Network Coding: An Instant Primer

Christina Fragouli
EPFL - IC
christina.fragouli@epfl.ch

Jean-Yves Le Boudec
EPFL - IC
jean-yves.leboudec@epfl.ch

Jörg Widmer
DoCoMo Labs
widmer@docomolab-euro.com

ABSTRACT

Network coding is a new research area that may have interesting applications in practical networking systems. With network coding, intermediate nodes may send out packets that are linear combinations of previously received information. There are two main benefits of this approach: potential throughput improvements and a high degree of robustness. Robustness translates into loss resilience and facilitates the design of simple distributed algorithms that perform well, even if decisions are based only on partial information. This paper is an instant primer on network coding: we explain what network coding does and how it does it. We also discuss the implications of theoretical results on network coding for realistic settings and show how network coding can be used in practice.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous

General Terms

Network Coding

Keywords

Network Coding

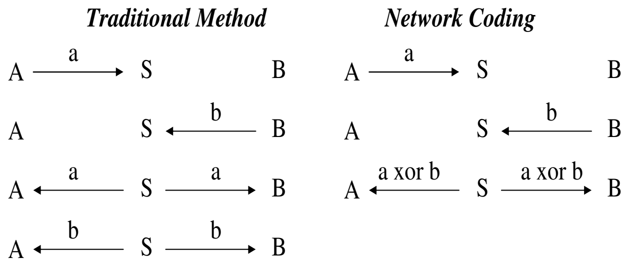

Figure 1: A simple network coding example. Nodes A and B want to exchange packets via an intermediate node S (wireless base station). A [resp. B] sends a packet a [resp. b] to B, which then broadcasts a `xor` b instead of a and b in sequence. Both A and B can recover the packet of interest, while the number of transmissions is reduced.

1. INTRODUCTION

Communication networks today share the same fundamental principle of operation. Whether it is packets over the Internet, or signals in a phone network, information is transported in the same way as cars share a highway or fluids share pipes. That is, independent data streams may share network resources, but the information itself is separate. Routing, data storage, error control, and generally all network functions are based on this assumption.

Network coding [2] is a recent field in information theory that breaks with this assumption. Instead of simply forwarding data, nodes may recombine several input packets into one or several output packets. A simple example in a wireless context is a three node topology, as shown in Figure 1. Linear network coding, in general, is similar to this example, with the difference that the `xor` operation is replaced by a linear combination of the data, interpreted as numbers over some finite field. This allows for a much larger degree of flexibility in the way packets can be combined. In addition to the throughput benefits evidenced in this example, network coding is also very well suited for environments where only partial or uncertain information is available for decision making. Similar to erasure coding, successful reception of information does not depend on receiving specific packet content but rather on receiving a sufficient number of independent packets.

Linear combining requires enhanced computational capabilities at the nodes of the network. However, according to Moore's law, processing is becoming less and less expensive. The bottleneck has shifted to network bandwidth to support the ever-growing demand in applications and QoS guarantees over large unreliable networks. Network coding utilizes cheap computational power to increase network efficacy.

The goal of this paper is to make the basic concepts of network coding available to the networking community.[1] Section 2 explains what it is and how it can be implemented. Section 3 discusses existing results on the performance benefits of network coding. Section 4 reviews proposals to use network coding in information and networking systems.

2. WHAT IS NETWORK CODING ?

2.1 Linear Network Coding

Consider a system that acts as information relay, such as a router, a node in an ad-hoc network, or a node in a peer to peer distribution network. Traditionally, when forwarding an information packet destined to some other node, it simply *repeats* it. With network coding, we allow the node

[1] For an exhaustive list of literature on network coding see `www.networkcoding.info`

to *combine* a number of packets it has received or created into one or several outgoing packets.

Assume that each packet consists of L bits. When the packets to be combined do not have the same size, the shorter ones are padded with trailing 0s. We can interpret s consecutive bits of a packet as a symbol over the field $\mathbb{F}_{2^s}$, with each packet consisting of a vector of L/s symbols. With *linear* network coding, outgoing packets are linear combinations of the original packets, where addition and multiplication are performed over the field $\mathbb{F}_{2^s}$ (see Section 2.5). The reason for choosing a linear framework is that the algorithms for coding and decoding are well understood.

Linear combination is not concatenation: if we linearly combine packets of length L, the resulting encoded packet also has size L. An encoded packet generally carries information about several original packets, but in contrast to concatenation, just by itself it does not allow to recover any part of the original packets. One can think of linear network coding as a form of information *spreading*.

2.2 Encoding

Assume that a number of original packets $M^1, ..., M^n$ are generated by one or several sources. In linear network coding, each packet in the network is associated with a sequence of coefficients $g_1, ..., g_n$ in $\mathbb{F}_{2^s}$ and is equal to $X = \sum_{i=1}^{n} g_i M^i$. The summation has to occur for every symbol position, i.e., $X_k = \sum_{i=1}^{n} g_i M^i_k$, where M^i_k and X_k is the kth symbol of M^i and X respectively. In the example of Figure 1, the field is $\mathbb{F}_2 = \{0, 1\}$, a symbol is a bit, and the linear combination sent by S after receiving $M^1 = a$ and $M^2 = b$ is $M^1 + M^2$ (the $+$ sign here is addition in $\mathbb{F}_2$, i.e., bitwise `xor`).

For simplicity, we assume that a packet contains both the coefficients $g = (g_1, ..., g_n)$, called *encoding vector*, and the encoded data $X = \sum_{i=1}^{n} g_i M^i$, called *information vector* [6]. The encoding vector is used by recipients to decode the data, as explained later. For example, the encoding vector $\mathbf{e}_i = (0, ..., 0, 1, 0, ...0)$, where the 1 is at the ith position, means that the information vector is equal to M^i (i.e., is not encoded).

Encoding can be performed recursively, namely, to already encoded packets. Consider a node that has received and stored a set $(g^1, X^1), ..., (g^m, X^m)$ of encoded packets, where g^j [resp. X^j] is the encoding [resp. information] vector of the jth packet. This node may generate a new encoded packet (g', X') by picking a set of coefficients $h_1, ..., h_m$ and computing the linear combination $X' = \sum_{j=1}^{m} h_j X^j$. The corresponding encoding vector g' is not simply equal to h, since the coefficients are with respect to the original packets $M^1, ..., M^n$; in contrast, straightforward algebra shows that it is given by $g'_i = \sum_{j=1}^{m} h_j g^j_i$. This operation may be repeated at several nodes in the network.

2.3 Decoding

Assume a node has received the set $(g^1, X^1), ..., (g^m, X^m)$. In order to retrieve the original packets, it needs to solve the system $\{X^j = \sum_{i=1}^{n} g^j_i M^i\}$ (where the unknowns are M^i). This is a linear system with m equations and n unknowns. We need $m \geq n$ to have a chance of recovering all data, i.e., the number of received packets needs to be at least as large as the number of original packets. Conversely, the condition $m \geq n$ is not sufficient, as some of the combinations might be linearly dependent. However, and this is a major appeal of network coding, this problem is easy, as we discuss next.

2.4 How to Select the Linear Combinations

The problem of network code design is to select what linear combinations each node of the network performs. A simple algorithm is to have each node in the network select uniformly at random the coefficients over the field $\mathbb{F}_{2^s}$, in a completely independent and decentralized manner [14].

With random network coding there is a certain probability of selecting linearly dependent combinations [14]. This probability is related to the field size 2^s. Simulation results indicate that even for small field sizes (for example, $s = 8$) the probability becomes negligible [29].

Alternatively, we can use deterministic algorithms to design network codes. The polynomial-time algorithm for multicasting in [24] sequentially examines each node of the network, and decides what linear combinations each node performs. Since each node uses fixed linear coefficients, the packets only need to carry the information vector. There also exist deterministic decentralized algorithms that apply to restricted families of network configurations [10].

2.5 Practical Considerations

Decoding: Decoding requires solving a set of linear equations. In practice, this can be done as follows. A node stores the encoded vectors it receives as well as its own original packets, row by row, in a so-called *decoding matrix*. Initially, it contains only the non-encoded packets issued by this node with the corresponding encoding vectors (if any, else it is empty). When an encoded packet is received, it is inserted as last row into the decoding matrix. The matrix of coefficients is transformed to triangular matrix[2] using Gaussian elimination. A received packet is called innovative if it increases the rank of the matrix. If a packet is non-innovative, it is reduced to a row of 0s by Gaussian elimination and is ignored. As soon as the matrix contains a row of the form $(\mathbf{e}_i, X)$, the node knows that the original packet M_i is equal to X. This occurs at the latest when n linearly independent encoded vectors are received. Note that decoding does not need to be performed at all nodes of the network, but only at the receivers.

Generations: For all practical purposes, the size of the matrices with which network coding operates has to be limited. This is straightforward to achieve for deterministic network codes, but more difficult with random network coding. For the latter, packets are usually grouped into so-called generations, and only packets of the same generation can be combined [6]. Size and composition of generations may have significant impact on the performance of network coding [11]. Similar considerations hold for the size of the finite field. Both parameters allow to trade off performance for lower memory requirements and reduced computational complexity.

In some cases, it is possible to enforce sparse decoding matrices [22] to reduce memory requirements. This is particularly important in case the matrices do not fit into main memory and slower I/O operations have to be used to fetch the data.

Delay: The fact that packets need to be decoded has a minor impact on delay. It is usually not necessary to

[2]The matrix can be transformed to a reduced row echelon form, where within each row, the leading term is 1, and the position of the leading terms moves to the right.

receive all encoded packets before some of the packets can be decoded (i.e., whenever Gaussian elimination leads to a row in the form $(\mathbf{e}_i, M_i)$). Together with a reduction in the number of required transmissions, the overall end-to-end delay with network coding is usually not larger than the normal end-to-end delay in realistic settings.

Finite field operations: Network coding requires operations in $\mathbb{F}_{2^s}$, i.e., operations on strings of s bits. Addition is the standard bitwise `xor`. For multiplication, one interprets a sequence $b_0, ..., b_{s-1}$ of s bits as the polynomial $b_0 + b_1 Z + ... + b_{s-1} Z^{s-1}$. Then one picks a polynomial of degree s that is *irreducible* over $\mathbb{F}_2$ (there are several of them, and each gives a different representation of $\mathbb{F}_{2^s}$; for example, Rijndael's representation of $\mathbb{F}_{2^8}$ uses $1 + Z + Z^3 + Z^4 + Z^8$). Multiplication is obtained by computing the usual product of two polynomials (which gives a polynomial of degree possibly larger than $s - 1$) modulo the chosen irreducible polynomial. Division is computed by the Euclidian algorithm. Both multiplication and division can be implemented efficiently with s shifts and additions [26].

If s is small (e.g., $s = 8$), a faster alternative is to use discrete logarithms. In a finite field there exists at least one special element α, called *generator* (for example, $\alpha =$ `0x03` $= 1 + Z$ is a generator in Rijndael's representation of $\mathbb{F}_{2^8}$). Any non-zero $x \in \mathbb{F}_{2^s}$ can be written in a unique way $x = \alpha^{l(x)}$; $l(x)$ is called the logarithm [20]. Since $l(xy) = l(x) + l(y)$, multiplication and division can be implemented by looking up the two tables that map x to $l(x)$ and vice-versa. For $s = 8$ these are two tables of size 255 bytes.

3. WHAT ARE THE BENEFITS OF NETWORK CODING ?

Theoretically proven results about network coding mainly concern performance improvements in static settings. We review these first and then discuss random distributed settings.

3.1 Throughput Gain in Static Environment

A primary result that sparked the interest in network coding is that it can increase the capacity of a network for multicast flows. More specifically, consider a network that can be represented as a directed graph (typically, this is a wired network). The vertices of the graph correspond to terminals, and the edges of the graph corresponds to channels. Assume that we have M sources, each sending information at some given rate, and N receivers. All receivers are interested in receiving all sources.

THEOREM 1. *Assume that the source rates are such that, without network coding, the network can support each receiver in isolation (i.e. each receiver can decode all sources when it is the only receiver in the network). With an appropriate choice of linear coding coefficients, the network can support all receivers* simultaneously *[2, 18].*

In other words, when the N receivers share the network resources, each of them can receive the maximum rate it could hope to receive, even if it were using all the network resources by itself. Thus, network coding can help to better share the available network resources (Figure 2).

Network coding may offer throughput benefits not only for multicast flows, but also for other traffic patterns, such as unicast. Consider again Figure 2 but assume now that source S_1 transmits to destination R_2 and S_2 to R_1. With network coding we can send at rate 1 to each receiver, while without, we can only send at rate 1/2 to each receiver.

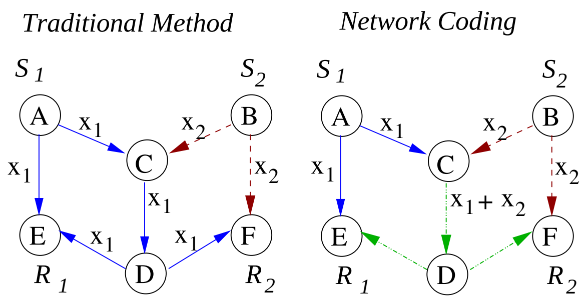

Figure 2: (Butterfly Network) S_1 and S_2 multicast to both R_1 and R_2. All links have capacity 1. With network coding (by `xor`ing the data on link CD), the achievable rates are 2 for each source, the same as if every destination were using the network for its sole use. Without network coding, the achievable rates are less (for example if both rates are equal, the maximum rate is 1.5).

There exist directed graphs where the throughput gains of network coding for multicasting can be very significant [24, 5]. However, in undirected graphs (e.g., a wired network where all links are half-duplex) the throughput gain is at most a factor of two [19]. Experimental results in [29] over the network graphs of six Internet service providers show a small throughput gain in this case.

An interesting point is that network coding allows to achieve the optimal throughput when multicasting using polynomial time algorithms. In contrast, achieving the optimal throughput with routing is NP-complete: this is the problem of packing Steiner trees in CS theory. Thus, even when the expected throughput benefits of network coding are not large, we expect to be able to achieve them using "simpler" algorithms. We expand on this point in the following.

3.2 Robustness and Adaptability

The most compelling benefits of network coding might be in terms of robustness and adaptability. Intuitively, we can think that network coding, similarly to traditional coding, takes information packets and produces encoded packets, where each encoded packet is "equally important". Provided we receive a sufficient number of encoded packets, no matter which, we are able to decode. The new twist that network coding brings, is that the linear combining is performed opportunistically over the network, not only at the source node, and thus it is well suited for the (typical) cases where nodes only have incomplete information about the global network state.

Consider again Figure 1 and assume that A and B may go into sleep mode (or may move out of range) at random and without notifying the base station S. If the base station S broadcasts a (or b), the transmission might be completely wasted, since the intended destination might not be able to receive. However, if the base station broadcasts a `xor` b, or more generally, random linear combinations of the information packets, the transmission will bring new information to all active nodes. This argument is the basis for the significant performance improvements reported in Section 4.2.

Coupon Collector Problem

This is a generic problem for which a theoretical result is available, and which explains the performance gains reported in Section 4.1. Consider a network with n nodes and $O(n)$ messages. All nodes would like to receive all messages. A centralized gossip-based protocol allows to disseminate the messages in $\Theta(n)$ rounds, where in each round $\Theta(n)$ pairs of nodes exchange one message. To perform the same task in a decentralized manner, we need $\Theta(n \log n)$ rounds. This is because, for a receiver a specific message may become "rare", i.e., hard to collect. Allowing codes to use network coding and propagate random linear combinations of their messages instead, makes all packets "equal": it is sufficient for a node to collect *any* n messages. This approach allows to disseminate all messages in $\Theta(n)$ rounds [7]. Thus, network coding allows to achieve the optimal performance using a simple decentralized algorithm.

Packet Erasure Networks

Network coding can offer benefits for delay sensitive and high data rate applications in networks where packets get dropped. Two main approaches are employed today: Automatic repeat request (ARQ) schemes that achieve the optimal rate at the cost of delay; and packet-level forward error correcting (FEC) schemes that achieve the optimal delay at the cost of rate, such as Fountain codes [25]. Fountain codes are end-to-end: packets are encoded at the source and decoded at the destination, while intermediate nodes are only allowed to replicate and forward packets. Applying ideas from network coding in this context, i.e., allowing intermediate nodes to also form linear combinations, allows to achieve both the optimal rate and the optimal delay at the same time.

For example, consider a source A that would like to transmit information to a destination C. On the path from A to C there exists a router B that can perform network coding operations. Assume that node A sends encoded packets, that are dropped on paths AB and BC with probability ϵ_{AB} and ϵ_{BC} respectively. Using an end-to-end FEC scheme, i.e., having the destination C decode the packets it receives, restricts the rate to $R_1 \leq (1-\epsilon_{AB})(1-\epsilon_{BC})$. If we allow the router B to perfectly decode and re-encode, we will achieve the optimal min-cut rate $R_2 \leq \min\{(1-\epsilon_{AB}), (1-\epsilon_{BC})\}$, but at the cost of additional delay: we have to wait at node B to receive sufficient encoded packets to be able to decode and re-encode the information. Using an ARQ scheme will again allow to achieve rate R_2, but again at the cost of increased delay.

Alternatively, node B can at each time instance form and send random linear combinations of the encoded packets it has received up to that time, without waiting for all encoded packets. We can then achieve the optimal rate R_1 without an additional delay. Moreover, it is sufficient to perform `xor` operations. This scheme in its full generality can be applied over an arbitrary network topology, and with diverse traffic load (multicasting, unicasting, broadcasting, etc.) [21, 22].

4. WHERE CAN NETWORK CODING BE USED ?

In the following, we list a number of applications of network coding and discuss how the benefits mentioned in Section 3 improve performance in concrete settings.

4.1 P2P File Distribution

Probably the most widely known application using network coding is Avalanche [12, 1]. Generally, in a peer-to-peer content distribution network, a server splits a large file into a number of blocks. Peer nodes try to retrieve the original file by downloading blocks from the server but also distributing downloaded blocks among them. To this end, peers maintain connections to a limited number of neighboring peers (randomly selected among the set of peers) with which they exchange blocks. In Avalanche, the blocks sent out by the server are random linear combinations of all original blocks. Similarly, peers send out random linear combinations of all the blocks available to them. A node can either determine how many innovative blocks it can transmit to a neighbor by comparing its own and the neighbor's matrix of decoding coefficients, or it can simply transmit coded blocks until the neighbor receives the first non-innovative block. The node then stops transmitting to this neighbor until it receives further innovative blocks from other nodes. Coding coefficients are transmitted together with the blocks, but since blocks usually have a size of hundreds of kilobytes, this overhead is negligible.

Network coding helps in several respects. 1) It minimizes download times; in a such large scale distributed peer-to-peer system, optimal packet scheduling is very complex, particularly if the participating hosts only have very limited information about the underlying network topology. With network coding, the performance of the system depends much less on the specific overlay topology and schedule. Consequently, very simple mechanisms that construct a random overlay can be used. The authors show that network coding outperforms traditional forwarding or FEC-based peer-to-peer systems by a significant margin. 2) Due to the diversity of the coded blocks, a network coding based solution is much more robust in case the server leaves early (before all peers have finished their download) or in the face of high churn rates (where nodes only join for a short period of time or leave immediately after finishing their download). 3) In contrast to forwarding based protocols, their network coding protocol suffers only a small performance penalty when incentive mechanisms to cooperate are implemented (e.g., tit-for-tat to prevent free-riding).

4.2 Wireless Networks

Bidirectional traffic in a wireless network: As shown in Figure 1, network coding can improve throughput when two wireless nodes communicate via a common base-station. This setting can be extended to the case of multi-hop routing in a wireless network (or any other network with physical layer broadcast) where the traffic between the two end nodes is bidirectional [30] and both nodes have a similar number of packets to exchange. Given a schedule that alternates between adjacent routers, after a few initial steps all intermediate routers have packets buffered for transmission in both directions of the path. Whenever a transmission opportunity arises, a router combines two packets, one for each direction, with a simple *xor* and broadcasts it to its neighbors. Both receiving routers already know one of the packets the broadcast is coded over, while the other packet is new. Thus, each broadcast allows two routers to receive a new packet, effectively doubling the capacity of the path.

In [30], the authors also discuss a distributed implementation that works when transmissions are not synchronized and the wireless channel is lossy and has random delay. Overhearing a packet of a neighbor that is coded over information previously forwarded to the neighbor serves as a passive acknowledgment. This allows to make better use of transmission opportunities at routers that only have new packets buffered for a single direction. In this case, one of these new packets is combined with an old packet for the opposite direction, for which no passive acknowledgment has been received.

Residential wireless mesh networks: Even a limited form of network coding which only uses *xor* to combine packets may significantly improve network performance in wireless mesh networks [17]. All transmissions are broadcast and are overheard by the neighbors. Packets are annotated with summary information about all other packets a node already heard. This way, information about which nodes hold which packets is distributed within the neighborhood. A node can xor multiple packets for different neighbors and send them in a single transmission, if each neighbor already has the remaining information to decode the packet immediately. In experiments with 802.11 hardware, the authors show that their mechanism almost doubles network throughput.

Many-to-many broadcast: Network-wide broadcast is used for a number of purposes in ad-hoc networks (e.g., route discovery) and can be implemented much more efficiently with network coding [27, 11]. Already a simple distributed algorithm for random network coding reduces the number of transmission by a factor of 2 or more, leading to significant energy savings. In such a setting, a larger transmit power directly translates into a reduction in the number of required transmissions, which allows for interesting energy tradeoffs. Energy expenditure is either evenly distributed among the nodes or covered by only a few nodes (maybe with longer battery life). There is a larger flexibility in the distribution the energy requirements compared to conventional algorithms.

Algorithms for challenging wireless networks which may be very sparse or highly mobile are investigated in [28]. Usually, algorithms for such networks employ some form of intelligent flooding to combat the adverse network conditions. With network coding, performance improvements are much larger than the ones reported above, indicating that network coding is particularly suitable when robust operation is of high importance. The authors present a simple mechanism for network wide broadcast that has a much lower overhead than flooding and also discuss practical implementation issues for network coding in wireless networks.

Similar performance benefits can be expected when nodes spend most of their time in sleep mode to save energy or use randomized beam-forming with directional antennas. All of these cases have in common that a specific pair of nodes is not likely to be able to communicate at a given time.

4.3 Ad-hoc Sensor Networks

Untuned radios in sensor networks: A novel and interesting application for network coding is, to use it to cope with untuned transceivers [23]. For sensor nodes which should be as simple and cheap to manufacture as possible, the quartz oscillator to tune the radio to a specific frequency makes up a significant fraction of hardware cost and design complexity. The authors propose to replace the analog oscillator by a much simpler on-chip resonator. As a consequence, the transceivers' radio frequencies depend to some degree on variations in the manufacturing process and two such devices are not very likely to be able to communicate directly. However, in dense sensor networks with thousands of tiny devices and multiple radios per device, a multi-hop path between information source and data sink will most probably exist. With random network coding, it is possible to use these paths without having to "explicitly find them", and without the excessive overhead of flooding.

In their paper, the authors do not propose a specific protocol but rather give a theoretical analysis, that, given their assumptions, untuned radios with network coding perform only a factor of $1/e$ worse than perfectly tuned radios without network coding. They conclude, that operating networks in such a randomized fashion may be preferable to the more traditional way of controlled operation, since it implies a much simpler radio architecture.

Data gathering in sensor networks: An interesting data gathering algorithm for sensor networks is presented in [8]. Nodes have storage space for one single packet. Overheard packets from neighboring nodes are multiplied with a random coefficient and added to the existing information. Sensor nodes are thus not able to decode, but in a sensor network the goal is only to make the data available to a (more capable) sink node. Sensor information is pro-actively distributed to a small number of other nodes, so that a sink node can reconstruct n data packets with a high probability by contacting only n sensor nodes anywhere in the network.

4.4 Network Tomography

In [9], network coding is used to infer the loss rates of links in an overlay network. For conventional active probing, packets are usually multicast to several receivers. The receivers experience the same loss event which provides information about losses in the underlying multicast tree. After a sufficiently large number of probe packets, shared links and their loss rates can be identified with reasonable accuracy. In such a setting, network coding provides additional flexibility since packets are not only duplicated at branching points of the multicast tree, but may also be merged. If multiple senders unicast packets to a single receiver, and these packets are combined within the network, it allows to infer the topology in much the same way as multicasting from one sender to multiple receivers. Furthermore, if the network code (i.e., the specific way in which packets are combined at the nodes) is known in advance, the coding coefficients contained in the probe packets provide additional information about the original packets that were combined (and consequently which packets were lost in which part of the tree). By exploiting these features, it is possible to significantly reduce the number of active probes and the link stress generated by these probes.

Network coding has also been proposed for passive network monitoring. As with active monitoring, a random (but fixed) network code allows receivers to determine which coefficients are expected under normal operation. When the obtained coefficients differ, the receiver can draw from that conclusions about the failure patterns [15].

4.5 Network security

We distinguish the following forms of network security: security against an eavesdropper that attempts to recover part of the data, security against a malicious attacker that attempts to misinform the receivers by modifying packets, and security against jamming attacks.

Network coding seems to facilitate protection from an eavesdropper, since information is more spread out and thus more difficult to "overhear". In [4], the authors investigate the problem of designing secure network codes for wiretap networks, where certain known links are tapped by attackers. The source combines the original data with random information and designs a network code in such a way that only the receivers are able to decode the original packets. Furthermore, the mutual information between the packets obtained by the eavesdroppers and the original packets is zero (security in the information theoretic sense). The authors in [3] investigate a weaker form of security, based on the fact that nodes can only decode packets if they have received a sufficient number of linearly independent information vectors, which an eavesdropper might not be able to do.

Network coding also simplifies the protection against modified packets in a network [16]. In a network with no additional protection, an intermediate attacker may make arbitrary modifications to a packet to achieve a certain reaction at the attacked destination. However, in the case of network coding, an attacker cannot control the outcome of the decoding process at the destination, without knowing all other coded packets the destination will receive. Given that packets are routed along many different paths, this makes controlled man-in-the-middle-attacks more difficult.

On the other hand, network coding seems to impede security against jamming attacks, since corrupting a few packets might affect a much larger set of data. A method to counteract such attacks has been recently proposed in [13].

5. CONCLUSION

Network coding may have an impact on the design of new networking and information dissemination protocols. By allowing to better spread the information content in the network environment, it can simplify distributed algorithms. Reliable multicast is an example where the existing solutions need to be re-thought; we have shown in our review that emerging areas such as ad-hoc networks, overlay infrastructures, and sensor networks are starting to benefit from network coding. What will be next?

6. REFERENCES

[1] Avalanche: File swarming with network coding. http://research.microsoft.com/ pablo/avalanche.aspx.

[2] R. Ahlswede, N. Cai, S. R. Li, and R. W. Yeung. Network information flow. *IEEE Transactions on Information Theory*, July 2000.

[3] K. Bhattad and K. R. Narayanan. Weakly secure network coding. In *NetCod*, Apr. 2005.

[4] N. Cai and R. W. Yeung. Secure network coding. In *ISIT*, 2002.

[5] C. Chekuri, C. Fragouli, and E. Soljanin. On average throughput and alphabet size in network coding. Accepted to *IEEE Trans. Inform. Theory.*

[6] P. A. Chou, Y. Wu, and K. Jain. Practical network coding. In *Allerton*, Oct. 2003.

[7] S. Deb and M. Medard. Algebraic gossip: A network coding approach to optimal multiple rumor mongering. In *Allerton*, Oct. 2004.

[8] A. G. Dimakis, V. Prabhakaran, and K. Ramchandran. Ubiquitous access to distributed data in large-scale sensor networks through decentralized erasure codes. In *IPSN*, Apr. 2005.

[9] C. Fragouli and A. Markopoulou. A network coding approach to overlay network monitoring. In *Allerton*, Sept. 2005.

[10] C. Fragouli and E. Soljanin. Decentralized network coding. *Information Theory Workshop*, Oct. 2004.

[11] C. Fragouli, J. Widmer, and J.-Y. LeBoudec. A network coding approach to energy efficient broadcasting: from theory to practice. Technical Report LCA-REPORT-2005-009, accepted at Infocom 2006, EPFL, July 2005.

[12] C. Gkantsidis and P. Rodriguez. Network coding for large scale content distribution. In *Infocom*, Miami, FL, Mar. 2005.

[13] C. Gkantsidis and P. R. Rodriguez. Cooperative security for network coding file distribution. Technical Report accepted at Infocom, 2006.

[14] T. Ho, R. Koetter, M. Medard, D. R. Karger, and M. Effros. The benefits of coding over routing in a randomized setting. In *ISIT*, July 2003.

[15] T. Ho, B. Leong, Y. Chang, Y. Wen, and R. Koetter. Network monitoring in multicast networks using network coding. In *ISIT*, 2005.

[16] T. Ho, B. Leong, R. Koetter, M. Mdard, M. Effros, and D. R. Karger. Byzantine modification detection in multicast networks using randomized network coding. In *ISIT*, 2004.

[17] S. Katti, D. Katabi, W. Hu, H. Rahul, and M. Medard. The importance of being opportunistic: Practical network coding for wireless environments. In *Allerton*, 2005.

[18] S.-Y. R. Li, R. W. Yeung, and N. Cai. Linear network coding. *IEEE Trans. Inform. Theory*, 49:371–381, Feb. 2003.

[19] Z. Li and B. Li. Network coding in undirected networks. *CISS*, 2004.

[20] C. H. Lim and P. J. Lee. More flexible exponentiation with precomputation. In *Advances in Cryptology*, Aug. 1994.

[21] D. S. Lun, M. Medard, and M. Effros. On coding for reliable communication over packet networks. *Allerton*, 2004.

[22] P. Pakzad, C. Fragouli, and A. Shokrollahi. On low complexity coding for line networks. *ISIT*, Australia, Sept. 2005.

[23] D. Petrovic, K. Ramchandran, and J. Rabaey. Overcoming untuned radios in wireless networks with network coding. In *NetCod*, Italy, Apr. 2005.

[24] P. Sanders, S. Egner, and L. Tolhuizen. Polynomial time algorithms for network information flow. *Proc. 15th ACM Symposium on Parallel Algorithms and Architectures*, 2003.

[25] A. Shokrollahi. Raptor codes. *Submitted to IEEE Trans. Information Theory*, 2004.

[26] N. R. Wagner. *The Laws of Cryptography with Java Code*. Available online at Neal Wagner's home page.

[27] J. Widmer, C. Fragouli, and J. Y. LeBoudec. Energy efficient broadcasting in wireless ad hoc networks. In *NetCod*, Apr. 2005.

[28] J. Widmer and J.-Y. LeBoudec. Network coding for efficient communication in extreme networks. In *WDTN*, Aug. 2005.

[29] Y. Wu, P. A. Chou, and K. Jain. A comparison of network coding and tree packing. *ISIT*, 2004.

[30] Y. Wu, P. A. Chou, and S.-Y. Kung. Information exchange in wireless networks with network coding and physical-layer broadcast. Technical Report MSR-TR-2004-78, Microsoft Research, Aug. 2004.