NTNU – Trondheim
Norwegian University of
Science and Technology

Performance enhancements in WiFi using Network Coding

Lars Kristian Johansen

Master of Science in Communication Technology
Submission date: December 2013
Supervisor: Harald Øverby, ITEM

Norwegian University of Science and Technology
Department of Telematics

NTNU – Trondheim
Norwegian University of
Science and Technology

Performance enhancements in WiFi using Network Coding

Lars Kristian Johansen

Submission date:	December 2013
Responsible professor:	Harald Øverby, ITEM, NTNU
Supervisor:	Muriel Médard, RLE, Massachusetts Institute of Technology
Co-supervisor:	Gergely Biczók, ITEM, NTNU

Norwegian University of Science and Technology
Department of Telematics

Title:	Performance enhancements in WiFi using Network Coding
Student:	Lars Kristian Johansen

Problem description:

A major challenge with WiFi is packet loss due to bit-errors. This loss has significant impact on the upper layer protocols, in which for example TCP performance can be severely degraded. Reducing the packet loss in WiFi would help to improve overall performance in the wireless network. In this thesis, the student will study how packet loss can be reduced using Random Linear Network Coding (RLNC) in a WiFi architecture by having RLNC replacing some MAC level ACK mechanisms. A study with similar approach has been done in WiMAX by disabling ARQ/HARQ which showed significant results. Owing to the rapidly changing channel conditions in WiFi it is difficult to do adaptive modulation and coding based on ACKs and hence it will be interesting to apply NC to this technology. The scenario will consist of access points and clients where the Network Coding will be applied directly between access points and clients. The architecture will include devices that uses the 802.11n standard and can send and receive aggregated frames and block acknowledgements. The thesis consists of a theoretical part and an experimental part including testing WiFi equipment with and without NC-capability, and comparing performance indicators such as packet loss, throughput and delay. The student will initially implement Network Coding in the IP layer on both the access point-side and the client-side, and if time allows, see if it is desirable to use Network Coding in other layers instead.

Responsible professor:	Harald Øverby, ITEM, NTNU
Supervisor:	Muriel Médard, RLE, Massachusetts Institute of Technology
Co-supervisor:	Gergely Biczók, ITEM, NTNU

Abstract

Packet loss is an inherent phenomenon in WiFi due to its underlying physical medium. At the time of writing, the newest WiFi standard, 802.11n, ensures reliable communication by retransmitting lost packets through a mechanism called ARQ, where each lost packet is retransmitted until it is received. If several clients are connected to the same AP, currently, the only way to achieve reliable communication for all clients is by applying this mechanism individually for each client using multiple unicast flows. By instead exploiting the nature of the physical channel, the present thesis proposes a more throughput efficient reliability mechanism using network coding.

In the wireless physical medium, all data transmitted by the access point is in fact inherently broadcasted to all clients, even when it is not intended. For no extra cost, will other clients receive data packets not addressed to themselves. In case where a packet is lost at the receiving client, other clients may receive it successfully. By having these clients keep those packets for some time, enables network coding to reduce the number of packets to be transmitted at the access point in order to recover the lost packets. The present thesis investigates possibilities of improving the throughput in WiFi for single client and multiple clients using network coding. In both cases, reliability mechanisms are suggested, analyzed theoretically, as well as tested practically through experiments. In order to execute the experiments, modifications in the driver and operating system were necessary. The modification were realized using both open source operating system and open source network drivers. In this work, experiments were carried out in a closed office using up to 4 stationary computers with 802.11n compatible network cards.

The results show that network coding does not achieve improvements in a single client scenario. However, with multiple clients, a proposed reliability mechanism using network coding allows for a 25% reduction of redundant packets transmitted. This gives significant improvement in the throughput as compared to ARQ. Since the results are based on both experiments and theory, the aspects of commercial use are discussed, and followed up by giving suggestions for future work.

Sammendrag

Pakketap er et naturlig fenomen i WiFi grunnet det underliggende fysiske mediumet. Per dags dato, den nyeste WiFi standarden, 802.11n, sikrer pålitelig kommunikasjon ved å overføre tapte pakker på nytt igjennom en mekanisme kalt ARQ, hvor hver tapte pakke blir overført på nytt helt til den er mottatt. Dersom mange klienter er tilkoblet samme aksesspunkt (AP) er dagens måte å sikre pålitelig kommunikasjon for alle klientene ved å individuelt bruke denne mekanismen for hver klient i mange unicast strømmer. Ved å isteden utnytte kanalens fysiske egenskaper, foreslår denne avhandlingen en pålitelighetsmekanisme ved bruk av nettverkskoding som er mer båndbredde effektiv sammenliknet med ARQ i WiFi.

I det trådløse fysiske mediumet er all overført data fra aksesspunktet per definisjon kringkastet til alle klientene, til og med når det ikke er intensjonen. For ingen ekstra pris vil klienter motta data pakker adressert for andre klienter enn dem selv. I et tilfelle hvor en pakke er tapt på den mottakende klienten kan noen andre klienter ha mottatt denne pakken med suksess. Ved å få disse klientene til å ta vare på de pakkene i en begrenset periode, kan aksesspunktet, ved help av nettverkskoding, redusere antall pakker sendt for å opprettholde pålitelig kommunikasjon. Denne avhandlingen undersøker mulighetene med båndbredde forbedringer i WiFi i både singel og multiple klient-situasjoner ved bruk av nettverkskoding. I begge tilfeller er pålitelighetsmekanismer foreslått, analysert teoretisk i tillegg til testet praktisk ved eksperimenter. De modifiseringene som er nødvendige for å gjøre eksperimentene er realisert igjennom bruk av åpen kildekode operativsystem og nettverksdrivere. I dette arbeidet er eksperimentene gjort i et lukket kontor med opp til 4 stasjonære datamaskiner med 802.11n kompatible nettverkskort.

Resultatene viser at i WiFi er det ingen båndbredde forbedring ved bruk av nettverkskoding i et singel klient scenario. Derimot, med multiple klienter viser en foreslått pålitelighetsmekanisme ved bruk av nettverkskoding opp til 25% reduksjon av sendte redundante pakker. Dette gir signifikante båndbredde forbedringer sammenliknet med ARQ. Siden resultatene er basert på både eksperimenter og teori er aspektene angående kommersielt bruk diskuert og fulgt opp ved å gi forslag til videre arbeid.

Acknowledgements

This work would not be possible without help from the following people, and I want to personally thank them for their kindness and support. Muriel Médard for her excellent advice, knowledge and constant support she has generously shared with me. Jason Cloud and Kerim Fouli, from the Network Coding and Reliable Communication Group at RLE, MIT, for sharing their experience and knowledge as well taking part in discussions and provide assistance throughout the experiments. Harald Øverby and Gergely Biczók for their constant availability and supportive discussions with me. The rest of the Network Coding and Reliable Communication Group at RLE, MIT for their support and insightful help. Ali Abedi and other people from the open-source community. I also want to thank my family and friends for their never-ending support.

Contents

List of Figures

List of Tables

List of Algorithms

List of Acronyms

ACK Acknowledgement.

AMPDU Aggregated MAC Protocol Data Unit.

AP Access Point.

ARQ Automatic Repeat reQuest.

B-ACK Block Acknowledgement.

CDF Cumulative Distribution Function.

CRC Cyclic Redundancy Check.

CTS Clear To Send.

DIFS DCF InterFrame Space.

HARQ Hybrid Automatic Repeat reQuest.

IEEE Institute of Electrical and Electronics Engineers.

IP Internet Protocol.

MAC Media Access Control.

MCR Modulation and Coding Rate.

MCS Modulation and Coding Scheme.

NACK Negative Acknowledgement.

NC Network Coding.

RLNC Random Linear Network Coding.

RSSI	Received Signal Strength Indication.
RTS	Request To Send.
RTT	Round Trip Time.
SIFS	Short InterFrame Space.
SNR	Signal to Noise Ratio.
XOR	eXclusive OR.

Chapter 1 Introduction

1.1 Motivation

Today, WiFi is extensively used in both mobile and stationary devices. Laptops have in many years integrated the newest WiFi standard available and more recently smart phones are doing this as well. According to [Cis13] will traffic from wireless and mobile devices exceed traffic from wired devices by 2016. In fact, in 2012 more than 2/3 of the data traffic generated by smart phones originates from WiFi [Inf12]. Moreover, video-on-demand traffic will nearly triple by 2017 compared to 2013 [Cis13]. This significant growth in mobile data traffic and specifically in WiFi will require higher capacity in the network. Although work is being done on improving the bandwidth and latency in the network by improving the physical network, this comes at a high price. It is no longer obvious that it will be cost efficient to keep up the improvement of networks with the current growth in usage. This problem gives the motivation to look into less traditional ways, like Network Coding (NC), to improve the bandwidth and latency in networks, in which is done in the present thesis.

In the present thesis, improvements of WiFi in infrastructure mode is considered. Specifically unicast point-to-point and multicast point-to-multipoint one-directed communication are being investigated. WiFi networks have packet loss due to bit-errors. This is a fact and WiFi networks employing NC could potentially be improved by taking advantage of this fact, which is the reason for the work done here.

Today, a WiFi unicast flow uses a reliability mechanism called Automatic Repeat reQuest (ARQ). ARQ works by having the sender retransmit the packet if it was not received by the receiver. In order to retransmit a packet, the sender either will receive a Negative Acknowledgement (NACK) from the receiver, indicating that the receiver did not receive the packet correctly, or a timer will expire at the sender before it retransmits the assumed lost packet. If the Round Trip Time (RTT) is long, this scheme can cause large portions of the channel to be unused for long periods of time and hence waste bandwidth. By using NC this can be avoided by having the sender know a-priori approximately how much loss there are in the channel and consequently send extra redundancy in advance without the need for per-packet feedback that cause the channel being idle. In this way the wasteful waiting time for feedback can be avoided and instead enable potential gains in throughput.

In a WiFi infrastructure environment with several clients, using many unicast flows is not necessarily the most efficient way to deliver data reliably. In this thesis, the situation where the Access Point (AP) has potentially different data to send to every client is investigated, i.e. several single-directed flows originating from the access point. A typical scenario for this is when the clients are streaming videos for example. This thesis investigates the throughput gain by having the sender broadcast the data. By having the clients overhearing the channel and exploit the overheard data, the redundancy packets sent by the access point can contain information enough to recover several lost packets per sent packet and thus improve throughput.

1.2 Theory and Experiments

This thesis investigates the performance gains that can achieved both in unicast and multicast WiFi by conducting experiments as well as finding the theoretical gains. There are done three different set of experiments, in which two of them are done in the unicast case, and one in the case of broadcast. This thesis investigates the case where there is only one-directed communication, for example video content streaming. In the unicast case, this means a single flow from one to another, and in the multicast case it means several flows, all originating from the same access point. Theoretical results are investigated and quantified before experiments are done. The end of this thesis combines theory and experimental results to propose an efficient reliability mechanism by multicasting several unicast flows.

1.3 Outline

The following chapters are organized as follows:

Chapter 2 gives detailed background information regarding several concepts and terms that are useful throughout the rest of the thesis. It also gives information regarding other related work.

Chapter 3 provides a study using network coding in a unicast scenario with a single client, which includes theoretical as well as experimental results.

Chapter 4 gives detailed design and theoretical information regarding a proposed reliability mechanism for multiple clients in WiFi by multicasting data.

Chapter 5 gives detailed experimental results and analysis for the preceding chapter.

Chapter 6 suggests an XOR-scheme using per-packet feedback in a scenario similar to that in Chapter 4 and 5. It shows how it can improve the performance with the use of the experimental data from those chapters.

Chapter 7 discuss several topics that are relevant for the work done in this thesis. It touches upon topics that should be discussed with the perspective of the whole scope of this thesis.

Chapter 8 concludes this thesis as well as suggests further work.

Chapter 2

Background and Related Work

The purpose of this chapter is to introduce several important concepts and terms that are of importance in the later chapters. It also discusses related work.

2.1 WiFi and Wi-Fi Alliance

The Wi-Fi registered trademark is owned by The Wi-Fi Alliance [All]. The Wi-Fi Alliance define WiFi (also written Wi-Fi and Wifi) to be any wireless local area network that is based on the Institute of Electrical and Electronics Engineers (IEEE) 802.11 standards [Web]. The 802.11 standard has been through a development starting out with 802.11a and 802.11b, then a few years later came 802.11g in 2003. 802.11n was standardized in late 2009, and 802.11ac is still in development assuming to finish in 2014 [Gro]. The WiFi Alliance certifies products based on the 802.11n standard, and all the WiFi cards used in the experiments in this thesis are certified 802.11n.

802.11n

The currently newest 802.11 standard is IEEE 802.11n-2009, commonly shortened to 802.11n. The development of this standard started in 2002 and ended 7 years later, with a lot of improvements over the previous 802.11a and 802.11g as was the goal of this new standard. The following list contains a selection of the most important new features introduced in 802.11n.

– MIMO - several spacial streams within the same spectral channel of bandwidth. This increases the bandwidth but will require as many antennas on both sender and receiver as there are spacial streams, which is indeed expensive.

– Aggressive Modulation and Coding Rate (MCR) - more aggressive MCR can achieve higher throughput in areas with high Signal to Noise Ratio (SNR). In areas with low SNR this will instead cause packet loss.

– 40 MHz spectral channel bandwidth - allowing for a slightly more than doubling in throughput compared to previous standards. This comes with the downside of having potentially more overlapping channels.

– Halved guard interval - In many cases it is unnecessary with the 800ns guard interval which is default in 802.11a and 802.11g, so 802.11n introduces 400ns intervals, and it can give a ~10% increase in throughput.

– Aggregated MAC Protocol Data Unit (AMPDU) - frame aggregation - This is a MAC-layer mechanism that aggregate frames and send them in bursts. This require the sender to acquire the channel only once for the whole burst of packets, instead of having to acquire the channel for each packet to transmit. It also gathers the acknowledgments into a single Block-ACK packet, having only 1 acknowledgment for the whole burst indicating which packets in the burst that was lost. If the channel is quite busy with other clients or have significant packet loss probability, this can improve the efficiency of the overall throughput in the network as well as efficiency per client.

Although the above list is only a part of the improvements in 802.11n, it will give an increase of max data rate from 54Mbit/s in 802.11g to 600Mbit/s. The improvement of throughput is huge, but there has not been that much focus on improvement of latency in the network, which would most likely require change in the Media Access Control (MAC) algorithm.

Media Access Control

In order for the clients to transmit to the channel without too many collisions, some media access mechanism must be implemented. This mechanism is called the MAC algorithm, which is a fair algorithm meaning that it provides equal access to the channel for each client. When a client is ready to send a packet, it waits a random amount of time, called the back-off time, before it tries to transmit to the channel. If the transmission fails, it waits again a random amount of time, but now on average this time should be twice as long. This mechanism is called the exponential back-off algorithm. By having the random amount of time increasing exponentially when failing, make the clients enable to adapt to the traffic, and does not end up with mostly collisions. 802.11 also provides a mechanism in the MAC that fragments the frames, which is not discussed because it is turned off in the experiments in this thesis. Likewise for the mechanism to ensure guaranteed access to the channel called Request To Send (RTS)/ Clear To Send (CTS).

There has been done a lot of work in 802.11n in order to enable sharing the same frequency spectrum with the previous standards, such that older technologies still can function properly. Most of the devices that support WiFi today ships with 802.11n, although there are a lot of devices with the older standard still in use.

802.11ac

Although the 802.11ac standard is still in development, it doesn't seem to introduce many important new concepts compared to 802.11n. It will allow for more spacial streams, more aggressive modulation and code rate, quadruple the spectral channel bandwidth to mention some of them. Since this standard seems conceptually similar to the 802.11n, the work being done in this thesis will be relevant not only in the near future, but also after the release of 802.11ac.

2.2 Automatic Repeat ReQuest (ARQ)

The way 802.11n ensure reliable communication in the presence of packet loss is by using a mechanism called ARQ. This mechanism is designed for only a single sender and receiver, so it is only used in unicast streams in WiFi, and not in broadcast streams. The mechanism works by getting feedback through acknowledgements from the receiver regarding packets that were correctly received and which were not correctly received. The receiver can verify whether a packet is received successfully by using the Cyclic Redundancy Check (CRC) code provided in the packet. By using the concept of acknowledging packets, there have emerged a few commonly known versions of ARQ. The most common are called Stop-and-wait, Go-back-N and Selective Repeat. WiFi uses the simplest version of ARQ called Stop-and-wait, which is explained in detail in the following section.

Stop-and-wait ARQ

This version of ARQ ensures both recovery of lost packets as well as in-order reception of packets. The Stop-and-wait ARQ mechanism ensure in-order delivery by forcing each packet in turn to be received at the receiver, and it provides this reliable communication as follows:

1. Send current packet and start the local timer.
2. Wait until you either receive an Acknowledgement (ACK), NACK or the local timer expires.
   - If ACK received, continue with the next packet, knowing that the packet is received successfully.
   - If NACK is received or local timer expires, go back to 1 again.

In WiFi there is a parameter that controls the number of consecutive times the algorithm is allowed to jump back to 1 before it should give up the transmission of the packet. This is called the maximum retransmission parameter, and a typical value is 6, meaning that if the packet is not received correctly within 7 attempts, the sender has failed to send it and will continue to try to send the next packet.

2.3 Network Coding

A network that apply NC can be defined as a network where the nodes in the network does operations on the received or generated packets rather than just store and forward them as in a traditional packet switched network.

Intra-Session and Inter-Session Network Coding

Network coded packets can be created in several ways, and when they are created based on multiple sessions, flows or connections, the network coding performed is called inter-session network coding, and consequently when network coded packets are created based on a single session, flow or connection, the network coding performed is called intra-session network coding.

Linear Network Coding

A particular direction of NC has gained a lot of attention, which is called linear network coding. In linear network coding, coded packets are created as a linear combination of a number of data packets, called a block of data packets, making each coded packet contain information about all the data packets in the block. The process of creating the coded packet is called encoding. If all but one of the data packets from the block are received, the coded packet can represent that missing packet and be calculated into it mathematically. The process of doing this is called decoding. In order to decode, the coefficients used in the encoding of the coded packet must be known to the receiver, which will be further explained in the next section. The number of coefficients will necessarily be the same as the size of the block the coded packet is a linear combination of.

Linear network coding can be beneficial in several situation, for example if there are several paths between the sender and receiver. Since each coded packet can represent any of the packets in the block, the receiver only cares about getting enough packets regardless of where they come from. Another example is when there is loss in the channel. By using linear network coding in that case, the sender need to know only the number of packets lost instead of which packets that were lost in order to recover the loss using coded packets. In some cases this can increase throughput. The trade-off by doing this is that on average it increases per-packet delay at the

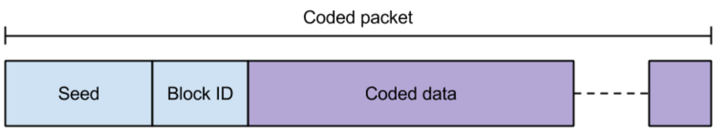


Figure 2.1: Structure of a coded packet using RLNC with seed.

receiver. The coded packets will not be decodable until the receiver has received at least the same number of packets as there are in the block the coded packets were a linear combination of. This delay-throughput trade-off has been subject to a lot of research [FLMP07, EOMA08, CMWB08, SES10], and a particular outcome of this is called Random Linear Network Coding (RLNC).

Random Linear Network Coding (RLNC)

RLNC is a type of linear network coding where the coefficients for encoding the coded packets are chosen at random from a Galois field. This has been shown to allow close to optimal throughput [HKM$^+$], yet with less complexity at the sender. Field theory is not covered in detail in this thesis, but is explained in great detail in a book called Finite Fields for Computer Scientists and Engineers [McE87]. By choosing the coefficients randomly also allows the sender and receiver to generate coded packets with little overhead. Since both encoder and decoder must use the same coefficients, it is sufficient for the encoder to transmit only a seed together with the coded packet, and both encoder and decoder can use the seed to generate the same psudo-random coefficients. The architecture of a coded packet can therefore be as simple as in Figure 2.1.

Encoding and Decoding

The encoding of a packet is illustrated in Figure 2.2. When transmitting data, say a file, it is divided up into blocks of packets. During the creation of a coded packet, each of the packets in a block are being multiplied individually with coefficients in a Galois Field. The multiplication is performed over a Galois Field and therefore the result is also in the same field. The multiplication between a coefficient and a data packet is performed by dividing each data packet into chunks with size of the coefficients such that the multiplication is done separately on each chunk as shown in Figure 2.3. When finished, the chunks are concatenated back into a packet size again. The XOR-ing of packets are done in a similar manner, where XOR-ing two packets together consists of XOR-ing the packets bitwise.

The decoding of a block of coded packets is easiest explained mathematically. By

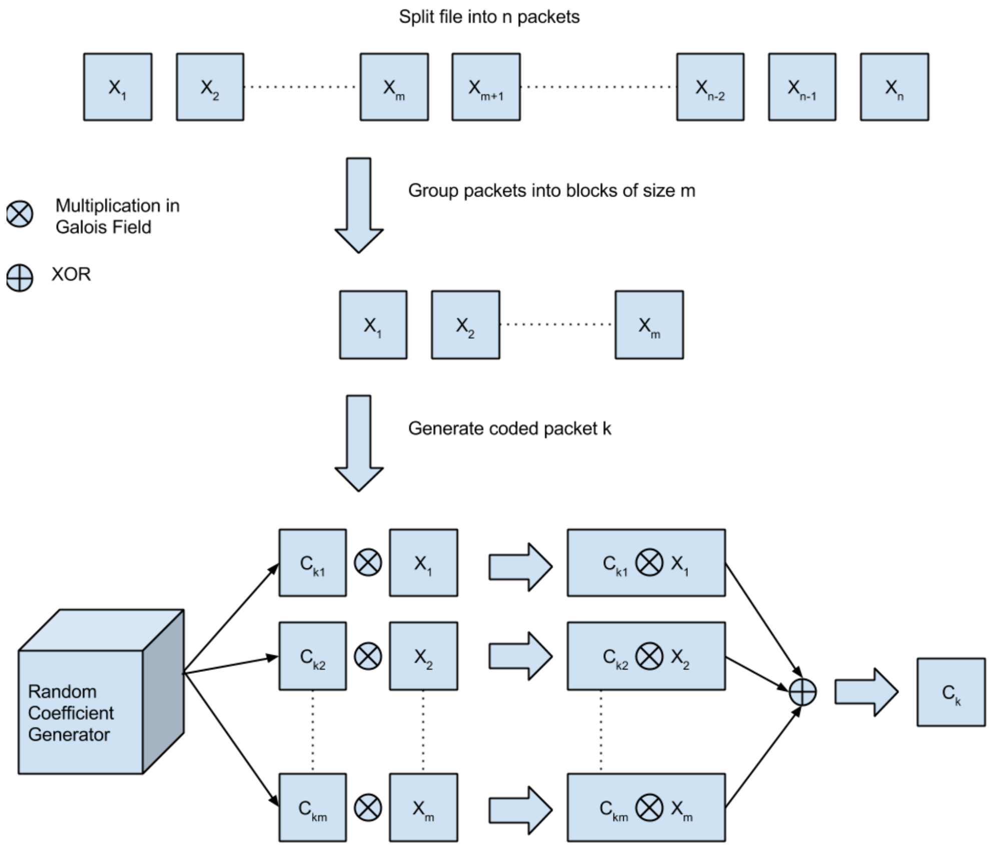

Split file into n packets
X_1 X_2 X_m X_{m+1} X_{n-2} X_{n-1} X_n
Group packets into blocks of size m
$\otimes$ Multiplication in Galois Field
$\oplus$ XOR
X_1 X_2 X_m
Generate coded packet k
Random Coefficient Generator
$C_{k1} \otimes X_1$
$C_{k2} \otimes X_2$
$C_{km} \otimes X_m$
C_k

Figure 2.2: Outline of how a coded packet is created using RLNC.

representing the coefficients as a matrix C where each row contains the coefficients chosen to generate each coded packet for the block, and representing the original data packets as a column vector X, where each element is one original data packet, the coded packets can be represented as the elements in the column vector $R = CX$, where multiplication is over Galois Field and addition is bitwise XOR-ing. The problem of decoding the original data packets can therefore be described as solving the equation $X = C^{-1}R$. The problem of decoding the block is therefore the problem of finding the inverse of the coefficients matrix, C^{-1}. Since the coefficients are chosen at random, there might not exist an inverse of C, and therefore more coded packets might be needed. This issue is discussed in a later section. Finding the inverse of C or whether it exist can be done efficiently with the use of Gauss-Jordan elimination for example.

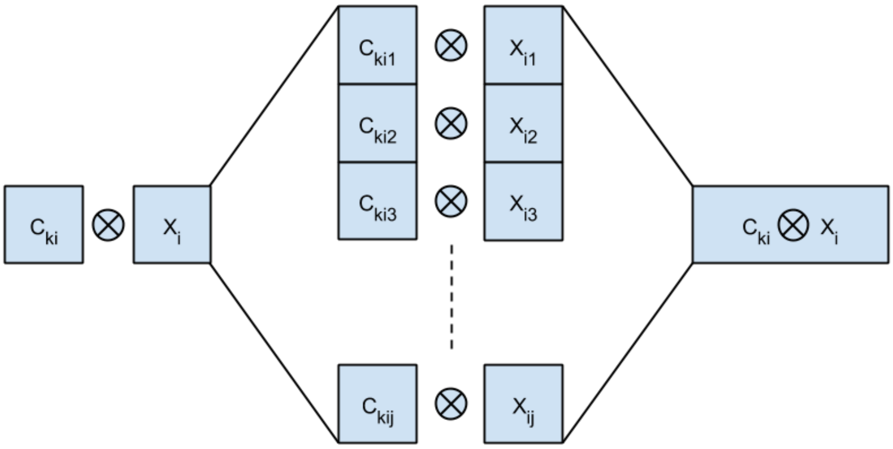


Figure 2.3: Multiplication in GF field of a data packet and coefficient.

Systematic vs. Non-Systematic Coding

Two well known separate ways of using RLNC have emerged, and they are called systematic RLNC and non-systematic RLNC. The difference between non-systematic RLNC and systematic RLNC is that in systematic RLNC the whole block of original data packets are sent in replacement of the coded packets and then extra coded packets are sent additionally as redundancy. [SMJ] has shown systematic RLNC to be as good as non-systematic RLNC in terms of throughput, yet has less decoding complexity and in some cases less per-packet delay. The decoding process can be applied in the same way, relating each uncoded packet with a coefficient vector consisting of only 0s and a single 1 placed in an increasing position as the uncoded packets are sent. This ensures the uncoded packet to be unmodified during decoding and all the uncoded packets will be linearly independent. Another neat feature about systematic RLNC is that even though the receiver did not receive sufficient packets to decode the block, the systematic packets that was received can still be used. Due to these performance enhancements in systematic RLNC, it is chosen as the way to use linear network coding in this thesis.

Decoding Probability of Packets in RLNC

The received coefficient matrix C might not be invertible, and thus cause the decoding to fail. This means that some coded packets must have been linearly dependent with each other, and did not add any new information to the receiver. Such packets are called non-innovative packets and all other packets are called innovative. The goal is to maximize the number of innovative packets with the constraint of having decoding to be fast. In this thesis, systematic RLNC with field size of 2^8 is being used for that reason. In [LMS09] it is proven that in RLNC, given the field size of q, the upper limit of the mean number of coded packets that needs to be received in order to decode M packets is:

$$\min\{M\frac{q}{q-1}, M+1+\frac{1-q^{-M+1}}{q-1}\} \tag{2.1}$$

This limit applies perfectly to situations where all packets in a block are coded, but in the situation with systematic RLNC, the first M packets are actually guaranteed to be linearly independent, and hence this limit does not help unless we introduce the fact that we need additional coded packets to make up for packet loss. This upper limit is still correct assuming M is the total of systematic packets and non-systematic coded packets. Although it is a correct upper limit, it is not a tight one because the systematic packets are guaranteed to be linearly independent. A tighter upper limit will be to calculate the probability given that the first systematic packets are linearly independent. With a field size of 2^8 and a constant packet loss probability, it is easy to show that with any reasonable block size and packet loss probability, the probability that the received packets are linearly independent are very close to 1.

Theoretical Probability given an Example

Assuming the scenario with a network with 20% packet loss. Further assume a field size of 2^8, 10 systematic packets, and following 2 coded packets to make up for the expected packet loss. Using (2.1) with $M = 12$ and subtracting with $M = 10$ (for the systematic packets) gives the mean number of required coded packets to be received as 2.0078 on average. The extra 0.0078 packets are because of linear dependency. Similarly with 100 systematic packets and 20 coded packets gives a mean number of required coded packets to be received as 20.078. Since this is an upper limit on the average number of packets to be received in order to decode, a field size of 2^8 will suffice without significant linear dependency. Therefore, for the rest of this thesis if not otherwise stated, the assumption will be that all packets in a block, both coded and uncoded, are linearly independent from each other.

Inter-Session Network Coding using XOR

In a wireless communication channel, clients can generally overhear the channel. By having the access point divide packets into groups, called blocks of packets, clients can store packets in the block not directly intended for them self, and these packets can help to improve the overall efficiency of the channel. Having the access point knowing which packets that are received and which are not for each client, it can XOR packets together such that clients have all but one of the packets, and from the coded packet can retrieve this packet. The scheme is all based on the fact that XOR-ing packets twice cancels them out. The structure of the coded packet can be

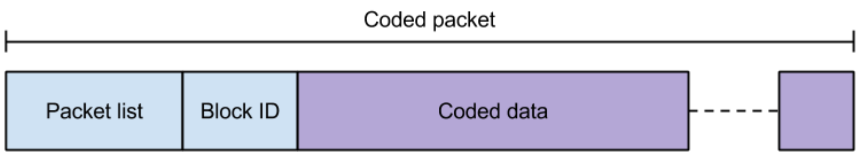


Figure 2.4: Structure of a coded packet using XOR.

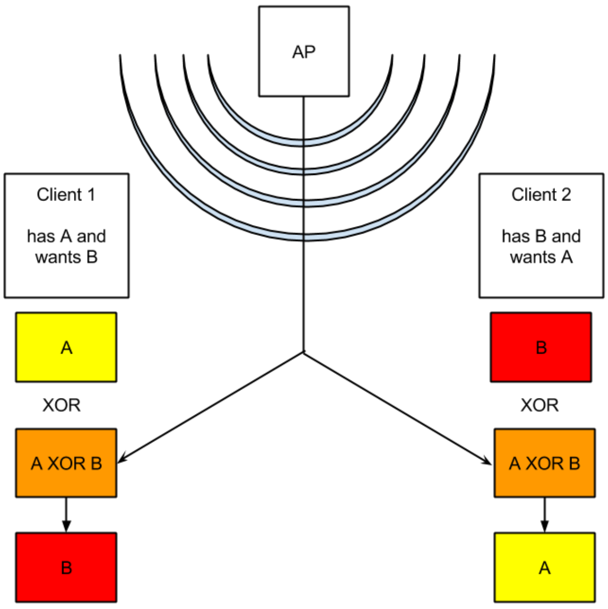


Figure 2.5: Conceptual benefit of XOR scheme. The access point broadcasts one packet and recovers two different packet losses.

seen in Figure 2.4, where the packet list contains the id for the packets the coded data is XOR-ed over. The benefit of this scheme is that only one transmission from the access point can recover different packet loss on several clients, which is illustrated in Figure 2.5. The overhead by using this scheme is the block id and the list of packets that must be included into the coded packet in order to decode it.

Related Work

Work on concepts like network coding was first proposed by [Met00] in 1984 by improving bandwidth in a broadcast network using simple XORs of packets, however the work done in [ACLSY00] is considered the introduction of network coding, which aim to improve bandwidth in wired networks. Applying network coding in wireless networks was first studied in [KRK+84] developing a system called COPE which showed significant improvements in wireless mesh networks. Although the introduction of network coding was not initially intended to improve performance in networks with significant packet loss, this has become of interest shortly after [YS00, Lar07, LJ06, Lar08, LMKE08, LMKE05, MTK08, DGP+06, SSM08, NTNB09, QFC10, Tee, CWJ03, EOMA08, JMM93]. Reliability schemes including ARQ and network coding has been widely studied [YS00, Lar07, LJ06, Lar08, LMKE08, LMKE05, MTK08, DGP+06, SSM08, NTNB09, QFC10, JMM93, Tee] and will be briefly summarized in the rest of the section.

Yong *et al.* [YS00] investigates multicast with XOR as a reliability mechanism, where the multicasted data is intended for all receivers. Larsson *et al.* [Lar07, LJ06] analyze multi-user ARQ with multiple unicast flows as well as XOR network coding schemes. Larsson *et al.* [Lar08] also analyzes adaptive linear network coding and multiuser ARQ in a multicast scenario where the coefficients for the network coded packets are adaptively selected.

Random linear network coding is studied extensively, and Lun *et al.* [LMKE08, LMKE05] shows that linear network coding achieves channel capacity when coded packets are created from previous received packets. Dana *et al.* [DGP+06] investigates a class of wireless networks called erasure networks, which exploits the broadcast nature of wireless channels and shows that the network achieve the capacity region by the use of linear network coding at the nodes in the network. Ghaderi *et al.* [MTK08] quantifies the reliability gain of network coding for reliable multicasting in wireless networks. Sundararajan *et al.* [SSM08] propose an online coding algorithm for packet erasure broadcast channels by extending ARQ to coded networks. Nguyen *et al.* [NTNB09] analyzes and simulates the performance of network coding compared to ARQ in a broadcast wireless scenario. Qureshi *et al.* [QFC10] propose a latency and bandwidth efficient reliability protocol called BENEFIT. It is improving efficiency using network coding in point-to-multipoint multicast networks.

A lot of theoretical and analytical work has been done in the area of reliability mechanisms using network coding, but less work has been with experimental data to back it up. S. Teerapittayanon [Tee] explores intra-session network coding in WiMAX and compares it to the existing ARQ/Hybrid Automatic Repeat reQuest (HARQ) mechanisms using experiments, which the first part of this thesis is highly motivated by. Furthermore Jolfaei *et al.* [JMM93] studied analytically reliability mechanisms in point-to-multipoint scenario using XORs and compared it to Selective Repeat, which the end of this thesis is inspired by.

2.4 Open Source Drivers

There are two open source drivers used during experiments in this thesis. They are both from Atheros, called ath9k and ath9k_htc.

ath9k and ath9k_htc

ath9k is compatible with several WiFi cards that has a certain chipset and is using the PCI bus. Similarly ath9k_htc is used for several WiFi cards that has a certain chipset and is using the USB bus. Both drivers supports most of the features in 802.11n. The drivers has a default rate control algorithm that dynamically adapts the modulation and coding based on the loss patterns. ath9k controls all MAC features in its software while ath9k_htc uses firmware for certain tasks.

Custom Rate Adaption Algorithm

The author developed a custom rate adaption algorithm for both drivers. This rate adaption algorithm ensures a fixed modulation and coding scheme regardless of loss patterns. The modulation and coding scheme can be manually changed, but it will remain the same until it is manually changed again. In the rest of this thesis, it is referred to as the fixed rate adaption algorithm.

Chapter 3
A Unicast WiFi Study

Inspired by the large gains found in [Tee] for WiMAX, a similar study is therefore conducted in WiFi in this thesis. By the use of intra-session RLNC as a replacement for ARQ/ HARQ, [Tee] showed a gain close to 6 times better than the current standard in terms of throughput by eliminating the need for per-packet feedback in WiMAX. Since WiMAX and WiFi has similar nature, employing the same concepts on WiFi is being investigated as a possible throughput enhancement in this thesis. The design and architecture is covered first, and then theoretical and experimental performance is covered in the end of this chapter.

3.1 Design and Architecture

In any WiFi unicast flow, there is default an ARQ mechanism implemented. This is implemented so that packets will be reliably delivered at the receiver, even when there is packet loss due to bit errors in the channel. Usually there is a limit on how many times the station will retransmit the packet before it gives up and continues with the next packet. The following sections will describe the current designs as well as the proposed design with network coding for both 802.11n and pre-802.11n WiFi systems.

Pre-802.11n WiFi Unicast Design

As shown in Figure 3.1, when a station is transmitting several packets, each packet in turn must first wait DCF InterFrame Space (DIFS) amount of time, then acquire access to the channel by using the back-off algorithm, then send the packet. The time it takes for the receiver to receive the incoming packet and find out whether it should send an ACK or NACK back is shortly referred to as the Short InterFrame Space (SIFS) time, and only after that time has elapsed, the acknowledgment can be sent back. After the data frame has been sent and an acknowledgement has been returned, access to the channel must be acquired again for the next packet. ARQ is also applied to the transmission, making use of the acknowledgments immediately

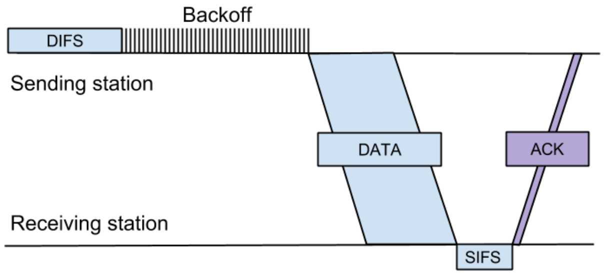


Figure 3.1: The WiFi pre-802.11n design.

returned after the data packet is sent so that the sender can be sure that the packet successfully arrived at the receiver or not. If the sender received an NACK or it took too long to receive any feedback, the same packet is transmitted again after acquiring access to the channel assuming that it was lost. It is a simple ARQ mechanism, which make sure each packet is successfully transmitted before it attempts to transmit the next packet.

Pre-802.11n RLNC Enhanced WiFi Unicast Design

The Pre-802.11n RLNC-enhanced design is similar to the design of the Pre-802.11n WiFi unicast design, but with an important difference. Packets are grouped together in what is called a block of packets. This is an ordered group of packets, usually just a collection of a certain number of packets that would normally be transmitted consecutively. The new design using this block of packets shows that it is unnecessary to collect information regarding exactly which packets that were lost, but instead only information about how many that were lost within each block. By estimating this number based on the channel quality, no per-packet feedback is necessary. Figure 3.2 shows the design of such NC-Enhanced WiFi architecture. Instead of requiring feedback from the receiver, the sender by default transmits extra coded packets that were coded over the whole block of packets that were just sent, assuming there was packet loss at the receiver. Since these coded packets can recover any lost packet in the block, the receiver can recover from the packet loss. Since the redundant coded packets are coded over the whole block of packets, these coded packets can immediately be transmitted, just with the knowledge that there are packet loss in the channel. This was not previously possible without the use of network coding and hence the per-packet feedback is eliminated and can instead be used for transmission of data packets.

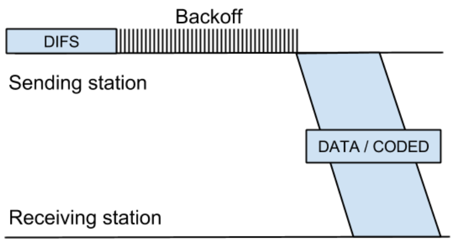


Figure 3.2: The pre-802.11n RLNC Enhanced WiFi design.

802.11n WiFi Unicast Design

Starting in 802.11n, as shown in Figure 3.3, aggregated frames and Block Acknowledgements (B-ACKs) are used. Instead of waiting DIFS time and acquiring access to the channel for each packet, the sender instead acquire access to the channel for several packets at a time, called a block of packets. Only after having sent the whole block, a B-ACK request is sent from the sender and an B-ACKs response is returned to the sender by the receiver. This acknowledgement, called B-ACKs, is a bit-map indicating which packets that were successfully received and which that were lost at the receiver. The sender now retransmits whichever packets that was indicated lost at the receiver. These retransmitted packets can be incorporated into the next block so the length of the block is maintained the same until the transmission is over. In this way, the throughput can be increased on the expense of increased per-packet delay.

802.11n RLNC Enhanced WiFi Unicast Design

The RLNC-enhanced design use the same design concepts as in the pre-802.11n RLNC Enhanced design. The difference now is that the block size in the network coded scheme is the same as the block size at the 802.11n scheme and the channel will

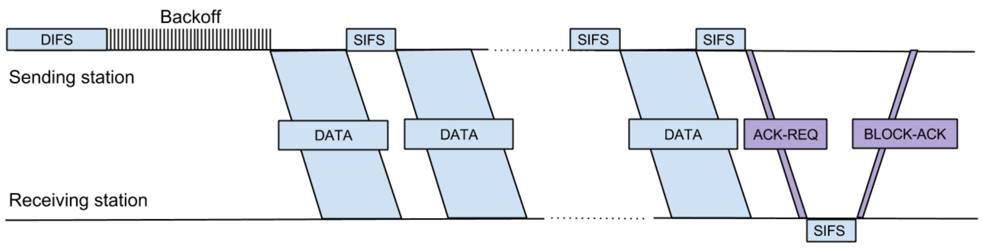


Figure 3.3: Describing the WiFi 802.11n design with the use of aggregated frames.

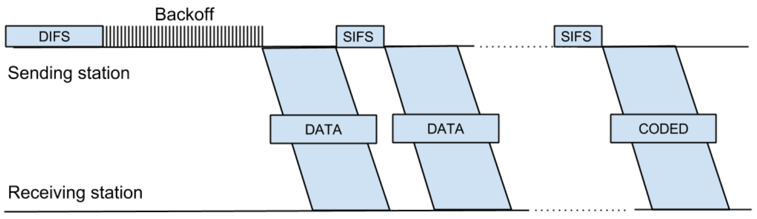


Figure 3.4: Describing the 802.11n RLNC Enhanced WiFi design.

be reserved for all those packets. Figure 3.4 shows the design of such RLNC-Enhanced WiFi architecture. As in the previous enhanced design, instead of requiring feedback from the receiver, it by default transmits extra coded packets over the block of packets that was just sent, assuming there was packet loss at the receiver. In this way the receiver can recover from packet loss as in 802.11n.

3.2 Theoretical Performance

To see whether the new designs look promising, theoretical performance is being quantified for each proposed design.

Variables used in Calculation

Theoretically, given that the assumptions are correct, the proposed designs should show gains in throughput with the trade-off of increased per-packet delay. Table 3.1 presents parameters that will be used when quantifying the performance of the different designs. Since the enhancements does not try to reduce the number of redundant packets needed to be transmitted, but instead tries to reduce the time to transmit all the packets, the average total number of redundant packets sent in all cases will be

$$num\_extra = \frac{num\_d}{1-p} - num\_d \ .$$

This accounts for the loss of redundant packets as well. It is important to notice that the proposed designs does not try to reduce the amount of redundancy, but instead tries to reduce the time it takes to send the packets.

Performance in Pre-802.11n

For the standard Pre-802.11n, this thesis assumes that half of the retransmissions are caused by the local timeout to expire, while the other half is caused by the reception of NACK packets. In reality this can vary based on the probability of packet loss at both stations as well as other channel properties.

The number of retransmissions caused by local timeout, as well as by reception of NACKs are

$$loc\_retr = nack\_retr = \frac{num\_extra}{2} \; .$$

The total time it will take to transmit d data packets on average is the following

$$total\_time = (num\_d + nack\_retr) \times$$
$$(difs\_delay + bo\_delay + dt\_delay + p\_delay + sifs\_delay + at\_delay + p\_delay) +$$
$$loc\_retr * (difs\_delay + bo\_delay + dt\_delay + lt\_delay) \; .$$

Performance in Pre-802.11n with NC

To estimate the performance in this design, it is assumed that the sender predicted the true packet loss at the receiver. The total time it will take to send d data packets is therefore

$$total\_time = (num\_d + num\_extra)(difs\_delay + bo\_delay + dt\_delay + p\_delay) \; .$$

Performance in 802.11n

To easily estimate the performance in this design, it is assumed that there are a lot of blocks to be transmitted, so the total time it will take to transmit d data packets can be approximated to

$$total\_time = (num\_d + num\_extra) \times (dt\_delay + sifs\_delay +$$
$$\frac{difs\_delay + bo\_delay + 2at\_delay + sifs\_delay + 2p\_delay}{b}) \; .$$

Table 3.1: Parameters used in the theoretical performance calculation for the proposed unicast designs.

Loss probability	p
Number of data packets	num_d
Block size	b_size
Propagation delay	p_delay
Data Transmission delay	dt_delay
ACK/B-ACK req./B-ACK resp./NACK Transmission delay	at_delay
Back-off delay	bo_delay
SIFS delay	$sifs\_delay$
DIFS delay	$difs\_delay$
Local timeout delay	lt_delay

Table 3.2: Selected parameter values in unicast WiFi.

Loss probability	0.2
Number of data packets	1000
Block size	60
Propagation delay	$1 \cdot 10^{-6}$s
Data Transmission delay	$1.43 \cdot 10^{-3}$s
(B)ACK/NACK Transmission delay	$7.6 \cdot 10^{-5}$s
Back-off delay	$7.2 \cdot 10^{-5}$s
SIFS delay	$1.6 \cdot 10^{-5}$s
DIFS delay	$3.4 \cdot 10^{-5}$s
Local timeout delay	$9.4 \cdot 10^{-5}$s

Performance in 802.11n with NC

In the 802.11n NC Enhanced design, the total time it will take to send d data packets, given that the sender assumed the true packet loss at the receiver is

$$total\_time = (num\_d + num\_extra)\times$$
$$(dt\_delay + sifs\_delay + \frac{difs\_delay + bo\_delay}{b}) \ .$$

Comparing the Designs Numerically

To compare these designs numerically, typical values are being replaced by the variables. Table 3.2 shows the typical values for all the variables used in the derivation of the theoretical performance for the designs. These numbers are based on the 802.11n standard, as well as the following assumptions:

- Transmission speed is 1Mbit/s and packet size of 1500bit.
- Distance between stations are 10 meters.
- There are no other clients competing for the channel.
- The overhead by using the seed and block number in RLNC is neglected because of large packet sizes.

Using the numbers from Table 3.2, Table 3.3 shows the time it takes to transmit the data using the different schemes. As seen, there is a potential of up to 6% decrease in transmission time for Pre-802.11n and up to 0.5% decrease in transmission time for 802.11n. Even though the numbers in Table 3.2 are mainly from the 802.11n standard,

Table 3.3: Theoretical results for unicast WiFi with and without network coding.

Type	Transmission time	% decrease from corresponding scheme
Pre-802.11n	2.05 s	-
Pre-802.11n w/NC	1.92 s	6.3% decrease
802.11n	1.81 s	-
802.11n w/NC	1.80 s	0.5% decrease

they are similar enough to be used in the pre-802.11n analysis as well. As seen, the NC designs should only perform slightly better than the corresponding standard without NC given that the assumptions were correct. To verify this, experiments are conducted, which will be the topic of the rest of this chapter.

3.3 Experiment

The motivation for retrieving experimental results is to verify the theoretical results, and after analyzing the results give motivation for what to investigate further. The experiment being conducted is written in the application layer where both coding and decoding happens. Prior to this experiment, the author conducted a similar experiment that was done at the Internet Protocol (IP) layer to confirm the same hypothesis. That experiment is only covered briefly, because it had a few potential flaws compared to the design of the experiment at the application layer.

Confirming No Throughput Improvements

To show that there are practically no gains by using intra-session network coding in a WiFi unicast scenario, the assumption of being able to predict correct number of packet losses for each block is being tested, and experiments will explore the number of extra packets needed to be transmitted in order to maintain reliable communication. The number of extra packets will be the number of coded packets sent when using network coding, and in the ARQ case it will be the number of retransmitted packets. If the number of required extra packets using network coding significantly exceeds the number of retransmitted packets in the ARQ case, then the decrease in performance because of those extra packets will outweigh the slight increase in performance by removing the per-packet feedback from the original designs. The experiment is therefore conducted with the assumption that there will be need for more coded packets than retransmitted packets and hence show that there will be no gain by using network coding.

Experiment at the IP Layer

As mentioned, an experiment at the IP layer, similar to the one covered in the rest of this chapter, was performed. Some important differences between the experiment at the IP layer and the one covered in the rest of this chapter made the IP layer experiment suboptimal. The following list compare the most important reasons for not including the IP layer experiment in this thesis, and instead focus on the application layer experiment. When no specific experiment is referred to, it is implicit that it is the application layer experiment.

- The encoding of packets worked by waiting until a whole block of packets arrived in the IP queue, then additional packets was encoded, and after that was done, the whole block plus the additional packets were released to the MAC layer. This caused at least two possible problems.
    - The MAC layer buffers got overflowed, and the MAC layer started to drop packets and cause packet loss.
    - Because of the packet bursts from the IP layer, the transmission rate at the MAC layer could come close to the physical rate of the channel, provoking unnecessary packet loss.

Instead in the new proposed software design, the packets will be sent with a constant rate all the way down to the physical layer when using network coding and none of these issues will be a problem. The rest of the chapter is therefore focusing on that experiment instead of the IP layer experiment.

3.4 Setup

This experiment is conducted in a small office with short distances between the nodes in the network. The topology will be further discussed in the next section. After that the architecture of the system is explained, as well as the hardware, software and WiFi configuration used during the experiment.

Topology

The topology of this experiment is simple. It is one access point connected directly to a client. Figure 3.5 shows the topology of the experiment. There are no other clients connected to the access point.

Architecture

The architecture used to send data reliably in the experiment is illustrated in Figure 3.6. This experiment is conducted in the application layer because of simplicity

Figure 3.5: Topology of the unicast experiment.

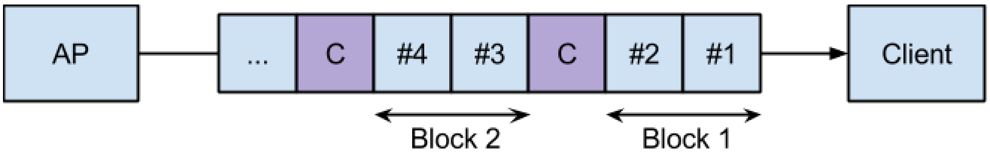


Figure 3.6: Architecture used to send data reliably using RLNC in a unicast WiFi. When a coded packet is sent, it is coded based on the previous block of packets.

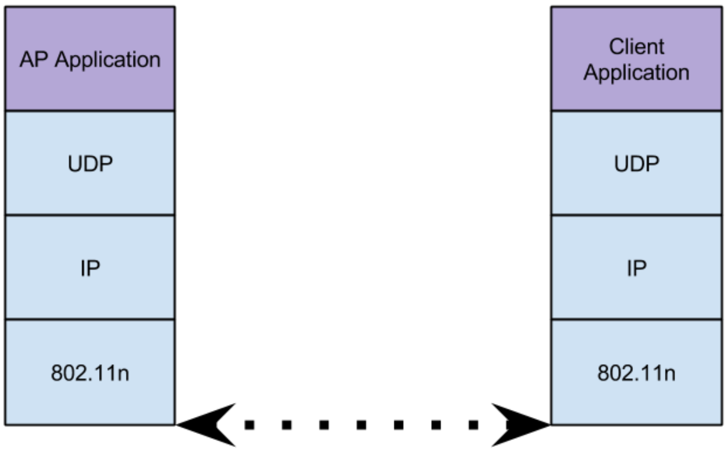


Figure 3.7: The network layered architecture used in the unicast experiment.

and efficiency. The structure of the experiment separated into network layers is illustrated in Figure 3.7. Even though it is possible to estimate the packet loss probability to a certain level of the underlying channel, it is complex and it is therefore decided that the coded packets are pre-coded with a certain percentage of redundancy for each block. This percentage should ideally be around the same as the packet loss percentage in the channel, but since that is unknown, varying this percentage in repeated executions of this experiment will clarify the best configuration. By doing it this way, potential delay caused by encoding is eliminated when the experiment is actually conducted, as was seen to be a potential problem in the IP layer experiment.

The structure of the coded, uncoded and normal packets that are sent is illustrated in Figure 3.8. When the ARQ is being tested, normal data packets are sent and no overhead is added by the application. When RLNC is being tested, there are 5 fields plus payload data that are included into the packet, regardless of whether it is an uncoded packet or a coded packet. When a packet is uncoded, the type field is set to a number that indicates an uncoded packet. The Seq ID is set to the number of sent packets before the current packet in the block. The block size and block ID is set to their respective values depending on which block and block size that is transmitted. When a coded packet is sent, the type is set to indicate that the packet is a coded packet. Furthermore, the block size and block ID is set to their respective values and the seed is set to the generated seed at the sender. Therefore, when an uncoded packet is sent, the seed field is not being used, and when a coded packet is sent, the Seq ID field is not being used. For the uncoded packets, the sequence number is being used to determine the position of the 1 in their coefficients vector.

The access point application sends a file of size 2281600 bytes to the receiving client. The file is divided into packets of total size 1472 bytes, which includes the overhead previously discussed. The total number of data packets to be sent using the ARQ mechanism is therefore 1550, and the total number of data packets to be sent using the network coding mechanism is therefore 1558, adding trailing 0s as padding for the last packet. The reason for the choice of 1472 bytes for the packets is because it is the maximum size for a packet at the application layer such that it will not split into several packets in any of the below layers.

Hardware

The hardware used is two Dell computers. One of them is used as the AP and the other as the client. Their specifications are the following:

- Motherboard: Dell 0C2KJT vA00
- CPU: 64 bit, Intel(R) Core(TM) i5 CPU 650 @ 3.20 GHz, Dual core
- RAM: 6 GB, 3×2 GB, 1333 MHz, Hyundai HMT325U6BFR8C-H9
- Operating System: Ubuntu 12.04 64-bit, Kernel version: 3.5.0-42-generic
- WiFi card:
    - Model: TP-LINK TL-WN781ND Wireless N150
    - Frequency: 2.4 GHz

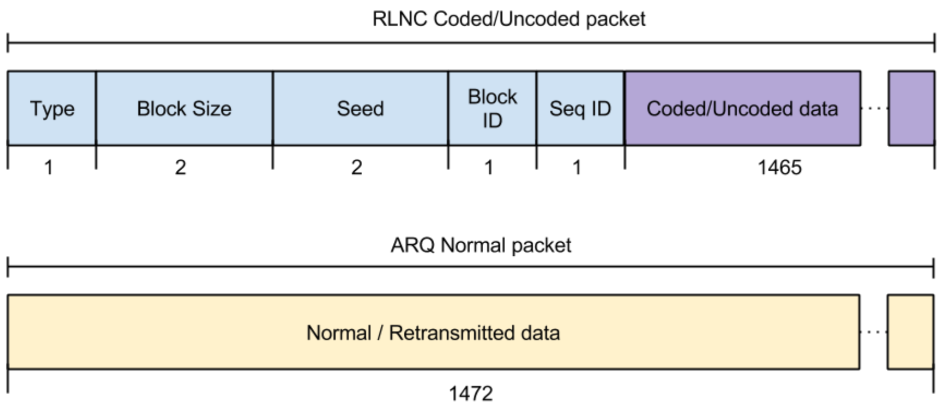


Figure 3.8: The structure of both the coded packet and uncoded packet in this experiment, as well as the packet structure when using ARQ. The numbers indicate the number of bytes.

- Max bandwidth: 150 Mbps
- Chipset: Atheros AR9485
- Driver: ath9k

Software

The software used to transmit and code the packets are written in the application layer as mentioned. The transmission rate from the application is fixed such that the net transmission of data packets is fixed. This will cause the rate to vary based on the level of redundancy in the transmission, but it will have a constant rate for each execution. The underlying protocols used is UDP and IP. The IP packets are then transported from the AP over to the client using the datagrams in unicast WiFi. The application is fully written by the author in C++ without the use of any third-party libraries except the standard system libraries.

The application fix the throughput of data packets to 5 Mbit/s. In the case when additional network coded packets are sent, the total throughput will increase accordingly so it always maintains 5 Mbit/s of data packets. Since the addition of coded packets are known in advance in this experiment, the throughput will be held at a constant but higher level depending on the amount of redundancy sent. An example is if the redundancy is 20%, the constant transmission speed will be set to 6 Mbit/s. In the case when ARQ is being used as a reliability mechanism in WiFi, the application throughput is not changed, and will constantly have a 5 Mbit/s throughput of data packets in addition to bursts of retransmitted packets.

WiFi Configuration

The WiFi configuration is changing during the experiment. This section will highlight which parameters that are constant and which are changing during the different execution rounds of the experiment. The following list contains information about the WiFi configuration that is staying the same throughout the whole experiment:

- WiFi frequency: 2.4 GHz
- WiFi channel: 6
- 802.11n configuration:
  - Channel width: 20 MHz (HT20)
  - 1 Spacial Stream (No MIMO)
  - Short Guard Interval disabled
  - Space-Time Block Coding disabled
  - AMSDU disabled

The WiFi configuration that stays the same during the different execution rounds, but will differ throughout the whole experiment is the following:

- AMPDU on/off
- Max Retransmissions
- Modulation and coding scheme
- Transmission power

Being able to fix the modulation and coding scheme as well as the retransmissions throughout whole execution of a round required implementation of a fixed rate adaption algorithm in the driver. Instead of separating the computers far apart from each other to create a decent packet loss probability for the experiments, metallic bottles are placed around the antennas to create attenuation of the signal. A picture of one of the antennas in the experiment is illustrated in Figure 3.9. The major reason to do this, compared to separating the computers far apart was to encapsulate the experiment as much as possible from the outside world, so that the loss percentage was kept with low fluctuation. If the computers were separated far apart, potential random interference could cause faulty results. The next section will explain how the execution rounds of the experiment are performed.

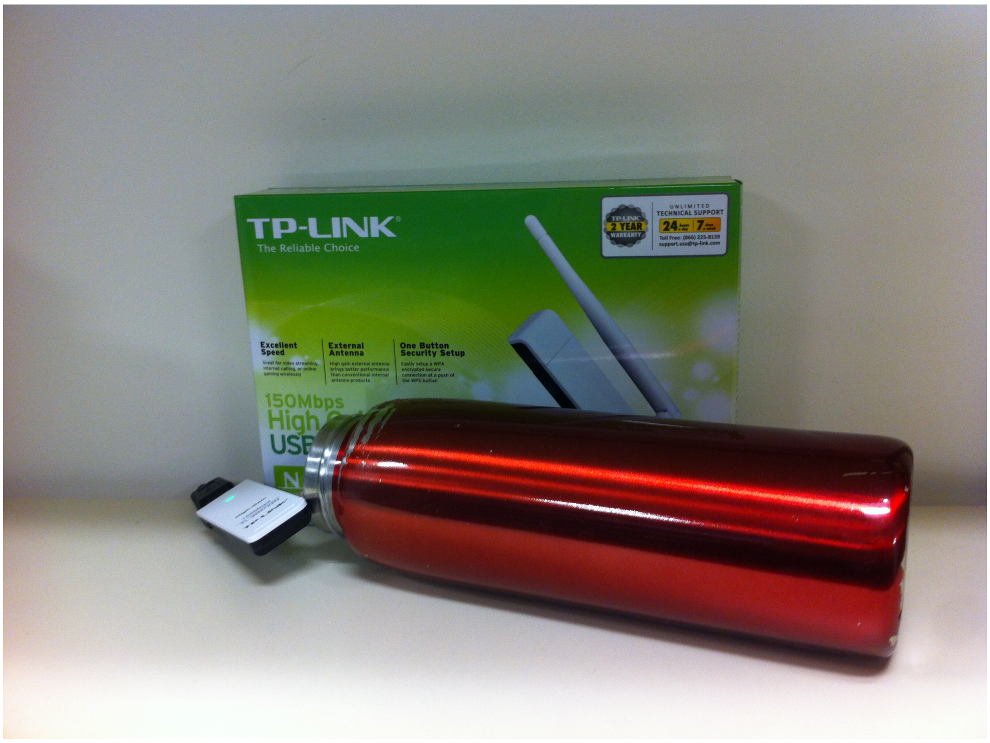


Figure 3.9: Description of how attenuation of signal is achieved, and consequently packet losses.

3.5 Execution

The experiment is executed such that comparing the number of packets retransmitted using ARQ to the number of redundant packets sent using network coding is easy and can be concluding. In order to do that, each round of execution adjusts the physical configuration of the channel to be fixed but different from previous rounds. Each round will consist of the AP repeatedly sending the file to the client using first two configurations of ARQ and then several configurations of network coding.

Execution Rounds using Different Channel Configurations

Each round of execution consists of fixing the channel parameters so that ARQ as a reliability mechanism can be compared to network coding as a reliability mechanism for different packet loss probabilities. The following list shows the different channel configurations used in the different execution rounds.

Table 3.4: Configurations for each round of execution in unicast WiFi.

Configuration	Network Coding	ARQ	AMPDU
Raw	Off	Off	Off
ARQ	Off	On	Off
ARQ-Aggregated	Off	On	On
NC-2.5%	On	Off	Off
NC-5%	On	Off	Off
NC-12.5%	On	Off	Off
NC-35%	On	Off	Off
NC-50%	On	Off	Off

– Channel Modulation: 64-QAM, Code Rate: 3/4, Transmission power: 6 dBm

– Channel Modulation: 64-QAM, Code Rate: 3/4, Transmission power: 1 dBm

– Channel Modulation: 64-QAM, Code Rate: 5/6, Transmission power: 4 dBm

– Channel Modulation: 64-QAM, Code Rate: 5/6, Transmission power: 3 dBm

– Channel Modulation: 64-QAM, Code Rate: 5/6, Transmission power: 2 dBm

These channel configurations are adjusted so that the packet loss probability should range between 1% and 35%.

Each Execution Round

For each round of execution, both ARQ and network coding will be tested as a reliability mechanism. When ARQ is enabled, the default 7 maximum transmissions attempts will be used. Table 3.4 shows all the configurations being run within each round. NC-X means network coding with X percent redundant packets. Using several different amounts of redundancy in network coding makes it possible to find just the right amount of redundancy needed in order to maintain reliable communication for the different rounds of execution.

Remote Execution via ssh

When executing the rounds, both client and the access point is connected with wire to the internet. In that way, it is possible to login to the computers via ssh and execute the code on the machines without being close by. The interference from people moving around while running the experiments is not preferable, so therefore remotely executing the code was chosen instead of physically go to the client and access point and start the application.

3.6 Results

The results are naturally sectioned into the different rounds of execution. The diagrams that follows describes the throughput, loss and redundancy in terms of average rate. The blue bar is the average throughput of data packets received at the client. The red bar is the average rate of lost data packets at the client. The orange bar is the average rate of redundancy sent at the access point.

Modulation: 64-QAM, CR: 3/4

Figure 3.10 shows the results of the experiment using 64-QAM modulation with 3/4 code rate and with 6 dBm and 1 dBm as transmission power. The results show the number of extra packets transmitted in the different ARQ and NC configurations in hope of eliminating loss in these rounds.

Modulation: 64-QAM, CR: 5/6

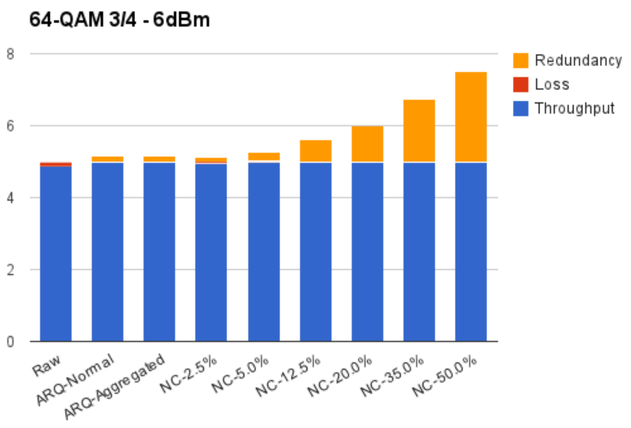
Figure 3.11 shows the result of the experiment using 64-QAM modulation with 5/6 code rate and with 4 dBm, 3 dBm and 2 dBm as transmission power. The results show the number of extra packets transmitted in the different ARQ and NC configurations in hope of eliminating loss in these rounds. When a configuration is denoted unstable, it means that with this configuration there were packet losses that were fluctuating so much that it almost lost whole blocks of packets at times.

3.7 Analysis

The results show that in the different physical configurations, network coding performs significantly worse than both ARQ using aggregated frames as well as when it is not. In order for the network coded schemes to achieve zero packet loss, a lot more redundancy had to be sent compared to in the case with ARQ. The reasons for this is discussed in the next sections. Because the theoretical gains using network coding compared to ARQ is larger in pre-802.11n compared to 802.11n with aggregated frames, the network coding is only applied without aggregated frames and not with aggregated frames as well.

Varying Channel Quality within each Execution

If the channel quality is varying enough between the transmissions of blocks in NC, some of the blocks will need a lot more redundancy to recover the packets within that block. Since the redundancy is kept constant throughout the transmission of all blocks, it causes the other blocks to receive unnecessary redundancy, and hence the results became as shown.



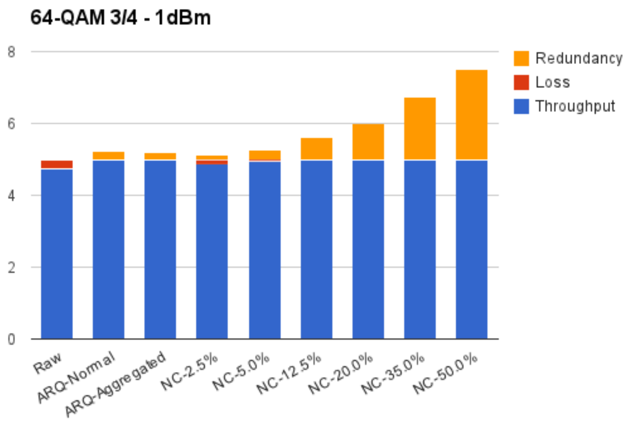


Figure 3.10: Experimental results with Modulation: 64-QAM, CR: 3/4 for both Tx-Power: 6 dBm and 1dBm.

The theoretical results showed that there were minimal gains by using network coding as opposed to ARQ in unicast WiFi. In addition to that, the theory had certain assumptions. One of them were that the WiFi system would be able to sense the channel quality good enough to perfectly predict the number of packets needed in order to get reliable communication. Even though the experiment did not sense the channel and used that information, this experiment showed that the quality of the channel is fluctuating so much that it would practically be impossible to predict exactly the number of redundant packets needed. Because of this fact, is it evident that using network coding in a unicast WiFi will not give any throughput gains.

Since the results showed no gains by using network coding in point-to-point unicast WiFi, the next chapter will explore network coding in a multicast setting using point-to-multipoint multicast in WiFi.

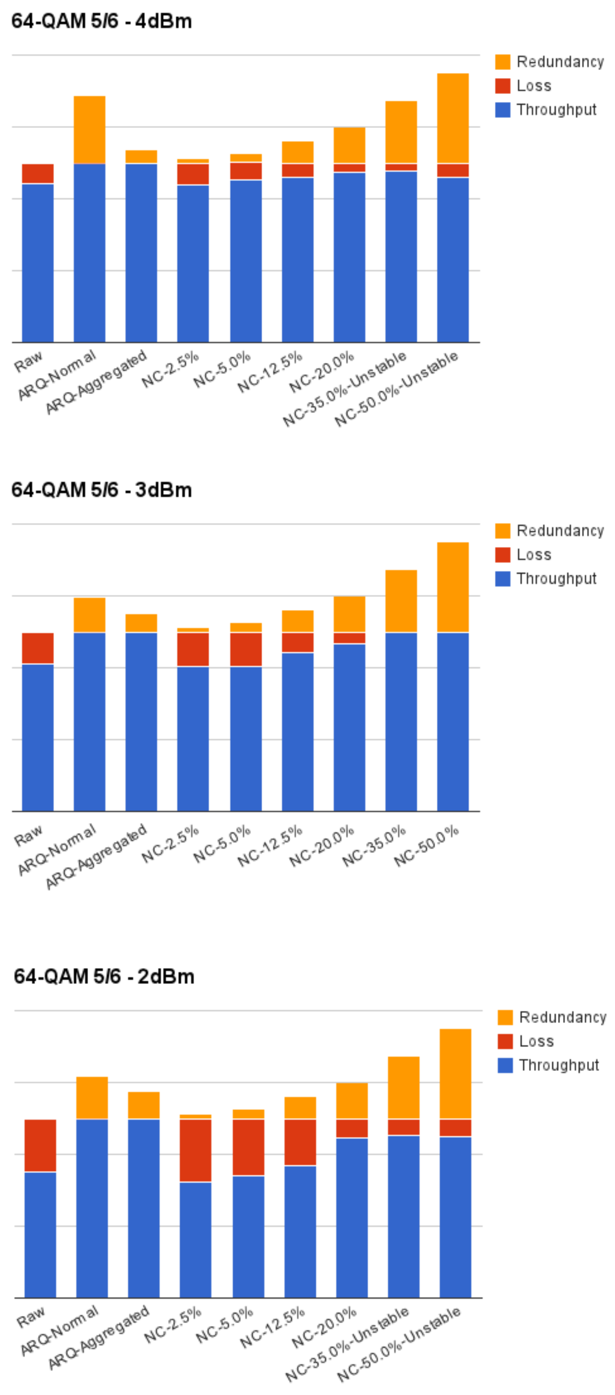


Figure 3.11: Experimental results with Modulation: 64-QAM, CR: 3/4, Tx-Power: 1 dBm.

Chapter 4

Multicast WiFi Design and Theory

The results from applying intra-session random linear network coding to a point-to-point unicast scenario showed no gains in terms of throughput compared to ARQ. That motivates looking into different scenarios. Here, this is chosen to be point-to-multipoint multicast networking. When several clients are using the same access point in WiFi, currently, the only way for each client to have reliable transmission is by using several unicast flows, where each unicast flow is using ARQ. This chapter proposes both design and theoretical performance of two new schemes for reliable data transfer among several clients using the same access point. The first section discusses the design and architecture of such schemes.

4.1 Design and Architecture

This section describes briefly the design of the traditional way of achieving reliable transmission with multiple clients. In addition, two other designs using NC and multicast are described, which still have reliable transmission.

Multiple Unicast Design

Figure 4.1 shows a typical design for an access point that transmits reliable traffic to several clients. The access point sends each flow of data with unicast to each corresponding client. The clients acknowledge the packets and if needed, get retransmitted by the access point. This is currently the only way to achieve reliable transfer of data in the current WiFi standard, 802.11n.

RLNC Multicast Design without Per-Packet Feedback

Figure 4.2 shows a design that allows multiple flows of data to be transferred reliably using RLNC on top of a broadcast transmission without per-packet feedback. This means that the access point does not know which particular packets were lost. The only feedback that is needed is an ACK indicating that a client is finished decoding

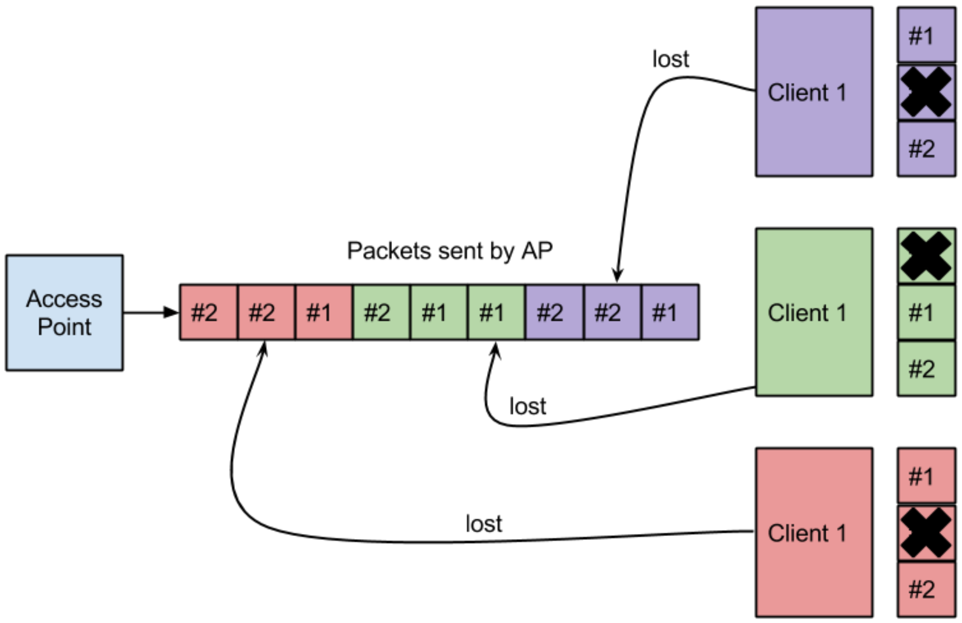


Figure 4.1: Design of an access point delivering reliable data to several clients in current WiFi standard.

a block. Since the access point has no information regarding which packets that are lost, inter-session network coding is being used. The coded packets are created using RLNC, where the block of packets includes equal amounts of data packets from each client. In the case of Figure 4.2, it means all 6 data packets. The next design will address the possible inefficiency of having to code over the whole block.

XOR-NC Multicast WiFi Design with Per-Packet Feedback

Figure 4.3 shows a design that allows multiple flows of data to be transferred reliably using network coding in multicast with per-packet feedback. By using per-packet ACK/NACK from each client, the access point can generate a map of which clients that have lost which packets. Since the access point have knowledge of which packets that are lost at each client, it does not have to code the network coded packets over the whole block, but can instead code in a more clever way. The details about the coding scheme is presented later in the chapter. The concept of the scheme is to XOR some of the packets that are lost together such that several clients that lost only one of those packets can decode the received packet into their lost one. By exploiting the fact that packet losses are not perfectly correlated among clients, the

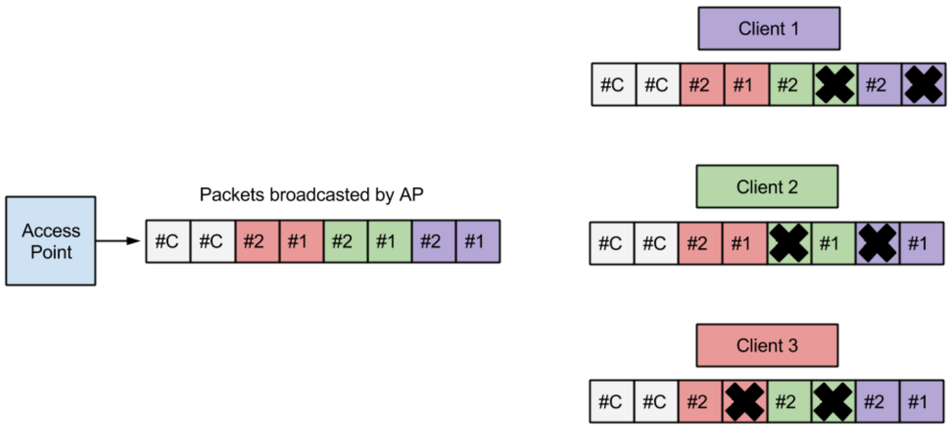


Figure 4.2: Design of an access point delivering reliable data to several clients using network coding in a multicast network without per-packet feedback.

Table 4.1: Key variables for performance measures in multicast.

Packet loss probability	p
Total data packets	d
Number of clients	c
Data packets per client	$d_c = \frac{d}{c}$

number of redundant packets needed to be transmitted can be reduced compared to pure retransmissions in ARQ.

4.2 Theoretical Performance

Analyzing the theoretical performance of the suggested new designs can show potential performance improvements. To verify the theoretical results, experiments were performed, as described in the next chapter. But first, the present chapter quantifies the theoretical performance of the suggested new designs, and then compare it to the one that currently exists. Table 4.1 contains some key variables that will be used in order to analyze the performance.

Performance in Multiple Unicast WiFi Design

In order to compare the suggested designs, the performance of the current default way of sending reliable streams must be analyzed. Using multiple unicast streams is

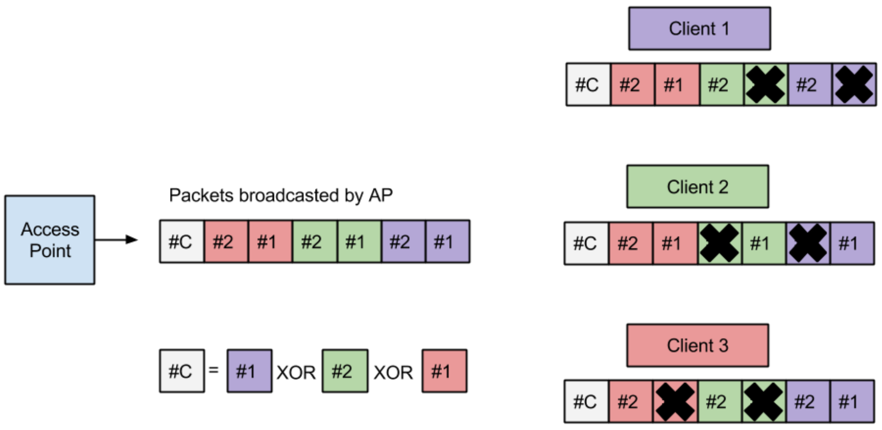


Figure 4.3: Design of an access point delivering reliable data to several clients using network coding in a multicast network with per-packet feedback.

currently the only way of achieving reliable data transfer to multiple clients. The rest of the section will analyze this design.

The average number of retransmissions given by the ARQ mechanism in a WiFi unicast flow is

$$r = d_c(\frac{p}{1-p}) \ .$$

Therefore the total number of retransmissions over all clients will on the average be

$$total = cr = d(\frac{p}{1-p}) \ .$$

Figure 4.4 shows the probability of receiving all 100 packets assuming a 10% packet loss and a variable restriction on the number of retransmissions and clients. From the expression, one sees that the number of clients has no impact on the probability of receiving all data packets. As will be shown later in this chapter, this is not necessarily the case in other designs.

Performance in Multicast WiFi Design without Per-Packet Feedback

Rather than retransmitting single packets that were lost at each client, consider now sending packets that are coded over a certain block size, where the block contains data packets addressed to all clients. Each block consists of the same number of data

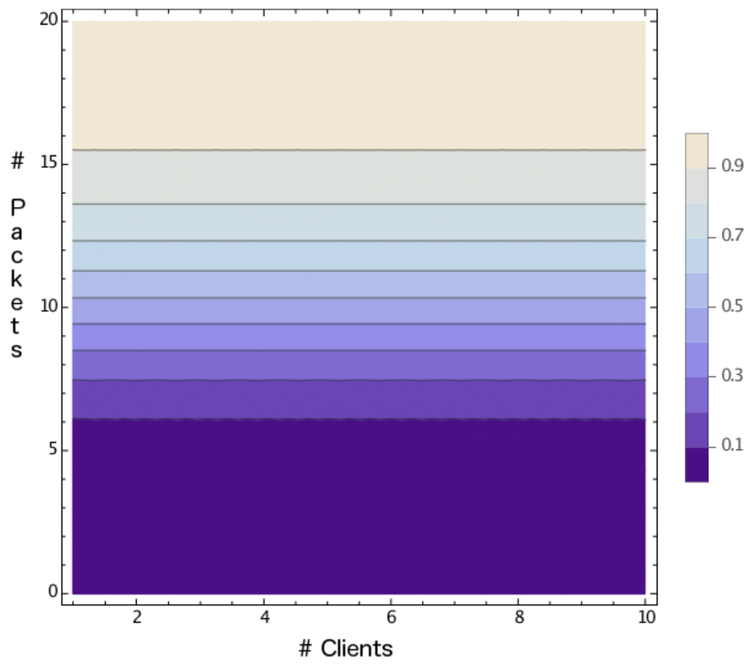


Figure 4.4: Probability of receiving all packets at all clients using unicast flows with ARQ for variable number of maximum total retransmission packets and clients in the system with packet loss probability of 10% and total of 100 data packets.

packets for each client, assuming they demand equal number of data packets. By the nature of WiFi, each client will receive the packets originally addressed to the other clients, in addition to themselves. A term self-addressed data packet is from now referring to a packet that contains data of direct interest for the client. In this approach, a coded packet is a linear combination of all the packets in the block. The following list shows the assumptions made in order to quantify the performance:

– The clients need to recover all the lost packets, even when they have not lost any of their own self-addressed packets. This is likely the case for sufficiently large block size and packet loss probability.

– The packet loss probability is equal for all clients.

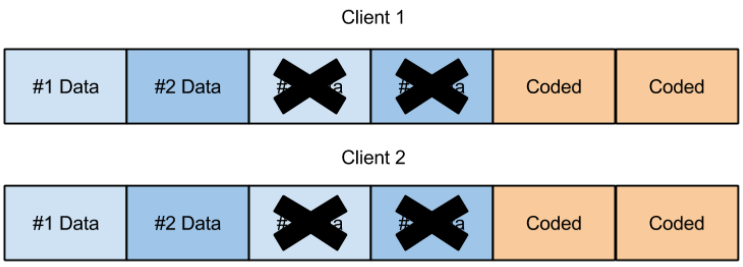


Figure 4.5: Perfect positively correlated packet loss among the clients.

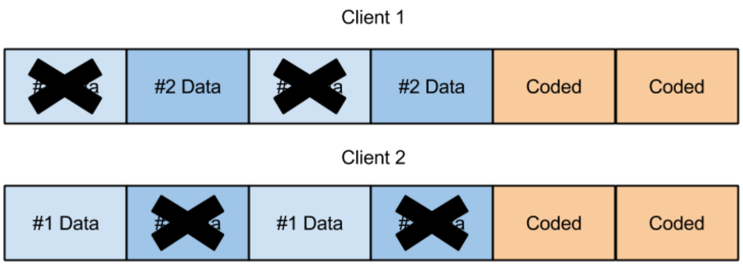


Figure 4.6: Perfect negatively correlated packet loss among the clients.

Packet Loss Correlation

Without doing experiments it is hard to know how the loss of packets are correlated. The reason for this is that the correlation may vary in time as well as with topologies. The correlation of loss between the packets sent, must be distinguished from the correlation of loss between clients for one packet. In the present thesis, packet loss correlation refers to the latter. If the former exist, it will be negligible due to the temporal dispersion of packets, as shown later in the chapter. Three cases of packet loss correlation are investigated.

Perfect Positively Correlated Loss When the packet losses are positively correlated, it means that every time any client loses a packet, also all the other clients do. Figure 4.5 shows a situation with positively correlated loss among two clients. On both clients, the two last packets that were sent by the AP are lost, and both clients need 2 coded packets to decode all of their self-addressed packets.

Perfect Negatively Correlated When the packet losses are negatively correlated, it means that every time a client loses a packet, all the other clients do not lose that packet. Figure 4.6 shows an example of this case. In this scenario, the AP need to

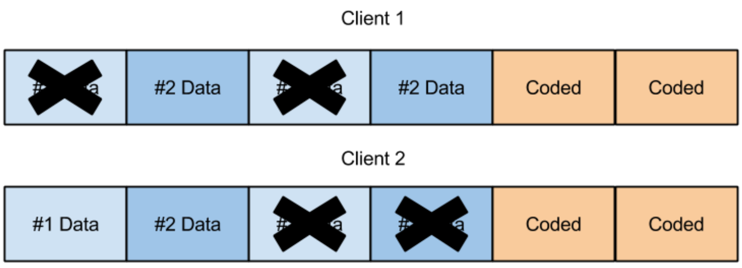


Figure 4.7: Non-correlated packet loss among the clients.

transmit two coded packets to decode all of their self-addressed packets.

Perfectly Non-Correlated In the non-correlated case, the packet losses are identically and independently distributed, and each packet loss probability follows a Bernoulli distribution with equal probability. An example can be seen in Figure 4.7. The Cumulative Distribution Function (CDF) for the binomial distribution with parameters l, d and p will give the probability that at most l number of packets are lost given the total number of sent data packets, d, at a client with a given packet loss probability, p. The coded packets that must be sent to recover this loss can be expressed as

$$r = \frac{l}{1 - \frac{l}{d}} ,$$

assuming the coded packets will be lost with the same probability as the data packets. The CDF for the binomial distribution will only give the probability that at most l number of packets are lost at one client. Hence, to find the probability that at most l number of packets are lost at all clients, the CDFs are multiplied,

$$P = \left(\sum_{i=0}^{l} \left[\binom{d}{i} (1-p)^{d-i} p^i \right] \right)^c .$$

This expresses the probability that all the clients lose at most l packets having d data packets in each block with packet loss of p. Using the expressions above and eliminating l, the function P is shown graphically in Figure 4.8, for the case where $d = 100$ and $p = 0.1$. The figure shows that with increasing number of clients, the number of coded packets that must be sent is also increasing. This is no improvement compared to multiple unicast flows, indeed it is doing worse than multiple unicast flows.

By not knowing how the packet loss is correlated among the clients, it is important to conduct experiments to measure the real correlation. Most likely, none of the three

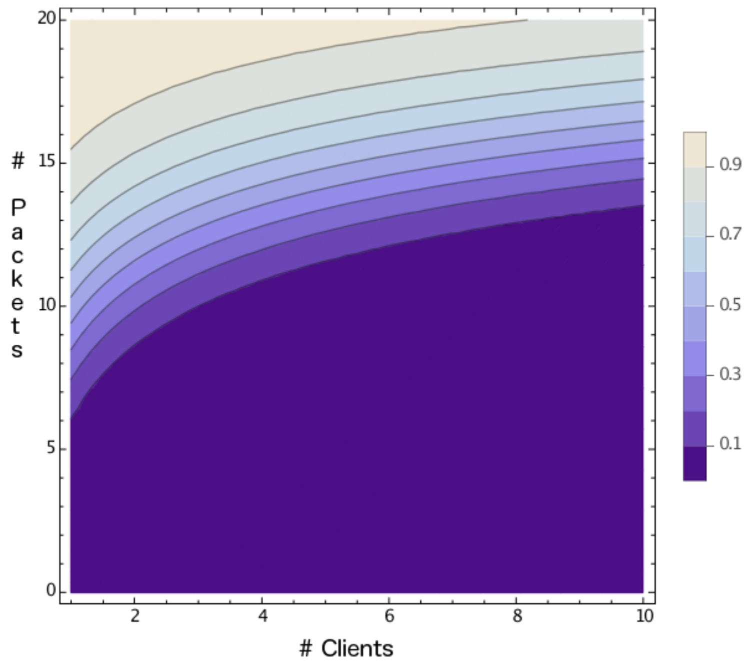


Figure 4.8: Probability of successful decoding at all clients as function of number of clients and maximum number of redundancy packets. The result is based on packet loss probability of 10% and block size of 100 data packets.

schemes of correlation is the actual case, but the real correlation lies somewhere in between. Whether it is closer to being correlated than uncorrelated is hard to say a priori.

Performance in Multicast WiFi Design with Per-Packet Feedback

Instead of coding the redundancy over the whole block, it is better to code each packet over a selected set packets that are both lost and self-addressed at each client. In that way, if a client loses a large number of packets that are not self-addressed, the client should not need to recover them. In order for this to work, it is required that the WiFi system uses NACKs so that the AP can find out which packets are lost at each client. In this way, the access point can use that information in the encoding of the coded packets. Several coding schemes can then be used based on XOR-ing selected packets together.

Table 4.2: Theoretical performance for XOR-scheme with two clients

Two Clients	# packets	affection rate	XOR factor
Lost by two	np^2	1	1
Lost by one	$2np(1-p)$	$\frac{1}{2}$	$\frac{1}{2}$

Schemes to be Analyzed

The gain using an XOR scheme depends on the number of clients in the system, block size as well as packet loss correlation. in order to analyze the performance, it is assumed now that the packet losses are uncorrelated. First an optimal scheme with 2 and 3 clients is discussed and then a general coding scheme is considered. The more clients in the system, the more complex is the derivation of optimal performance. The theoretical performance is assuming an infinitely large block size.

Optimal XOR Performance with two Clients

Table 4.2 shows the parameters used to calculate the average number of coded packets that must be sent in order to decode all the self-addressed packets on all the clients. Since there are only two clients, a single transmitted packet can only be lost by either 1 client or by 2 clients. With a total of n packets to send, the average number of packets that are lost by both clients is np^2. For each of those losses, there is a client that had that packet self-addressed. This client is always affected by the loss, since the loss is on both clients, hence the affection rate is 1. Since those packets were lost by all the clients, there are no other packets that can be XOR-ed with them. Therefore, it is not possible to reduce the number of total coded packets, which gives an XOR factor of 1.

The average number of packets that are lost by one client equals $2np(1-p)$. For each of those losses, there is on average only half of them that are lost and self-addressed at the same time. Therefore, only half of the losses affects the clients, which gives an affection rate of 1/2. By definition, each packet that was lost by one client was received by the other. Moreover, since each client loses on average the same number of packets, all the packets that were lost by one and addressed to client A can be XOR-ed with all the packets that were lost by one and addressed to the other client, B. Therefore, this gives an XOR factor of 1/2.

The total number of redundant packets that on average must be sent to deliver all the packets for all the clients is therefore equal to,

$$np^2 \cdot 1 \cdot 1 + 2np(1-p) \cdot \frac{1}{2} \cdot \frac{1}{2} = np[p + \frac{1}{2}(1-p)] \ .$$

Table 4.3: Theoretical performance for XOR-scheme with three clients

Three Clients	# packets	affection rate	XOR factor
Lost by three	np^3	1	1
Lost by two	$3np^2(1-p)$	$\frac{2}{3}$	1
Lost by one	$3np(1-p)^2$	$\frac{1}{3}$	$\frac{1}{3}(1-\frac{2p}{(1-p)})$

Optimal XOR Performance with three Clients

Table 4.3 shows parameters used to calculate the average number of coded packets that must be sent in order to decode all the self-addressed packets on all the clients. Since there are 3 clients, a packet can only be lost by either one, two or by three clients. Having a total of n packets needed to be transmitted, the average number of packets that are lost by three clients is np^3. For each of those losses, there is a client having that packet self-addressed, and that client is always affected by the loss, since the loss is on all three clients, hence the affection rate of 1. Since those packets were lost by all the clients, there are no other packets that can be XOR-ed with these packets and reduce the number of total coded packets, giving an XOR factor of 1.

The average number of packets that are lost by 2 is $3np^2(1-p)$. For each of those losses, there is on average $\frac{2}{3}$ of them that were both lost and self-addressed at the same time, and hence only $\frac{2}{3}$ of the losses affected the clients, giving an affection rate of $\frac{2}{3}$. By restricting the packet loss to less than a third, $p < \frac{1}{3}$, there are on average more packets that are lost by one client and self-addressed, than lost by two clients and self-addressed. The packets that were lost by two clients can only be XOR-ed with a packet that was lost by one client, where each client received only one of the two packets. Since there are on average more packets that are lost by one client and self-addressed, than lost by two clients and self-addressed, one can, on average, XOR all the packets that are self-addressed and lost by two client with some of the packets that are self-addressed and lost by one client. Even though all the packets will be XOR-ed with packets that were lost by 1, the benefit of XOR-ing is reflected in the group of packets that were lost by 1, hence XOR factor of 1.

The average number of packets that are lost by one client is $3np(1-p)^2$. For each of those losses, there is on average only 1/3 of them that were lost and self-addressed, and therefore only 1/3 of the losses affect the clients. Hence, the affection rate equals 1/3. Since already $\frac{2}{3} * 2np^2(1-p)$ of the packets that are self-addressed and lost by one client are incorporated into other coded packets, the remaining information from the packets that were lost by 1 must be sent. By definition, each of the packets that were lost by one client were received by all the others. Since any packet that was lost by one client was received by all the others, and each client loses

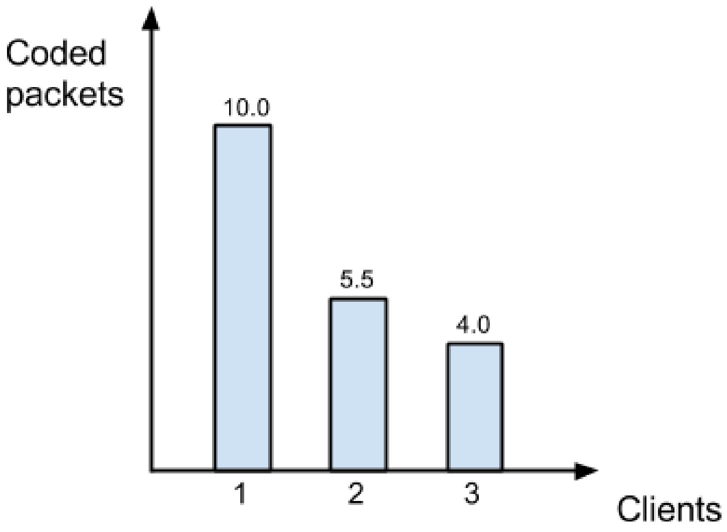


Figure 4.9: The number of redundancy packets needed to be transmitted with block size of 100 and packet loss probability of 10% using XOR-schemes assuming the redundant packets cannot be lost.

on average the same number of packets, the remaining packets that were lost by one client and contain information yet to be sent, can be XOR-ed with 2 other packets also lost by one client. The XOR factor in which the affected packets that were lost by 1 is therefore given by

$$\frac{1}{3} \cdot \frac{\frac{1}{3} \cdot 3np(1-p)^2 - \frac{2}{3} \cdot 3np^2(1-p)}{\frac{1}{3} \cdot 3np(1-p)^2} = \frac{1}{3} \cdot (1 - \frac{2p}{(1-p)}) \ .$$

The total number of redundant packets that on average must be sent to deliver all the packets for all the clients is then

$$\begin{aligned} & np^3 \cdot 1 \cdot 1 + 3np^2(1-p) \cdot \frac{2}{3} \cdot 1 + 3np(1-p)^2 \cdot \frac{1}{3} \cdot \frac{1}{3} \cdot (1 - \frac{2p}{1-p}) \\ & = np[p^2 + 2p(1-p) + \frac{1}{3}(1-p)^2(1 - \frac{2p}{1-p})] \ . \end{aligned}$$

Performance in XOR scheme Using an Example

So far, the optimal XOR scheme has been considered with maximum 3 clients. Figure 4.9 shows the number of coded packets needed to be transmitted in order to deliver the self-addressed packets reliably in case of having 100 packets to send and an average packet loss probability of 10%. The result is based on the assumption of no loss of the redundant packets. The data show a strong decay in the number of redundant packets needed to transmitted with increasing number of clients. In the perfect negatively correlated case, the number of redundant packets would be 5 and $3\frac{1}{3}$ with 2 and 3 clients, respectively. In the present results, the required number of redundant transmitted packets is higher. This is because the packet losses are assumed uncorrelated, and any packet can be lost by more than one client.

4.3 General Solution of XOR Coding

Assuming the packets sent to recover packet loss cannot be lost, the upper bounds on the number of redundant packets required to be sent from the access point using XOR is the number of lost and self-addressed packets in total at all clients. This is the same as ARQ will have. The lower bound is the number of lost and self-addressed packets at the client that has the most loss in each block. This gives strong incentive to look further into this scheme. The larger block size, as well as the more clients involved, should therefore be beneficial compared to ARQ. For an arbitrary number of clients, the problem of XOR-ing lost packets at the access point such that only the minimum number of packets must be transmitted is an NP-complete problem. Therefore, the optimal solution will be infeasible practically, but theoretically it is important to understand in order to create a practical good heuristic solution.

Optimal Solution

The problem can be formalized to consist of clients that have a subset of all the packets, and wants a specific subset of all the packets from the access point. Instead of partitioning the packets into blocks, the solution is based on a single large block, where all packets needed to be transmitted at the access point is in that block. The design of an optimal solution is illustrated in Figure 4.10. The colored arrows in the figure illustrate a packet needed at the client the arrow is pointing at. In order to find the minimum number of transmissions, partitioning the nodes into the fewest number of cliques must be determined. Partitioning a graph into as few cliques as possible is known to be NP-complete and was one of Richard Karp's original 21 problems shown NP-complete [Kar72]. The name Karp used for the problem was clique cover, and the NP-completeness of clique cover comes from the reduction of graph k-colorability, as explained further in [Wik]. In order to see the link between clique cover and graph k-colorability, a transformation of the graph, G, from graph colorability into its inverse graph, G', must be created. Now, being able to partition G' into k cliques corresponds exactly to finding a partition of the nodes of G into k independent sets. After that is done, each of the independent sets can be assigned one color, to create a k-coloring. This shows that solving the clique cover problem is equivalent to solving the graph coloring problem.

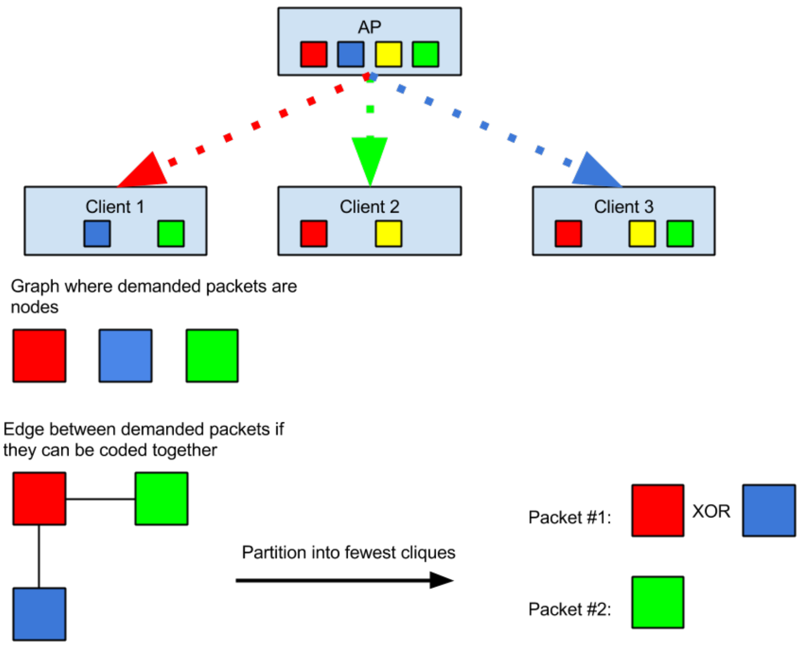

AP
Client 1
Client 2
Client 3
Graph where demanded packets are nodes
Edge between demanded packets if they can be coded together
Partition into fewest cliques
Packet #1:
XOR
Packet #2:

Figure 4.10: Illustration of the problem of finding the minimum number of required transmissions using XORs in a multicast scenario with clients overhearing the channel. Each client has a specific subset and needs a specific subset of packets.

Practical Solution

Since the optimal solution is NP-complete, it makes it practically unsuitable. Work in [JMM93] has been done to find good heuristic algorithms for a problem close to the XOR problem discussed in this chapter. Compared to ARQ, they are they all performing well based on simulation. The difference between the presently discussed XOR problem and the one investigated in [JMM93] is that in the latter, all the clients wanted all packets instead of a subset of them. The heuristic solutions in [JMM93] are still relevant, and need only being adjusted slightly to be applicable in the present XOR problem. In later chapters, one of the solutions from [JMM93], called Degree scheme, will be tested based on data gathered in the experiment explained in the next chapter. Although XOR-schemes have been studied extensively, a scheme developed by the present author is also being investigated. This heuristic-based scheme is explained in Algorithm 4.1. In this algorithm, it is given that the access point receives a list of the packets that were successfully and unsuccessfully received from each client. The sequential order of these lists are chosen to be ascending in the client number, an arbitrary choice. Given these lists, Algorithm 4.1 creates the set of coded packets that will enable the clients to recover their packet losses. The size of this set is therefore the number of packets needed to be sent. Typically, a

Algorithm 4.1 Proposed XOR Scheme

```
input: ordered lists from clients that have elements
             marked as either "received" or "lost" from the whole block

declare and initialize an empty set of coded packets
declare and initialize a coded packet to 0
for each list, called i
| while there still are self-addressed packets marked as "lost" in i
| | for the remaining lists including list i, called j
| | | find the first element in j that is self-addressed,
| | | |     marked as "lost" in j, marked as "received" in previous
| | | |     lists that are part of the coded packet and where none of
| | | |     the packets that are part of the coded packet is lost in j
| | | if such element is found
| | | | xor it with the coded packet and include index number from list
| | | | mark that element as "received"
| | if coded packet is not 0
| | | add a copy of the coded packet to the set of coded packets
            and set the coded packet to 0

output: the set of coded packets
```

heuristic solution is hard to analyze regarding performance. Intuitively, the algorithm should be able to combine packets using XOR and therefore perform better than ARQ. However, since it is a heuristic solution it should not necessarily be optimal. In the work [JMM93], simulations were used to verify performance, and based on those results, all of the heuristic solutions presented performed far better than ARQ. The simulation assumed uncorrelated packet loss among the clients as well as equal packet loss probabilities, which seems like the most critical assumptions to make. In general, the performance is very much dependent on those two factors, since if the losses are perfectly positively correlated, there would be no packets to XOR together. Moreover, if a client has no loss, it will be no performance improvement by including it. Because of that, both the algorithm Degree scheme from [JMM93], and Algorithm 4.1 are going to be tested experimentally with real data in later chapters. Algorithm 4.1 is relatively fast with a worst case running time of $O(b^4)$ where b is the block size. Practically it will be a lot faster, since the number of received packets is usually much larger than the number of lost packets. Furthermore, there are also usually a lot fewer clients in a system than packets in a block. With relatively low packet loss probability, Algorithm 4.1 would be suitable as an algorithm to be tested practically in an experiment with per-packet feedback, which goes beyond the scope of the present thesis.

Comparing the Multicast Schemes

As shown previously, the scheme with multiple unicast flows theoretically is more promising than the RLNC multicast scheme with no per-packet feedback. However, if per-packet feedback is used through NACKs, the multiple unicast scheme is outperformed significantly by the XOR scheme, even with only a few clients. The general result is that the scheme with per-packet feedback improves its performance the more clients there are in the system. Moreover, it seems to outperform both the multiple unicast scheme as well as the other proposed scheme. Even though this scheme seems to be the best theoretically, it is also the most complex out of the proposed schemes. Since it will require per-packet feedback, it must either be implemented in the MAC-layer or by its own mechanisms in other layers. On the opposite side, the scheme with no per-packet feedback can easily be implemented in the application space using ACKs when the client has decoded. At the same time, the scheme with no per-packet feedback can gather valuable data at each client, and a combined experimental and theoretical result can be derived regarding the XOR schemes with per-packet feedback. Experiments using the design where no per-packet feedback is required is therefore presented in the next chapter.

Chapter 5

Multicast WiFi Experiment and Analysis

This chapter presents a WiFi multicast experiment using RLNC where no per-packet feedback is required. The motivation for doing this experiment is to verify the theoretical results as well as provide experimental data to verify performance in a setting where per-packet feedback is available using XOR schemes.

5.1 Setup

The experiment is conducted in the same office as the unicast experiment, having small physical distances between the nodes in the network. The following sections will discuss the topology, architecture, software, hardware and WiFi configuration used in the experiment.

Topology

The topology of this experiment is shown in Figure 5.1. It includes one access points and up to 3 clients. They are separated into a T-shape, where the access point is in the middle. The reason for separating them in such a way is to make the packet losses uncorrelated, so the data collected will give the best results as possible for the XOR scheme discussed in the next chapter.

Architecture

The architecture used to send data reliably in the experiment is illustrated in Figure 5.2. The experiment is conducted in the application space, for simplicity as well as efficiency. The structure of the experiment separated into network layers is illustrated in Figure 5.3. The access point broadcasts the data to all clients and each client responds with an acknowledgement when the whole block has been received successfully. To eliminate any encoding delays during the experiment, the coded packets are pre-coded in the following way. Each block of data packets also contains a large amount of coded packets ready to be sent, such that the acknowledgements

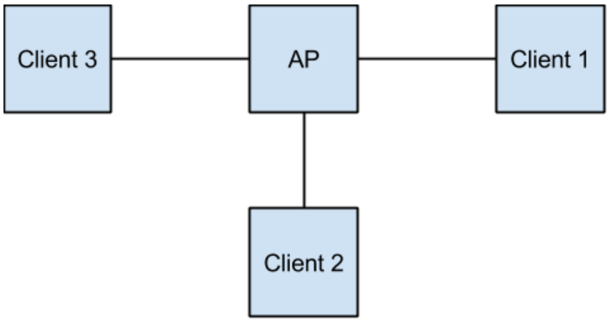


Figure 5.1: Topology of the multicast experiment. The access point communicates with the all the clients, while each client only communicates with the access point.

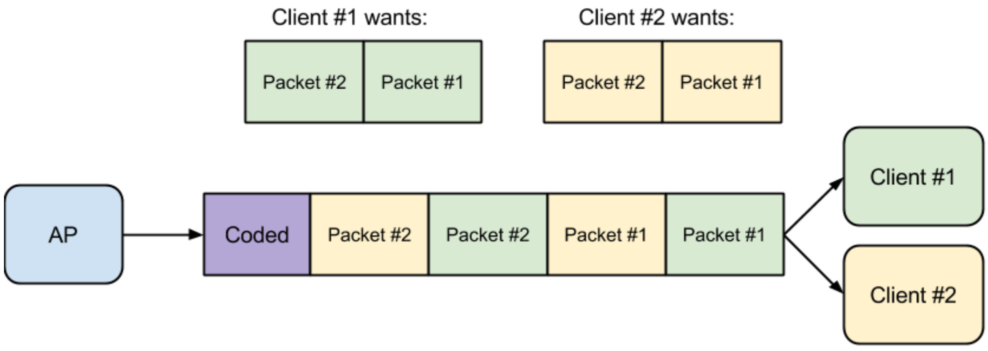


Figure 5.2: Architecture used to send data reliably using RLNC in a multicast WiFi. When a coded packet is sent, it is coded over the whole previous block of packets, which involves packets addressed to different clients.

will be received before the block runs out of redundant packets. The access point will stop sending the coded packets for a block when it has received acknowledgements from all clients.

The structure of all packets that are sent is illustrated in Figure 5.4. The meaning of the fields in the coded and uncoded packet is explained in Section 3.3 under Architecture, as they have the same meaning as in the unicast experiment. The only exception is the type field. In this experiment, it also incorporates the client number. The reason is that the type will determine not only if it is a coded packet or uncoded packet, but also to which client it is directly addressed for. The acknowledgement packet is returned from the clients with the block ID for the corresponding block that was successfully decoded and their own client number.

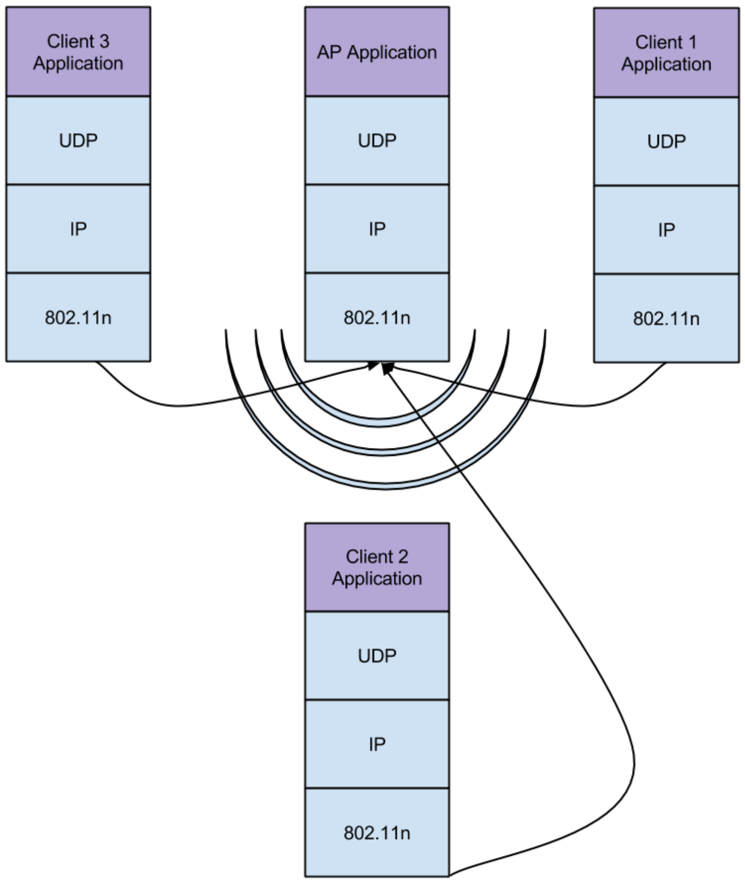


Figure 5.3: The network layered architecture used in the multicast experiment.

The AP has files of size 2281600 bytes it wants to transmit to each client. These files are divided into packets of total size 1472 bytes, which includes the overhead. The total number of data packets to be sent using the ARQ mechanism is therefore 1550 per client, and the total number of data packets to be sent using the network coding scheme is therefore 1558 per client, adding trailing 0s to the last packet as padding. Regardless of the type of reliability mechanism used, the packets are sent in an interleaving pattern as shown in Figure 5.2.

Hardware

The hardware from the unicast experiment is reused in this experiment as hardware for the access point and client 1. In addition to that, two additional clients use two

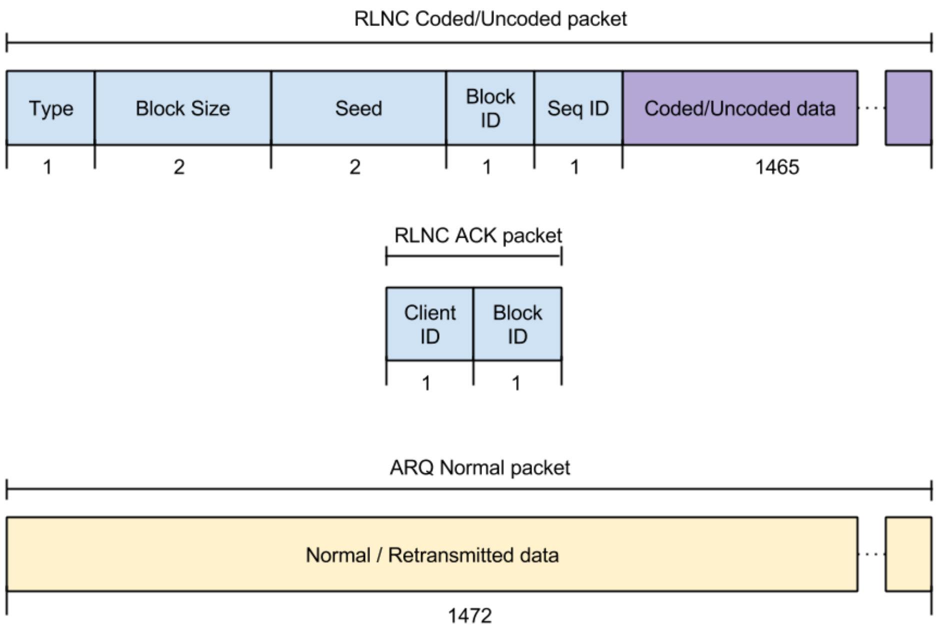


Figure 5.4: The structure of the coded, uncoded and acknowledgement packets when using network coding, and the normal packet when ARQ is used in the multicast experiment. The numbers indicate the number of bytes.

other computers. The specifications for client 2 and 3 are the following:

- Motherboard: Dell 0WF887
- CPU: 32 bit, Intel(R) Pentium 4, 2533 MHz-3600 MHz
- RAM: 512 MB, 2·256 MB, 333 MHz SDRAM
- Operating System: Ubuntu 12.04 32-bit, Kernel version: 3.8.0-31-generic
- WiFi card:
  - Model: Atheros Communications, Inc. AR9271 802.11n
  - Frequency: 2.4 GHz
  - Max bandwidth: 150 Mbps
  - Chipset: Atheros AR9271
  - Driver: ath9k_htc

Software

The software used to conduct this experiment is written in the application layer. The underlying protocols used are UDP and IP, just like in the unicast experiment. The IP packets are then transported between the access point and the clients using datagrams in WiFi. As in the unicast experiment, the software is fully written by the present author in C++, and the only third-party libraries used are the system libraries.

In the case where the data is sent with multiple unicast flows and ARQ is used as a reliability mechanism, the throughput of the data packets is held constant at 5 Mbit/s, with additional bursts of retransmissions. In the case where the data are multicasted, the application sets the throughput of data and redundancy in total to be constant at 5 Mbit/s. Since the amount of redundancy needed is unknown during the execution, the time to perform the experiment varies depending on the channel conditions. Keeping the throughput of data and redundancy to a constant rate ensures that the performance is not affected by physical constraints of the channel.

WiFi Configuration

The WiFi configuration has many similarities to the unicast experiment, and all the items on the list of constant parameters in the paragraph about WiFi configuration in Section 3.4 will remain constant throughout this experiment as well. In addition, the physical configuration is kept constant throughout the whole experiment, as opposed to the unicast experiment. In order to ensure constant Modulation and Coding Scheme (MCS), fixed rate adaption algorithms were developed. Since this experiment uses both ath9k and ath9k_htc drivers, two rate adaption algorithms were developed. Both algorithms kept the MCS at a constant number, as well as kept the number of retransmissions constant.

In order to adjust to a preferred packet loss probability at each client given a MCS, metallic bottles are placed around the antennas of the WiFi cards for clients and the for the access point, as illustrated in the previous experiment. In addition to the list of constant parameters in the paragraph about WiFi configuration in Section 3.4, the following list includes other parameters that are also kept constant throughout the experiment:

- AMPDU is turned off
- MCS is kept constant
    - At the AP: Modulation 16-QAM, Code Rate: 1/2 (MCS 3)
    - At the Clients: Modulation BPSK, Code Rate: 1/2 (MCS 0)

- Transmission power is kept constant
    - At the AP: 6 dBm
    - At the Clients: 20 dBm

The only elements of the WiFi configuration that vary during the experiment are the number of clients and whether it is multiple unicast or broadcast setting. When using the unicast setting, the maximum number of transmission attempts is set to 7.

5.2 Execution

The execution of the experiment is conducted in the following way. The experiment is divided into rounds of execution, where each round is generally a configuration of a block size and number of clients. When it is not a configuration of a block size, the purpose of the round is to test the corresponding multiple unicast performance. For each round of execution, all the files are being broadcasted to the clients. That is the way to multicast packets in this experiment. The number of clients in this experiment is 2 and 3. The number of retransmitted packets in multiple unicast flows is collected and is later compared to the number of redundant coded packets needed in multicast.

Rounds using Multiple Unicast Flows and ARQ

The only way to compare the proposed scheme is to quantitatively measure their performance. The standard way of achieving reliable communication is through multiple unicast flows using ARQ, so that is considered first. The data collected when executing the multiple unicast rounds are the number of times retransmissions occurred for each client at the access point.

There are only two rounds using multiple unicast flows, one with 2 clients and one with 3 clients. For each of those rounds, the access point sends the packets using the respective 2 and 3 UDP unicast connections. The number of retransmissions are being logged for each client at the access point. When the transmission of all the files to all the clients is completed, the numbers of retransmissions are stored.

Rounds using Multicast and RLNC

The rest of the rounds of execution is performed using multicast and using network coding as a reliability mechanism. Each round consist of sending all the files to either 2 or 3 clients using a specified block size. When all data packets within a block are sent, the access point continues to send redundancy. It continues until it has received an acknowledgement from all clients indicating they have received enough packets to

Table 5.1: Block sizes in different rounds given the number of clients

Round	Block size w/2 clients	Block size w/3 clients
1	8	12
2	12	18
3	16	24
4	20	30
5	24	36
6	28	42
7	32	48
8	36	54
9	40	60
10	44	66

decode their self-addressed packets in the block. Each client acknowledges the access point immediately when it has received all of its self-addressed packets. In the worst case when they have received enough packets to decode the whole block.

Table 5.1 shows the different block sizes for all the rounds given the number of clients. The number of redundant packets needed to be sent for each block is collected at the access point during each round of execution. In addition, also which packets were lost in each block at each client is also collected at each client. This is crucially important to be able to demonstrate the gains in the next chapter when a different reliability scheme is proposed.

Remote Execution via ssh

Similarly to the unicast experiment, the present experiment was also executed remotely for each round. All the clients and the access point are connected with wire to the internet, such that it is possible to login with ssh. This is done for the same reasons as in the unicast experiment; to avoid interference while running the experiments.

Repeating the Multicast Experiment

The whole experiment is repeated 3 times in order to get a more reliable results. The fluctuating environment in WiFi makes it very important to have several repetitions of the whole experiment. The experiment was conducted with approximately 5 hours separation, with a total execution period of 24 hours.

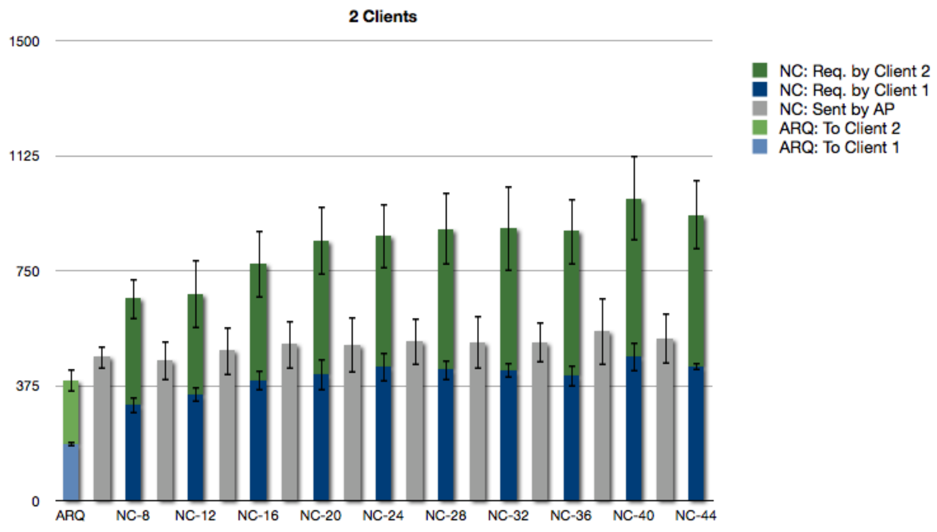


Figure 5.5: Experimental results for the multicast experiment with 2 clients. The vertical axis shows the number of packets and the horizontal axis shows the different reliability configurations.

5.3 Results and Analysis

Two Clients

Figure 5.5 shows the number of packets sent to recover from packet loss in the rounds using 2 clients. Only the additional redundant packets are shown, and not the total number of packets. In the situation with multiple unicast flows, the retransmitted packets from ARQ are seen as the first bar divided by color coding to show the amount each client required, see the legend of Figure 5.5. All the next bars are forming pairs of one grey and one colored bar. Each pair is marked by NC-X, which stands for RLNC with block size of X. Shown as grey bars are the number of coded packets needed to be sent at the access point to recover from the packet losses at the clients. The total demand for redundant packets at each client to recover all self-addressed data is also color coded in each network coding configuration.

Three Clients

Figure 5.6 shows a smilier graphic presentation of the results for the experimental rounds using three clients. Similar to the previous case, the number of retransmitted

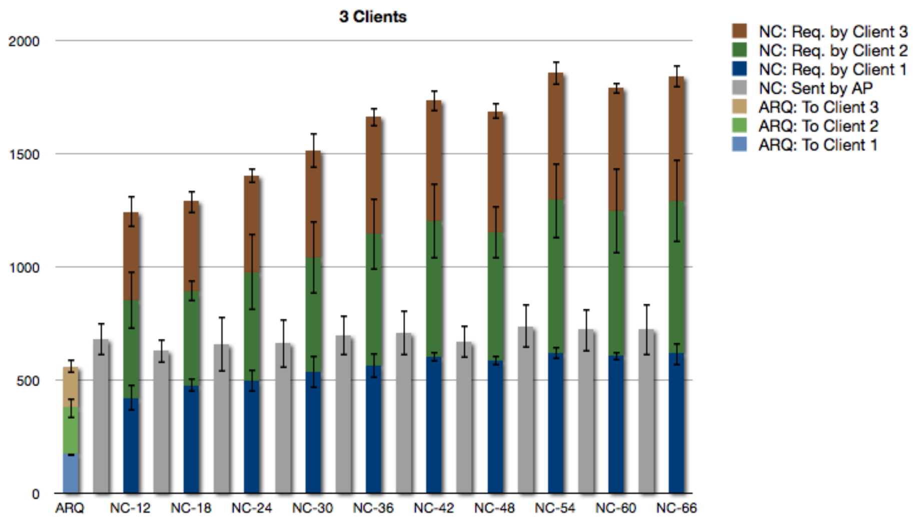


Figure 5.6: Experimental results for the multicast experiment with 3 clients. The vertical axis shows the number of packets and the horizontal axis shows the different reliability configurations.

packets is shown in the ARQ case, as well as the number of coded packets required to be sent and required at each client are shown.

Discussion

For both client configurations, the number of required transmissions using network coding exceeds the amount using ARQ. This is expected, as the theory showed similar results, see Section 4.2 under Performance in Multicast WiFi Design without Per-Packet Feedback. The theory also predicted an increase in difference between ARQ and RLNC when the number of clients increased, but this is hard to verify because of the large variability in the results and the small range of number of clients.

Mismatch of Required Packets at Clients and Packets Sent by AP

Intuitively, the number of transmitted packets at the access point should equal the number of required packets at the client that required the most coded packets. This is because the lost packets at the other clients should be covered by these packets. In the experimental results, there is a mismatch between the total required packets at the client that required the most packets, and the coded packets sent by the access point. Regardless of the network coding configuration, one sees from the graph that the number of coded packets sent by the access point is more than the required redundant packets at the client that required the most packets. However, this difference diminish as the block size increases, a trend which will be explained later. The reason for the mismatch can be explained by the combination of fluctuating channel conditions and similar packet loss percentage across the clients. With these properties, different clients will require the most packets for different blocks in a single round of execution. This means that a single client does not necessarily require the most packets for all the blocks, making the access point transmit the highest number of required coded packets for each block. In the end, this gives more than the total required coded packets at any client for a round. If the round consisted of only one block, this mismatch should not have occurred. Therefore, from theory one expects that the smaller block size, the larger the mismatch as found experimentally.

Increase of Required Packets at each Client

As can be seen by the experimental results, the required packets by each client increases with the block size. The reasons for this is the way the network coding is performed on each block. The larger the block size, the more packets not addressed for the specific client are needed at the client when a client lose a packet. This is due to the fact that the coding is done over the whole block so each client must recover all lost packets in each block if it had one or more self-addressed packet loss. Having larger blocks will increase the probability of requiring these packets not addressed for themselves, compared to having a larger number of smaller blocks.

NC Performance Not Affected by Block Size

The experimental results also show that the number of required packets to be transmitted at the access point seems fairly unaffected by the block size. This can be understood as a result of combining the two previously discussed phenomena. The increase of required coded packets at each client should increase the total required coded packets sent at the access point. However, at the same time when the block size increases, fewer blocks will be transmitted. Therefore, the mismatch between the total required coded packets sent at the access point and the required packets at the client that required the most diminishes. This compensates for the negative effect of having more required packets at the client that required the most packets.

As one can conclude from the experimental results, the number of packets required to recover from all packet loss is unnecessary high. This is because each client needs to recover lost packets that are not even self-addressed. This issue can be solved with per-packet feedback, in which the next chapter will discuss.

Chapter 6

Multicast WiFi using XOR Scheme

To solve the major problem in the previous multicast scheme, which required recovery of packets that are not self-addressed in order to recover self-addressed packet loss, per-packet feedback can be introduced. With per-packet feedback, the problem can be eliminated by applying network coding over only the packet losses that were self-addressed, rather than all types of packet losses at each client. By simply using intra-session RLNC with some slight modifications in the scheme would most likely be an improvement compared to having to code over the whole block. However, since per-packet feedback is available to the access point, another scheme can also be used. This scheme is called XOR and is an inter-session network coding scheme that can intelligently select which packets to XOR together, and send those coded packets to recover packet loss.

6.1 XOR versus RLNC

Using Intra-Session RLNC with Per-Packet Feedback

Using intra-session RLNC with per-packet feedback enables the access point to generate coded packets that are coded over fewer packets than the whole block. In fact, the access point needs only to code over the packets that are lost and self-addressed at each client. This will create groups of coded packets. Since the number of lost and self-addressed packets are the same as the packets RLNC will have to send, RLNC will not achieve higher throughput than ARQ. Instead, pure retransmissions using ARQ would achieve the same result with less computation.

Using XOR with Per-Packet Feedback

By XOR-ing certain packets together, better performance should be achieved, as suggested theoretically in Chapter 4. The required number of redundant packets highly depends on the correlation between the packet loss at the clients, as well as the similarity of packet loss probabilities between the clients. Since this scheme is theoretically never worse than using RLNC or multiple unicast streams with ARQ in terms of the number of transmissions, XOR schemes will be further discussed.

6.2 Performance of XOR using Experimental Data

As discussed in Chapter 4, the performance of an XOR-scheme is highly dependent on the correlation of packet loss between the clients. The theoretical results were based on having independent and identically distributed losses. These conditions might not be representative of a real situation. In order to obtain realistic results in the present thesis, each client in the previous experiment collected per-packet loss for each block. The overall performance of an XOR scheme can now be determined based on real data without assumptions regarding packet loss correlation. By using the experimental data, as well as XOR coding schemes discussed in Chapter 4, improvements compared to inter-session RLNC can be determined quantitatively.

Results using XOR Schemes

By applying the two algorithms, Algorithm 4.1 and an adjusted version of Degree [JMM93], the number of required redundant packets is measured based on data collected in the multicast experiment. Although being able to eliminate the assumption of packet loss correlation, the only assumption made here is that the loss percentage and loss correlation of the redundant packets are the same as the percentage and correlation for the data packets.

XOR Performance with Two Clients

The performance of the XOR scheme using two clients is presented in Figure 6.1. Only the additional redundant packets are shown. In the situation with multiple unicast flows, the retransmitted packets from ARQ are seen as the leftmost bar. As before, the bar is divided by color coding to show the amount each client required. All the next bars are forming groups of 4 with the same color coding. Each group represents the results using a network coding reliability mechanism. The grey bar represents the number of coded packets sent using the authors XOR scheme at the access point. Similarly, the beige bar shows the number of coded packets sent using the Degree scheme at the access point. It is evident from the figure that both XOR schemes significantly reduce the number of required redundant packets compared to ARQ. The red/purple bar combines the total loss based on whether the packets are lost by a single or both clients, see legend of Figure 6.1. Finally, the blue/green bar shows the number of packet that are self-addressed and lost by only one client. The bar is divided according to the respective clients.

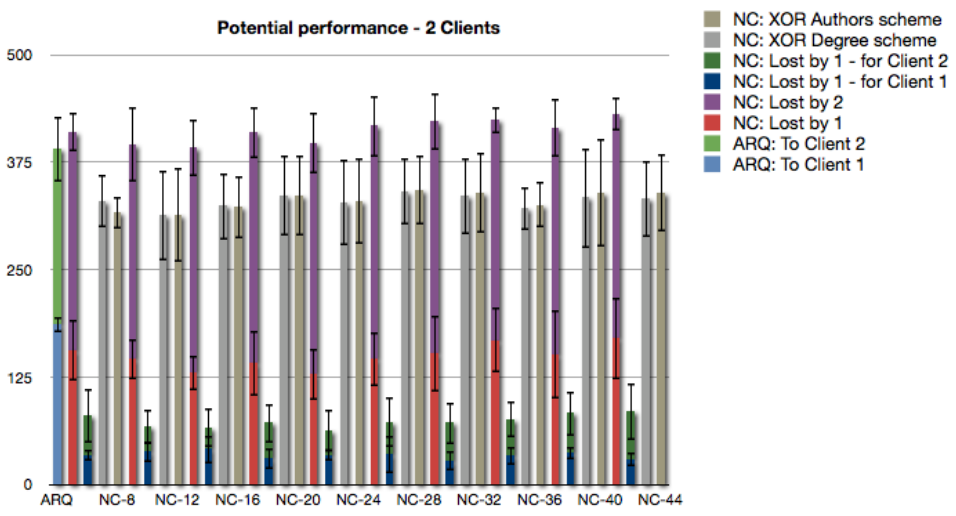


Figure 6.1: Combined experimental and theoretical results for XOR multicast scheme with 2 clients.

XOR Performance with Three Clients

Similarly, the performance with three clients is shown in Figure 6.2. As with two clients, the number of retransmitted packets is shown in the ARQ case. The number of redundant packets transmitted using the author's scheme and Degree scheme are represented in the same way as for two clients. The total lost packets and the self-addressed packets that are only lost by one client is represented in a similar way as with two clients.

6.3 Analysis of XOR Schemes

As seen from Figure 6.1 and Figure 6.2, there is a clear reduction of redundant transmitted packets using XOR compared to using ARQ. In fact, using block size of 16 and 24 when having 2 and 3 clients respectively, reduced the number of redundant packets with 20% and 25% using the best of the two XOR schemes. The theory predicted a reduction more than what is shown experimentally, and the next sections will explain why this is the case.

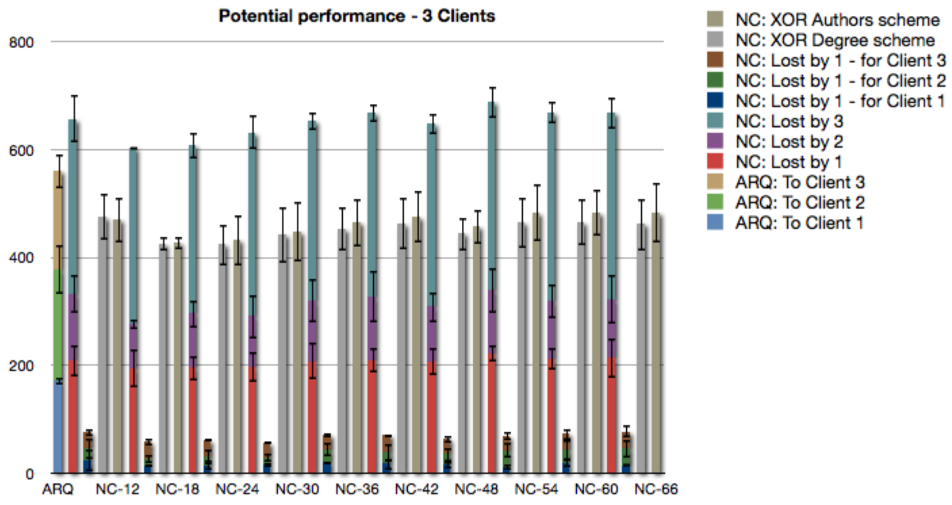


Figure 6.2: Combined experimental and theoretical results for XOR multicast scheme with 3 clients.

Correlated Packet Loss

The experimental results are based on the data that were presented in Chapter 5, showing a packet loss in the range 1-35%. Given this loss percentage and the assumption of uncorrelated packet loss, Table 4.2 shows that the number of packets lost by one client should be significantly larger than the number of packets lost by both. Comparing this with the experimental results suggests that the packet losses are correlated. This is evident by comparing the number of packets lost by all and the packets that are lost by only one client. In fact, the results show that whenever a packet is lost at a client, it is most likely lost at all the other clients as well. This is unfortunate, since the more negatively correlated the packet losses are, the more gain the XOR scheme can achieve. To use the assumption that the packet losses are uncorrelated is therefore not advisable when predicting the efficiency of any XOR schemes in WiFi.

Fewer Exploitable Packet Losses than Expected

An even more unfortunate consequence of the positive correlation is that the packets that are lost by all clients always affects the client that the packet was originally addressed to. This is because there is always one client that needs the lost packet. In the case with 3 clients, a packet that is lost by only 1 client might not need to be transmitted again, in fact it has a chance of affecting the client it is lost at with probability of 1/3. To illustrate the benefit of any XOR scheme, Figure 6.1 and Figure 6.2 includes the number of packets that were self-addressed and lost at only one client. They are denoted as NC: Lost by 1 - for Client X, where X is the client number. In the case of 2 clients, these losses are the ones that can be XOR-ed together and give the improvement in performance. In fact, with the optimal XOR scheme, the total number of needed transmissions would be the maximum of the losses described above for all clients, plus the packets that were lost by both clients. It can therefore be seen that both the authors Algorithm 4.1 as well as Degree [JMM93] performed close to optimal with few clients. It is therefore clear that the packets that were lost at all clients are the ones that hurts the most in terms of performance of the XOR scheme. It is therefore very unfortunate having the strong positive correlation between the packet losses.

XOR Performance Not Affected by Block Size

Theoretically the performance of the XOR scheme should improve with larger block size up to a certain point, because of the larger possibilities of XOR-ing packets together. Despite that, the results show that the performance is fairly unrelated to the block size, just as in the inter-session RLNC experiment. Two major reasons for this are discussed below.

Highly Correlated Packet Losses

The packet losses among the clients are highly correlated and thus causes the gains in general to be very small. This is because it enables few packets to be coded together regardless of block size. However, it does not fully explain why XOR-ing packets does not improve with the increase of block size, but it explains why it should not improve significantly.

High Fluctuation of Packet Losses at each Client

Inspecting the temporal variation of packet losses at any client for large and small block sizes, shows that the variance of the number of lost packets in a block is proportional to the size of the block regardless of block size. In addition, the experiments have few clients in the system, which gives many self-addressed packets in a block, even when the block size is small. Combining the previous observations shows that regardless of block size, the total number of exploitable packets, and therefore gains, will remain the same for one large block as it will for many smaller blocks with the same total number of transmitted packets. In case the number of clients increases, fewer self-addressed packets will be incorporated in each block, and the result may look different. A possible increase in performance might then be evident with larger block sizes in that case.

Chapter 7 Discussion

In this thesis, network coding applied to both point-to-point unicast and point-to-multipoint multicast networks are investigated. One scheme using network coding in point-to-point unicast is investigated, as well as two schemes using network coding in point-to-multipoint multicast. The scheme using network coding in point-to-multipoint multicast with per-packet feedback showed actual improvements by making use of experimental data, thereby providing trustworthy results. This chapter discusses the obtained results, as well as the aspects of using the improved scheme in a commercial system.

7.1 Point-To-Point Unicast using Intra-Session RLNC

The performance of RLNC in a single unicast flow was found experimentally to not increase the throughput as compared to using the traditional ARQ mechanism. This is due to the high efficiency of the ARQ mechanism. Moreover, implementing the RLNC in a single unicast flow would require complex software in order to predict accurate packet loss for each block of packets. This is necessary to enable the scheme to work optimally.

WiMAX vs. WiFi Experimental Results

The reasons why the similar experiment done in WiMAX [Tee] had significant improvement in contrast to the present WiFi unicast experiment can be many. Some of them are the following.

– The WiFi ACKs are sent immediately after a packet is received and the physical medium is reserved for these ACKs. Unless it is a B-ACK, this would be even more efficient due to the infrequent requests of ACKs.

– The WiMAX physical distance is longer than in WiFi, causing the propagation delay, and therefore RTT to become longer.

  - The WiMAX experiment showed that 3 copies of the same packet were always sent when the channel condition was sufficiently poor. This is not the case in WiFi.

Because of the above reasons, the WiMAX experiment using intra-session network coding showed a great performance improvement compared to what is shown in WiFi.

The ARQ mechanism in WiFi performs nearly optimally in terms of throughput, with only negligible time the channel is idle because of the rapid acknowledgements. After the introduction of aggregated frames, this time the channel is idle is reduced even more because of the infrequent acknowledgements. Hence, there is almost no theoretical improvement by using network coding. The small improvement using network coding comes with assumptions that are unrealistic in a practical scenario. The assumption that the channel quality can be measured to allow that the exact right amount of redundancy to be sent, is so fragile that even being off by one packet in a large block would be enough to remove the whole advantage of the scheme compared to ARQ. Practically it is therefore a vast undertaking trying to perform better than ARQ in 802.11n with or even without aggregated frames.

7.2 Multicast using RLNC without Per-Packet Feedback

In order to confirm theoretical results and provide data for XOR-scheme, reliable communication for several clients through a multicast experiment without per-packet feedback was conducted. It confirmed what was predicted theoretically, namely, that using inter-session RLNC over the whole block of data packets is never better than using multiple unicast flows and ARQ as a reliability mechanism. This scheme is not advisable for practical use, since it shows both experimentally and theoretically no improvements compared to using ARQ in a multiple unicast situation.

7.3 Multicast using XOR with Per-Packet Feedback

To confirm the theoretical gains using XOR as a multicast throughput improvement in WiFi, the data from the experiment with inter-session RLNC were used. The result is therefore a combined theoretical and experimental result, where two major assumptions are no longer needed, namely uncorrelated packet loss, and equal packet loss probability among the clients.

Packet Loss Probability at each Client

The experimental results show that the total packet loss at each client is quite similar, which is consistent with the theoretical assumptions. However, the variation in packet loss is considerable, and sometimes one client loses a lot more than another. This makes the experimental results differ from the theoretical expectations.

Optimal Variation of Packet Loss Probabilities

The theory assumes there is no variation of packet loss among the clients, which is not true in practice. Since the losses affect clients at different times, the blocks of packets will not have the same number of packet loss at each client, even though the packet loss for all blocks in total can be nearly to the same. The gain of the XOR-scheme comes from the possibility of XOR-ing packets together. This will decrease the larger variation there is in the packet loss in each block. Therefore, the optimal situation is when the clients have the same total loss distributed evenly over time. Unfortunately, in practice, the variation in time is significant, and reduces the overall performance of the scheme.

Effect of Block Size

As seen in the point-to-multipoint multicast experiments, the block size had minimal impact on the performance, for reasons explained earlier. However, block size has an impact also on other aspects.

Too Large The bigger the block size is, the more possibilities there are to combine packets with each other. In terms of throughput, there should therefore be no upper limit on the block size. However, the trade-off that comes with a large block size is the increased time it takes to decode at the receiver. Depending on the type of data, this can cause problems, for example in real-time applications. For other types of data, this can be negligible, for example in streaming of video.

Too Small For smaller block size, there are fewer possibilities to combine packets, and the throughput gain will decrease the smaller the block size. However, the decoding at the receiver will be more frequent and therefore the per-packet delay will be smaller. This can be beneficial for real-time data, but the decrease of throughput will be unfortunate for types of data like streaming of video.

Implementing XOR in a Commercial System

The results from the unicast experiment demonstrated that it is very efficient per-packet feedback in WiFi. Based on this observation, implementing the XOR scheme in a full commercial system should therefore be done in the driver. This is because the driver is where the per-packet feedback is available in an efficient way. Recalling that the per-packet feedback needs to be available only when the whole block is sent, the block acknowledgements in 802.11n using aggregated frames will cause no problems.

Even though this thesis presents results based on theory and experiments, it has only given a proof of concept. The rest of the sections in this chapter will discuss different topics relevant for implementation of the XOR scheme in commercial systems.

Number of Clients in the System

For the experimental results, it was not possible to conclude whether the XOR scheme in practice would scale well in terms of number of clients. By including a large number of clients, it is reasonable to expect that the packet loss probabilities will broaden. In this case, the performance improves less rapidly with the number of clients as compared to the theoretical results which is based on having the same packet loss probability at each client.

In the case that all clients care only about throughput, and not about per-packet delays, the system should never exclude a client. However, grouping clients with similar packet loss probabilities may be more beneficial in terms of throughput. Client grouping is further investigated in the next section.

Different Packet Demands at each Client

In a commercial system, it is probably not correct to assume that there is the same demand for packets at all clients. In case there are different demands for packets, the blocks of packets can be restructured. The blocks must not necessarily contain the same number of packets from each client, but can be dynamically changed from block to block. In order to maintain high probability of being able to XOR packets together, the blocks should contain equal number of packets for the different clients in the system. Therefore, dynamically changing content of a block will potentially hurt performance. Another way to deal with the different demands of packets is to group packets that are addressed to a subset of clients together. This subset consists of clients that have similar demand of packets. The content of a block should therefore be determined based on both packet loss probabilities and the demand of packets at clients.

Different Type of Data for each Client

Some clients might have demand for time-sensitive data, and some not. Since the XOR scheme improves the throughput at the cost of increased latency, including clients demanding time-sensitive data is generally not a good idea. Theoretically, it is simple to exclude certain clients from coding with this scheme. However, in practice the access point will have to know in some way which packets from the clients are time sensitive and which are not. One way to determine this, is to dive into the higher layers to find out whether the data is likely to be time-sensitive or not. One way for the access point to achieve this is to have a list of protocols indicating whether the packets are time sensitive or not. Each incoming packet can be classified as time sensitive or not, and based on this, decide whether to be included into the XOR scheme or not.

Chapter 8 Conclusions

In the present work, possibilities for improving the throughput in WiFi for single client and multiple clients using network coding were investigated theoretically and experimentally. It was found experimentally that the unicast case did not achieve improvements as compared to ARQ. Due to this result, it became more interesting and important to focus on finding improvements in multicast cases.

The present thesis reports the development of two network coding schemes for multicast. First, an inter-session RLNC scheme was designed and tested, and later an XOR scheme was developed. The former scheme did not lead to improvements in throughput, whereas the latter was shown to provide significant advantages. In particular, it was found that the XOR scheme in the multicast case decreases the number of transmitted packets due to bit errors. As compared to ARQ, the reduction was up to 25%. This is a significant result, and the developed XOR scheme should be of great interest to implement in throughput demanding systems. As far as the author knows, this is the first experimental effort addressing this XOR problem in WiFi.

Other theoretical works have analyzed and quantified the gains of XOR-ing packets, and in most cases it is assumed uncorrelated packet losses. In the present work, it was found experimentally that the packet losses are actually correlated. This strongly suggests that theoretical modeling and numerical simulations should consider also the possibility of having correlated packet losses.

Commercially, whether the XOR scheme will scale well with the number of clients, for different topologies, packet loss probabilities, demands and types of data remain open questions. These are all important issues, and deserve to be investigated in future work.

8.1 Further Work

This section provides suggestions to further work that can be done in order to improve and confirm unanswered questions regarding the XOR scheme provided in this thesis.

Whether other topologies with far more clients will work well, should be investigated. Although the probability of packet loss among the clients in the XOR scheme was very similar in the experiments done in this thesis, including a lot more clients could show different packet loss characteristics. Having different packet loss probabilities may well have a strong impact on the performance, and should therefore be looked further into.

In a real system, there will most likely be clients with different demands of packets. This should also be investigated further. A possible separation of clients into groups with same demand should also be investigated. Another solution to the problem with different demands is to exclude certain clients from the scheme, This will imply testing the XOR scheme together with other unicast flows at the same access point.

In a commercial system, clients will use the WiFi for different purposes, and will have packets of different types. Some might be time-sensitive, while others not, and this problem should be investigated further.

The experiments done in the present work tried to eliminate all interference in WiFi. However, many practical WiFi systems overlap, and interfere. This problem can easily affect the performance and should also be addressed in future work.

References

[ACLSY00] R. Ahlswede, N. Cai, and R. Li Shuo-Yen. Network information flow. *IEEE Transactions on Information Theory*, 46, July 2000.

[All] Wi-Fi Alliance. Wi-fi alliance trademarks. http://www.wi-fi.org/about/organization. Accessed: 5.12.2013.

[Cis13] Cisco. Cisco visual networking index: Forecast and methodology, 2012-2017. May 2013.

[CMWB08] R. Costa, D. Munaretto, J. Widmer, and J. Barros. Informed network coding for minimum decoding delay. *IEEE International Conference on Mobile Ad-hoc and Sensor Systems*, pages 80–91, 2008.

[CWJ03] Philip A. Chou, Yunnan Wu, and Kamal Jain. Practical network coding. 2003.

[DGP$^+$06] A.F. Dana, R. Gowaikar, R. Palanki, B. Hassibi, and M. Effros. Capacity of wireless erasure networks. *Information Theory, IEEE Transactions on*, 52(3):789–804, 2006.

[EOMA08] A. Eryilmaz, A. Ozdaglar, M. Médard, and E. Ahmed. On the delay and throughput gains of coding in unreliable networks. *IEEE Transactions on Information Theory*, 54(12):5511–5524, December 2008.

[FLMP07] C. Fragouli, D. Lun, M. Médard, and P. Pakzad. On feedback for network coding. *Annual Conference, Information Sciences and Systems*, pages 248–252, 2007.

[Gro] IEEE Working Group. Official ieee 802.11 working group project timelines. http://grouper.ieee.org/groups/802/11/Reports/802.11_Timelines.htm. Accessed: 4.11.2013.

[HKM$^+$] T. Ho, R. Koetter, M. Médard, D. R. Karger, and M. Effros. The benefits of coding over routing in a randomized setting.

[Inf12] Informa. Understanding today's smartphone user: Demystifying data usage trends on cellular & wi-fi networks, 2012.

[JMM93] M. Jolfaei, S. Martin, and J. Mattfeldt. A new efficient selective repeat protocol for point-to-multipoint communication. *Communications, 1993. ICC '93 Geneva. Technical Program, Conference Record, IEEE International Conference on*, 2:1113–1117, 1993.

[Kar72] R. Karp. Reducibility among combinatorial problems. 1972.

[KRK$^+$84] S. Katti, W. Rahul, D. Katabi, M. Médard, and J. Crowcroft. Xors in the air: practical wireless network coding. *ACM SIGCOMM Computer Communication Review*, 36:243–254, 1984.

[Lar07] P. Larsson. Analysis of multi-user arq with multiple unicast flows under non-iid reception probabilities. *Wireless Communications and Networking Conference, 2007. WCNC 2007. IEEE*, pages 384–388, 2007.

[Lar08] P. Larsson. Multicast multiuser arq. *Wireless Communications and Networking Conference, 2008. WCNC 2008. IEEE*, pages 1985–1990, 2008.

[LJ06] P. Larsson and N. Johansson. Multi-user arq. *Vehicular Technology Conference, 2006. VTC 2006-Spring. IEEE 63rd*, 4:2052–2057, 2006.

[LMKE05] D.S. Lun, M. Médard, R. Koetter, and M. Effros. Further results on coding for reliable communication over packet networks. *Information Theory, 2005. ISIT 2005. Proceedings. International Symposium on*, pages 1848–1852, 2005.

[LMKE08] D.S. Lun, M. Médard, R. Koetter, and M. Effros. On coding for reliable communication over packet networks. *Physical Communication*, pages 3–20, 2008.

[LMS09] D. Lucani, M. Médard, and M. Stojanovic. Random linear network coding for time-division duplexing: Field size considerations. *Global Telecommunications Conference, 2009*, November 2009.

[McE87] R. J. McEliece, editor. *Finite Fields for Computer Scientists and Engineers.* Kluwer Academic Publishers, 1987.

[Met00] J. Metzner. An improved broadcast retransmission protocol. *IEEE Transactions on Comminications*, 32(6):679–683, 2000.

[MTK08] Ghaderi. M., D. Towsley, and J. Kurose. Reliability gain of network coding in lossy wireless networks. *INFOCOM 2008. The 27th Conference on Computer Communications. IEEE*, pages 2171–2179, 2008.

[NTNB09] D. Nguyen, T. Tran, T. Nguyen, and B. Bose. Wireless broadcast using network coding. *Vehicular Technology, IEEE Transactions on*, 58(2):914–925, 2009.

[QFC10] J. Qureshi, C.H. Foh, and J. Cai. An efficient network coding based retransmission algorithm for wireless multicast. *CoRR*, 2010.

[SES10] B. Swapna, A. Eryilmaz, and N. Shroff. Throughput-delay analysis of random linear network coding for wireless broadcasting. *International Symposium on Network Coding (NETCOD)*, pages 1–6, June 2010.

[SMJ] B. Shrader and N. M. Jones. Systematic wireless network coding.

[SSM08] J.K. Sundararajan, D. Saha, and M. Médard. Arq for network coding. *ISIT 2008. IEEE International Sumposium on*, pages 1651–1655, 2008.

[Tee] S. Teerapittayanon. Performance enhancements in next generation wireless networks using network coding: A case study in wimax.

[Web] Webopedia. Wi-fi. http://www.webopedia.com/TERM/W/Wi_Fi.html. Accessed: 4.11.2013.

[Wik] Wikipedia. Graph coloring. http://en.wikipedia.org/wiki/Graph_coloring. Accessed: 12.10.2013.

[YS00] S. Yong and L.B. Sung. Xor retransmission in multicast error recovery. *Networks, 2000.(ICON 2000). Proceedings. IEEE International Conference on*, pages 336–340, 2000.