Layer 2.5 Network Coding to Improve 802.11 Network Efficiency

Jarrett Gaddy

A thesis

submitted in partial fulfillment of the

requirements for the degree of

Master of Science in Electrical Engineering

University of Washington

2016

Reading Committee:

Sumit Roy, Chair

Thomas R. Henderson

Program Authorized to Offer Degree:

Electrical Engineering

University of Washington

Abstract

Layer 2.5 Network Coding to Improve 802.11 Network Efficiency

Jarrett Gaddy

Chair of the Supervisory Committee:
Professor Sumit Roy
Electrical Engineering

The principles of Network Coding have been developed conceptually for over a decade (starting with key results c. 2003 on network coding for butterfly networks). Despite several subsequent extensions to wireless networks, there exists remarkably few *implementations* of Wireless Network Coding (WNC) techniques within existing wireless networking technology. In this thesis, we present a novel Layer 2.5 implementation for WNC system that can be retrofitted to existing 802.11 nodes (APs and client) with some minimal changes (i.e. is backward compatible with legacy nodes that do not implement WNC) to improve network throughput, when there exist significant intra-network symmetric traffic flows. We implement L2.5 WNC on an 802.11 software defined radio test-bed to experimentally collect data and demonstrate feasibility. We experiment using a simple network with 2 clients and 1 access point. The clients saturate the network by sending messages to each other through the access point whenever possible. Our experimental results show that on average, adding L2.5 network coding to an 802.11 network increases the network efficiency by 11.7% and throughput by 13.6%.

TABLE OF CONTENTS

CONTENTS

Chapter 1. INTRODUCTION

The IEEE 802.11 standard (or Wi-Fi) has become the most common broadband wireless access infrastructure. Due to its widespread use, the inefficiencies of the base Distributed Coordination Function (DCF) MAC protocol has become apparent, particularly for *dense* network deployments. This thesis proposes a partial solution to this problem by using wireless network coding (WNC) techniques for scenarios whereby there exist bi-directional traffic flows within clients in a single network. The proposed WNC scheme involves combining layer 3 datagrams at Access Points (APs) and layer 2 broadcast as described, to control the flow of layer 2 data and acknowledgment. This scheme is implemented in a new layer between 2 and 3 of the standard OSI stack, and hence we refer to it as a layer 2.5 (L2.5) solution.

This thesis will describe the operating scenarios with a few examples, for which we propose to use wireless network coding. We discuss the advantages and disadvantages of using layer 2 or layer 2.5 network coding designs and why we decided to use layer 2.5 implementation. The implementation details of layer 2.5 WNC will be explored, including the modifications to clients and access points. Finally, this thesis will discuss the experiment setup – using BladeRF software defined radios (vendor: nuand.com) and the results form adding layer 2.5 network coding to the simplest of 802.11 networks, as benchmarked against operation without network coding. The throughput gains are studied as a function of different network parameters and their implications exposed.

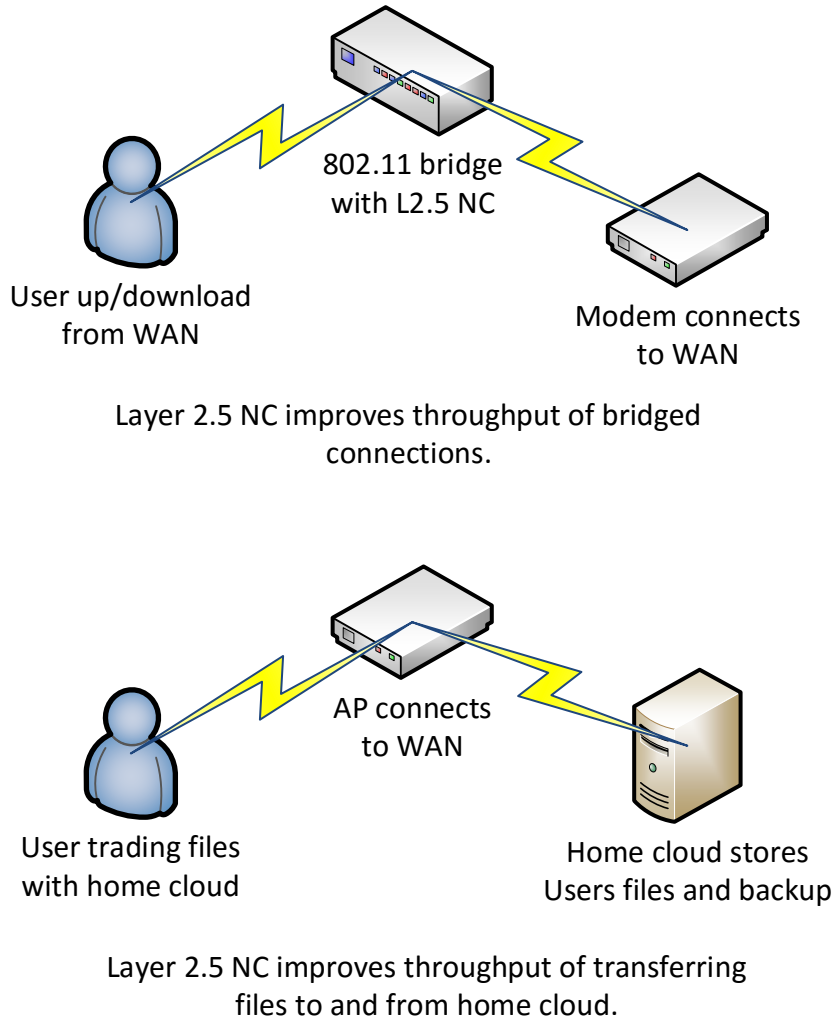


Figure 1: Layer 2.5 NC home use cases.

In our experiment we will use very specific conditions in which L 2.5 network coding are optimal, a symmetric, bidirectional, high data rate flow between 2 clients, with no other traffic on the network, but, we believe that L2.5 NC can still make significant improvements to 802.11 throughput in several common home use scenarios. Figure 1 shows two home use cases in which layer 2.5 network coding will improve throughput significantly. First, there is the case of a bridged connection. Today, many home 802.11 networks have extended range using an 802.11 bridge. WAN traffic comes in to the network at what can be viewed as a client in layer 2.5 which means L2.5 NC can be performed at the wireless bridge on all upload/download traffic to the internet. Second, the case of a home cloud setup. With the increasing need for data storage, many home users are creating their own home cloud systems for file storage and backup. Layer 2.5 NC can be

performed at the AP as information is being traded between the user's PC and their home cloud server.

Principles of Network Coding: Brief Recap

Network coding (NC) is the notion that data meant for transmission through a packet switched network can be mapped to new data (`coding') at intermediate nodes (routers) as opposed to typical store and forwarding (i.e. traditional routing) [1]. Thus network coding has been touted as a significant component in the development of new protocols to respond to needs for quick and reliable data transfer that scales better with increase in multicast/broadcast traffic (than legacy protocols) [6]. While many researchers focus on developing entirely new network designs using physical layer coding [10] or redesigning layer 2 of 802.11 [5] to realize improvements over baseline 802.11 network performance, we instead focus on improving 802.11 with only minor changes to remain compatible with the existing 802.11 b/g/n/ac infrastructure. In order to remain compatible with legacy 802.11, our NC protocol must be designed carefully so as to not disrupt legacy users that do not have NC capability [9]. Prior art has suggested using XOR coding to reduce the number of transmissions required [2, 4], which we will implement in our experiments and measure resulting improvements.

There is significant literature describing network coding proposals at the physical or MAC layers (requiring significant changes to existing 802.11 stack), and a few implementations of network coding within the application layer. Via redesign of the physical layer to add physical or analog network coding [3, 11], 802.11 networks can be made much more efficient. With Physical layer NC, it is possible to combine individual symbols transmitted from different nodes at an intermediate node (Access Point), turning interference into an advantage via multicasting. Unfortunately, any physical layer coding scheme will not be compatible with legacy devices as the entire radio system must be redesigned in order to accommodate the coding mechanism.

There is also prior work with adding network coding based multicast schemes at the application layer. In [7] the author suggests that the applications collecting data can establish a peer-to-peer network when nearby clients want similar information. They can then establish a system where each client will obtain parts of the information and then distribute the rest to each other. This type of network coding will certainly improve the efficiency of multiple clients downloading the same

file and using the same application. Our research focuses on improving the efficiency of traffic flow that originates and ends in the 802.11 network, while the scheme put forth in [7] improves traffic flow from outside the network but going to multiple clients within the network. Both MAC layer and the application layer described in [7] could be used together to improve multiple aspects of 802.11 under different traffic flow conditions. Another application layer coding system is described in [8]. Their system creates an acyclic graph for delivery of data through a network. This system focuses on transmitting data around a network edge instead of requiring transmissions to and from the network center when edge devices want application data. This type of network coding does not work to solve the issue that we attempt to solve with our NC design. We have designed a system that can code any information traversing a single 802.11 network, while the systems described in [7, 8] code specific application data through the internet as a whole.

Finally, there have been some attempts at creating network coding systems at the MAC layer, these rework the MAC layer in order to support multicasting of sharing of data [5]. While this can be expected to lead to more significant improvements over 802.11 than our design, our approach is unique in its backward compatibility with legacy 802.11 devices on the same network. Some previous work has focused only on simulating a network in which frames are coded together [12] in a similar way to our system but these papers do not focus on the issue of implementing a network coding system in to the 802.11 MAC. Other researchers have focused on the theory of MAC layer coding, showing the limits of how much it can benefit a network [13]. In [14] it is shown that network coding can bring benefits to frequently used applications like video conferencing, but we believe that network coding can be ubiquitously applied at layer 2.5 and can then code any type of data from any application. In this thesis we develop a WNC solution that, unlike previous WNC theories, focuses on integrating with pre-existing 802.11 infrastructure instead of on maximizing throughput increases. Using the 802.2 logical link control header used by the 802.11 specification we have created a system for WNC that conforms completely with IEEE 802.11 layers 1 and 2. Our system relies on the 802.11 layer 2 broadcast features which do not include acknowledgment of data frames sent as a broadcast. The lack of acknowledgment is seen as an issue by many researchers as all 802.11 layer 2 unicast data is guaranteed reliable transmission using ACK frames and retransmissions. We have solved the lack of broadcast ACK problem by further introducing a layer 2.5 acknowledgment frame which is piggybacked on to unicast data frames. It is expected that layer 2.5 network coding will improve the throughput of bidirectional data between to clients

on an 802.11 network by up to 25%, while also reducing the number of transmissions used by the network by up to 25%.

Chapter 2. SYSTEM ARCHITECTURE

2.1 802.11 MAC

Under normal operation, an 802.11 MAC frame has a header, payload, and frame check sequence. A frame as shown in Figure 2 will typically carry a logical link control (LLC) header and IP packet in its payload. For network coding we will use a LLC header field to specify whether a frame is a network coding frame or a typical 802.11 MAC frame. The MAC header consists of a frame control field, which contains the protocol version, frame type, frame subtype, and a number of informational flags. The three main types of frames are control, data, and management. The next field in the MAC header is the MAC Duration/ID field which contains a number corresponding to the number of timeslots the station needs reserved for the carrier sense multiple access channel usage. The next two fields in the MAC header are the source and destination addresses. The 5th field in the header is room for a 3rd address used in some 802.11 network architectures. The sequence control field has a sequence number, which in typical 802.11 use case is used for reassembling the order of frames. This field will be used much more significantly in our network coding scheme. The final field before the payload is a 4th address field. This field is often not used, but can be used in some networks for additional functionality. After the payload is a four byte frame check sequence that is computed using a cyclic redundancy check. This is used to ensure the accuracy of the frame.

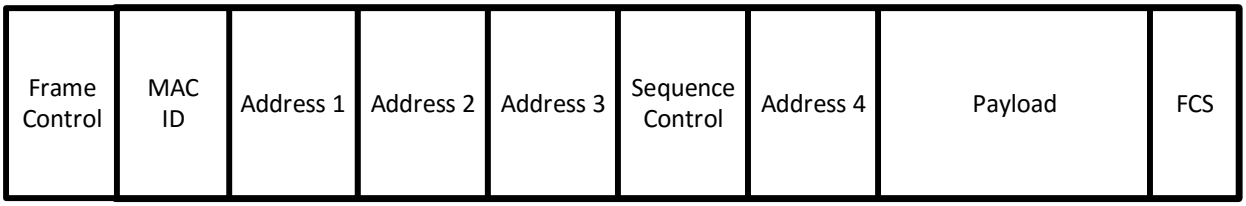


Figure 2: Typical MAC frame structure.

The MAC layer payload consists of two pieces, the logical link control and the encapsulated data. The logical Link Control layer is specified by IEEE 802.2 and contains information about how a frame is encoded in layer 2 for proper decoding. In most cases with 802.11 use, the LLC contains s standard set of data that represents the use of an ether type value corresponding to the use of IP. The LLC layer allows for specifying proprietary protocols. We will use this field to specify that a

frame is coded using layer 2.5 network coding by using the value 0xAA for the DSAP and SSAP then creating our own organizational identifier and protocol id (need figure of LLC). The second part of the payload after the LLC is the encapsulated datagram. This datagram will often be an IP packet.

2.2 LAYER 2 VS LAYER 2.5 NETWORK CODING

MAC layer network coding can be done at one of two locations in the 802.11 network stack. The network coding can be done at the MAC layer by using extra pieces of the MAC header in order to store the information required for network coding. Alternatively, network coding can be implemented using IEEE 802.2 Logical Link Control, an extra header that exists between layers 2 and 3 to pass extra information from layer 3 down to layer 2, this will be called layer 2.5 of the network stack shown in Figure 5. The goal behind network coding is shown in Figure 3 where we can see that on the left side in a legacy network four transmissions are used to send two messages mA and mB while on the right in either a layer 2 or 2.5 network coding system only three transmissions are used to send the same two messages. Because each client must know one of the messages used in creating the coded message to decode the other message, any messages introduced at the AP (ie: traffic downloaded from WAN) cannot be coded, meaning we will see no positive effect on data coming from or going to outside the immediate 802.11 network.

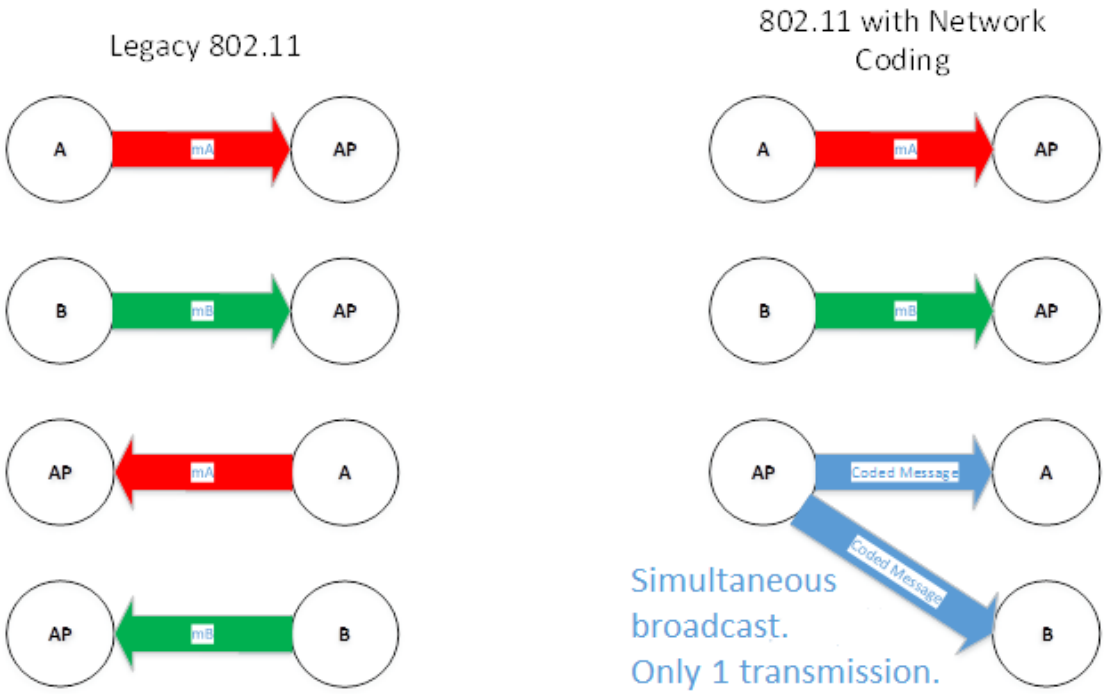


Figure 3: Message transmissions in a legacy network vs network coding network.

In layer 2 network coding the information for the extra destination address and frame identification numbers are stored in the extra address fields 3 and 4 which are not frequently used in IEEE 802.11, the frame header is shown in Figure 4. In layer 2 network coding the entire payload of the layer 2 frame is coded data. The payload contains two different IP packets that have been combined as well as combined LLC headers. The decoding process in this case is simple as the entire payload of the MAC frame can be passed to the decoding function. The design of layer 2 network coding is simple in that there is no need to design a new layer of the internet stack because all the information we need to do the decoding is stored in pre-existing fields of the MAC header. The problem with this design is that we have not collected enough information to determine how replacing address 3 and 4 in a MAC frame will affect legacy 802.11 systems in which network coding may operate. Layer 2 network coding is also not scalable in that it can support at most 1 additional receiver for a frame for a total of 2. While this paper will not discuss network coding with $n > 2$ messages coded together in a single packet, the system should be designed with the possibility for $n > 2$ being considered.

Frame Control	MAC ID	Address 1	Address 2	NC Address 1	Sequence Control	NC Address 2	Network Coded IP Packets	FCS

Figure 4: Layer 2 network coding MAC frame format.

Layer 2.5 network coding is performed using extra information placed in the layer 2 payload that surrounds the coded IP packets. This is called layer 2.5 because it uses information between the OSI defined MAC at layer 2 and IP at layer 3. In IEEE 802.11 there is already a small amount of information in this area of the network stack. It is called the Logical Link Control Layer and is defined in IEEE 802.2. The Logical Link Control layer can be used to pass layer 2 information to layer 3. In the case of network coding the information passed will be the address of recipient nodes, as well as frame identification information. The MAC frame for layer 2.5 is shown in Figure 6. The network stack structure with L2.5 network coding is shown in Figure 5. The upper layers of the network stack and the physical layer are shown in blue as they require no modification or extra information for layer 2.5 to be inserted. Shown in yellow is the MAC layer and the IP layer which are both used by network coding and require some implementation, but not protocol, modifications in order to insert layer 2.5. Layer 2.5 is shown in red and is inserted between the MAC layer and IP layer in a typical network stack.

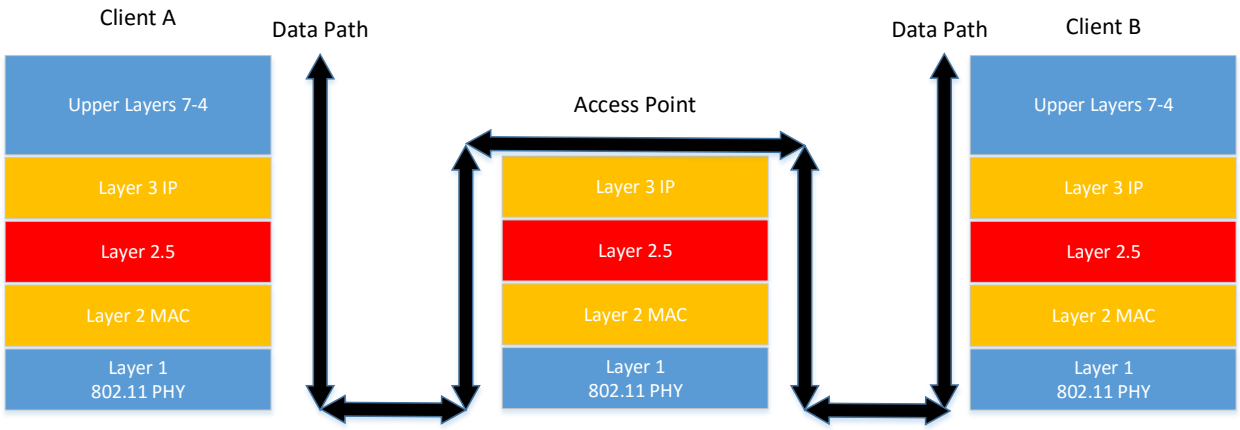


Figure 5: Network layer structure with layer 2.5 inserted.

The section labeled “NC Info” can be either at the front or end of the payload, but the LLC header specifying network coding must be at the front of the payload to conform to IEEE 802.11 MAC. The implementation for this system is more difficult than for layer 2 network coding as an entirely

new layer must be designed and implemented in code. Layer 2.5 is better than layer 2 in compatibility with legacy 802.11 systems because any legacy 802.11 node that receives a network coding packet will discard a MAC frame with an unfamiliar LLC header. This allows us to send the network coding frame as a broadcast to all nodes which check the LLC and NC Info section of the payload to see if they are a recipient of the coded data encapsulated. Additionally, layer 2.5 allows for scalability to n > 2 recipient nodes for coding. This can be done by simply adding more information to the NC Info portion of the frame. The drawback to this is that part of the payload's capacity is used by network coding information and not actual data. This could cause a need for fragmentation if the data concatenated with the network coding info is longer than the payload size, however, the maximum layer 3 frame length is still larger than 1500 bytes, the maximum length of an Ethernet frame, meaning fragmentation will be unlikely.

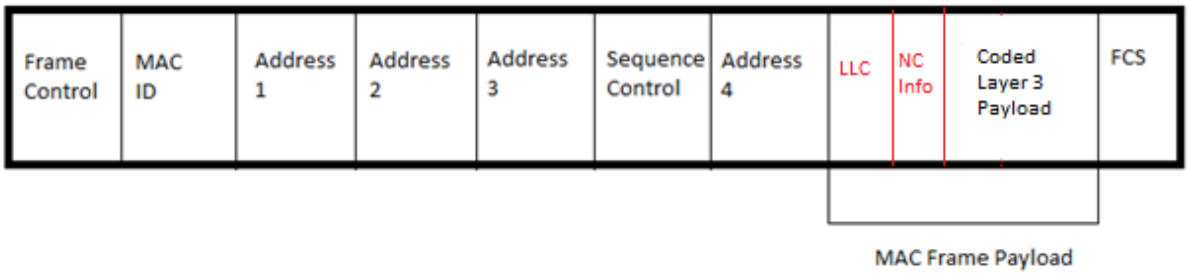


Figure 6: Layer 2.5 network coding MAC frame format.

The layer 2.5 network coding system we have developed increases the efficiency of peer to peer communication between two clients associated with one access point. To show how much layer 2.5 NC effects the efficiency of the 802.11 network we design a scenario in which two clients are constantly sending data frames to each other at the same constant rate. The network will have a client A sending a message Ma to a client B which sends a message Mb to client A as shown in Figure 7. Figure 7 and Figure 8 show how this network scenario behaves in a traditional 802.11 network that does not perform any network coding. In traditional 802.11 clients can not directly communicate with each other, as such each client must transmit their message to the access point(AP) which will then send the message on to the appropriate destination client. To exchange messages Ma and Mb between clients A and B requires 4 layer 2 unicast data frames are required.

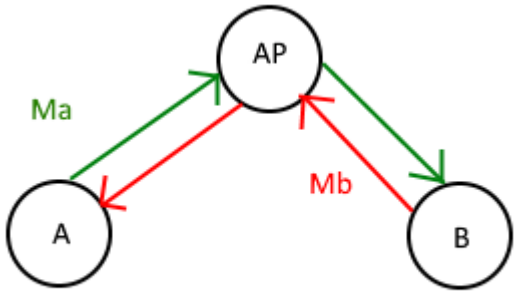


Figure 7: Network architecture without network coding.

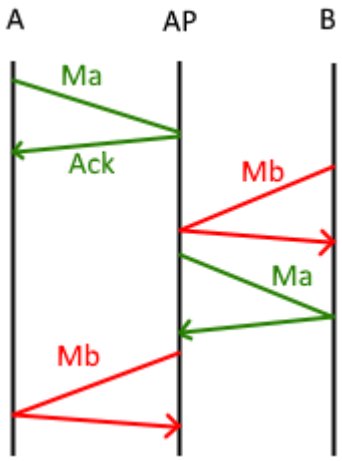


Figure 8: Timing diagram of non-network coded information transfer

When we add layer 2.5 network coding to the network shown in Figure 7 we can code messages Ma and Mb together at the access point and send a single broadcast frame to both clients A and B as shown in Figure 9. By combining the Ma and Mb from the access point we can reduce the number of layer 2 frames required to exchange the information by 1 as shown in the timing diagram (add fig). The tradeoff of preforming layer 2.5 network coding is that each client receiving a frame from the access point in a coded layer 2 frame must individually acknowledge receiving its frame because 802.11 broadcast frames are not acknowledged.

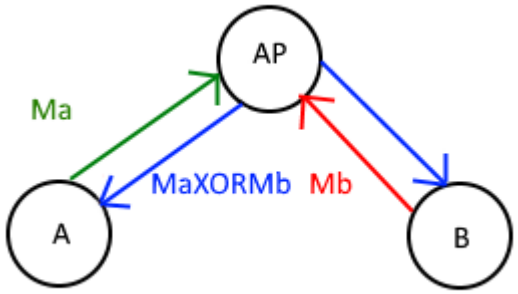


Figure 9: Network architecture under network coding.

2.3 ACKNOWLEDGEMENT OF BROADCAST

One feature of the 802.11 MAC is that all layer 2 data frames must be acknowledged(ACK) by their recipient or the source will immediately attempt to retransmit. Traditionally, broadcast is not used for sending data frames as data frames must be immediately acknowledged, but if each client receiving the broadcast immediately tries to acknowledge the data frame the acknowledgments will cause collisions and the source of the broadcast will not receive any acknowledgments. 802.11 does not currently support ACK of broadcast and multicast frame [15] but some ACK system is required for network coding to be viable. The scenario in which acknowledgments to a broadcast collide is called ACK explosion and is a significant problem that must be solved for any proposal to send data using broadcast. In this thesis we discuss three acknowledgment systems to solve the ACK explosion problem in regards to layer 2.5 network coding, and we determine which is best to implement in the final experiment.

The first plan for acknowledgement is the simplest and will be used for initial experiments. This scheme will use no acknowledgment for frames sent by a network coded broadcast. Any ACK scheme for network coding has significant difficulties because we need two different clients to ACK the reception of their data frame at the same time. In this case we will simplify network coding by assuming all coded frames are received and decoded without error. In practice this is not likely to be the case, so using this ACK strategy we will lose significant amounts of data. This could however be practical for some types of data transmission that rely more and speed and throughput than accuracy.

The second strategy for acknowledgement is an explicit ACK for each frame that is decoded after a NC broadcast. The ACK will be a layer 2.5 frame that contains no data and is sent immediately

to the AP from each client that received a frame as part of the NC broadcast. The layer 2.5 ACKs will be sent as layer 2 data frames which means they will be sent using normal 802.11 CSMA/CA which will circumvent the issue of ACK explosion. The advantage of this strategy is that it is not too complex to implement and it guarantees that all data is transmitted to the correct destination without error. The downside to this plan is that two additional messages must be sent after a NC broadcast before any messages are acknowledged as having been received. Each of these messages are sent as unicast 802.11 frames which means they must vie for contention under DCF and must each be ACKed by normal 802.11 ACKs. This will take a significant amount of extra time to send all the frames required which will significantly reduce the gains of doing network coding in the first place. Because the ACK's are very short messages with no data, it is likely that there would still be significant gain in throughput even when using this ACK strategy, but a more efficient strategy would improve the throughput gain. A timing diagram of this ACK strategy is shown in Figure 10. In this diagram the explicit ACKs are shown after the broadcast NC frame. These ACKs are not guaranteed to be in the order shown below, and may even have other unicast transmissions in between the broadcast frame and any of the ACKs. While the access point is waiting for ACKs from the destinations of its NC frame it cannot send any more NC broadcast frames and may have limited functionality of sending unicast data or management frames which is another significant drawback of this plan.

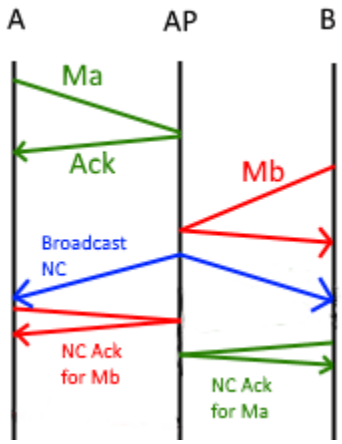


Figure 10: Timing diagram of network coded information transfer.

The final ACK strategy is to use a piggybacked acknowledgement in future unicast data frames from the recent destination of broadcast NC frames. The idea for this ACK strategy is shown in Figure 11 and uses the next message coming from a client to acknowledge broadcast NC frames. When the access point sends a network coded frame as a broadcast, the two sequence numbers

from the original frames that were coded together are sent in the NC header for the broadcast frame. Upon receiving a NC frame a client will attempt to decode the message intended for it. If the broadcast is successfully received, then a field in the client that holds the most recent decoded sequence number is set to the sequence number of the frame just decoded. Every time a client sends a data frame the client will send a slightly modified normal 802.11 data frame. The frame will have a modified LLC header that indicates we are using the layer 2.5 NC protocol. The LLC header will be followed by the value in the most recently decoded sequence number field. This value will be the sequence number of the data frame contained in the last NC frame received by this client as long as it was received correctly. This will then be followed by the layer 3 data as the payload. It may be necessary to force an explicit acknowledgment frame to be sent if there are no data frames that are ready to be sent on a client that needs to send an ACK. In this case we can use the same explicit ACK from the 2nd ACK strategy instead of piggybacking on a data frame that may not be ready until after a timeout has occurred.

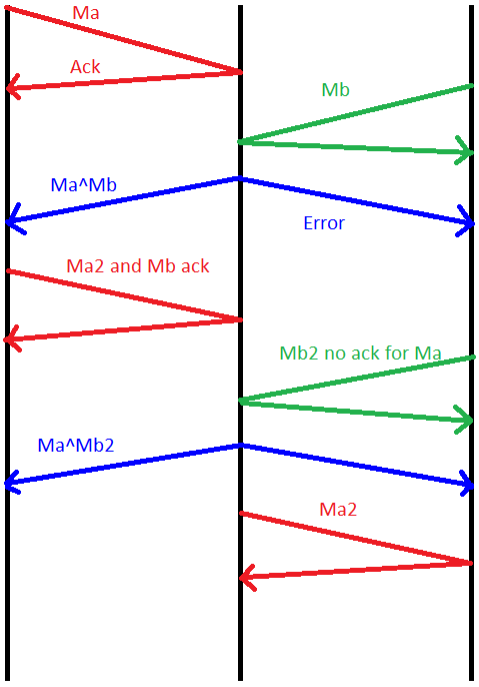


Figure 11: Timing diagram of network coding with implicit ACK and an error receiving the NC frame.

In either the 2nd or 3rd ACK strategy we have the possibility of having to wait for some time to receive an ACK. The access point will keep track of each client's ACK status. If a client's status

is waiting for an ACK then that client cannot be used with network coding. Eventually if no ACK is received a timeout will occur and act as a negative acknowledgement. Additionally, in the 3rd strategy, a message from a client involved in a past NC broadcast can act as a negative ACK if it does not contain the most recently coded sequence number. When a negative acknowledgment happens, the frame that was meant to be sent is immediately placed in the front of the transmit queue for its intended destination. This can happen for the coded frame at either 1 or both intended receiving clients. That frame can be coded again if possible. When a positive ACK occurs, the frame that was coded and intended for the receiving client that sent the ACK can be forgotten at the access point and the client can move on to sending new data as either unicast or network coded without restriction.

For the purpose of experimenting with layer 2.5 network coding we will implement the 3rd ACK strategy described above. We hypothesize that the piggyback ACK scenario will result in maximum efficiency gain over traditional 802.11 while also minimizing the number of frames that are not transmitted successfully. The first plan of ignoring ACKs would result in the most efficient scenario, because there would be no wasted time sending or waiting for ACK frames, however this would sacrifice reliability as frames could be lost and never retransmitted. The second plan of sending explicit layer 2.5 ACK frames ensures reliability but requires the AP to wait for ACK frames and each client to take time sending an ACK frame before sending more data. By spending time only ACKing the efficiency gained by doing network coding is reduced. Therefore, the implicit ACK scenario of the 3rd strategy is the most sensible to use in layer 2.5 NC experiments.

2.4 Summary of Layer 2.5 Design

2.4.1 *Design Features*

- Operates in 802.11 network between MAC and network layers.
- Uses 802.2 LLC to distinguish L2.5 frames from non L2.5 frames.
- Frames that are layer 2.5 frames have added layer 2.5 header between LLC and layer 3 header.
- MAC layer does not change, however, sequence number of frames and must be passed to layer 2.5 and remembered for decoding.
- Layer 2.5 remembers data and sequence number of sent frames at client.

- Layer 2.5 combines messages at the access point and sends sequence numbers of the combined frames in the layer 2.5 header.
- Client uses remembered data and sequence numbers to decoded combined frames.
- Client sends acknowledgments when it receives a layer 2.5 coded frame.
- Layer 2.5 ACK is just a layer 2.5 header containing the sequence number being ACKed.
- Layer 2.5 ACK header is placed in a normal message headed to the access point to increase efficiency. LLC header is also modified for one of these L2.5 piggyback ACKs.

2.4.2 *Pros*

- Easy to implement on existing 802.11 infrastructure with software update to clients and access points.
- Has no negative impact on and can coexist in the same network with legacy 802.11 devices.
- Does not affect traffic flow outside of coding symmetric traffic flows.
- Allows for up to 25% efficiency gain over traditional 802.11 network.
- No disruption of data or loss of efficiency for frames introduced at access point (ie: internet download) or frames that leave the 802.11 network.

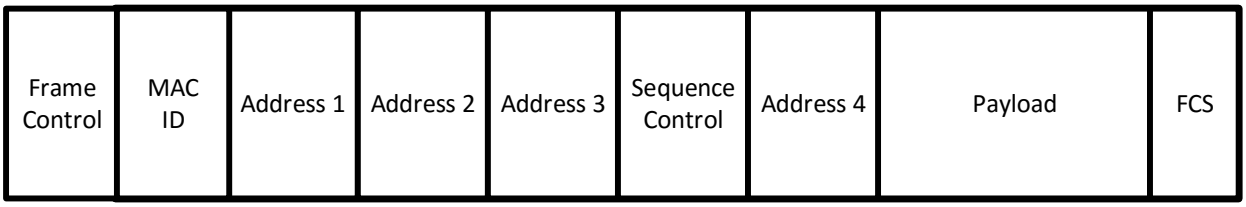
2.4.3 *Cons*

- Requires symmetric data flow between clients on the network for efficiency gain.
- Requires complicated ACK system to ensure reliable data transmission.
- Introduces extra processing and memory requirements on clients and access points.
- Requires at least one access point and two clients to be updated for any efficiency gain.
- Broadcast data frames require processing at all nodes on a network even if they are not the frames destination.
- Has no positive effect on data downloaded from or uploaded to outside the 802.11 network.

2.5 SPECIFICATIONS FOR USING LAYER 2.5 NETWORK CODING

In this section we layout the specifications for using layer 2.5 network coding as it is implemented in the experiment described later in this document. Layer 2.5 can be implemented using these guidelines while changing the organizational unique identifier and protocol id used in the LLC header for layer 2.5 frames as appropriate. We use the values shown below in our implementation.

2.5.1 *MAC Frame*



A layer 2 MAC frame contains the following in order:

- 2 bytes of Frame Control information
- 2 bytes for Duration and ID number
- 6 bytes for Address 1
- 6 bytes for Address 2
- 6 bytes for Address 3
- 2 bytes for sequence control
- 6 bytes for Address 4
- 0 to 2312 bytes for Payload
- 4 bytes for frame check sequence

The variable length from 0 to 2312 byte payload contains the 802.2 logical link control header. If the MAC frame contains coded data then a layer 2.5 datagram follows the logical link control header, if the data is not coded, a layer datagram will immediately follow the logical link control header. The frame check sequence FCS is used to ensure that there are no bit errors in the received MAC frame.

2.5.2 *Logical Link Control*

The logical link control header is the upper part of the data link layer which is layer 2 of the OSI model. The logical link control header can be used to notify a layer 2 service point what higher layer protocol is encapsulated by the layer 2 frame. This is important to layer 2.5 network coding because a node in the 802.11 network needs to be informed of whether a received frame is a network coded frame or not. The logical link control frame is mandatory in all 802.11 transmissions. The logical link control 802.2 specification allows vendors to specify their own protocol to operate on top of the data link layer. Layer 2.5 network coding is specified as a proprietary protocol operating on top of the 802 data link layer. As an experimental value, the three byte organizational unique identifier OUI used for layer 2.5 network coding is 0xAAAAAA. The OUI is not registered with any registration authority and therefore is not a valid OUI but for experimenting any value other than 0x000000 which specifies ether type can be used.

The logical link control header contains the following values in order for a coded frame:

- 1 byte for Source Service Access Point SSAP containing 0xAA
- 1 byte for Destination Service Access Point DSAP containing 0xAA
- 1 or 2 bytes for Control field (still working on understanding this field. uses high-level data link control specification) should contain 1 byte 0x03
- 3 bytes for OUI containing 0xAAAAAA
- 2 bytes for Protocol ID containing 0xAAAA
- 0 to 2304 bytes for layer 2.5 payload

2.5.3 *Access Point Layer 2.5*

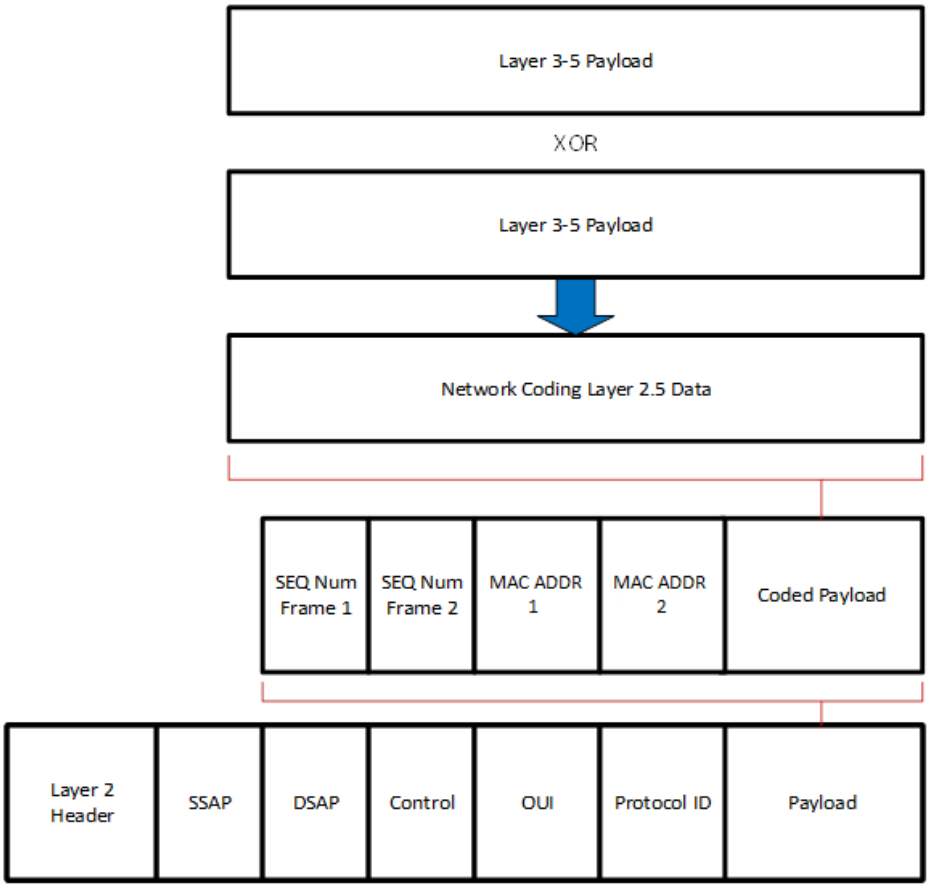


Figure 12: Layer 2.5 Data Coded Frame encapsulation.

When the access point on an 802.11 network intends to send data it must determine whether or not there is enough data stored in the network to perform network coding. If the access point determines that it can do network coding it will first create a layer 2.5 frame.

The layer 2.5 frame created will have a layer 2.5 header with the following information in order:

- 6 bytes for MAC address of destination 1
- 2 bytes for sequence control taken from frame intended for destination 1
- 6 bytes for MAC address of destination 2
- 2 bytes for sequence control taken from frame intended for destination 2
- 0 to 2288 bytes for payload which will contain 2 layer 3 datagrams XOR together

The layer 2.5 frame is then encapsulated with a logical link control header as described in section 3 of this document. The logical link control header and layer 2.5 frame is then encapsulated into

the payload of an 802.11 MAC frame. The MAC frame is given a MAC address of FF:FF:FF:FF:FF:FF in the Address 1 field in order to send a broadcast transmission. The MAC address of the access point is placed in Address 2. Address 3 contains the BSSID of the 802.11 network and Address 4 is not used. The frame type is set to be a data frame and the remaining fields are set as specified in IEEE 802.11 depending on the network configuration. The access point must remember which higher layer frames were sent and which clients have been recipients to network coded frames to ensure reliable data transmission. The access point must remember the most recent coded frame sent to each associated client until that client has acknowledged receiving the decoded frame intended for it. The access point may send a network coded frame to a client that currently has an un-acknowledged network coded frame previously sent.

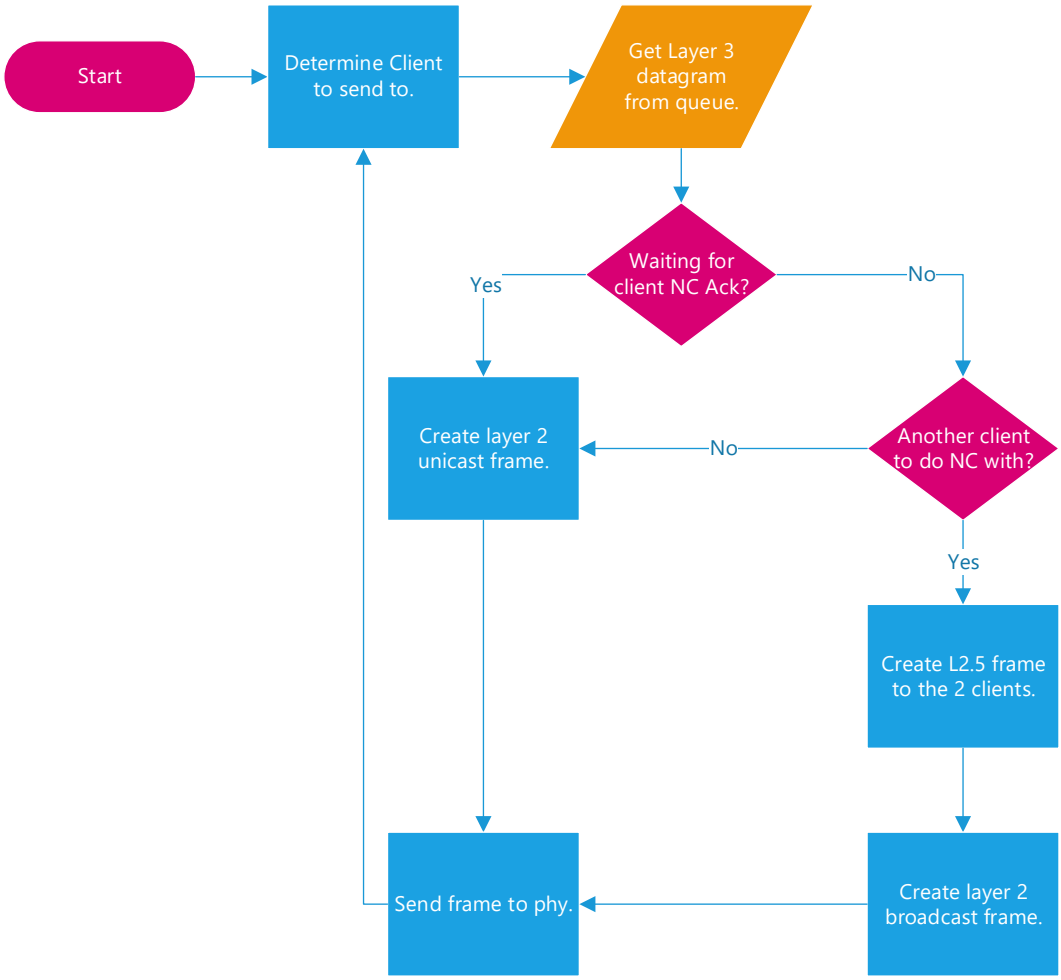


Figure 13: Flowchart of AP operation.

Figure 13 shows a flowchart of the operation of the transmit loop for an access point using layer 2.5 network coding. The AP must first determine which data it will transmit on the next

transmission. The AP then checks if it has recently sent layer 2.5 coded frames to the destination of the data. If the AP has sent a coded frame recently and it has not been Acked yet, then the destination is not available to use with network coding, so the AP will send a unicast layer 2 frame. If the destination is available for network coding, then the AP must find a suitable candidate data to code with the current data. A candidate is suitable if its destination is available for network coding, and its destination is the source of the current data. The candidate data must also have been sent from the client that the current data is intended for which is checked against the source address of the candidate data. If the coding criteria is met, then the AP can construct a layer 2.5 header, and send the layer 2.5 coded frame in a layer 2 broadcast. If there are no valid candidate data, the current data will be sent as a unicast layer 2 frame.

2.5.4 *Client Layer 2.5*

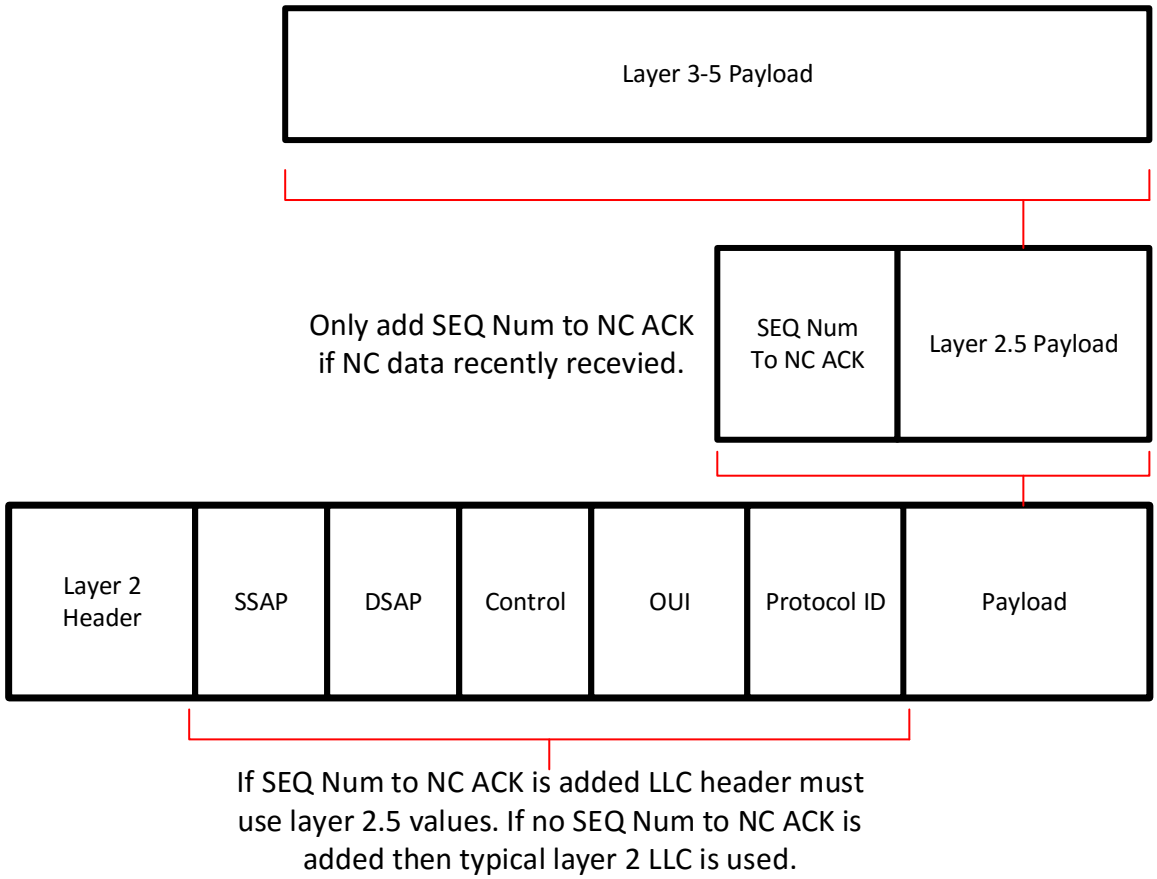


Figure 14: Piggyback Ack Frame encapsulation.

A client on the 802.11 network does not code data frames together, however, it can send a layer 2.5 frame in order to acknowledge receiving and correctly decoding a layer 2.5 frame sent to it by

the access point. When an access point sends a layer 2.5 coded frame, all clients on the network will receive the frame because it is sent as a broadcast. Each client will use the layer 2.5 header to determine if it is an intended destination of the frame. When a client is an intended destination the client must try to decode the coded layer 3 datagram and then acknowledge that it has received the frame. Acknowledgment is important because all unicast data frames sent over 802.11 must be acknowledged, or else they are retransmitted until acknowledged successfully or until a maximum number of retries is reached. In 802.11, broadcast signals do not follow the same acknowledgement strategy because all clients would attempt to acknowledge at the same time and many collisions would occur, as such, 802.11 does not have acknowledgment for broadcast frames. In order to ensure data is not lost, layer 2.5 network coded frames coming from an access point to a client must be acknowledged by the intended destination clients. The client cannot use an 802.11 MAC acknowledgment to avoid collisions so layer 2.5 acknowledgments must be used. To improve throughput and reduce number of transmissions, a layer 2.5 acknowledgment will be included in the next unicast data transmission from the client to the access point. The Layer 2.5 payload will contain a single un-coded layer 3 datagram.

The layer 2.5 acknowledgment frame will have a layer 2.5 header as follows in order:

- 2 bytes for ACK number field sequence control field taken from the frame being acknowledged
- 0 to 2302 bytes of payload for layer 3 un-coded datagram

The layer 2.5 frame is then encapsulated by a logical link control header as specified in section 3 of this document. The logical link control and layer 2.5 frame is encapsulated in the payload of an 802.11 MAC frame. The MAC frame is given the MAC address of the access point for Address 1. The MAC address of the client sending the frame is placed in Address 2. Address 3 contains the BSSID of the 802.11 network and Address 4 is not used. The frame type is set to be a data frame and the remaining fields are set as specified in IEEE 802.11 depending on the network configuration.

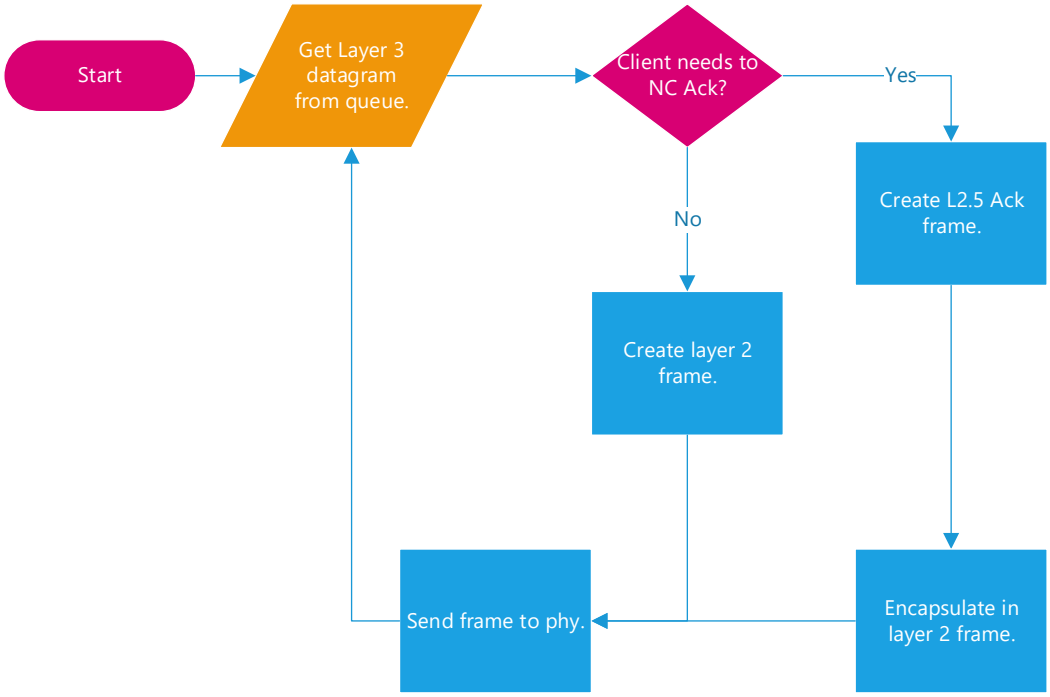


Figure 15: Layer 2.5 Client operation flowchart.

Figure 15 shows a flowchart of the operation of the transmit loop for a client using layer 2.5 network coding. The client is responsible for determining if it needs to send a frame with a layer 2.5 Ack piggybacked. An Ack is only required if the client has received a network coded frame that it has not Acked yet. If an Ack is not required, then the data can be sent as a typical 802.11 frame. When a client with layer 2.5 enabled connects to an AP it must immediately send a L2.5 ACK frame with a SEQ number of 0 and an empty data field to inform the AP that it is a L2.5 client.

2.5.5 *Layer 2.5 Acknowledgement*

Only the third proposed Ack strategy from section 2.4 will be discussed in depth in this document. Option 1 is to simply never acknowledge and accept the errors. Option 2 involves immediately sending a layer 2.5 Ack with empty data field. This section will discuss the specifications for piggyback acknowledgment.

All layer 2.5 data frames must be acknowledged (ACK/ACKed) by both of the destinations of the coded layer 2.5 data frame. Each destination individually ACKs upon receiving and decoding the message within the coded payload section of the layer 2.5 data frame.

The process of a client node ACKing starts upon receiving and decoding a layer 2.5 data frame and is as follows:

1. The client will construct a layer 2.5 header as described in section 5 of this document for a layer 2.5 ACK frame, while leaving the payload empty and filling the Ack number field with the sequence control number from the layer 2 frame that contained the information being Acked.
2. The client will wait up to ½ the ACK Timeout parameter number ms before transmitting the layer 2.5 Ack with an empty payload.
3. If the client has upper layer information to send within the ½ ACK Timeout ms window, the client will put the upper layer datagram in the payload of the layer 2.5 ACK frame and send the layer 2.5 Ack frame as a layer 2 frame as described in section 2 of this document.
4. If the client does not have upper layer information to send within the window, then the layer 2.5 frame constructed in step 1 is transmitted as a layer 2 frame.

At the Access Point when a layer 2.5 data frame is sent to two destinations, both of those destinations are locked from being used in layer 2.5 network coding until either:

1. A layer 2.5 Ack with the correct sequence control number has been received at the access point from the destination.
2. The ACK Timeout value number of ms have passed since the last layer 2.5 data frame was sent to the destination.

Each destination associated with the access point is locked individually (two at a time though) as layer 2.5 data frames are sent to them and unlocked when a correct Ack is received from a locked destination, or after a destination has been locked for 1 second. While a destination is locked, no layer 2.5 frames can be sent with that client as its destination. Layer 2 frames sending non layer 2.5 frames can still be sent to the locked destinations. If a destination is unlocked by a timeout, the next frame that is sent to that destination must contain the upper layer datagram that was coded in the most recent layer 2.5 data frame to that destination. This is a retransmission of the information that was not Acked. The retransmission can be a unicast Layer 2 frame without a layer 2.5 data frame encapsulated, or it can be retransmitted in a layer 2.5 data frame with the upper layer

datagram coded with some other destinations upper layer datagram and sent as a layer 2.5 data frame as described in section 4.

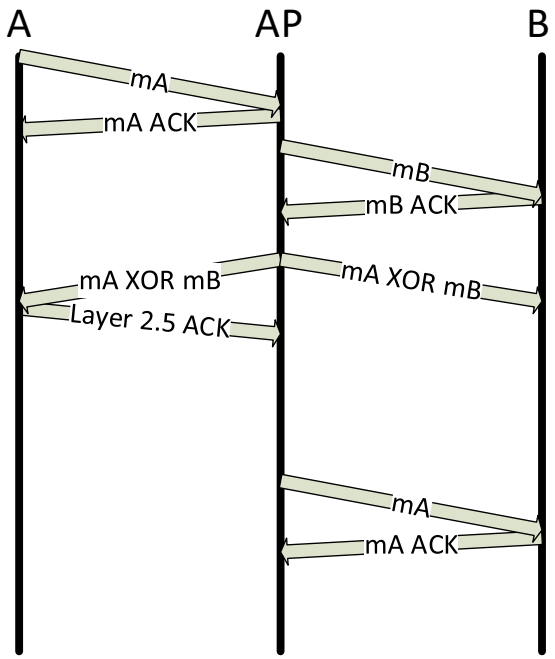


Figure 16: Timing diagram of ACK strategy with no ACK received at the AP from client B, so AP must retransmit mA to client B.

The diagram in *Figure 16* shows two clients exchanging messages through the access point. The access point then codes the two messages together and sends a layer 2.5 data frame to both clients. The clients must Ack the layer 2.5 data frame. Client B does not Ack the layer 2.5 data frame so the access point retransmits the message intended for client B after 1 second.

2.6 Access Point and Network Coding scheduling

One of the key algorithms that must be designed for this experiment is the access point scheduling algorithm. In traditional 802.11, this algorithm determines when frames will be sent from the access point and in what order they will be sent. We will add additional components to this algorithm to determine when the access point can send a layer 2.5 network coded frame encapsulated in a layer 2 broadcast frame instead of just a layer 2 unicast data frame. An example timing diagram Figure 17 shows how layer 2.5 datagrams are sent through the network under our AP scheduling algorithm. In this example, which node will win contention of the channel at any

time is decided randomly. The example shows that first client A sends two frames to the AP by winning contention twice in a row. Meanwhile after receiving the first frame the AP starts vying for contention to send the frame it received from A to B. The AP then wins contention. The AP initially would have sent a message to client A but it has no messages for client A so instead sends a message to client B. The AP receives a few more frames from clients A and B and can wait up to the Frame Wait Time parameter before trying to send another unicast frame for a network coding opportunity to appear. After receiving a transmission from both clients A and B, the AP now has a network coding opportunity, so it combines the messages and sends a L2.5 NC broadcast frame to the layer 2 queue which will then contend for channel access and be sent when the AP wins contention, in our example case this happens immediately. Now that an NC broadcast frame has been sent, each client must send an L2.5 ACK before coding can occur again. This is shown when the AP sends messages 4 and 5 as unicast frames instead of broadcast NC frames. Neither m4 nor m5 can be sent as coded frames because both clients must send a L2.5 ACK before they can be used as an NC destination again. After client B has sent m7 with a piggyback L2.5 ACK, the AP could send another NC broadcast frame to clients A and B, however, in this example, the AP does not have another message to code with m7 before the example ends.

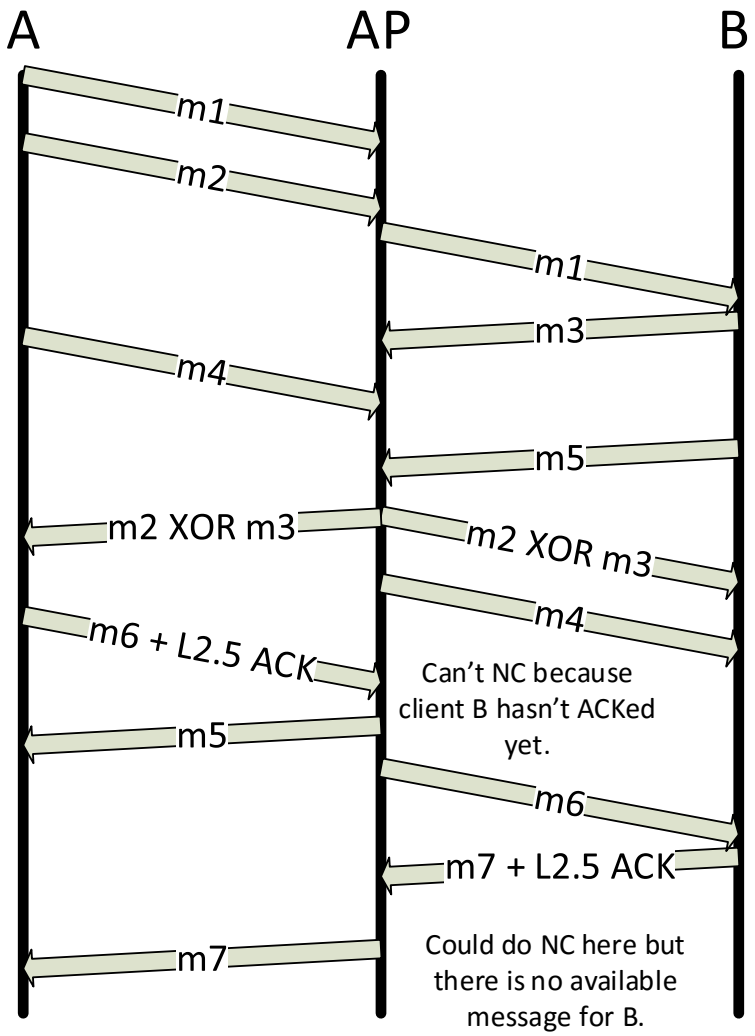


Figure 17: Timing diagram of Layer 2.5 messages moving in a 2 client 1 AP network.

The basis for this algorithm is a traversal of a circular linked list containing nodes for each client associated to the access point. For simplicity we implement the AP scheduling by having the destination client nodes will be visited in a round robin fashion. Each will be visited once in order until the last client node is visited and then the next node visited will be the first client node again. There exist many implementations of more complicated AP scheduling functions which can prioritize certain types of traffic flows under some conditions to improve usability of an 802.11 network, however, we are will not be looking in to scheduling in this thesis. Each client node is a data structure as shown in Figure 18. The data structure contains a queue of frames to be sent to the client that this node corresponds to, a list of SEQ numbers associated with the frames in the queue as well as those frames source client. These SEQ numbers and source client are remembered by the AP from the layer 2 header when the frame was received. This is shown in Figure 18 where

the access point remembers the source address and SEQ number and are placed in the data structure once the layer 3 header is decoded to find the destination for the frame, which corresponds to the client node that the current frame will be added to. Additionally, the data structure has fields for holding information regarding acknowledgment if it is being used. The first ACK field is a two bit flag field that holds the current ACK state of either waiting for ACK, positively ACKed, or negatively ACKed. The next ACK field contains the SEQ number of the last NC broadcast frame sent to the client. This field is used for checking against the SEQ number in a piggyback ACK sent from the client. If the SEQ fields match, it is a positive ACK. The final ACK field holds the entirety of the last frame that was sent as a NC frame in its un-coded form. This field is used in the case of a timeout or negative ACK to retransmit the data. The data structure also contains a queue of the sequence numbers that the client remembers for use in network coding as well as the destination of the frame with the corresponding sequence numbers. These fields are set whenever a frame is received at the access point from the current client. The queue of sequence numbers is of length M, where M is the maximum number of frames that can be remembered by the access point. Every time a frame is received, a frame from the front of the queue is dequeued and the received frame's sequence number and destination are added to the end of the queue. This field is used to find a frame that is compatible for network coding with the client and frame we are currently on.

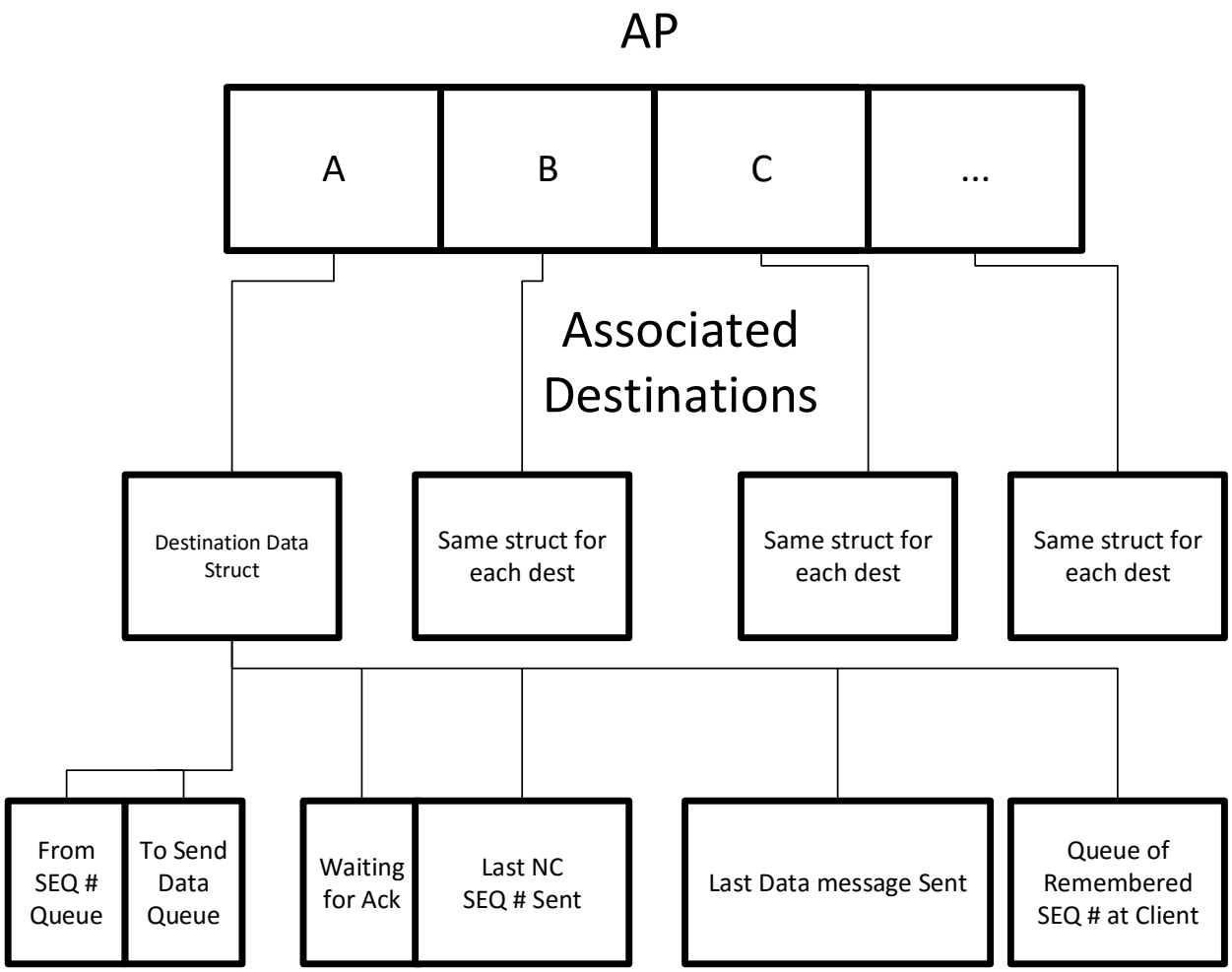


Figure 18: The AP data structure with one client node data structure expanded.

When a client node is visited by the algorithm there are three possible outcomes shown in Figure 19. First, the node can be skipped with no transmission. Second, the access point will transmit a unicast 802.11 data frame to the client. The final possibility is that the frame to be sent from the current client will be coded with a frame coming from the current client and going to the source of the current frame to be sent to create a NC broadcast frame to transmit. The first outcome happens only when the AP data structure for a destination client has an empty transmit queue. This means that the access point does not currently have any data destined for the current client. In this case, no data will be transmitted to the current client and the algorithm will visit the next client node in the AP data structure list immediately. The second scenario occurs when we do have data in the transmit queue or the transmit queue is empty, but a recent NC frame to the current client was not ACKed before its ACK timeout expired. If the ACK status shows waiting for ACK, the access point will immediately take option 2 with the frame in the front of the transmit queue. Additionally,

if the access point cannot find a valid frame to code the current frame to transmit with within the Frame Wait Time, then the access point will take this option and send the frame in unicast mode.

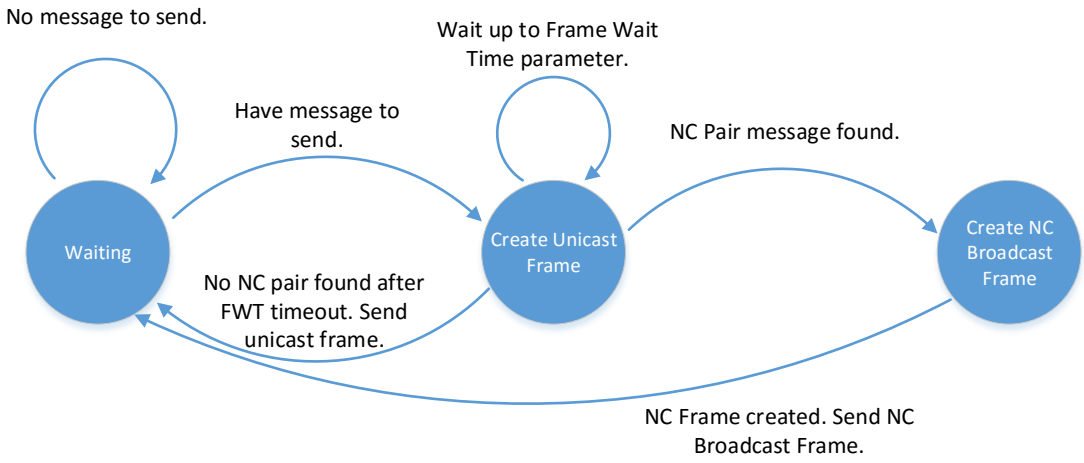


Figure 19: State diagram of Access Point possible outcomes of visiting a client in the AP data structure.

The third state, Create NC Broadcast Frame, is the network coding option unique to this experiment. This option will only be taken when all of the following criteria for establishing a valid network coding frame pair has been met. The first check for network coding is the ACK status of the source and destination clients of the current frame to send. If either are waiting for an ACK then network coding cannot occur and option 2 is taken instead. If both are not waiting for an ACK then the access point will next check the remembered frames field of the current client node to see if the client remembers any frames with a destination address equal to the source address of the current frame to send. If this condition is met, then the next field to verify is that the destination of the frame to send currently remembers it. This is checked by accessing the remembered frames field of the client node for the source client. If the SEQ number of the current frame to send is not in its remembered frames SEQ numbers, then network coding can't occur and the frame to send will be unicast. If all of previous criteria for network coding is met, then a network coding broadcast frame can be created from the 2 frames that we have found to be a valid NC pair. The 2 frames will then be coded together and the layer 2.5 header including the destination address of both frames and SEQ number of both frames will be created. This will then be encapsulated by a layer 2 broadcast data frame that will be transmitted when possible.

2.7 LAYER 2.5 IMPLEMENTATION

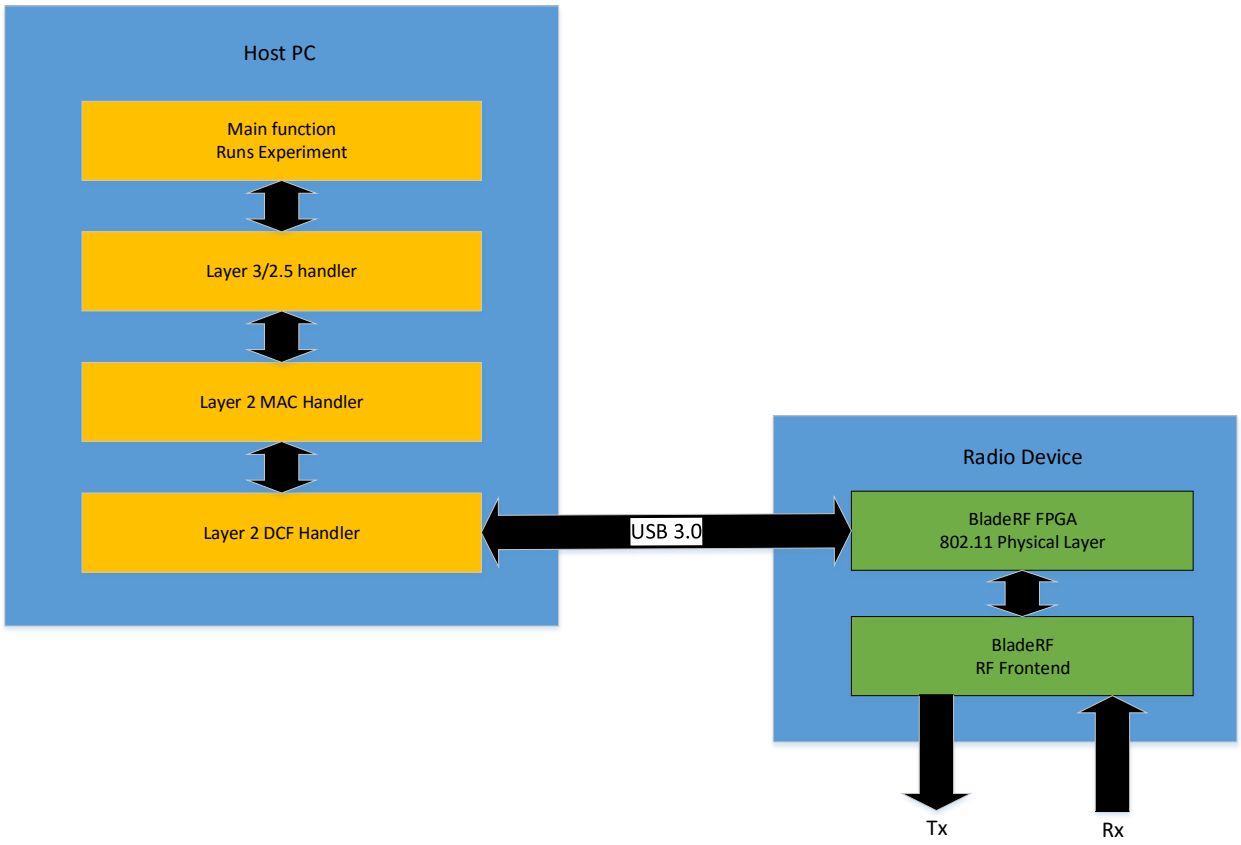


Figure 20: Software architecture showing the structure of implementation of network layers.

Our layer 2.5 system is implemented on an 802.11 layer 1 and 2 to verify that layer 2.5 as specified in this document is compatible with the 802.11 MAC. To have the desired level of control over all aspects of the communication system the University of Washington Fundamentals of Networking Lab(FUNLab) has implemented the 802.11 layer 1 using the Nuand BladeRF software defined radio. Layer 1 and 2 were mostly implemented by others in the UW FUNLab and are not the main focus of this document. For this experiment layer 2.5 was implemented independently to run using the layer 2 interface provided by FUNLab with a few modifications to the lower layers in order to add functionality not included in the previous FUNLab experiments and to add SEQ number passing up from layer 2.

Layer 1 is mostly implemented directly on the radio's FPGA which communicates back to a host PC through USB 3.0. The 802.11 layer 2 is implemented on the host PC in 2 distinct sections. The lower part of layer 2 is the DCF controller which runs the 802.11 distributed coordinate function(DCF) rules and communicates with the physical layer on the FPGA to determine when a

message is being received and when the communication channel can be accessed for sending frames. The DCF layer on the host PC determines when the physical layer is allowed to transmit. This is currently an issue with the FUNLab implementation of 802.11 as the DCF needs very quick access to the physical layer in order to function properly, but in the current implementation the DCF on the host PC must wait for data from the physical layer to be sent via USB and scheduled by the host PC operating system.

The second portion of layer 2 is the MAC layer. The MAC code controls how the nodes talk to each other on a network. The MAC layer complies with the 802.11 media access control(MAC) rules but does not implement the 802.11 MAC fully. Not all pieces of the 802.11 MAC are required for our experiment because we are only creating a predetermined network amongst our own radios. Because our network can be initialized by hand, the MAC implementation does not have the functionality required for a user to identify or connect to the network (ie: beacons and SSID). The MAC code controls the creation of layer 2 frames and, once a frame is constructed, it sends the frame directly to the DCF code so that the DCF controller can determine when layer 1 should send the frame. The MAC layer accepts data and a destination address to send a frame in the 802.11 network. The MAC layer determines the rest of the fields of the MAC header as shown in Figure 4 but layer 2.5 requires knowledge of the sequence number that the MAC code chooses for a given frame. In our implementation of the MAC there is added functionality for layer 2.5 to find the sequence number of the most recently sent frame.

Layer 2.5 is implemented in two main pieces, the client layer 2.5 controller and the access point layer 2.5 controller. Unlike layer 2 and layer 1 the client and the access point have very different functionality in layer 2.5 so the code is implemented independently and the controller required for a node in the network is specified on boot of the node. In addition to implementing all of the layer 2.5 requirements as specified in section 2.6 on page 19 of this document layer 2.5 also implements the logical link control header specified by IEEE 802.2 and implements a small portion of layer 3 for routing messages from 1 client to another through the access point. Both layer 2.5 controllers are implanted on the host PC and run as a controller that interfaces with the layer 2 code, which then interfaces with layer 1 on the radio.

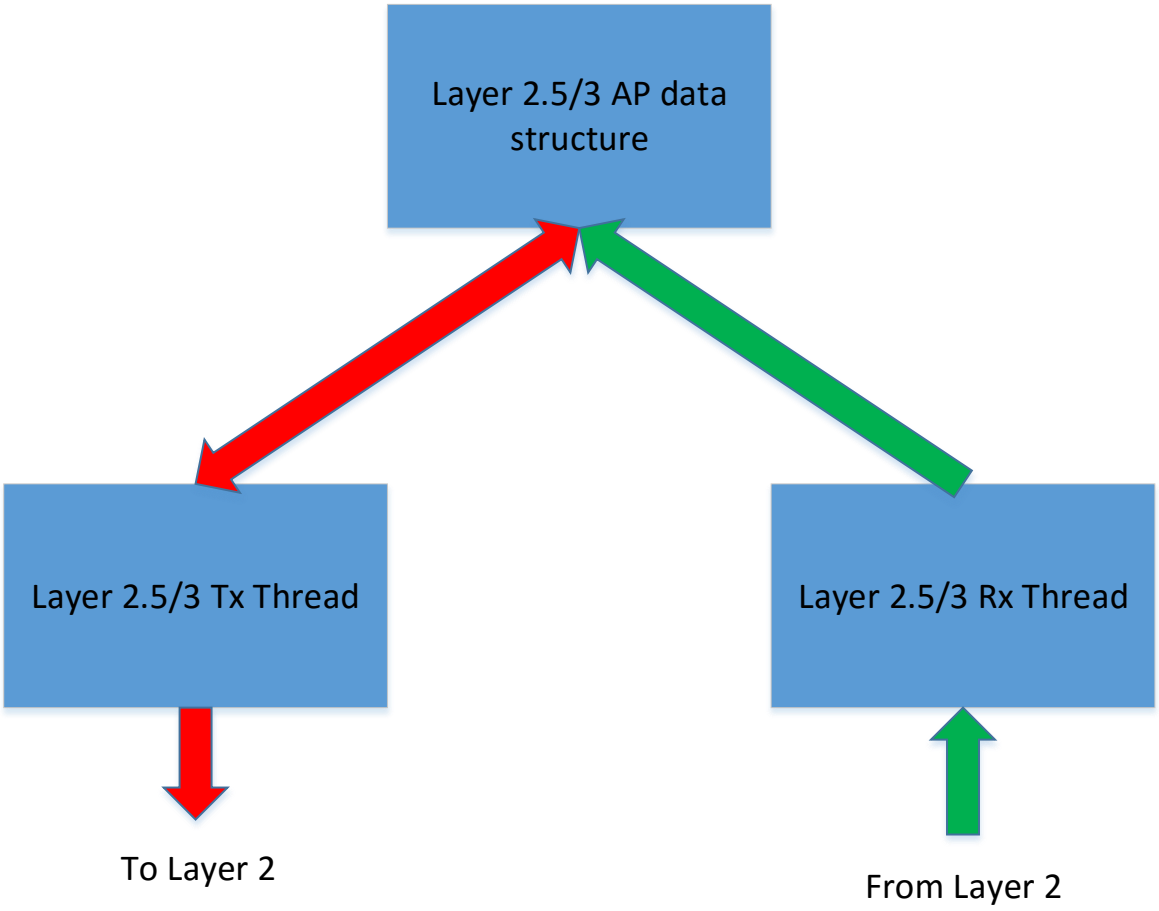


Figure 21: Data flow in layer 2.5 and 3 access point implementation.

For the access point layer 2.5 controller the main responsibility is managing when frames will be sent, what frames will be sent, determining whether or not to code frames, and creating layer 3 and LLC header information for the frame to be sent. Layer 2.5 must manage incoming frames by sorting them in to their intended destination as well as remembering pertinent information for coding such as a frame's layer 2.5 sequence number as it is received. Layer 2.5 controller is also responsible for checking each message received for a network coding ACK and managing network coding ACK timeouts for reliable data transfer through broadcast frames. When the layer 2.5 controller sends a frame it will first determine which frame to send, then check if a frame can be coded with that frames. Next the controller will fill the layer 3 header information of the frame and the data field of the frame with either coded or uncoded information. The layer 2.5 controller finishes creating a frame by filling in the LLC header with either typical layer 2 information or layer 2.5 information and creating a layer 2.5 header if necessary. The frame will then be pushed down to the MAC controller.

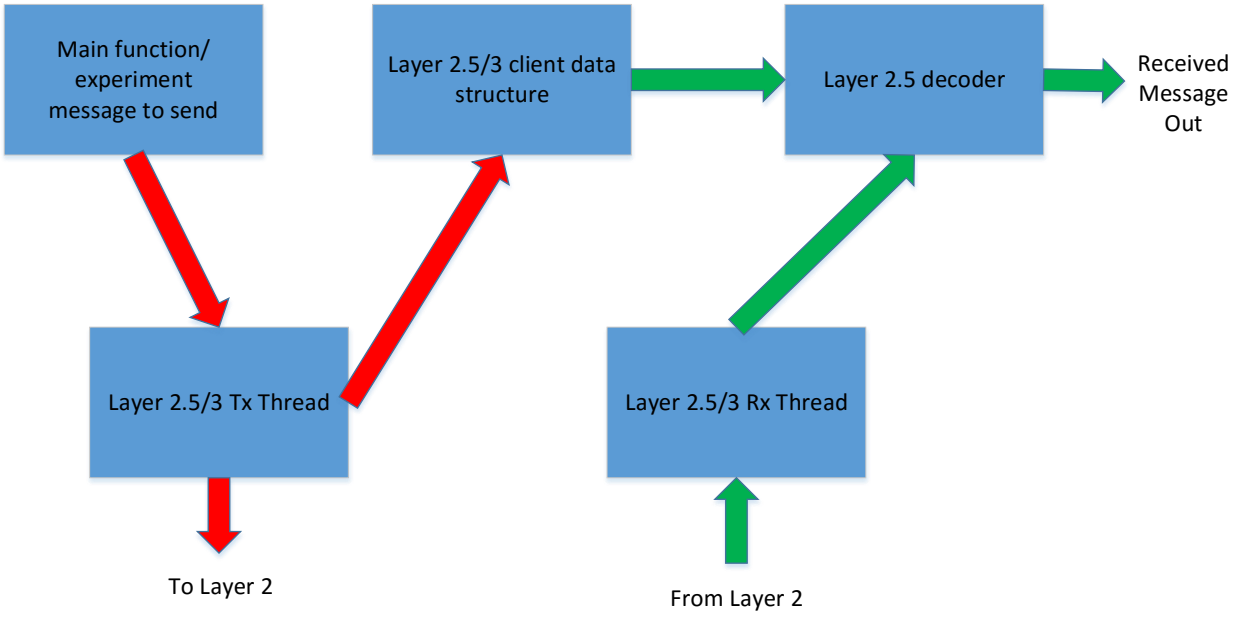


Figure 22: Data flow in layer 2.5 and 3 client implementation.

The layer 2.5 controller on the client plays the same role as the controller in the access point but the function it performs is somewhat different. The controller in the client has the responsibility of determining when a received frame is a coded frame and decoding it, in addition to filling the LLC header and layer 2.5 header and adding the layer 3 source and destination addresses. The layer 2.5 client controller is also responsible for remembering the frames it sends and their layer 2 sequence number. The layer 2.5 client controller receives frames from the layer 2 controller, determines if the frame is coded and decodes it if required. To send a frame the layer 2.5 controller first checks if it needs to send a layer 2.5 ACK frame, then constructs the layer 3 and LLC headers. A layer 2.5 header with ACK information is added if the frame is a layer 2.5 ACK. The controller then passes the constructed frame to the MAC controller with the destination MAC address as the access point's MAC address.

2.8 Layer 2.5 Testing

In order to test that layer 2.5 is functioning properly before collecting experimental data we must design a test that shows layer 2.5 transmits data correctly and receives data correctly at the clients, remembers frames correctly at the AP and clients, and acknowledges layer 2.5 frames accurately.

We must also test layer 1 and layer 2 for accurate and efficient data transfer to ensure that issues with layer 1 or 2 are not causing issues with layer 2.5 functionality.

The initial test setup involves placing the three nodes to be used in an equilateral triangle formation. The nodes are then each assigned a designation of client 1, client 2, or access point. The initial test to be run is the layer 1 test as layer 2 will not function if there is a failure at layer 1. The layer 1 test consists of setting one node to send phy data while the other two listen and report the number of frames that were received correctly according to the frames CRC value. To pass this test both clients should be receiving almost every frame correctly. This test must be repeated two more times alternating which node is the transmitter each time. This ensures a good quality layer 1 connection between all nodes in both the Tx and Rx directions.

Once the phy test is passed, layer 2 can be tested. The nodes should all be left in the same position for the layer 2 test as the layer 1 conditions can change when nodes are moved. In the layer 2 test each node will be given a MAC address and each node will transmit frames as fast as possible to the node with the next MAC address. Node A sends to node B, node B sends to node C and node C sends to node A. The layer 2 test reports number of MAC frames that were transmitted successfully as determined by the sender receiving ACK frame. The MAC allows for retransmission of frames that were not received correctly so we also record the number of retries required to send all the frames in the test.

The final testing will verify that layer 2.5 operates correctly. This test will use the same nodes in the same position. Now 1 node will be an access point with 2 clients connected to it. The clients each send a specific message destined for the other but must send this message through the access point. The access point will relay messages and when possible do layer 2.5 network coding on the messages. To verify this system works, we check that network coding does occur by keeping track of how many times the access point encodes frames. We then verify at each client that the message received decodes to the correct message that was sent by the other client. Each message consists of “From” followed by the original senders first MAC address byte followed by a counter value. We also keep track of how frequently the access point receives network coding ACKs, and how often there is a network coding ACK timeout to ensure these systems work correctly.

Chapter 3. EXPERIMENT

3.1 Experiment Design and Principles

In Chapter 2 of this thesis we have developed layer 2.5 network coding, explained its integration with 802.11, and shown our plan for implementing and testing layer 2.5 to improve the efficiency of a network. In order to show that layer 2.5 network coding is effective we must create an experiment that utilizes layer 2.5 network coding, emulates a scenario that exists in a typical 802.11 network, and can be measured. We design an experiment with two clients A and B and an access point, with the 802.11 network set to infrastructure mode. The experiment is designed to emulate client A having a large file to share with client B and client B having a large file to share with client A. Each of the clients A and B can't communicate directly with each other, so they must send data to the other client through the access point as shown in Figure 23. The access point should have many opportunities for network coding.

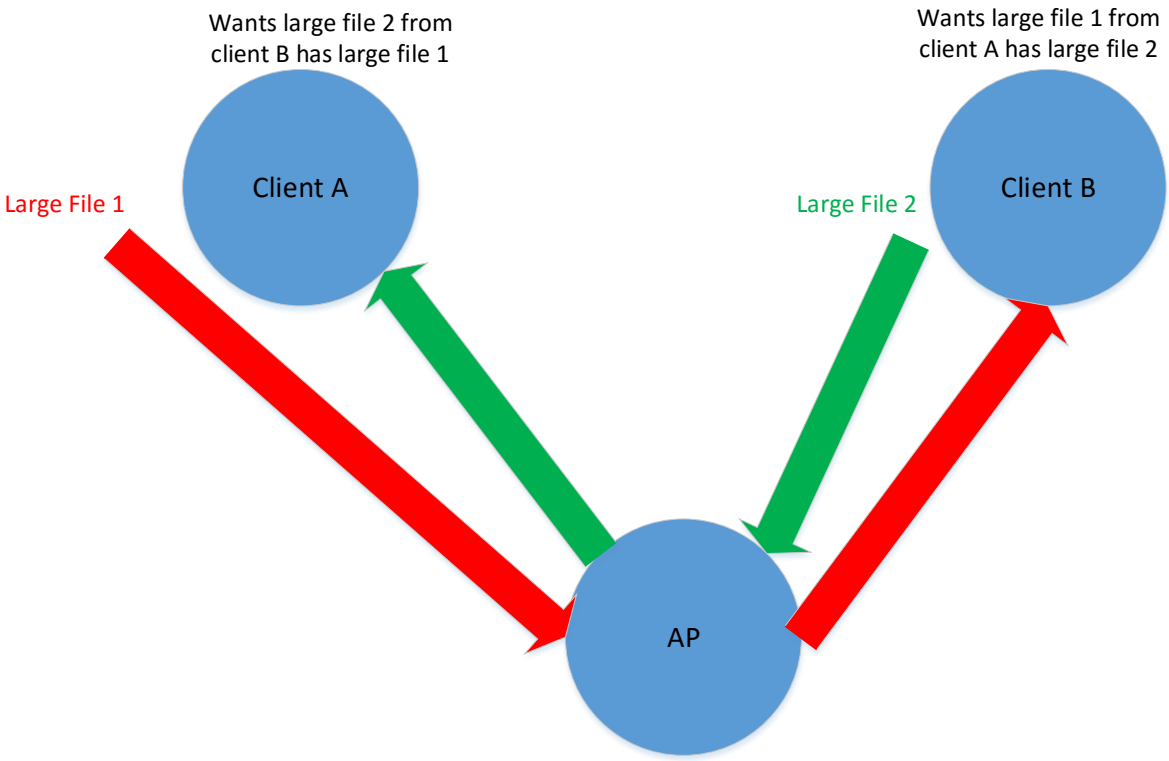


Figure 23: experiment design

In our experiment, we will be sending a specific message instead of a large file but the behavior in the network is the same. Client A will send the message "From AA #" to client B where # is a

counter that increments each frame. Client B will send “From BB #” to client A. Each of these messages must pass through the access point and the access point will perform network coding whenever possible. The clients will be responsible for sending NC piggybacked ACK frames on messages after receiving a coded data frame as specified in section 2.5. Each node in the network will be using the same MAC, DCF, and physical layer implementation as discussed in section 2.7. The clients will constantly be attempting to send their message whenever they win contention, while the access point will only contend for the channel when it has a message to send. No data will be introduced at the access point. All nodes will be hard coded to use MCS 0 ensuring that all nodes will send at the same data rate. This experiment will show how effective layer 2.5 NC is under the conditions for which it is specifically designed to improve but with the real world randomness of 802.11 interference, DCF, and scheduling introduced.

While the experiment is running, the access point keeps track of 2 values. The access point counts the total number of frames that it has sent, and the total number of layer 2.5 NC data frames it has sent. Whenever the AP sends a coded frame it is assumed that this is equivalent to a legacy 802.11 network having sent 2 frames. We can use these values to find the ratio of frames that were sent in the network coded scenario to the number of frames a legacy network would have used to send the same amount of information. The ratio of frame reduction is given below in terms of the number of coded frames and number of uncoded frames sent by the access point.

$$Reduction = \frac{\#Coded + 2X\#Uncoded + 2X\#Coded}{2X(2X\#Coded + \#Uncoded)} \quad \text{(Equation 1)}$$

The efficiency gain caused by layer 2.5 network coding can then be calculated using the below equation.

$$Efficiency\ Gain = 1 - Reduction \quad \text{(Equation 2)}$$

The efficiency gain can range between 0.0 and 0.25 with larger numbers indicating that layer 2.5 network coding is making a larger positive impact on the network.

There are two tunable parameters designed in to the layer 2.5 network coding system. We have designed two experiments that will be run independently to test the effectiveness of layer 2.5 network coding and how the two tunable parameters effect how well network coding works.

Our 2 system parameters for L2.5 network coding are:

- NC ACK timeout: The amount of time in milliseconds the AP will wait before retransmitting a message that was sent as a layer 2.5 coded message and has not been acknowledged by its destination client.

- Frame Wait Time(FWT): The time window in milliseconds in which a network coding opportunity can occur. If the AP does not find a network coding opportunity when it creates a frame to send, it will wait up to this amount of time for a new frame to arrive giving the AP a network coding opportunity, before passing the original message down to layer 2 for a unicast transmission. If a network coding opportunity becomes available at any point during this window, the AP immediately takes the NC opportunity and passes the L2.5 coded frame down to be sent as a layer 2 broadcast frame.

3.1.1 *Experiment 1*

In our experiment we will do several trials collecting data with a few different sets of parameters. The first experiment will focus on changing the first tunable parameter the NC ACK timeout value, which is the number of milliseconds the AP will wait to receive a layer 2.5 ACK from a client after sending it a layer 2.5 coded data frame. If the AP has not received a layer 2.5 ACK in this time, the AP will assume the client did not receive the frame and will retransmit it as discussed in section 2.5. We will increase the NC ACK timeout parameter and record the number of frames that are coded at the AP over several hundred total frames sent by the AP. We then calculate efficiency gain using equations 1 and 2 and take the mean efficiency gain over several trials. We will use this experiment to investigate whether or not NC has a significant impact on network efficiency and how the NC ACK timeout affects the efficiency gain from network coding.

3.1.2 *Experiment 2*

The second experiment will investigate the effect of the second parameter, the Frame Wait Time parameter. In this experiment, we will set the second parameter, the Frame Wait Time value, to be non-zero. Unlike the NC ACK timeout this parameter adds a latency that does not exist in traditional 802.11 access points. By introducing this latency, the ratio Reduction is no longer a direct comparison of how much more efficient 802.11 with layer 2.5 NC is, as the AP will waste time waiting for NC opportunities. In order to determine how the parameter affects the efficiency of a NC network we must use a metric other than the Reduction ratio from equation 1. We will determine the throughput of the system by measuring the rate at which the AP receives frames from each client and sends frames. The throughput, unlike the Reduction ratio, is not a direct comparison to a network with and without NC so we must collect throughput data without NC to

compare to the throughput with NC and with the Frame Wait Time parameter set to several values including the baseline value of 0ms which is identical to the setup of experiment 1. Throughput, being the measure of how fast data flows through the system, will show us whether NC is actually an improvement over traditional 802.11 no matter the latency added by the NC algorithms. We will separately calculate the throughput of the clients as well as the access point. We calculate the AP throughput separately to ensure that the NC algorithm and parameters are not causing the throughput specifically from the AP to drop significantly. It is possible that increasing Frame Wait Time would improve the overall network throughput but lower the throughput of data sent by the AP. This may not be desirable if the AP throughput drops significantly as the AP needs to be able to send the data it receives in a reasonable amount of time. The AP will calculate both throughput values. The AP throughput will be calculated as shown in Equation 3 and the combined throughput of the clients will be calculated at the AP as shown in Equation 4.

$$Throughput_{AP} = \frac{(\#uncoded\_sent + \#NC\_ACKS\_Received) * 2000}{Time\_Elapsed\_seconds}\ bps \qquad \text{(Equation 3)}$$

$$Throughput_{Client} = \frac{(\#Frames\_Received) * 2000}{Time\_Elapsed\_seconds}\ \text{bps} \qquad \text{(Equation 4)}$$

3.2 RESULTS

Figure 24: Picture of experiment set up. Radio 1 and 3 are clients, Radio 2 is AP.

Our experiment is set up with each of the three nodes as a point on a triangle as shown in Figure 24 and each node connected to its own host PC. The 3 nodes each run a different function in the experiment code. The first node is client A and runs layer 2.5 client code with MAC address AA-AA-AA-AA-AA-AA and layer 3 address 1. The 2nd node is client B and also runs client layer 2.5 code with MAC address BB-BB-BB-BB-BB-BB and layer 3 address 2. The 3rd node runs layer 2.5 and layer 3 code specific to the access point and has MAC address 00-00-00-00-00-00. The AP does not have a layer 3 address in our experiment. Client A is set to send messages to layer 3 address 2 and client B is set to send messages to layer 3 address 1. All nodes are set to wait one millisecond after sending a message before trying to transmit another message. This delay is introduced to help mitigate the DCF implementation issues mentioned above.

3.2.1 *Experiment 1*

Data is collected at several different values of NC ACK timeout parameters. The first experiment is run with ACK Timeout at 300 ms and it is increased by 100 ms in each successive set of trials until we reach 600 ms. At each value of NC ACK timeout 10 trials of data are collected and aggregated in the table below.

ACK Timeout	300 ms	400 ms	500 ms	600 ms
Number of trials	10	10	10	10
Total Frames Sent	4293	4311	4589	4383
Mean Efficiency Gain	11.55%	11.30%	10.72%	10.64%
Std Dev Efficiency Gain	.44%	.54%	0.85%	.95%

3.2.2 *Experiment 2*

For experiment 2, we fix the NC ACK timeout parameter at 300 ms shown to be the most effective value in experiment 1. We then vary the Frame Wait Time parameter starting at 0 ms, then 5, 10, 20, and 30 ms and measure the Mean throughput of the system over several trials. We will also measure the throughput of the system when NC is not used. The non-NC throughput data will give us a baseline to compare how utilizing network coding effects the throughput of a network. We then compare how changing the Frame Wait Time effects throughput and efficiency gain.

Frame Wait Time	Non-NC	0 ms	5 ms	10 ms	20 ms	30 ms
Number of Trials	10	10	10	10	10	10
Mean AP Throughput	48316 bps	67097	64685	63152	61489	54943
Mean Client Throughput	101151 bps	102633	105133	108201	110720	110814
Total Frames Sent	4010	4010	4010	4010	4010	3010
Mean Efficiency Gain	0	11.7%	13.3%	14.6%	16.8%	18.7%
Mean Total Throughput	149467 bps	169730	169818	171354	172210	166449

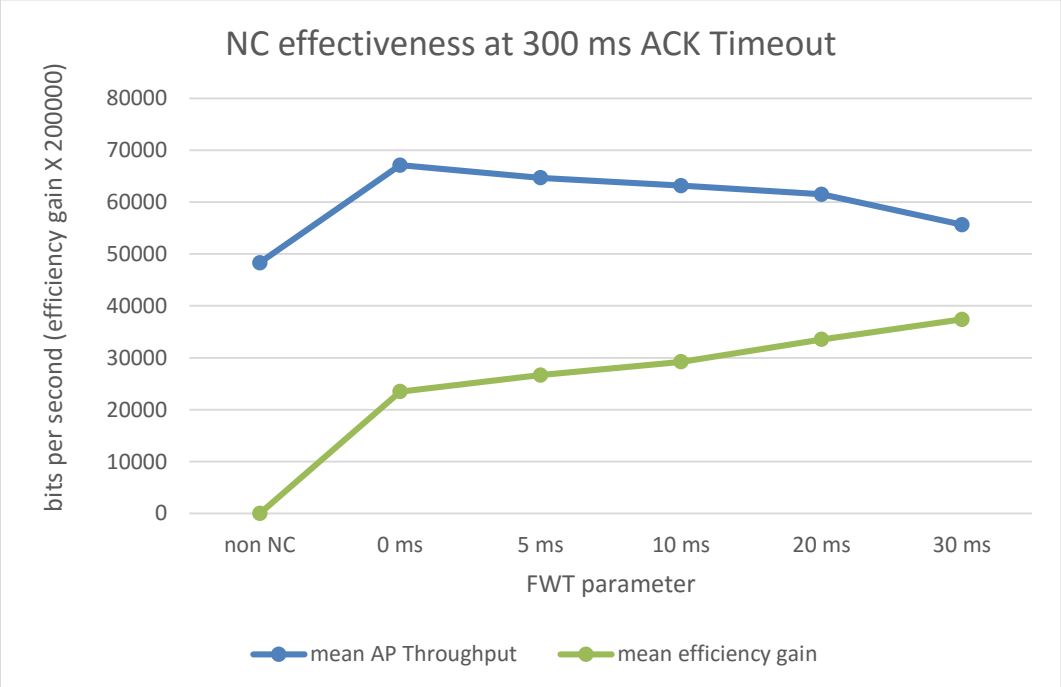


Figure 25: Experiment 2 results as FTW parameter changes.

3.3 Discussion of Results

In a perfect scenario with non-random channel access and no errors, scheduled such that client A sends a message, then client B sends a message, then the access point sends a message, it is possible to achieve a rate of 100% frames coded. This is possible because each time the access point gets a chance to send a frame it will always have exactly one message from each of the two clients, allowing it to network code those two frames together and send a broadcast layer 2.5 data frame. Then client A will send a new message along with a piggyback layer 2.5 ACK as will client B. Since client A and B have both ACKed and there is a new message in the access point data structure for each of them the access point can do network coding again. When 100% of frames sent by the access point are coded we have the result from our reduction in frames formula of $\frac{3X\#Coded}{4X\#Coded}$ which results in a reduction to 75% of the frames required to send the same amount of information in a traditional 802.11 network.

A 25% reduction in frames sent in an 802.11 network could be a huge boost to throughput especially in high density deployment areas where collisions happen very frequently. Unfortunately, any real world scenario will not be likely to approach a 100% coding rate as traffic patterns are not often going to be in the perfect scenario. This is why we must run experiments with actual 802.11 networks to see how effective coding is in more of a real world scenario. We must also consider how real world 802.11 traffic patterns will interact with layer 2.5 network coding. In many home use 802.11 networks most traffic is sourced from the access point. Our experiment does not introduce traffic from the access point as layer 2.5 network coding can't be done on that traffic. While layer 2.5 network coding will not cause any side effects on traffic that is sourced at the access point we will also find that there is no reduction in frames required to send information sourced at the access point by using it. Layer 2.5 network coding works best in high rate symmetric traffic flows between 2 clients connected to the same 802.11 access point. There are many real work applications where this traffic pattern does occur (file sharing) but activities like internet browsing will be unaffected.

Achieving a 25% efficiency gain is not possible under the randomness of normal 802.11 DCF. We would expect to achieve some significant efficiency gain that varies from trial to trial but not ever reaching the maximum efficiency gain. The experimental data shows that we have achieved a significant efficiency gain in all of our trials. The resulting best mean efficiency gain of 11.55%

with NC ACK timeout at 300 ms from experiment 1 is a very significant improvement to Wi-Fi. While 11.55% is less than half the maximum efficiency gain, it is expected that as the FUNLab 802.11 implementation is improved and DCF timing issues resolved, the experimental efficiency gain will increase further.

Experiment 2 shows us that Network coding does positively increase the throughput of an 802.11 network. We can see that the mean total throughput of our experimental network goes from 149467 bits per second to 169730 bits per second by adding network coding with 0 Frame Wait Time. This is an increase of 13.6% in throughput which is even larger than the 11.7% efficiency gain from the same data set. We can further see the effect of NC on throughput by looking at the client and AP specific throughputs. With 0 Frame Wait Time the client throughput is nearly unaffected by adding NC to the system which is to be expected because NC should not negatively impact the clients and coding is only done at the AP. The AP throughput shows a huge jump when NC is added clearly showing that the AP coding frames allows the AP to send data to clients more quickly and efficiently.

Experiment 2 continues by exploring the throughput as Frame Wait Time(FWT) is increased. Increasing Frame Wait Time does significantly increase the efficiency gain as expected, however, because having FWT introduces latency at the AP we can no longer compare how well the system works using efficiency gain. We look instead to the client, AP, and total throughput of the system as we increase FWT. We have shown that as FWT increases, the client throughput generally increases while the AP throughput decreases. The client throughput is explained by the clients sending more frames during the time interval that the AP is waiting for matching NC frames and not contending for the channel. Meanwhile the AP loses throughput while it is waiting for frames. The important thing to note is that the total throughput does increase, however, this is not necessarily desirable. The total throughput increasing does mean more data is being sent through the network in less time, but, the AP throughput decreasing means that less data is actually being delivered to its end destination in the same amount of time. All data must travel through the AP, so AP throughput is the limiting factor in delivering data to its endpoint. We suggest using FWT of 0 ms which results in a large gain in total throughput over non-NC and a higher AP throughput than larger FWT values. This paper does not investigate how efficiency gain may be more effective in different scenarios, however, it is possible that increasing efficiency gain has desirable effects, in which case there may be scenarios where increasing FWT is useful even though it may lower

AP throughput. For example, in a high density deployment area where there are many collisions, increasing the efficiency gain may have a greater effect than in our experiment where there is only the one experimental network operating in range of the experiment.

From the collected data, we can see there is significant sources of variance in the experiment. The variance is caused by randomness in the nodes on the network. Each node contends for the physical layer channel and the nodes should win in an approximate uniform distribution. Unfortunately, there is instability in layer 2 timing, and the DCF carrier sense multiple access algorithm which causes further randomness in the DCF contention. This variance will cause the ratio Reduced to vary significantly from trial to trial which is why we take the mean value over 10 trials for each set of parameters.

One issue that has been discovered with the FUNLab 802.11 implementation is that the layer 2 protocol has significant timing issues when more than two nodes are in a network. Once a 3rd node is added to a network the DCF timing issues will frequently cause one or two nodes to dominate the channel while one node is left unable to gain access to the physical layer medium. When one node cannot win contention of the channel we will find that the number of network coded frames falls off dramatically. For network coding to take place we must have an available frame from both clients and we also need to be receiving new messages from each client to acknowledge network coded frames. When one node is being dominated it will not be able to send new messages to the access point meaning it can't ACK network coded frames nor can it provide new frames for the access point to use for network coding. In this case we will find that only one client is receiving messages frequently and the percentage of frames that are coded will be near 0% and therefore the efficiency gain will also be 0%. We have mitigated the impact of the issue in our experiments by introducing a one ms wait after a node sends a frame before it can send another frame. This one ms wait time ensures that no node will ever win contention two times in a row. While this is not exactly how traditional DCF works, it is much better than the alternative of one node winning many times consecutively. The wait time of one ms is short enough that after one frame has been sent by another node the node that was waiting will vie for contention again.

Chapter 4. CONCLUSIONS AND FUTURE WORK

4.1 CONCLUSIONS

In this thesis, we first discussed the applications and possible benefits of performing MAC layer network coding. We then determine that while coding in the MAC layer may be effective, it leads to backwards compatibility issues with legacy 802.11 nodes. We designed a system that fits between the MAC layer and the network layer, which we call layer 2.5. By creating layer 2.5 for network coding we have developed a system that can hopefully be implemented and updated on to any 802.11 Wi-Fi hardware via a manufacturer software update, without disrupting operation of legacy 802.11 systems. The design of fitting layer 2.5 in to 802.11 is described and the specifications for implementing layer 2.5 network coding are developed. Our layer 2.5 NC system was implemented on an experimental 802.11 radio network. We developed an experiment to show that layer 2.5 NC is effective in improving the efficiency of traffic flowing between two nodes in a wireless network. Our experimental results show that layer 2.5 network coding can significantly improve the efficiency of an 802.11 client to client communication by reducing the number of frames the access point must send to distribute some amount of data. Our design includes two tunable parameters, NC ACK Timeout and Frame Wait Time, and experiments showed that network coding is most effective with NC ACK timeout at 300 ms and Frame Wait Time 0 ms. With the optimal parameters, on average, layer 2.5 network coding has shown to improve the efficiency of the network by 11.7%, and the throughput by 13.6%, showing that wireless network coding is effective and may be considered as an extension to the 802.11 standard and deployed to consumers as a software update to clients and routers.

One drawback to deployment of L2.5 NC is that both clients and AP need to be updated, but many users do not have clients and Aps manufactured by the same vendor. Deployment will require either multiple vendor's involvement, or a vendor that has market share in both 802.11 clients and APs for significant improvements to be realized. An additional drawback to L2.5 NC is that the use of broadcast frames for sending data may negatively impact the battery life of low power devices as they must briefly check broadcast frames for information they might require, not knowing that they are L2.5 NC frames destined for other users. Layer 2.5 serves as a partial solution to the inefficiencies of 802.11 when 2 clients on the network want to communicate with

each other. Wi-Fi Direct is another technology recently released to improve communication directly between 2 devices, however, Wi-Fi direct only solves this issue for temporary communication between only 2 devices as a new network must be set up each time 2 devices want to communicate. Layer 2.5 allows improved efficiency on a pre-existing network without changing network setup when you want to communicate with your other device, while also allowing simultaneous access to the WAN and connection to other clients, which Wi-Fi Direct does not allow.

This thesis only covers the design and a few simple experiments using our designed layer 2.5 network coding system. A huge source of issues in our experiments, and in designing further experiments is the timing problems with DCF described in section 2.7 and section 3.1. While we were able to avoid significantly impacting our results using a delay after transmission to avoid domination by 1 node, we were unable to resolve this issue for more complex networks. As the FUNLab 802.11 implementation is improved these problems should be resolved. Once the timing issues are fixed, there will be many opportunities to further stress the layer 2.5 network coding design. We are currently limited to a network with only two clients and one AP. In the future it will be possible to design experiments to test our layer 2.5 design with more clients. Currently there is one flow of data, back and forth between clients A and B. An important future experiment will be to add more pairs of clients with data flows through the same access point. First the system would be scaled to five nodes, one AP and two pairs of bidirectional data flows from clients communicating back and forth. This can then be scaled to add even more pairs of clients. The hypothesis is that layer 2.5 network coding can still improve efficiency over typical 802.11 as the number of client pairs scales up. Additionally, an experiment should be designed to test layer 2.5 network coding when there are not specific pairs communicating but a group of N clients communicating randomly amongst themselves. This experiment could start at with three clients randomly sending messages to each other and further increasing the number of clients to show how layer 2.5 network coding scales with more clients not in a bidirectional communication pair. In addition to exploring new experiments, the original experiment should also be run using the new implementation of 802.11. Once the timing issues are solved, the experiment can be rerun in a more realistic simulation of an 802.11 network to show that our results would still be valid on consumer 802.11 hardware that is not subject to our implementation’s DCF issues.

4.2 FUTURE CONSIDERATION

In the future the code produced for these experiments can be used as a basis for further experiments into network coding that are not considered in this thesis. Some ideas for future investigation are using multicast groups instead of pairs, and investigating network coding under less favorable network conditions. In the experiments reported in this thesis only pairs of frames are coded together using a simple XOR function. In a multicast group, instead of coding 2 frames together, we can code 3 or more frames together and send sets of combinations of frames. It may be that additional efficiency gain can be achieved by coding more than two frames together. In this case a more sophisticated coding function and data distribution network will need to be created. However, the layer 2.5 header and how it fits in with 802.11 layer 2 can be re used from this experiment. In addition to multicast, it would be very useful to create experiments for network coding when we have network flows that are more difficult to work with. In this paper's network coding experiment, it is assumed that all 802.11 frames contain the same amount of data. This makes it much easier to code frames together as the frames are the same length. With differing lengths of data, there may need to be additional coding rules so that the receiving clients know how much data is meaningful. This experiment also assumes that the transmission rate (modulation used) is the same for all connections. This may not be the case in some network configurations as a node with a weaker signal will use a different modulation and therefore transmit data more slowly. This can affect how network coding performs and how to avoid any penalty from this effect when using network coding should be investigated. We also should conduct experiments in which not all data is cross network traffic. We Future experiments can inject WAN traffic at the AP in order to see how effective L2.5 NC is when not all traffic can be coded because it did not originate from a client on the network. Finally, it would be useful to experiment on the impact of L2.5 NC on legacy device battery drain.

Bibliography

[1] T. Ho and D. Lun, *Network coding*. Cambridge: Cambridge University Press, 2008.

[2] Y. Wu, "Network Coding for Wireless Networks", *msr-waypoint*, 2007. [Online]. Available: http://www.msr-waypoint.net/pubs/70463/tr-2007-90.pdf. [Accessed: 18- May- 2016].

[3] P. Ostovari, J. Wu and A. Khreishah, "Network Coding Techniques for Wireless and Sensor Networks", *Citeseerx.ist.psu.edu*, 2013. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.380.5419&rank=1. [Accessed: 18- May- 2016].

[4] C. Fragouli, "Wireless Network Coding: Opportunities & Challenges", 2016. [Online]. Available: http://people.csail.mit.edu/rahul/papers/nc-milcom2007.pdf. [Accessed: 18- May- 2016].

[5] S. Katti and e. al, "XORs in The Air: Practical Wireless Network Coding", 2008. [Online]. Available: http://nms.csail.mit.edu/~sachin/papers/copesc.pdf. [Accessed: 18- May- 2016].

[6] C. Fragouli and E. Soljanin, "Network Coding Applications", 2007. [Online]. Available: http://ect.bell-labs.com/who/emina/papers/NCApp.pdf. [Accessed: 18- May- 2016].

[7] M. Yang, "Network Coding Application Layer Multicast", 2009. [Online]. Available: https://dspace.sunyconnect.suny.edu/bitstream/handle/1951/52389/000000877.sbu.pdf?sequence=1. [Accessed: 18- May- 2016].

[8] Y. Zhu, B. Li and J. Guo, "Multicast With Network Coding in Application-Layer Overlay Networks", *IEEE J. Select. Areas Commun.*, vol. 22, no. 1, pp. 107-120, 2004.

[9] M. Firooz, Z. Chen, S. Roy and H. Liu, "Wireless Network Coding via Modified 802.11 MAC/PHY: Design and Implementation on SDR", 2012. [Online]. Available: http://arxiv.org/pdf/1210.1326.pdf. [Accessed: 18- May- 2016].

[10] S. Zhang, "Physical Layer Network Coding". [Online]. Available: https://arxiv.org/ftp/arxiv/papers/0704/0704.2475.pdf. [Accessed: 18- May- 2016].

[11] S. Katti, S. Gollakota and D. Katabi, "Embracing Wireless Interference: Analog Network Coding", 2007. [Online]. Available: https://homes.cs.washington.edu/~gshyam/Papers/anc.pdf. [Accessed: 18- May- 2016].

[12] F. Kao, K. Tan and X. Li, "XOR Rescue: Exploiting Network Coding in Lossy Wireless Networks",*iit.edu*, 2009. [Online]. Available: http://www.cs.iit.edu/~xli/paper/Conf/xorr-secon09.pdf. [Accessed: 23- May- 2016].

[13] A. Keshavarz-Haddad and R. Riedi, "Bounds on the Benefit of Network Coding for Wireless Multicast and Unicast", *IEEE Transactions on Mobile Computing*, vol. 13, no. 1, pp. 102-115, 2014.

[14] Y. Pu, "On the Benefits of Network Coding in Multi-party Video Conferencing", Masters of Applied Science, University of Toronto, 2014.

[15] M. Sun, L. Huang, A. Arora and T. Lai, "Reliable MAC layer multicast in IEEE 802.11 wireless networks", *Proceedings International Conference on Parallel Processing*.

Appendix

UW FUNLab 802.11 implementation available at https://github.com/farzadh/Wlanphy

Layer 2.5 network coding code available at https://github.com/jarrett155/Layer_25

For use of layer 2.5 code all files in Layer_25 project replace all files of the same name in Wlanphy/host/utilities/wlan/src

Data for each experiment trial can be viewed in the following Google sheet https://docs.google.com/spreadsheets/d/1tmSytv5hExQ2fQs_CaL9eT0V3muxavIf1Rya0LG8M-c/edit?usp=sharing