCOMM Symposium on Software Defined Networking Research, 2015, p. 16.

[Schulz15b] J. Schulz-Zander, C. Mayer, B. Ciobotaru, S. Schmid, A. Feldmann, and R. Riggio. "Programming the Home and Enterprise WiFi with OpenSDWN." In Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication, 2015, pp. 117-118.

[Toh97] C. Toh. "Associativity-based Routing for Ad Hoc Mobile Networks." In Wireless Personal Communications 4, no. 2, 1997, pp. 103-139.

[TokenBus16] "IEEE 802.4 Token Bus MIB". Tools.ietf.org. [Online] Available: https://tools.ietf.org/html/draft-ietf-snmp-tokenbusmib-00 [Accessed February 23, 2016].

[TORA16] "Temporally-Ordered Routing Algorithm (TORA) Version 1 Functional Specification", Ietf.org. [Online] Available: http://www.ietf.org/proceedings/53/I-D/draft-ietf-manet-tora-spec-04.txt [Accessed 23-February, 2016].

[Vara15] M. Vara and C. Campo, "Cross-Layer Service Discovery Mechanism for OLSRv2 Mobile Ad Hoc Networks." Sensors, vol. 15, no. 7, 2015, pp. 17621-17648.

[WiFiCalling16] "How to Enable WiFi Calling on Your Iphone 6S or 6S Plus". Imore.com. [Online] Available: http://www.imore.com/how-enable-wi-fi-calling-your-iphone-6s-or-6s-plus. [Accessed: February 22, 2016]

[Zapata02] M. G. Zapata. “Secure Ad Hoc On-demand Distance Vector Routing.” In ACM SIGMOBILE Mobile Computing and Communications Review vol. 6, no. 3, 2002, pp.106-107.

[80211IEEE2008] “IEEE Standard for Information Technology-- Local and Metropolitan Area Networks-- Specific Requirements-- Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 1: Radio Resource Measurement of Wireless LANs.” IEEE Standard 802.11, 2008.

[80211IEEE2012] “Information Technology--Telecommunications and Information Exchange between Systems Local and Metropolitan Area Networks--Specific Requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications.” IEEE Standard 802.11, 2012.

[8023IEEE2000] “IEEE Standard 802.3. Part 3: Carrier Sense Multiple Access with Collision Detection (CSMA/CD) Access Method and Physical Layer Specifications.” IEEE Standard 802.3, 2000.