



ICT - Information and Communication Technologies



Network Coding for Robust Architectures in Volatile Environments
Collaborative Project

Grant Agreement Number 215252

**WIRELESS PROTOCOLS USING NETWORK CODING:
REPORT ON ON STATE-OF-THE-ART**

Due Date of Deliverable: 31/12/08

Actual Submission Date: 22/01/09

Revision: Final

Start date of project: January 1st 2008

Duration: 36 months

Organization name of lead contractor for this deliverable:
THOMSON Research

Authors: Dan-Cristian Tomozei

Contributors: Theodoros Salonidis, Laurent Massoulie, Leandros Tassiulas,
Leonidas Georgiadis, Ioannis Broustis, Savvas Gitzenis

Project Information

PROJECT	
Project name:	Network Coding for Robust Architectures in Volatile Environments
Project acronym:	N-CRAVE
Project start date:	01/01/2008
Project duration:	36 months
Contract number:	215252
Project coordinator:	Leandros Tassiulas – CERTH
Instrument:	STREP
Activity:	Theme Challenge 1: Pervasive and Trusted Network and Service Infrastructures
DOCUMENT	
Document title:	Wireless Protocols Using Network Coding: Report on State-of-the-Art
Document type:	Report
Deliverable number:	D2.1
Contractual date of delivery:	31/12/08
Calendar date of delivery:	22/01/09
Editor:	Dan-Cristian Tomozei
Authors:	Theodoros Salonidis, Laurent Massoulie, Leandros Tassiulas, Leonidas Georgiadis, Ioannis Broustis, Savvas Gitzenis
Workpackage number:	WP2
Workpackage title:	Network Coding Experimentation
Lead partner:	THOMSON
Dissemination level:	Public
Date created:	15/09/2008
Updated:	22/01/2009
Version:	v1
Total number of Pages:	29
Document status:	final

Contents

1	Introduction	4
2	Description of existing protocols using Network Coding	6
2.1	Inter-session coding below network layer	6
2.1.1	Analog Network Coding [KGK07]	6
2.1.2	COPE [KRH ⁺ 06]	8
2.1.3	NoCoCo [SHC07]	9
2.1.4	CLONE [RSW ⁺ 08]	10
2.1.5	ER [RIM ⁺ 07]	12
2.1.6	XOR-SYM [CP07]	14
2.2	Intra-session coding	14
2.2.1	MORE [CJJK07]	15
2.2.2	MIXIT [KKBM08]	16
2.2.3	Ho & Viswanathan [HV03]	18
2.2.4	Multipath Code-Casting [RGK ⁺ 07]	20
2.3	Coding for distributed storage systems [DGWR07]	20
3	Comparison	23
3.1	Deployment complexity	23
3.2	Computational complexity	24
3.3	Importance of coding and wireless	24
3.4	Innovation	25
3.5	Multicast	26
4	Conclusion and future considerations	27

Abstract

We survey a representative subset of proposed protocols that use network coding in wireless environments. We provide a comparison of these with respect to the following aspects: supported communication primitives, backward compatibility, layers involved (from physical to transport layer), and implementation complexity. The report concludes with directions for future research.

1 Introduction

Recently many protocols have been proposed to exploit the benefits of network coding. These target environments going from wired to wireless and communication primitives going from unicast and multicast to data storage. They involve modifications from the physical to the transport and application layer, may be backward-compatible or not (depending on the layers they affect), have different levels of complexity, and exploit network coding to various degrees (in the coding operations allowed and the finite fields used).

In the present document we provide a critical survey of a representative subset of these proposed protocols. We further compare them along the above characteristics, in an attempt to identify main issues that remain to be addressed to enable the full benefits of network coding.

Specifically, we consider inter-session network coding, exemplified by Analog Network Coding [KGK07], COPE [KRH⁺06], NoCoCo [SHC07], CLONE [RSW⁺08], ER [RIM⁺07] and XOR-SYM [CP07]. Such protocols recode packets from distinct flows locally at sending nodes, and exploit the local broadcast nature of wireless transmission.

We next consider intra-session network coding, exemplified by MORE [CJKK07], MC2 [RGK⁺07] and MIXIT [KKBM08]. In the latter, only packets belonging to the same session are recoded together.

For both inter- and intra-session network coding, the supported communication primitive is that of unicast, typically using wireless lossy transmissions. Intra-session network coding incorporates multi-path transmissions. Extensions to multicast are described as natural in some proposals (e.g. MORE). However, multicast is not the main focus in the proposals mentioned in this survey.

We finally consider the work of Dimakis et al. [DGWR07] on distributed resilient storage. It specifically addresses minimization of bandwidth used in the maintenance of stored content.

After describing the salient features of each of the above proposals, we provide a comparison of their characteristics, in terms of complexity of implementation, deployment and required computations, as well as the performance benefits resulting from their use. The proposals are diverse and their contributions range from the formulation of optimization frameworks accounting for various phenomena in wireless networks to actual proof of concept deployments on real testbeds.

All of the cited papers show significant improvements over “traditional” wireless transmission methods (achieving on average throughput improvements of up to a factor of 4 in some settings – see [KRH⁺06] or [KKBM08]). However, these gains are not to this day fully understood. We formulate open questions (or reinforce existing ones) related to this issue at the end of the document. We inquire upon the following:

- Do the inter- and intra-session coding benefits showed in their respective proposals provide cumulative gains? Thus, if both techniques were to be adapted to be used in the same network, what benefits would be obtained?

- What impact does signalling accuracy have on throughput in an intra-session coding scheme?
- Is network coding able to provide better delay guarantees?

Finally, we set future work guidelines for the multicast communication primitive in the context of bandwidth-greedy peer-to-peer applications (such as video on demand, or file swarming). We are concerned with multicast flow control and ISP-friendliness issues.

The outline of the document is as follows. In Section 2 we give brief descriptions of various wireless protocols involving coding.

In Section 3 we compare the described protocols and we emphasize their strong points and weaknesses. We consider such criteria as deployment complexity, computational complexity, the impact of performing coding in a wireless environment.

We conclude with a description of directions for future work in Section 4.

2 Description of existing protocols using Network Coding

This Section provides a short description of proposed wireless communication protocols (implemented at various levels in the networking stack) which make use of coding techniques for data transmission. The selection below aims to include representatives of all types of proposals that we are aware of, covering both intra- and inter-session network coding, deployments at all layers from physical to application. For each proposal we start by specifying which layer they involve and which field is used for coding.

2.1 Inter-session coding below network layer

The proposals described next all target multiple unicast sessions, and aim to perform network coding below the network layer.

2.1.1 Analog Network Coding [KGK07]

Layer:

MAC		
PHY		ANC

; Coding field: \mathbb{C}

Analog network coding operates at the physical layer. It consists of letting the physical medium take care of the coding operation: a receiver directly measures the superposition of complex signals, hence receiving the sum over \mathbb{C} of the sent signals.

In paper [KGK07], the authors propose a method to reduce the number of time slots used for packet forwarding by using physical properties of the wireless medium. A typical example regards two flows intersecting at a node. More specifically, two nodes (Alice and Bob) want to forward packets to each other, but they are not in range. They do however both have a third “relay” node in range. This is called the “Alice-Bob topology”. When using “digital” coding, the operation would require 3 steps: the two nodes transmitting their respective packets and subsequently the relay broadcasting the xor-ed packets. With Analog Network Coding (ANC) only two steps are required: (1) both nodes transmit roughly at the same time and (2) the relay broadcasts the packet it has received (namely sum of the two physical signals) without trying to decode it. Normally such a packet received at the relay would normally be classified as “collision” and get discarded.

Another example regards the chain topology in which nodes interfere with distance-1 neighbours. In the case of an unidirectional stream of packets, using ANC can improve long term throughput by a factor of 1.5.

A third considered topology is the “X” topology, which consists of two sources $S1$ and $S2$ sending to two destinations $D1$ and $D2$ with a relay R amplifying the “colliding” transmissions. $D1$ can overhear $S2$ and $D2$ can overhear $S1$. Using the relay’s transmissions, the destinations can decode the packets they’re interested in by subtracting the directly overheard packets.

Below we give a more detailed description of the protocol:

A wireless signal transmitted by the sender is represented as $s[n] = A_s[n]e^{i\theta_s[n]}$, where $A_s[n]$ and $\theta_s[n]$ are the amplitude and the phase of the n -th sample. The MSK modulation is employed. This modulation maps “1” to a phase difference of $\frac{\pi}{2}$ and “0” to a phase difference of $-\frac{\pi}{2}$.

On the receiver side, the received signal takes the following form:

$$y[n] = hA_s[n]e^{i(\theta_s[n]+\gamma)}.$$

Computing the phase difference between the n -th and the $n + 1$ -th sample boils down to computing the argument of the ratio $r = \frac{y[n+1]}{y[n]} = e^{i(\theta_s[n+1]-\theta_s[n])}$. Because of noise, $\arg(r)$ will not be precisely $\frac{\pi}{2}$ or $-\frac{\pi}{2}$. Thus, a positive argument will be translated to a “1” and a negative one to a “0”.

ANC aims at decoding interfered signals from two sources:

$$y[n] = y_A[n] + y_B[n] = h'A_s[n]e^{i(\theta_s[n]+\gamma')} + h''B_s[n]e^{i(\phi_s[n]+\gamma'')}.$$

Namely, in the Alice and Bob topology, Alice receives $y[n]$ and knows $s[n]$. The question is how to decode the phase differences in Bob’s signal $t[n]$. Denoting $A = h'A$, $B = h''B$, $\theta[n] = \theta_s[n] + \gamma'$, $\phi[n] = \phi_s[n] + \gamma''$, $\mu = \frac{1}{N} \sum_n |y[n]|^2$ and $D = \frac{|y[n]|^2 - A^2 - B^2}{2AB}$ it is shown that solving the following system suffices for Alice to recover Bob’s bits:

$$\theta[n] = \arg(y[n](A + BD \pm iB\sqrt{1 - D^2})) \quad (1)$$

$$\phi[n] = \arg(y[n](B + AD \pm iA\sqrt{1 - D^2})) \quad (2)$$

$$A^2 + B^2 = \mu \quad (3)$$

$$A^2 + B^2 + 4AB/\pi = \frac{2}{N} \sum_{|y[n]|^2 > \mu} |y[n]|^2. \quad (4)$$

Equation (3) holds for a random bit sequence. Thus, the senders use a pseudo-random sequence which they use to XOR outgoing bits. Receivers XOR the received bits with the same sequence, thus recovering the sent bits. Since equations (1,2) have two solution pairs (denoted by (θ_1, ϕ_1) and (θ_2, ϕ_2)), Alice uses her knowledge of $\theta_s[n]$ to select the correct solution. Namely, to select the correct pair of differences $(\Delta\theta_{k\ell}[n], \Delta\phi_{k\ell}[n]) = (\theta_k[n+1] - \theta_\ell[n], \phi_k[n+1] - \phi_\ell[n])$, for $k, \ell \in \{1, 2\}$, the pair (k, ℓ) minimizing the error $|\Delta\theta_{k\ell}[n] - \Delta\theta_s[n]|$ is selected.

There are several practical issues described in the ANC paper. First, detection of the start of an interference is done by measuring the variance of the received signal energy. No interference translates to a low variance (ideally, 0). Interference breaks this property of the signal.

Second, the lack of synchronization is dealt with by imposing a fixed packet size. The packets are padded with a pre-defined bit sequence called a “pilot sequence” (the sequence at the end of the packets is the mirrored sequence found at the beginning of the packets). Thus, the first and last few bits of the transmission will be interference free and will be used for detecting the alignment of Alice and Bob’s packets. In fact, a slight de-synchronization is “encouraged”, as nodes start their transmission after a random delay.

Moreover, the frequency offset Δf originating in the fundamental limitations of oscillators, which causes a displacement in the phase differences of $2\pi\Delta f\Delta t$ (where Δt is the known sampling period), is taken into account. The pilot sequence is chosen such that the mean of the transmitted phase differences is zero. Thus, Δf is estimated by measuring this phase difference for the interference-free part of the transmission.

Last, considerations are made regarding channel distortion and sampling offset.

In [KGK07], the authors deployed ANC over a three-node testbed, using a Software Defined Radio (SDR) implementation. GNURadio software is used, resulting in a bit

rate of 500kb/s. The SNR is 20-30dB. The system uses Universal Software Radio Peripheral (USRP) with RFX2400 daughterboards.

The observed gain in throughput in the Alice-Bob topology is about 30% compared to COPE and about 70% compared to the traditional approach. For unidirectional flows in the X topology an improvement of 36% is observed when compared to the traditional approach (COPE does not deal with this case).

ZigZag A follow-up work on ANC is ZigZag [GK08]. It is not covered in this document, as it does not fall in the category of protocols using network coding. ZigZag uses techniques similar to ANC, thus the methodology of decoding signals summed at the physical medium is applicable to general wireless networks, not only to the ones using network coding.

2.1.2 COPE [KRH⁺06]



The authors of [KRH⁺06] propose COPE, a network coding-based forwarding algorithm for wireless mesh networks. COPE performs inter-session coding of multiple unicast sessions. Coding is used for one hop transmissions only, thus making it transparent to the network layer.

COPE relies on the wireless medium to seize coding opportunities. In particular, consider several unicast flows intersecting at a specific node i . Then, due to a relatively “dense” spatial distribution of the nodes, packets transiting through i might be overheard by other nodes in the neighbourhood. Instead of picking a single packet to forward, node i will select a subset of n received packets with distinct next hops among its neighbouring nodes, such that each of the n next hops already possesses all but one of the packets (with high probability). The n packets will then be xor-ed together and the resulting coded packet broadcast over the wireless medium. Each of the n receivers will be able to decode the packet for which they are the next hop simply by performing a xor operation with the $n - 1$ packets they already detain. Typical cases in which COPE improves throughput are bidirectional flows ($n = 2$). However the framework takes into account more general cases.

In order to benefit from coding opportunities, COPE requires each node to have a fairly accurate image of its neighbours’ list of packets. For this reason, control packets are broadcast regularly by each node, or piggy-backed on outgoing transmissions. These control packets contain updates about recent “overheard” or decoded packets, as well as acknowledgements for received native packets. In addition to this deterministic method, nodes use the ETX metric to estimate the probability of a node overhearing a packet. This enables to try and exploit coding opportunities “with high probability” in the absence of exact feedback.

Acknowledgements provide further clues to missing packets. A timeout mechanism insures packet delivery. Thus, if a packet is not followed by an ACK then it is retransmitted after a period of time (potentially coded with other native packets).

In order to further improve throughput, nodes should try to combine packets of similar sizes. To this end, they distinguish between “small” and “large” packets, and, in addition to a global queue of packets, they maintain 2 virtual queues per neighbour, corresponding to “large” and “small” packets.

Denote by M the number of such neighbours. The forwarding algorithm works as follows:

- Pick the first packet in the global queue and denote by s its size (it can be “large” or “small”)
- Consider the packets at the head of the remaining $M - 1$ virtual queues of packets of size s ; find the largest set of packets such that all next hops can decode with high probability, as described above and xor them together
- Consider in addition the heads of the remaining M virtual queues for potential incorporation in the encoded packet under construction
- Broadcast the resulting coded packet

The authors define two different types of theoretical performance metrics:

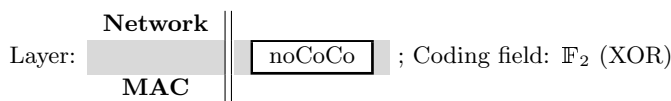
- The “Coding gain”, due to exploiting coding opportunities, equal to the ratio of the number of transmissions required by the current non-coding approach, to the minimum number of transmissions used by COPE to deliver the same set of packets. By definition it is greater than 1.
- The “Coding + MAC gain”, produced by the interaction between coding and MAC. To understand this metric, consider the case of two transmitting nodes and an intermediary router. In the non-coding approach, the router would need to send twice as many packets as the two nodes. If the two nodes transmit at the maximum rate allowed by the MAC, the router becomes a bottleneck and half the packets from each of the receivers gets dropped at the router’s queue. COPE allows the router to XOR pairs of packets, thus draining the queue twice as fast. In this case, the “Coding + MAC gain” is 2.

COPE has been implemented in a testbed of 20 wifi-equipped nodes. In an ad-hoc network setting, with TCP in the presence of hidden terminals, the benefits observed are low (2-3% on average), as high loss rates induce low TCP rates and reduce coding opportunities. Significant benefits are observed with UDP, where COPE greatly improves throughput (by a factor of 3-4 on average).

In a mesh access network setting when UDP flows are used, gains of using COPE increase with the upload to download traffic ratio, going from 5-15% when there is little amount of uplink traffic to 70% when uplink traffic increases.

Last, COPE’s main advantage is its transparency to the other elements of the protocol stack, which makes it easy to deploy in a legacy environment. Possible improvements to COPE are the noCoCo, CLONE and ER protocols, presented below.

2.1.3 NoCoCo [SHC07]



This extension to COPE specializes on bidirectional flows. The authors of [SHC07] introduce backpressure rules on routes. Specifically, no forwarding is done if the next hop already has an enqueued packet. The aim of noCoCo is to maximise the number of coding opportunities à la COPE for two opposite direction routes. Simulations of this cross-layer approach show improvements over COPE at high levels of congestion (up to 4 times greater throughput). No real experiments however have been reported.

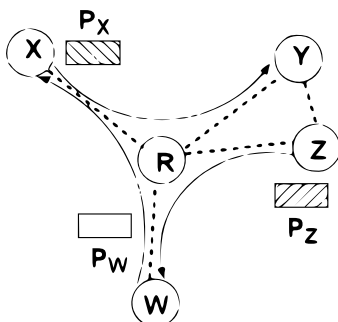


Figure 1: Example for CLONE

2.1.4 CLONE [RSW⁺08]



CLONE [RSW⁺08] is a generalization of COPE, also addressing multiple unicast sessions. This system takes into account lossy links as well as cases in which COPE does not take advantage of certain more complex coding opportunities. CLONE introduces redundancy to deal with losses. Typically, nodes will receive several coded packets and they will be able to decode in more than one way the packet which is of interest to them. Coding is done simply by performing xor operations.

An example is shown in Figure 1. The nodes connected with dotted edges can overhear each other's transmissions. The arrows next to the packets and their sources indicate the destination for these packets. In this specific scenario COPE does not provide any gains. Assuming a loss probability of p on each of the links, the classic approach provides a rate of $\frac{1-p}{6}$.

However, if the relay codes $P_Q = P_W \oplus P_X$ and $P_R = P_X \oplus P_Z$ and broadcasts these, each of the three destinations will be able to recover their packets: X will compute $P_W = P_X \oplus P_Q$, Y will compute $P_X = P_Z \oplus P_R$ and W will compute $P_Z = P_W \oplus P_R \oplus P_Q$. Thus, only 5 transmissions are required instead of 6.

Another coding method which can be applied for this example is the so called LOOP-code (originally introduced in [DWHC07]). In this case the relay broadcasts P_Q , P_R and $P_S = P_Z \oplus P_W$. Although six transmissions occur, each node is able to decode their packet in two possible ways and throughput gain is obtained through redundancy in a lossy environment. The achieved rate of the LOOP-code is $\frac{1-2p^2+p^3}{6}$.

For the general case, the authors formalize the problem of selecting the "best" packets to code in an interesting fashion. Given a fixed threshold probability p , the problem is how to find a minimal coding strategy (i.e. minimum number of transmissions) such that packets are delivered with probability at least p . The design space for the problem is quite large, hence the authors go through a series of approximations and heuristics to construct efficient coding strategies of low complexity.

First, observe that the number of possible ways to combine n packets is exponential

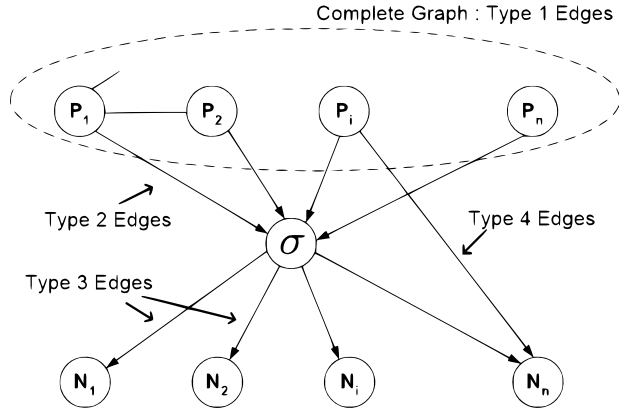


Figure 2: The “binary” coding graph H

(about 2^n)—a problem present at COPE, too. Due to the high cost of exploring all possible ways of coding, the authors consider the “binary network coding” by limiting coding to pairs of packets only. This is a much simpler problem, which, however, is shown to be still $\#P$ -complete.

Specifically, consider a relay node that needs to send n packets $P = (P_i)_{i=1}^n$ to n nodes $N = (N_i)_{i=1}^n$. Binary network coding regards the decision at the relay of which packets to code and which to transmit natively, and can be formulated as follows:

We construct the graph $H = (P \cup N \cup \{\sigma\}, E)$ in Figure 2, with σ being a special vertex to model packet availability at different next-hop nodes. It has 4 types of edges. Type-1 edges (P_i, P_j) are bidirectional edges corresponding to the possibility of transmitting a coded package $P_i \oplus P_j$. Type-2 edges connect each P_i to the central node σ and denote the possibility of sending the native packet P_i . Type-3 edges connect σ to each of the N_i . Type-4 edges directly connect packages P_i and nodes N_j , denoting the fact that N_j already possesses P_i .

A coding strategy S corresponds to a selection of type-1 and type-2 edges. These edges represent the packets transmitted by the relay node (i.e. the presence of an edge of type-1 (P_i, P_j) in S denotes the fact that the relay will send package $P_i \oplus P_j$, while a type-2 edge (P_i, σ) denotes the transmission of “native” packet P_i). The authors show that upon completion of packet transmissions by the relay as indicated by a strategy S , a neighbouring node N_i will be able to decode a packet P_i if and only if there exists a directed path from P_i to N_i in the subgraph induced by S .

For a strategy S and an index i , the computation of the probability $q_i(S)$ that a packet P_i can be successfully decoded at node N_i , is modelled in the following manner: weights w_e are assigned to the edges e in a specific way. Namely, for a type-4 edge (P_ℓ, N_j) its weight will be the probability that packet P_ℓ is already present at N_j . Type-3 edges have weight 1. Every edge in S (which is either a type-1 edge or a type-2 edge) is assigned weight r_i , which is equal to the probability that a packet transmitted from the relay is successfully delivered to node N_i . Then, the weight of an edge is interpreted as the probability that the edge is present in the graph. In this setting $q_i(S)$ represents the probability that there is a path from P_i to N_i in the graph induced by S . Computing q_i on a general graph is $\#P$ -complete. The authors conclude

that the problem at hand

$$\min_S |S|, \quad \text{s.t. } q_i(S) \geq p, \forall i \tag{5}$$

is also $\#P$ -complete.

To make the computation of $q_i(S)$ more tractable, the authors restrict themselves to sets \mathcal{P} of paths disjoint in their edges, except for type-3 edges. Since the probability $w(\pi)$ of a path π being present is the product of the weights of the edges it is made of, in this case q_i can be computed as $q_i(S) = 1 - \prod_{\pi \in \mathcal{P}(S)} (1 - w(\pi))$.

As binary network coding is still too hard to solve optimally (it is shown to be NP -hard), the authors propose two heuristics to design efficient coding strategies S : CLONE-kDSP and CLONE-kLoop.

The CLONE-kDSP algorithm builds the strategy S as follows: first the nodes are sorted increasingly according to the delivery probabilities r_i from the relay to node N_i . In this order, the successive shortest paths algorithm is run for the pairs (P_i, N_i) . For successive pairs, edges already used are assigned cost 0, and 1 otherwise. The decoding probability q_i is computed on the fly and the algorithm moves on to the next pair (P_i, N_i) when the threshold p is reached or no more edge-disjoint paths can be selected. In the end, S will be the set of type-1 edges selected in the whole process.

The CLONE-kLOOP algorithm outputs type-1 edges forming loops of length k . The last coding-enabled solution presented in the case in figure 1 is an example of a CLONE-3Loop code.

Returning back to the general case in which any number of packets can be XOR-ed in a single packet the authors propose the CLONE-MultiXOR heuristic. This is a greedy approach which defines the “usefulness” of a coded packet d with respect to an existing set of coded packets S as the additional number of ways in which each packet can be decoded by its destination. At each step, the coded packet maximizing the usefulness is added to the set S until $|S| = n$.

Simulations are shown for the three technique. In practice, however, only the CLONE-kLoop was implemented, as processing power constituted a bottleneck when using the other two strategies.

The authors raise an interesting problem with CLONE. They also show better gains than COPE over the traditional approach in heavily lossy environments (around 43%).

2.1.5 ER [RIM⁺07]



The authors in [RIM⁺07] develop and implement ER, a network coding scheme that opts to improve the efficiency of MAC layer retransmissions that are required to resend lost packets. The idea is to encode together packets lost at different destinations, so that those packets are received with a single transmission. Although the paper focuses on WLAN scenarios, ER can be easily extended to multihop topologies. The authors show that the problem of selecting the set of packets to code together towards minimizing the number of retransmissions is NP -hard! Hence, they discuss a set of practical heuristics and evaluate their efficacy through simulations and measurements.

ER includes mechanisms for (a) informing the relay about which packets have been successfully overheard by every node, (b) which packets, when and how should be retransmitted, and (c) which packets to encode together at each decision instance.

ER is implemented as a light modification of COPE and therefore adopts the acknowledgement mechanism of the latter. In particular, each node sends cumulative ACKs using a bit-map, contained in the additional COPE header that is transferred with each data packet. The authors have increased the bit-map size to 8 bytes (whereas in COPE it is 1 byte) to increase resiliency. With this, the effect of ACK losses becomes less pronounced, even though the overhead due to increased header length is higher. Another quite significant improvement with regards to COPE is that when nodes retransmit multicast packets, the destination nodes of these packets (and only those) are required to send feedback about successful reception/decoding. This reduces the overhead due to sending ACKs, especially in dense deployments, where the number of potential members in the multicast group can be large.

In order to determine which packets to transmit and when, ER estimates the retransmission timeout (RTO). In particular, nodes measure the time between the packet transmission and the reception of the corresponding ACK (referred to as the “data unit transmission delay” in other studies); this measurement is performed for every packet that has not been retransmitted. Furthermore, to efficiently decide whether to transmit a new or a lost packet, ER tries to balance the effects of low delay and high coding gain. This is achieved by employing the following heuristic: retransmit the packet when the retransmission queue reaches a certain threshold or when the packets in the retransmission queue timeout. The authors suggest values for timeout and retransmission queue threshold by performing simulations and experiments.

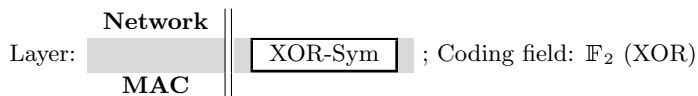
Additionally, the paper presents an analytical proof of the NP-hardness of the problem “which packets to code together”. Alternatively, a set of heuristics are discussed for approaching a solution to the problem.

1. COPE’s greedy approach “sort the packets as per their arrival time” is examined. With this strategy, the transmitter starts with the first packet in the queue (oldest packet) and proceeds with subsequent packets, taking into consideration the ability of the intended receivers to successfully decode the XOR packet.
2. With the “sort by utility” strategy, each packet is tagged with a utility, which reflects the number of receivers that would need to receive it. Priority for transmission is given to the packets with the higher utility value. Similarly as with the previous heuristic, the transmitter starts with the highest-utility packet and proceeds with further packets, considering the ability of the receivers to perform successful decoding.
3. Performing “max-clique partitioning” is another heuristic. We search the graph to identify the maximum clique. We remove it and proceed with finding another one, etc. Due to the NP-hardness of finding maximum clique, one may employ a simple heuristic by starting with the highest-degree vertex and adding more vertices while maintaining the clique property.
4. Finally, the authors propose an “exhaustive-search” algorithm in order to minimize the number of retransmissions. The idea is to discover the smallest number of packet combinations, which converts the current state to the state where every node happily receives the desired packet. Although this algorithm is computationally expensive (and therefore inappropriate in practical settings) it can serve as the baseline for comparison with the other aforementioned heuristics.

These heuristics are thoroughly evaluated and are shown to perform well in practical settings. ER was implemented on the publicly available COPE source code, and therefore inherits all the properties of this implementation. The 802.11b WLAN setting with one AP and 6 clients is considered throughout the experiments; the bit rate

is fixed. Overall, the presented experimental and simulation results suggest that ER effectively retransmits lost packets in different scenarios.

2.1.6 XOR-SYM [CP07]



Chaporkar and Proutière in [CP07] perform an analytical study to investigate the use of network coding in wireless multihop networks with unicast sessions. They demonstrate (by referring to certain topological scenarios and schedules) that if network coding and scheduling are designed separately, the throughput with network coding may not be improved; it may actually even degrade! Next, they investigate the combined problem of coding and scheduling and propose an optimal scheduling framework that considers various wireless network metrics, towards maximizing the throughput.

Furthermore, the authors propose XOR-Sym, a network coding scheme that can provide similar benefits to COPE, while requiring a lower implementation complexity than the latter. The idea with XOR-Sym is to decode packets only at their destinations and not at intermediate relays. With this, relays do not need to perform computationally expensive look-up searching in order to identify encoding opportunities with previously received/stored packets. In contrast, they simply need to perform bitwise XOR of HoL (head-of-line) packets, something that increases the scalability of the network coding system. The name “XOR-Sym” comes from the feature that XOR operations are performed on packets that belong to symmetric sessions only, where the source of one end of a router is the destination of the other end; to this extent, opportunistic listening is ignored.

More specifically, consider a scenario where nodes A and B are the two ends of a 4-hop route, and that they maintain two opposite flows (i.e., from A to B and reverse); the two intermediate relays simply forward the packets for these two flows. With XOR-Sym, the relays simply perform XOR operations with packets that belong to these two flows only. Since intermediate relays do not perform any decoding operations, the already encoded packets can also be further XOR-ed again as they travel to their destinations. As an example for this scenario, node A would receive a XOR packet that is comprised of 4 packets in total; three of these packets would have been transmitted by node A in the past (and are stored in node A for decoding purposes), while the 4th packet is the one sent from node B. Note that XOR-Sym requires that node A stores all the packets that it has recently transmitted. In addition, the relays are required to perform XOR operations only for the specific two flows, and for no other flows that traverse the particular route. Otherwise, unknown (not stored) packets would mix while encoding, and this would make it impossible for nodes A and B to successfully decode each other’s packets.

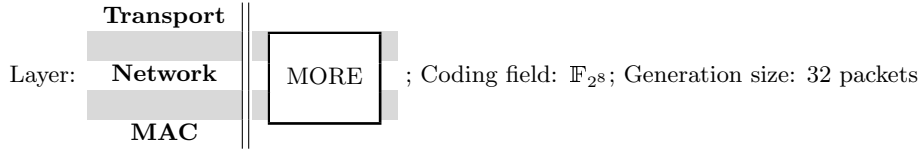
The authors do not implement their scheme in a real network. However, they perform thorough numerical experiments that verify their analytical results. Moreover, XOR-Sym cannot work in scenarios with non-symmetric flows.

2.2 Intra-session coding

The proposals described here use coding only on packets belonging to the same session. They typically divide data into “generations”. Coded transmissions deliver generations

successively and coding involves only packets within the same generation. For proposals in this section we point out the generation size in addition to information about the layers concerned and the coding field size. We note the following interesting fact: MORE and MIXIT are deemed “insensitive” to the generation (batch) size. To support this affirmation, the authors of MIXIT compare batch sizes of 8, 12, 16 and 32, while the authors of MORE look at batch sizes of 8, 16, 32, 64 and 128.

2.2.1 MORE [CJJK07]



MORE [CJJK07] (MAC-independent Opportunistic Routing and Encoding) is a proposed system relying on intra-session coding. It targets wireless multipath unicast sessions (an extension to multicast is also provided in [CJJK07]). The authors place MORE between the MAC and NETWORK layers. However, MORE provides routing, as well as reliable end-to-end packet transfer which are network and transport layer roles.

MORE works as follows: The flow source codes “batches” of K “native” packets together. It keeps sending linear combinations from the current batch until receiving an ACK from the flow destination. The ACK is given priority along a “reliable” shortest path from the flow destination to the flow source.

Upon receiving the ACK the source moves on to the next batch. MORE packets contain a list of “forwarders” sorted decreasingly with respect to the ETX metric to the destination. Intermediary nodes listen to the channel. If they are in the forwarders list, they process the packet. If the packet is “innovative” (i.e. not linearly dependent with the formerly received packets), a forwarder will store it regardless of its ETX.

If a forwarder has a lower ETX than the previous hop, then it will increase its “credit counter” by a certain number of “transmission credits” (TX_credits). These transmission credits are computed in a distributed fashion, relying on link-loss probabilities.

The credit scheme seeks to solve the following optimization problem [CJJK06]:

$$\begin{aligned}
 & \arg \min \sum_i z_i \\
 & \sum_j x_{ij} - \sum_j x_{ji} = \begin{cases} 1 & \text{if } i \text{ is the source} \\ -1 & \text{if } i \text{ is the destination} \\ 0 & \text{otherwise} \end{cases} \\
 & x_{ij} \geq 0, \quad \forall i, j \\
 & z_i \cdot p(i, J) \geq \sum_{j \in J} x_{ij},
 \end{aligned}$$

where z_i is the number of transmissions made by i , x_{ij} is the optimal number of packets that routing should deliver from i to j and $p(i, J)$ is the probability that at least one node in J receives a packet transmitted by i . Denoting by p_{ij} the delivery probability from i to j , we get $p(i, J) = 1 - \prod_{j \in J} (1 - p_{ij})$.

The authors propose performing measurements using ping probes to compute the latter. A forwarder will only transmit a recoded packet when its credit counter is positive. Each transmission will incur a decrement of this counter. Forwarders do not need to receive the ACK, as they are implicitly notified of a new batch becoming available upon reception of a packet from this new batch.

Measured link loss probabilities are distributed to the entire set of nodes and thus every node has a global image of the network. Moreover, building a “forwarders” list requires ETX knowledge of all nodes by the source. In this sense, MORE behaves like a link state routing algorithm (as the authors themselves state). Thus in larger networks MORE might become inefficient.

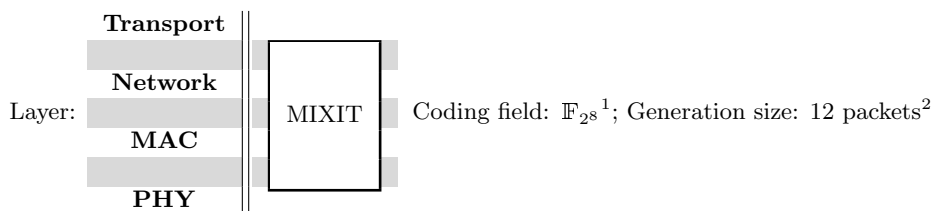
“Natural” modifications to MORE extend it to the multicast case.

Network coding allows MORE to use minimal signalling. Thus, by sending recoded packets instead of native ones the choice of a specific packet to forward is no longer an issue.

The authors compare MORE to existing routing algorithms in mesh networks. MORE achieve the best performance among the tested approaches, even when adaptive 802.11 rates are used.

MORE uses a 20 node testbed, each of them equipped with a wireless card transmitting at a power level of 18dBm operating in 802.11 ad-hoc mode with RTS/CTS disabled. The Click toolkit [MKJK99] is used as well as the Roofnet software package [ABB⁺04].

2.2.2 MIXIT [KKBM08]



MIXIT [KKBM08] is a cross-layer system relying on an approach which is similar to MORE. There are two main differences between MIXIT and MORE: (1) MIXIT uses a finer granularity and codes symbols rather than packets, (2) with MIXIT, the “credit” assignment scheme which is used to decide how many transmissions a node will perform is more advanced. Moreover, due to MIXIT’s tolerance to corrupted symbols, a more aggressive MAC improves performance, as it increases the amount of coding opportunities.

Like in MORE, the source codes batches of K packets together. A packet is viewed as a sequence of “symbols”. A MIXIT symbol is a group of PHY symbols (symbols from the physical layer). Forwarder nodes use “hints” from the physical layer to infer which MIXIT symbols within a packet are “clean” (i.e. correctly received with high probability) and which are “dirty” (i.e. incorrectly received). The distinction between “clean” and “dirty” MIXIT symbols is done at the network layer. MIXIT uses the SoftPHY [JB07] interface which annotates PHY symbols with confidence values. If at least one PHY symbol has a confidence value lower than a certain fixed threshold, then

¹Possibly, it is not specified

²These represent original packets. However, after adding error-correction, 16 resulting packets are used for coding.

the whole MIXIT symbol is marked “dirty”. Unless otherwise stated, in what follows we will use the word “symbol” to refer to a MIXIT symbol.

Forwarders code several packets together in the following way: they choose a vector of coefficients at random (one coefficient per packet); then they code the packets symbol by symbol, using only the “clean” symbols at a certain position. For example, if two packets $a = (a_i)_{i=1}^I$ and $b = (b_i)_{i=1}^I$ are to be coded together with coefficients α and β chosen at random, then the resulting packet c will be composed of symbols c_i such that $c_i = \alpha a_i \mathbb{1}_{\{a_i = \text{“clean”}\}} + \beta b_i \mathbb{1}_{\{b_i = \text{“clean”}\}}$.

The encoding is specified in the header of the coded packet as a sequence of “runs” (i.e. consecutive symbols in the form “start”:“end”), their associated coefficients and the identifiers of the original packets to which these coefficients were applied (namely those packets that did not have “dirty” symbols in that specific run).

If all the “clean” symbols are to be transmitted as described above, the overhead of specifying the runs might become too large. In order to minimize the encoding description overhead, the authors propose a dynamic programming (DP) scheme to determine “on the fly” the “smallest” number of runs, while promoting “innovative” symbols. In other words, when packets need to be coded together, some symbols will temporarily be marked artificially “dirty” in such a way that “innovative” information with respect to previous transmissions is still conveyed. This allows longer runs and thus a lower overhead. The header of a coded packet is always “clean”, as it is protected with FEC.

Similarly to MORE, forwarders use a “credit”-based scheme to determine the number of transmissions they should perform. However credit assignment is more complex and takes into account both link quality and congestion. The authors define the C-ETS metric at an intermediary node i to a destination d as follows: $\text{C-ETS}(i \rightsquigarrow d) = PQ(i \rightsquigarrow d) + kQ(i)$, where PQ denotes the path quality and is approximated using the ETS metric (expected number of transmissions for delivering a symbol) and $Q(i)$ is the size of the backlog at node i weighted by a factor k . The last term enables a “backpressure”-style congestion control.

Nodes also have estimations of link symbol delivery probabilities $p(i \rightarrow j)$. Credits are assigned to downstream nodes on a per-flow basis. The assignment procedure is as follows: for a specific flow, a node n will sort its downstream neighbours according to the C-ETS metric (a lower value of C-ETS indicates a better path quality). Denote the symbol delivery probabilities for the sorted nodes by p_1, p_2, \dots . Then node 1 will be instructed to forward all symbols it receives from node n and will thus get 1 credit, node 2 will forward only a fraction $(1 - p_1)$ of the symbols it receives from node n and will only get $(1 - p_1)$ credits, node 3 will receive $(1 - p_1)(1 - p_2)$ credits and so on. The probability that any of the downstream nodes will receive some symbol is $P = 1 - \prod_j (1 - p_j)$. Thus, a node would have to send on average $\frac{1}{1-P}$ symbols per queued symbol.

Each node maintains the following per-flow quantities:

- The batch buffer, containing clean received symbols
- The credit counter, which accumulates credits received from upstream nodes
- The transmission counter, which is increased upon credit assignment by $\frac{1}{1-P}$

When the credit counter is greater than 1, a node will run the credit assignment procedure and will create as many coded packets as the transmission counter indicates. Then the transmission counter and the credit counter are decremented (the details are a bit unclear...).

The C-ETS values are piggy-backed on transmitted packets overheard by upstream nodes.

MIXIT works on any MAC, the authors claim. However, a more aggressive MAC is proposed which creates more “concurrency”. The benefit of two nodes transmitting in parallel is evaluated. If this benefit is positive, then the transmissions take place simultaneously and the number of “dirty” symbols ideally remains “sufficiently” low such that coding remains possible.

Namely, consider two nodes n_1 and n_2 transmitting packets from two flows k and ℓ , respectively. For all $(n, q) \in \{(n_1, k), (n_2, \ell)\}$ delivery likelihoods $D(n, q)$ and $D^c(n, q)$ are computed both in the case of concurrent and non-concurrent transmissions:

$$D^c(n, q) = 1 - \prod_{j \in \text{downstream}(n, q)} (1 - p^c(n, j))$$

$$D(n, q) = 1 - \prod_{j \in \text{downstream}(n, q)} (1 - p(n, j)),$$

where the symbol delivery probabilities on link (i, j) are denoted by $p^c(i, j)$ when n_1 and n_2 transmit concurrently and by $p(i, j)$ when they do not.

Nodes decide to use concurrent transmissions if the following condition is verified:

$$D^c(n_1, \ell) + D^c(n_2, k) > \frac{D(n_1, \ell) + D(n_2, k)}{2}.$$

The per-link symbol delivery probabilities are evaluated as a function of the SNR³ in low activity periods of the system. Each node takes turns to broadcast probe packets while the other nodes perform measures of SNR and delivery probabilities. To evaluate the benefit of using parallel transmissions in the modified MAC an approximation is made: Namely, when two nodes n_1 and n_2 transmit in parallel, the delivery probability at a third node m is computed in dB as a function of the SINR⁴: $p(\text{SINR}(n_1, n_2, m))$, where $\text{SINR}(n_1, n_2, m) = \text{SNR}(n_1, m) - \text{SNR}(n_2, m)$. Intuitively, the signal having the lower SNR will be perceived as noise.

The issue of inaccurate hints describing symbol reliability is also considered. Thus, transparently to the whole procedure described so far, symbols are pre-coded using a rateless error correcting code. This code relies on the vector space preserving property of random linear coding. It provides the following guarantees: if m symbols were wrongly marked as “clean” among the B original symbols, then $B + 2m$ symbols are required to correctly perform decoding.

MIXIT was deployed on a 25 node testbed. Zigbee software radio is used over USRP with a 2.4GHz daughterboard. The latter provides the “hints” necessary to determine whether a symbol is clean or not. The peak data rate on the link is 250Kbits/s when no other transmissions are in progress.

2.2.3 Ho & Viswanathan [HV03]

The article [HV03] proposes network models for both wired and wireless networks. It discusses multicast application, and considers both correlated and uncorrelated data sources per session.

It characterizes the set of per session data rates achievable in this model. It then goes on to describe algorithms for effectively delivering traffic, given feasible data arrival

³Signal to Noise Ratio

⁴Signal to Interference and Noise Ratio

rates. These algorithms are of so-called back-pressure type, where weights of competing transmissions are computed as differences of backlogs, and power selection / scheduling decisions are chosen so as to maximize a sum of transmission rates weighted by corresponding back-pressure coefficients.

We now provide additional details for the model of wireless networks considered, which has the interesting feature of capturing the “local broadcast” nature of wireless transmissions.

The model specifies for each transmission power vector P and channel gain matrix S the transmission rate $\mu_{ij}(P, S)$ between sender i and receiver j . For a given set Z of potential receivers of transmissions from i , it then introduces the broadcast transmission rate from i to all receivers j in Z , $\mu_{iZ}(P, S)$. As an example, it suggests using the formula

$$\mu_{iZ}(P, S) := \min_{j \in Z} \mu_{ij}(P, S)$$

for defining such rates⁵.

Based on this, a primary capacity region Γ is identified, representing the sets of long term rates $\{r_{iZ}\}_{iZ}$ that can be achieved, as

$$\Gamma = \sum_S \pi(S) \overline{\{(\tilde{\mu}_{iZ}(P, S))_{iZ}, P \in \mathcal{P}\}}^{cx}$$

where $\pi(S)$ represents the fraction of time where the channel gains are equal to S , \mathcal{P} is the set of feasible power allocation vectors, and $\overline{\mathcal{S}}^{cx}$ is the convex hull of set \mathcal{S} . In the above, $(\tilde{\mu}_{iZ})_Z$ represents the vector of instantaneous rates $\tilde{\mu}_{iZ}$ at which sender i can simultaneously send to distinct receiver sets Z .

It then proceeds to characterize the vectors of feasible session rates λ_c , for all multicast sessions c , as follows.

- For each session c , every pair of nodes (u, v) and each destination j of c , there is a flow $\{f_{uv}^{cj}\}_{uv}$ transferring the injected rate λ_c from the session sources to the destination j .
- Each flow variable f_{uv}^{cj} can be split further into non-negative terms f_{uvZ}^{cj} , one per set Z of nodes containing node v .
- Denote by \mathcal{T}_c the set of receivers for session c . There is a vector $r = (r_{iZ})_{iZ}$ in the primary capacity set Γ such that:

$$\sum_{\text{sessions } c} \max_{j \in \mathcal{T}_c} \sum_{v \in Z} f_{uvZ}^{cj} \leq r_{uZ}.$$

Feasibility of multicast under these conditions (with strict inequality in the last display) is established by proving stability of the following back-pressure scheme.

Each node i maintains, for each session c and each receiver for c , j , a virtual backlog U_i^{cj} . For each such node i , and each set of listeners Z , a weight w_{iZ} is defined as follows.

For each session c and each receiver j for session c , a weight w_{iZ}^{cj} is defined as

$$w_{iZ}^{cj} = \max(\max_{v \in Z} (U_i^{cj} - U_b^{cj}), 0). \quad (6)$$

Then the weight w_{iZ} is defined as

$$w_{iZ} := \max_c \sum_{\text{receiver } j} w_{iZ}^{cj}. \quad (7)$$

⁵An alternative definition of such quantities, is used in [RGK⁺07] to model lossy links with independent losses.

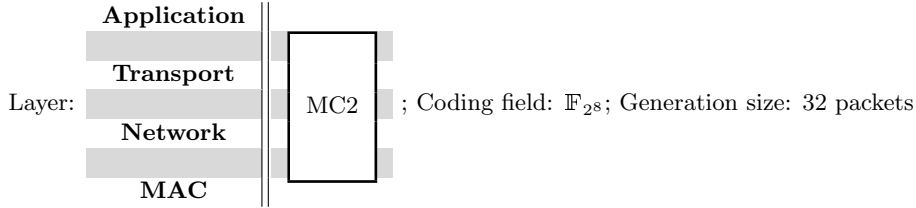
Power vector P is then specified as the one that maximises

$$\sum_{i,Z} w_{iZ} \mu_{iZ}(P, S)$$

where S captures the current channel gains.

Transmission is then done as follows. The session c achieving the maximum in the definition (7) is scheduled for broadcast link (i, Z) . For each receiver j of the elected session c , virtual queue $U_i^{c,j}$ is decremented by the corresponding amount $\mu_{iZ}(P, S)$, and virtual queue $U_v^{c,j}$, where v achieves the maximum in (6), is incremented by the same amount.

2.2.4 Multipath Code-Casting [RGK⁺07]



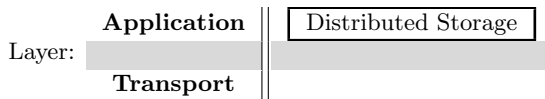
MC2 targets the same application scenario as MORE, namely multipath unicast sessions. However it also addresses the transport layer, and incorporates rate control mechanisms. It finds its motivation in an optimisation formulation, capturing the *maximal welfare* that the system can provide. This formulation builds upon the one in Ho and Viswanathan, described in the next section, which provides a general characterization of the set of feasible rates per session, also known as the schedulable region.

Based on this characterization, it formulates welfare maximisation over the following set of variables: power, rate and scheduling (choice of sessions to schedule at each sender). It considers, as in MORE, lossy wireless channels with independent losses over receivers. To achieve the optimal allocation, it proposes back-pressure mechanisms based on credits. These are rather intricate; let us only say that each relay node j maintains a credit count $C(j, J)$ for each subset of potential over-hearers J .

A comparison to MORE (in its initial version [CJJK06]) is performed. It is shown that MORE maximizes flow rate for a single flow scenario and when only one link can transmit at a time, hence achieves the same performance as MC2 under these assumptions. However it is shown that MC2 can outperform MORE still in a single flow scenario with several non-interfering links.

Simulations as well as experiments of a simplified version of MC2 have been performed, which support the benefits of MC2 over MORE. The MC2 prototype is built on top of the VRR system [CCNR06]. The “broadcast” mode of 802.11 is used (as opposed to the “pseudo-broadcast” mode in COPE) at a rate of 6Mbps in 802.11a.

2.3 Coding for distributed storage systems [DGWR07]



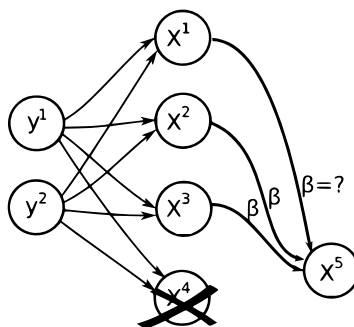


Figure 3: The repair problem

Dimakis et al use network coding to optimize repair bandwidth of erasure codes in distributed storage systems [DGWR07]. Previous work had shown that erasure codes are optimal for redundancy-reliability trade-off. However the repair bandwidth is high (M bits to recreate a block of size M/k). When a node fails, an incoming node downloads subsets of data from different surviving nodes, reconstructs a coded block using the downloaded data and stores it. For example, in Figure 3 content y^1, y^2 of total size M is encoded into 4 fragments distributed at nodes x^1, x^2, x^3 and x^4 , with the property that any 2 nodes can reconstruct the data (i.e. using a $(4, 2)$ MDS⁶ erasure code). Upon failure of x^4 , the problem is to find the minimum amount of information β that a newcomer x^5 needs to recover from the three active nodes such that it can replace x^4 . All previous coding constructions required access to the original data object (thus, in this case a value of $\beta = \frac{M}{3}$).

The authors introduced regenerating codes where nodes download functions (linear combinations) of stored data from surviving nodes to significantly reduce the repair bandwidth. The main idea is that if each node stores slightly more than M/k bits, the repair bandwidth can be significantly reduced.

The analysis in the paper is based on the notion of information flow graph, which describes how the information of each data object is communicated through the network, stored in nodes of limited memory and reaches reconstruction points at the data collectors. Using this formulation, the code repair problem is translated to a multicast problem in the information flow graph. Known results for network coding can then be readily used to show that a code repair can be achieved if the underlying information flow graph has enough connectivity. More specifically, the analysis boils down to computing minimum cuts on arbitrary graphs and solving an optimization problem to minimize storage α subject to a sufficient flow constraint.

Figure 4 provides an illustration of the information flow graph G corresponding to the $(4, 2)$ code of Figure 3. Assume the size of the original content is $M = 2\text{Mb}$. Node x_{in}^5 is connected to the $d = 3$ active storage nodes. Assuming β bits communicated from each active storage node, as previously stated, of interest is the minimum β required. The min-cut separating the source and the data collector must be larger than M for reconstruction to be possible. For this graph, the min-cut value is given by $1 + 2\beta$, implying that $\beta \geq 0.5\text{Mb}$ is sufficient and necessary. Thus, it suffices and it is necessary to communicate 1.5Mb instead of 2Mb to node x^5 in order for it to replace x^4 .

⁶Maximum Distance Separable

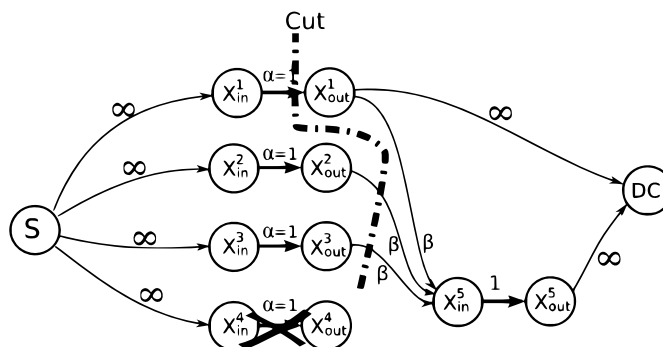


Figure 4: The information flow graph corresponding to the $(4, 2)$ code of Figure 3

Regenerating codes are characterized by an optimal trade-off curve between storage and repair bandwidth and there exist codes that can achieve any point in this curve. The two extremal points in the tradeoff curve are minimum-storage regenerating codes (MSR) and minimum-bandwidth regenerating codes (MBR)

Using a set of trace datasets, the authors compare the performance of MSR, MBR regenerating codes with Replication, Ideal Erasure codes, Hybrid in terms of expected repair bandwidth and expected storage availability. These two performance indices are computed based on two parameters f and α specific to each trace dataset, where f is the fraction of nodes that permanently fail per unit time, causing data transfers to repair the lost redundancy; α is the probability that a node is temporarily available.

The authors also compare MSR and MBR codes to the Hybrid strategy proposed by Rodrigues and Liskov [RL05]. Compared to Hybrid, MSR codes offer slightly less maintenance bandwidth and storage and a simple system architecture since only one type of redundancy needs to be maintained. In Hybrid, the disk I/O can become a bottleneck during repairs. This is because the disk storing the full replica and generating the encoded fragments needs to read the whole data object and compute the encoded fragment.

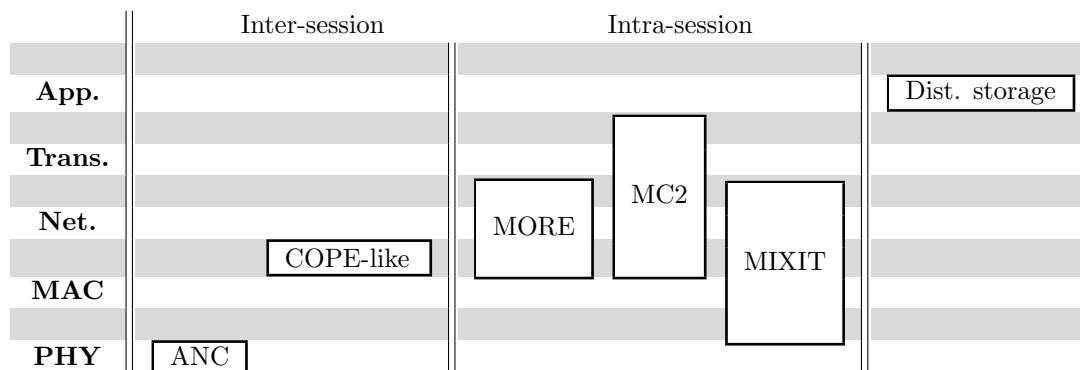


Table 1: Placement of proposed protocols in the networking stack. The “COPE-like” protocols are COPE, noCoCo, CLONE, ER and XOR-Sym

3 Comparison

In this section we provide a critical overview of the described protocols. We take several viewpoints in order to emphasize the advantages and shortcomings of the proposed methods. We begin by dividing the protocols into “intra-session” (MORE, MIXIT, MC2) and “inter-session” (COPE, CLONE, ANC), according to the type of coding they perform. Thus, the “intra-session” protocols tend to insure end-to-end reliability for data transfer within a “session” (be it unicast or multicast), while “inter-session” protocols are transparent to the network layer and only consider a local coding problem (between neighbours within transmission range).

3.1 Deployment complexity

Deployment is a delicate point when proposing a new protocol. Since deployment cost is high, replacing an existing working protocol is hard. One would need to leverage performance gains against component replacement. “Intra-session” protocols will affect several layers of the protocol stack, thus incurring a higher implementation cost, while “inter-session” methods stay below the network layer and are completely transparent to existing routing and transport protocols. We illustrate this in Table 1.

Below we provide implementation details about the presented protocols as found in the corresponding cited papers.

COPE was implemented on a 20 node testbed using the Click toolkit [MKJK99]. The nodes run Linux and use 802.11a. The “pseudo-broadcast” mode is preferred (a destination is picked for a packet transmission, while every other node in range uses the “promiscuous” mode to intercept the packet; the overhearing nodes then check to see if they are in the receiver list found in the packet header). Routing is performed using Srcr [BABM05].

ANC uses a Software Defined Radio (SDR) implementation. The signal is represented as complex numbers. GNURadio software is used, resulting in a bit rate of 500kb/s. The SNR is 20-30dB. The system uses Universal Software Radio Peripheral (USRP) with RFX2400 daughterboards.

CLONE was implemented on a 12-node Linux testbed. The Click toolkit is used [MKJK99]. The only deployed heuristic was the proposed CLONE-k-Loop, while the other two proved to be too demanding in terms of processing power. The underlying routing protocol is Srcr [BABM05], based on the ETX metric.

MORE uses a 20 node testbed, each of them equipped with a wireless card transmitting at a power level of 18dBm operating in 802.11 ad-hoc mode with RTS/CTS disabled. The Click toolkit [MKJK99] is used as well as the Roofnet software package [ABB⁺04].

MIXIT was deployed on a 25 node testbed. Zigbee software radio is used over USRP with a 2.4GHz daughterboard. The peak data rate on the link is 250Kbits/s when no other transmissions are in progress.

MC2 prototype is built on top of the VRR system [CCNR06]. The “broadcast” mode of 802.11 is used (as opposed to the “pseudo-broadcast” mode in COPE) at a rate of 6Mbps in 802.11a.

3.2 Computational complexity

Due to the somewhat complex mathematical operations involved in the process of coding and decoding (such as matrix inversion), most proposed protocols have a high computational overhead. Authors use various methods to alleviate the implicit complexity of coding. For instance,

- COPE-style protocols use the smallest non-trivial finite field for coding, leading to exclusively employing XOR operations, both for coding and for decoding. Consequently, no explicit matrix inversion needs to be performed.
- MORE uses a lookup table instead of actually computing products in a finite field.

MC2 authors find the time required for coding packets to be less than 2ms on a machine having a 847MHz CPU and 256MB of RAM.

However complexity may also arise for different reasons than coding per se. For instance, MIXIT employs a form of overhead minimization. The authors claim that due to the size of the problem instances, the dynamic programming-based proposed optimization takes less than one millisecond. The authors of CLONE are clearly facing an implicitly computationally harder problem, that of selecting which packets to code. Two of the three proposed heuristics are even too complex to run in real-time.

3.3 Importance of coding and wireless

For “intra-session” techniques like MORE the benefits obtained by the use of coding and by the fact transmissions are done in a wireless environment are orthogonal in the following sense: On the one hand, coding allows a relaxation in end-to-end signalling constraints. Since random linear combinations of the original packets are broadcast, the destination will eventually receive enough linearly independent combinations and it will not require the retransmission of a specific packet as it would need to do in the absence of coding and therefore it will not send a packet over the network in this purpose. On the other hand, the wireless medium is a substitute for multiple sources (or

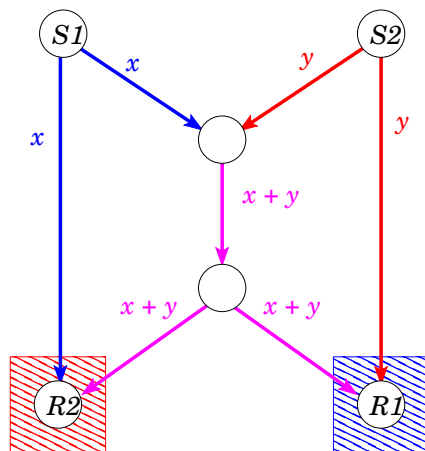


Figure 5: Inter-session coding in the butterfly network

multipath routing) sending randomly coded blocks in a wired environment (similarly to the Avalanche peer-to-peer system [GR]).

For “inter-session” techniques however (such as COPE), the gain of using coding cannot be obtained without the broadcast nature of the wireless medium. The two (coding and wireless) go “hand in hand” to provide a more efficient way of transmitting packets.

We raise the following question. Consider the MORE setting in which no coding is performed and there is perfect signalling in the wireless network (i.e. every node knows at any time whether a packet is present at another node in the network). How does network performance of a set of unicast sessions in this case compare to the case of a real wireless network where coding is employed (such as MORE)? Intuitively the rate at which data is received should be the same in both cases. Network coding might still provide an advantage in terms of delay, as a single lossy link might hold off a specific packet at an intermediary node in the first case, while in the second case coding implicitly goes around this problem (since linear combinations arrive on all paths from the source to the destination).

Another open question concerning inter-session coding techniques naturally emerges from the previous remarks. What is the relevance of a technique such as the one described in COPE for wired networks? For instance, for the classic butterfly network, considering two different sessions (S_1, R_1) and (S_2, R_2) as depicted in Figure 5, inter-session coding manages to provide a rate of 1 instead of a rate of $1/2$ provided by the classical routing scheme.

3.4 Innovation

There are two major approaches proposed for network coding in wireless, the exponents of which are COPE and MORE. While they bring performance improvements (in some scenarios even impressive improvements – e.g. ANC), the other existing proposals do not really innovate with respect to network coding.

- CLONE proposes to introduce resilience to deal with lossy links, but still uses

roughly the same idea as COPE. It explores more complex coding opportunities, translates them ingeniously to a graph problem which unfortunately proves to be computationally hard. Subsequently some heuristics are proposed.

- ANC takes COPE one step further, by reducing the number of wireless slots used for transmission. It is implemented on simple topologies and performance gains over previous proposals seem encouraging.
- MIXIT is in fact a better implementation of MORE which makes smart use of potentially corrupted packets and which deploys a better transmission credit scheme. Although the improvements seem minor at a first glance, the throughput gain in experiments is surprisingly high (in the multiple flows scenario MIXIT manages to convey 3 times as much information as MORE).
- MC2 provides a complex theoretical framework which approaches optimality, yet its implementation complexity remains a drawback due to the signalling constraints imposed through the proposed credit management system.

3.5 Multicast

Network coding was first introduced as a method of attaining the theoretical capacity limit for multicast sessions. In the context of wireless networks the presented protocols deal mostly with multipath unicast sessions. The proposals that do mention multicast claim the extension from unicast is natural. In some works actual results are shown for multicast sessions: for example, Multicast-MORE is claimed to achieve a high throughput gain over Srcr[BABM05] (100-300%) and ExOR [BM03] (35-200%). Despite these encouraging results, multicast is not the central issue in the studied proposals.

4 Conclusion and future considerations

In this document we presented several proposed protocols which make use of network coding in wireless environments. The proposals range from theoretical approaches which seek an optimal resource allocation by formulating complex optimization problems (such as MC2 and the work of Ho and Viswanathan) to practical approaches (such as COPE, MORE and MIXIT).

We observe that all the network layers have been considered for network coding. In ANC [KGK07] the authors tackle even the physical layer and provide a proof of concept implementation using Software Defined Radios, showing encouraging results for specific topologies.

All the proposed solutions show important improvement in throughput over traditional techniques, even under scenarios involving heavy losses (e.g. CLONE). This leads us to the conclusion that network coding will become an important element in the design and deployment of wireless networks in the near future. However, although existing preliminary approaches are very promising, the general problem of what is the optimal way to perform rate control in network coding remains open. Moreover, it is difficult to provide a precise characterisation of the capacity of wireless networks.

There are many aspects of network coding and its applicability in wireless communications which remain poorly understood to this day. An open question for instance is the following: how do network coding-techniques compensate for poor signalling in the network? Otherwise put, is perfect signalling over a traditional network equivalent to a network in which packets are coded (such as MORE)? A related question: what signalling should be performed to maximise the benefits of network coding? And what are the implications on delay performance, in environments à la MORE?

Another open question relates to inter-session coding, a field in which there has been little advancement: What is the relevance of a method like COPE in the context of wired networks?

It would also be interesting to consider the nature of performance gains in the two cases, namely inter- and intra-session coding. If they were to be deployed together in the same network, would the gains be cumulative? Or, on the contrary, would the two techniques exclude each other, as either one of them exploits to the maximum the coding opportunities? Such a deployment is not straightforward and implies modifications below the network layer for the intra-session protocol in order for it to accommodate the inter-session one (as seen in Table 1).

Finally, it is not clear how coding should be performed to approach the optimal rate. For instance, we do not yet understand what is the impact on rate of performing non-linear coding. Dynamically changing conditions add further to the complexity of these problems.

As future work, we will try to make progress on these questions. We will specifically try to identify efficient signalling mechanisms for both unicast and multicast communications, and evaluate the resulting delay and rate performances. Application to specific wireless networks, such as sensor networks, might require optimization with additional operational constraints in mind. We will pay particular interest to the multicast scenario in general networks in the context of bandwidth-greedy peer-to-peer applications. Efficient signalling in this case is the key to addressing several hot topics, such as ISP-friendliness and flow control. One of the more difficult aspects is using only local adaptation techniques at the participating nodes instead of assuming global knowledge about the network.

References

- [ABB⁺04] Daniel Aguayo, John Bicket, Sanjit Biswas, Glenn Judd, and Robert Morris. Link-level measurements from an 802.11b mesh network. *SIGCOMM Comput. Commun. Rev.*, 34(4):121–132, 2004.
- [BABM05] John Bicket, Daniel Aguayo, Sanjit Biswas, and Robert Morris. Architecture and evaluation of an unplanned 802.11b mesh network. In *MobiCom '05: Proceedings of the 11th annual international conference on Mobile computing and networking*, pages 31–42, New York, NY, USA, 2005. ACM Press.
- [BM03] S. Biswas and R. Morris. Opportunistic routing in multi-hop wireless networks. In *Proceedings of HotNets '03*, 2003.
- [CCNR06] Matthew Caesar, Miguel Castro, Edmund B. Nightingale, and Antony Rowstron. Virtual ring routing: network routing inspired by dhds. In *In Proc. of ACM SIGCOMM*, pages 351–362. ACM Press, 2006.
- [CJKK06] S. Chachulski, M. Jennings, S. Katti, and D. Katabi. More: Exploiting spatial diversity with network coding. Technical report, MIT CSAIL, 2006.
- [CJKK07] Szymon Chachulski, Michael Jennings, Sachin Katti, and Dina Katabi. Trading structure for randomness in wireless opportunistic routing. *SIGCOMM Comput. Commun. Rev.*, 37(4):169–180, October 2007.
- [CP07] Prasanna Chaporkar and Alexandre Proutiere. Adaptive network coding and scheduling for maximizing throughput in wireless networks. In *MobiCom '07: Proceedings of the 13th annual ACM international conference on Mobile computing and networking*, pages 135–146, New York, NY, USA, 2007. ACM.
- [DGWR07] Ros G. Dimakis, P. Brighten Godfrey, Martin J. Wainwright, and Kannan Ramch. Network coding for distributed storage systems. In *In Proc. of IEEE INFOCOM*, 2007.
- [DWHC07] Qunfeng Dong, Jianming Wu, Wenjun Hu, and Jon Crowcroft. Practical network coding in wireless networks. In *MobiCom '07: Proceedings of the 13th annual ACM international conference on Mobile computing and networking*, pages 306–309, New York, NY, USA, 2007. ACM.
- [GK08] Shyamnath Gollakota and Dina Katabi. Zigzag decoding: combating hidden terminals in wireless networks. *SIGCOMM Comput. Commun. Rev.*, 38(4):159–170, 2008.
- [GR] Christos Gkantsidis and Pablo Rodriguez. Avalanche: Peer-assisted content distribution. <http://research.microsoft.com/en-us/projects/avalanche/default.aspx>.
- [HV03] Tracey Ho and Harish Viswanathan. Dynamic algorithms for multicast with intra-session network coding. In *Proceedings of the 43rd Annual Allerton Conference on Communication, Control and Computing*, 2003.
- [JB07] Kyle Jamieson and Hari Balakrishnan. Ppr: partial packet recovery for wireless networks. In *SIGCOMM '07: Proceedings of the 2007 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 409–420, New York, NY, USA, 2007. ACM.

- [KGGK07] Sachin Katti, Shyamnath Gollakota, and Dina Katabi. Embracing wireless interference: analog network coding. In *SIGCOMM '07: Proceedings of the 2007 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 397–408, New York, NY, USA, 2007. ACM.
- [KKBM08] Sachin Katti, Dina Katabi, Hari Balakrishnan, and Muriel Medard. Symbol-level network coding for wireless mesh networks. *SIGCOMM Comput. Commun. Rev.*, 38(4):401–412, 2008.
- [KRH⁺06] Sachin Katti, Hariharan Rahul, Wenjun Hu, Dina Katabi, Muriel Medard, and Jon Crowcroft. Xors in the air: practical wireless network coding. In *SIGCOMM '06: Proceedings of the 2006 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 243–254, New York, NY, USA, 2006. ACM Press.
- [MKJK99] Robert Morris, Eddie Kohler, John Jannotti, and M. Frans Kaashoek. The click modular router. *SIGOPS Oper. Syst. Rev.*, 33(5):217–231, 1999.
- [RGK⁺07] Bozidar Radunovic, Christos Gkantsidis, Peter Key, Pablo Rodriguez, and Wenjun Hu. An optimization framework for practical multipath routing in wireless mesh networks. Technical report, Microsoft Research, July 2007.
- [RIM⁺07] Eric Rozner, Anand Padmanabha Iyer, Yogita Mehta, Lili Qiu, and Mansoor Jafry. Er: efficient retransmission scheme for wireless lans. In *CoNEXT '07: Proceedings of the 2007 ACM CoNEXT conference*, pages 1–12, New York, NY, USA, 2007. ACM.
- [RL05] Rodrigo Rodrigues and Barbara Liskov. High availability in dhds: Erasure coding vs. replication. pages 226–239. 2005.
- [RSW⁺08] Shravan Rayanchu, Sayandeep Sen, Jianming Wu, Suman Banerjee, and Sudipta Sengupta. Loss-aware network coding for unicast wireless sessions: design, implementation, and performance evaluation. *SIGMETRICS Perform. Eval. Rev.*, 36(1):85–96, 2008.
- [SHC07] Björn Scheuermann, Wenjun Hu, and Jon Crowcroft. Near-optimal coordinated coding in wireless multihop networks. In *CoNEXT '07: Proceedings of the 2007 ACM CoNEXT conference*, pages 1–12, New York, NY, USA, 2007. ACM.