Chapter 5

NETWORK CODING IN WIRELESS NETWORKS

A survey of techniques for efficient operation of coded wireless packet networks

Desmond S. Lun
Massachusetts Institute of Technology
dslun@mit.edu

Tracey Ho
California Institute of Technology
tho@caltech.edu

Niranjan Ratnakar
University of Illinois at Urbana-Champaign
ratnakar@uiuc.edu

Muriel Médard
Massachusetts Institute of Technology
medard@mit.edu

Ralf Koetter
University of Illinois at Urbana-Champaign
koetter@uiuc.edu

Abstract: The advent of network coding promises to change many aspects of networking. Network coding moves away from the conventional approach to networking, where packets are treated as inviolable, atomic units to be transported through

Frank H. P. Fitzek and Marcos D. Katz (eds.), Cooperation in Wireless Networks: Principles and Applications, 127–161.

the network, and instead allows packets to be mixed and combined, so packets outgoing from a node are allowed to be arbitrary, causal functions of packets received at that node. This approach has shown much promise in multi-hop wireless networks, affording gains including more efficient use of resources and the ability for decentralized operation. In this chapter, we overview some of the issues and technical approaches associated with network coding in the wireless domain. We hope, thereby, to provide the reader with a firm theoretical basis from which practical implementations and theoretical extensions can be developed.

Keywords: ad hoc networks, forward error correction, multicast communication, multi-hop wireless networks, network coding.

1. Introduction

The notion of coding at the packet level—commonly called network coding—has attracted much recent interest. In this survey, we provide an overview of some of the issues and technical approaches associated with network coding in the wireless domain. Considering the wireless applications is not merely an extension or simple modification of the wireline case. One could argue that most wireless routing schemes have indeed sought to replicate, in the wireless domain, topologies that resemble wireline networks. This approach has tended to neglect characteristics of wireless transmissions, such as inherent broadcast, interference, fading and mobility, in order to re-use the vast algorithmic and protocol knowledge established for routing in wireline networks. However, limited acknowledgment, in protocol design, of wireless transmission peculiarities has generally led to inefficient use of limited resources, such as spectrum and battery life, as well as to considerable complications in deployment. It is therefore our goal for this paper to afford some insights, in these early stages of the development of network coding, for the application of network coding to wireless environments, in such a way that identifies techniques that are common to both wireless and wireline networks, but embraces the peculiarities of wireless media.

To illustrate the differences and similarities between network coding for wireless and wireline applications, we commence with the simple canonical example from the initial work on the topic of network coding by [Ahlswede et al., 2000]. Figure 5.1 shows this example. Each link is assumed to have unit capacity, be error-free, and provide a unidirectional link which does not interfere with other links emerging from or incident upon a common node. The simple coding shown in the figure affords a multicast connection conveying two bits, b_1 and b_2, from the sender at node 1 to the receivers at nodes 6 and 7. We consider bits rather than packets because, from bits, it is clear how packets should be coded.

Figure 5.2 considers the same topology but seeks to represent the behavior of wireless links sharing bandwidth and enabled by a single omnidirectional

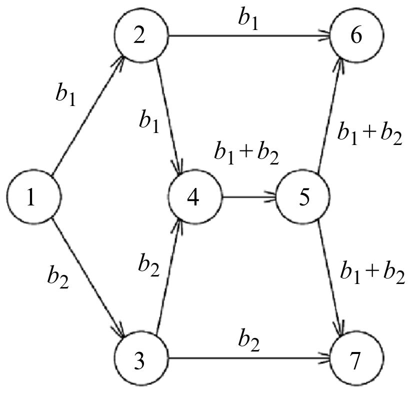

Figure 5.1. Canonical example of network coding in wireline networks given by [Ahlswede et al., 2000]. We denote by $b_1 + b_2$ the binary sum of bits b_1 and b_2.

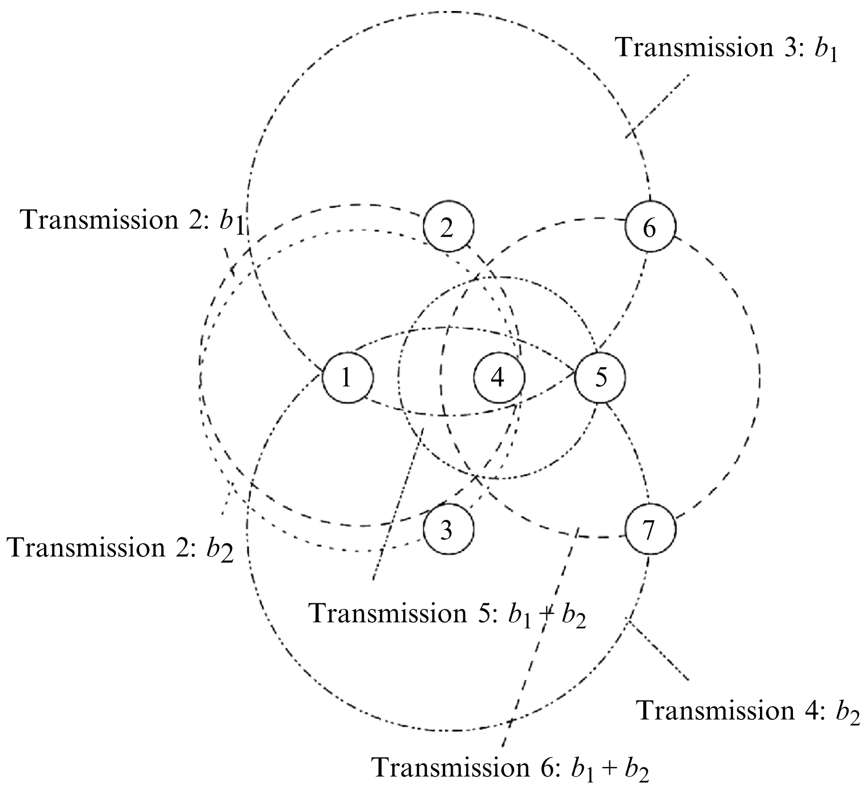

Figure 5.2. Figure 5.1 redrawn for wireless links.

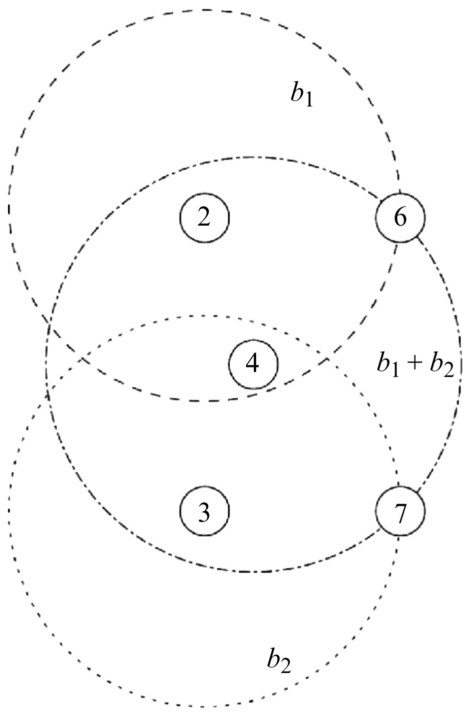

Figure 5.3. A simple five-node wireless network employing network coding.

transmit antenna and a single omnidirectional receive antenna at each node. Rather than using arcs to represent links, we draw circles whose radii indicate the range of a transmission. Moreover, the constraints of shared spectrum preclude a node transmitting and receiving simultaneously, as well as simultaneous reception of more than one transmission at a node. The sequential transmissions represented in Figure 5.2 indicate a possible instantiation of a schedule of transmissions and their associated ranges, which satisfy the simple wireless transmission modalities we detailed above. Note that network coding is still useful at transmission 5, in which node 4 communicates directly to node 5. In the absence of network coding, nodes 4 and 5 would need to perform two transmissions, one for b_1 and one for b_2. Note, however, that other instances of radii and schedule besides that represented in Figure 5.2 are possible. For example, at transmission 5, node 4 could have selected a wider range of transmission, including the receivers, nodes 6 and 7. In such a scheme, network coding would still be advantageous. Some different choices of transmission ranges would obviate the usefulness of network coding altogether, for example, if the source node, node 1, transmitted b_1 and b_2 successively over wide enough regions to include both receivers.

Network size	Approach	Average multicast energy			
		2 sinks	4 sinks	8 sinks	16 sinks
20 nodes	MIP algorithm	30.6	33.8	41.6	47.4
	Network coding	15.5	23.3	29.9	38.1
30 nodes	MIP algorithm	26.8	31.9	37.7	43.3
	Network coding	15.4	21.7	28.3	37.8
40 nodes	MIP algorithm	24.4	29.3	35.1	42.3
	Network coding	14.5	20.6	25.6	30.5
50 nodes	MIP algorithm	22.6	27.3	32.8	37.3
	Network coding	12.8	17.7	25.3	30.3

Table 5.1. Average energy of random multicast connections of unit rate for various approaches in random wireless networks of varying size. Nodes were placed randomly within a 10×10 square with a radius of connectivity of 3. The energy required to transmit at unit rate to a distance d was taken to be d^2. Source and sink nodes were selected according to an uniform distribution over all possible selections.

The dependence of connectivity on choice of transmission radii renders operation highly dependent on physical repartition of nodes. The simple example of Figure 5.3 illustrates this dependence. The topology is similar to that of Figures 5.1 and 5.2, except that nodes 1 and 5 have been removed and, so, b_1 originates at node 2 and b_2 at node 3. The representation of the transmission radii and schedules follow naturally from Figure 5.2. In this case, node 6 can be seen as "overhearing" the transmission of b_1 to the center of the network and, similarly, node 7 receives b_2 as part of the requisite transmission of b_2 to a distance sufficient to establish connectivity. Network coding in this case is a natural relaying with combining. The coding establishes a multicast of two sources originating at nodes 2 and 3 to two receivers at nodes 6 and 7. It can also be used to unicast a single source (b_1) from node 2 to node 7 and a single source (b_2) from node 3 to node 6. Thus, the coding establishes a natural multicast scenario from two unicast scenarios. The cause for this natural multicast is that nodes, because of the intrinsic properties of wireless transmission, overhear communications which may not be of direct interest to them but which will allow them to infer the transmission that are.

The examples that we have discussed clearly show that there is some potential for improving the performance of wireless networks by using network coding. We therefore wish to generalize the technique and make it broadly applicable. The remainder of this chapter is largely devoted to discussing such a generalization. We give a general prescription for the operation of coded wireless networks, *i.e.* we give a method for determining which node should send what when. The techniques that we discuss are very recent and have not yet been throughly tested, but preliminary results are promising. For example, in the problem of minimum-energy multicast, these techniques have been found to produce reductions in average energy consumption ranging from 13%

to 49% when compared to the Multicast Incremental Power (MIP) algorithm by [Wieselthier et al., 2002]—one of the few good approaches to minimum-energy multicast in non-coded, routed wireless networks (see Table 5.1). The results in Table 5.1 are for lossless networks, where links are, owing presumably to some underlying retransmission scheme, effectively lossless. We believe that performance gains for lossy networks, where network coding can be used as a means of ensuring reliable transmission and retransmission is obviated, may be even more significant. Before proceeding any further with the prescription for operation, however, we first present a model for wireless packet networks.

2. Model

We model the network with a directed hypergraph $\mathcal{H} = (\mathcal{N}, \mathcal{A})$, where $\mathcal{N}$ is the set of nodes and $\mathcal{A}$ is the set of hyperarcs. A hypergraph is a generalization of a graph, where, rather than arcs, we have hyperarcs. A hyperarc is a pair (i, J), where i, the start node, is an element of $\mathcal{N}$ and J, the set of end nodes, is a non-empty subset of $\mathcal{N}$.

Each hyperarc (i, J) represents a wireless broadcast link from node i to nodes in the non-empty set J. This link may be lossless or lossy, *i.e.* it may or may not be subject to packet erasures. Let A_{iJK} be the counting process describing the arrival of packets that are injected on hyperarc (i, J) and received by exactly the set of nodes $K \subset J$, *i.e.* for $\tau \geq 0$, $A_{iJK}(\tau)$ is the total number of packets that are injected on hyperarc (i, J) and received by all nodes in K (and no nodes in $\mathcal{N} \setminus K$) between time 0 and time τ. For example, suppose that three packets are injected on hyperarc $(1, \{2, 3\})$ between time 0 and time 1 and that, of these three packets, one is received by node 2 only, one is lost entirely, and one is received by both nodes 2 and 3; then we have $A_{1(23)\emptyset}(1) = 1$, $A_{1(23)2}(1) = 1$, $A_{1(23)3}(1) = 0$, and $A_{1(23)(23)}(1) = 1$.

We assume that A_{iJK} has an average rate z_{iJK}; more precisely, we assume that

$$\lim_{\tau \to \infty} \frac{A_{iJK}(\tau)}{\tau} = z_{iJK} \tag{5.1}$$

almost surely. When links are lossless, we have $z_{iJK} = 0$ for all $K \subsetneq J$.

Let $z_{iJ} := \sum_{K \subset J} z_{iJK}$ be the average rate at which packets are injected into hyperarc (i, J). The rate vector z, consisting of z_{iJ}, $(i, J) \in \mathcal{A}$, is called the coding subgraph and can be varied within a constraint set Z dictated to us by lower layers (for examples of such constraint sets, see [Cruz and Santhanam, 2003; Jain et al., 2003; Johansson et al., 2003; Xiao et al., 2004; Kodialam and Nandagopal, 2005; Wu et al., 2005]). We reasonably assume that Z is a convex subset of the positive orthant containing the origin. We associate with the network a cost function f that maps feasible coding subgraphs to real numbers and that we seek to minimize. For wireless networks, it is common for the cost function to reflect energy consumption, but it could also represent, for example, average latency, monetary cost, or a combination of these considerations.

We focus on network coding within a multicast session consisting of one or more sources multicasting to the same set of receiver nodes, or sinks. Coding within individual multicast sessions is a reasonable practical approach. There are cases where capacity can be increased by coding across sessions, such as the one mentioned in Section 1, but much less is known at present about such codes, which are discussed briefly in Section 5. We consider multicast sessions as they are the most general type of session, including unicast and broadcast as special cases. We denote the source processes by $X_1, X_2, \ldots, X_r$. Source X_k is generated at node $a(k)$, where $a : \{1, \ldots, r\} \to \mathcal{N}$ is an arbitrary mapping. The source processes are multicast to a set $T \subset \mathcal{N}$ of sinks. For simplicity, we assume subsequently that $a(k) \notin T$ for all $k \in \{1, \ldots, r\}$. Processes $X_1, \ldots, X_r$, mapping a, and set T specify a set of *multicast connection requirements*.

3. Distributed random network coding

In this section, we discuss how distributed random network coding can achieve any feasible set of multicast connection requirements in a given coding subgraph z. As a consequence, in setting up a single multicast session in a network, there is no loss of optimality in separating the problems of subgraph selection and coding, *i.e.* separating the optimization for a minimum-cost subgraph, which we discuss in Section 4, and the construction of a code for a given subgraph, which we now discuss. We ultimately aim for practicable distributed solutions for both problems, which can be simultaneously and, to a large degree, independently implemented.

We first consider idealized, lossless, "static" networks before considering the "dynamic" packet networks of our model in Section 2.

Static networks

To illustrate the basics of algebraic network coding, let us first consider the simple five-node network of Figure 5.2. Following our model, we represent the network using the hypergraph shown in Figure . We have $a(1) = 2$, $a(2) = 3$, and $T = \{6, 7\}$. We suppose that the source processes X_1 and X_2 are bits, so node 2 transmits X_1, node 3 transmits X_2, and node 4 transmits $X_1 + X_2$, the binary sum of X_1 and X_2. Recall that this coding permits both node 6 and node 7 to recover X_1 and X_2, establishing a multicast session consisting of two sources and two receivers.

We can generalize these basic ideas to more complex networks, with coding functions that can be arbitrary functions instead of just binary sums of bits. More specifically, we can consider the random process transmitted on hyperarc

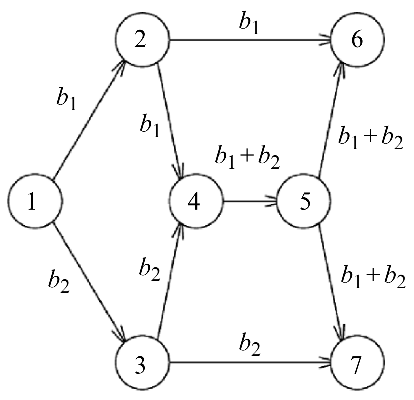

Figure 5.4. Figure 5.3 redrawn in its hypergraph representation.

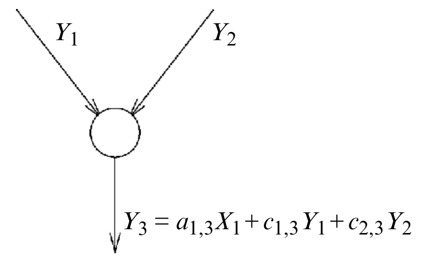

Figure 5.5. Illustration of linear coding at a node.

(i, J), denoted Y_{iJ}, to be a binary sequence obtained as a function θ_{iJ} of hyperarc (i, J)'s ***inputs***, ***i.e.*** source processes X_k for which $a(k) = i$ and random processes $Y_{i'J'}$ for which $i \in J'$, if any. Functions $\{\theta_{iJ} | (i, J) \in \mathcal{A}\}$ specify a network code. The kth output process $Z_{t,k}$ at a receiver node t is the binary sequence obtained as a function $\phi_{t,k}$ of the random processes $Y_{i'J'}$ for which $t \in J'$. If, for a given network code, functions $\{\phi_{t,k}\}$ exist for which the output processes $\{Z_{t,k}\}$ are equal to the corresponding source processes $\{X_k\}$ (possibly with some delay), the network code is called *permissible* for t. A network code that is permissible for all receivers simultaneously is called *valid*, and the network code together with the corresponding decoding functions $\{\phi_{t,k}\}$ form a *solution* to the multicast connection problem. If a solution exists for a multicast connection problem, it is called *feasible*, and the connection requirements are said to be feasible on the network.

It turns out that for any feasible multicast connection problem, there exists a linear network coding solution, in which coding operations are done on length-u vectors of bits in a finite field $\mathbb{F}_q$, $q = 2^u$. We can express the coding functions in compact mathematical form by considering unit rate sources and unit capacity hyperarcs, allowing for multiple sources at a node and multiple hyperarcs with the same origin and destination nodes. In an acyclic network where each node waits for inputs on all its incoming hyperarcs before sending an output, the formation of the process Y_{iJ} transmitted on a hyperarc (i, J) is represented by the equation

$$Y_{iJ} = \sum_{\{k\,:\,a(k)=i\}} a_{k,iJ} X_k + \sum_{\{(i',J')\,:\,i\in J'\}} c_{i'J',iJ} Y_{i'J'}.$$

This is illustrated in Figure 5.5. Each sink receives as many linearly independent input processes as the number of source processes, and is able to decode by taking a linear combination of its input processes:

$$Z_{t,k} = \sum_{\{(i',J')\,:\,t\in J'\}} b_{t,k,i'J'} Y_{i'J'}.$$

For general networks with cycles, we need to explicitly consider transmission delays to ensure stability. For instance, if each link has the same delay, the linear coding equations are

$$\begin{aligned} Y_{iJ}(t+1) &= \sum_{\{k\,:\,a(k)=i\}} a_{k,iJ} X_k(t) \\ &\quad + \sum_{\{(i',J')\,:\,i\in J'\}} c_{i'J',iJ} Y_{i'J'}(t), \\ Z_{t,k}(t+1) &= \sum_{u=0}^{\mu} b'_{t,k}(u) Z_{t,k}(t-u) \\ &\quad + \sum_{\{(i',J')\,:\,t\in J'\}} \sum_{u=0}^{\mu} b''_{t,k,i'J'}(u) Y_{i'J'}(t-u), \end{aligned}$$

where $X_k(t), Y_{iJ}(t), Z_{t,k}(t), b'_{t,k}(t)$ and $b''_{t,k,i'J'}(t)$ are the values of the corresponding variables at time t respectively. The variable μ represents the memory required at receiver t for decoding when link delays are considered. These equations and random processes can be represented algebraically in terms of a delay variable D:

$$\begin{aligned} Y_{iJ}(D) &= \sum_{\{k\,:\,a(k)=i\}} D a_{k,iJ} X_k(D) \\ &\quad + \sum_{\{(i',J')\,:\,i\in J'\}} D c_{i'J',iJ} Y_{i'J'}(D), \\ Z_{t,k}(D) &= \sum_{\{(i',J')\,:\,t\in J'\}} b_{t,k,i'J'}(D) Y_{i'J'}(D), \end{aligned}$$

where

$$b_{t,\,k,\,i'J'}(D) = \frac{\sum_{u=0}^{\mu} D^{u+1} b''_{t,k,i'J'}(u)}{1 - \sum_{u=0}^{\mu} D^{u+1} b'_{t,k}(u)}$$

and

$$\begin{aligned} X_k(D) &= \sum_{t=0}^{\infty} X_k(t) D^t, \\ Y_{iJ}(D) &= \sum_{t=0}^{\infty} Y_{iJ}(t) D^t, \qquad Y_{iJ}(0) = 0, \\ Z_{t,k}(D) &= \sum_{t=0}^{\infty} Z_{t,k}(t) D^t, \qquad Z_{t,k}(0) = 0. \end{aligned}$$

Furthermore, for a given feasible coding subgraph, choosing the code coefficients $\{a_{k,iJ}, c_{i'J',iJ}\}$ uniformly at random from a sufficiently large field $\mathbb{F}_q$ gives, with high probability, a solution to any multicast connection problem that is feasible on the subgraph. The field size q must be at least greater than the number of receivers d. Some of the coefficients can be fixed rather than randomly chosen, as long as there exists a solution to the network connection problem with the same values for these fixed coefficients. For instance, if a node i receives linearly dependent processes on two incoming links (i_1, J_1), (i_2, J_2), it can fix $c_{i_1 J_1, iJ} = 0$ for all its outgoing links (i, J).

While not the only way to find a valid network code, this random linear coding approach offers a particularly convenient distributed way to set up a network code, in which each node makes independent random choices of coding functions. The coefficient vectors needed for decoding can be sent through the network with each data block over which the code remains constant. The coefficient vector sent with each block of data from source X_k, $k = 1, \ldots, r$, is the length-r unit vector with a single nonzero entry, 1, in the kth position. Each coding node applies the same linear mappings to the coefficient vectors as to their corresponding data. In this way, the inputs received at a sink are accompanied by coefficient vectors specifying their composition as a linear combination of the original source processes.

To see why random linear network coding is sufficient to solve any feasible multicast connection problem with high probability, consider the hypergraph $\mathcal{H}$ used by some (possibly nonlinear) solution to a feasible multicast problem. Note that the multicast rate can be no larger than the minimum of the rates that can be sent to each sink separately on $\mathcal{H}$. We show that, with high probability, random linear network coding achieves this rate, which implies that it is sufficient.

Suppose we do random linear network coding over $\mathcal{H}$. Consider a subgraph $\mathcal{H}_t$ of $\mathcal{H}$ that transmits the desired rate to sink t separately. The network coding solution can be reduced to a flow solution to sink t over $\mathcal{H}_t$ if the code coefficients

$\{a_{k,iJ}, c_{i'J',iJ}\}$ associated with $\mathcal{H}_t$ take the value 1 and the rest of the code coefficients take the value 0. In this case, sink t receives a set of inputs whose coefficient vectors together form an identity matrix. Now consider the matrix of coefficient vectors of the same set of inputs, but with the code coefficients $\{a_{k,iJ}, c_{i'J',iJ}\}$ as indeterminate variables. The determinant of this matrix is a polynomial in $\{a_{k,iJ}, c_{i'J',iJ}\}$ (and in D, if we consider link delays) which we know, from considering the flow solution to t, is not identically zero. Each sink similarly has a set of inputs whose associated coefficient vectors form a matrix with a nonzero determinant polynomial. Multiplying these matrices together, we obtain a polynomial in $\{a_{k,iJ}, c_{i'J',iJ}\}$ that is not identically zero. By the Schwartz-Zippel theorem (see, for example, [Motwani and Raghavan, 1995]), if we choose the code coefficients uniformly at random from a finite field $\mathbb{F}_q$ where q is greater than the degree of the polynomial, then the polynomial takes a zero value with probability inversely proportional to q.

Owing to the particular structure of these polynomials, the probability of obtaining a valid random code is actually higher than that given by the Schwartz-Zippel theorem. We can bound this probability more tightly as follows. We denote by η the number of hyperarcs (i, J) with associated random coefficients $\{a_{k,iJ}, c_{i'J',iJ}\}$.

THEOREM 5.1 *Consider a multicast connection problem on an arbitrary static network and a network code in which some or all network code coefficients $\{a_{k,iJ}, c_{i'J',iJ}\}$ are chosen uniformly at random from a finite field $\mathbb{F}_q$ where $q > d$, and the remaining code coefficients, if any, are fixed. If there exists a solution to the network connection problem with the same values for the fixed code coefficients, then the probability that the random network code is valid is at least $(1 - d/q)^{\eta}$.*

For a proof of Theorem 5.1, see [Ho et al., 2003]. Recall that $q = 2^u$, so the error probability decreases exponentially with u. Thus, random linear coding achieves maximum multicast capacity with probability exponentially approaching 1 with the number of bits in the coding field.

We can extend the basic network coding model and results for static networks described above to dynamic packet networks with bursty sources and varying link delays and capacities. In the static case, the code is the same for all vectors of bits originating at the same source or traversing the same hyperarc. In the dynamic case, the code may change from packet to packet, but is the same for all vectors of bits in the same packet. Thus, we may draw an analogy between sources and hyperarcs in the static case, and source packets and coded/forwarded packets respectively in the dynamic case. In the dynamic case, each packet would contain a coefficient vector of length equal to the number of source packets that may be coded together in the network.

One way to operate a dynamic network is for each coding node to wait until it receives a packet corresponding to each of its inputs, before coding them together and transmitting packets on its outgoing hyperarcs. But such waiting seems unnecessary. An alternative approach divides packets formed around the same time into batches. Each packet is labeled with a batch number and some information specifying its intended sink node(s), and coding is done, without waiting, only across packets of the same batch. We consider applying such a batched approach to dynamic networks in the following section.

Dynamic networks

The specific coding scheme we consider, which we hereafter refer to as distributed random network coding, is as follows. We suppose that, at the beginning of each batch, all nodes flush their memories of packets associated with previous batches. Now, at node $a(k)$, source X_k generates a batch of K' message packets $X_{k,1}, X_{k,2}, \ldots, X_{k,K'}$, which are vectors of length u over the finite field $\mathbb{F}_q$. (If the packet length is b bits, then we take $u = \lceil b/\log_2 q \rceil$.) These message packets are placed in node $a(k)$'s memory.

The coding operation performed by each node is simple to describe and is the same for every node: Received packets are stored into the node's memory, and packets are formed for injection with random linear combinations of its memory contents whenever a packet injection occurs on an outgoing link. The coefficients of the combination are drawn uniformly from $\mathbb{F}_q$.

Since all coding is linear, we can write any packet x in the network as a linear combination of the $K := rK'$ message packets; namely, we have $x = \sum_{k=1}^{r} \sum_{l=1}^{K'} \gamma_{kl} X_{k,l}$. We call γ the *global encoding vector* of x, and we assume that it is sent along with x as side information in its header. The overhead this incurs (namely, $K \log_2 q$ bits) is negligible if packets are sufficiently large.

Nodes are assumed to have unlimited memory. The scheme can be modified so that received packets are stored into memory only if their global encoding vectors are linearly-independent of those already stored. This modification keeps our conclusions regarding the scheme unchanged while ensuring that nodes never need to store more than K packets.

A sink node collects packets and, if it has K packets with linearly-independent global encoding vectors, it is able to recover the message packets. Decoding can be done by Gaussian elimination, and the scheme can be operated ratelessly, *i.e.* it can be run indefinitely until successful reception (at which stage that fact is signaled to other nodes).

We suppose that sink $t \in T$ wishes to achieve rate arbitrarily close to R_t, *i.e.* to recover the K message packets, sink t wishes to wait for a time Δ_t that is only marginally greater than K/R_t. The main result that we have in relation to distributed random network coding is as follows.

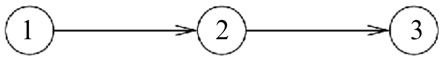

Figure 5.6. A network consisting of two links in tandem.

THEOREM 5.2 *Distributed random network coding achieves capacity for multicast sessions; more precisely, for K sufficiently large, it can satisfy, with arbitrarily small error probability, the set of multicast connection requirements $(X_1, \ldots, X_r; a; T)$ at rate arbitrarily close to R_t packets per unit time for each $t \in T$ if there exists, for all $t \in T$, a non-negative flow vector $x^{(t)}$ satisfying*

$$\sum_{\{J|(i,J)\in\mathcal{A}\}} \sum_{j\in J} x_{iJj}^{(t)} - \sum_{\{j|(j,I)\in\mathcal{A}, i\in I\}} x_{jIi}^{(t)} = \sigma_i^{(t)}, \tag{5.2}$$

for all $i \in \mathcal{N}$, and

$$\sum_{j\in K} x_{iJj}^{(t)} \leq \sum_{\{L\subset J|L\cap K\neq\emptyset\}} z_{iJL}$$

for all $(i, J) \in \mathcal{A}$ and $K \subset J$, where

$$\sigma_i^{(t)} := \begin{cases} \frac{|\{k|a(k)=i\}|}{r} R_t & \text{if } i \neq t, \\ -R_t & \text{if } i = t. \end{cases}$$

We see from Theorem 5.2 that distributed random network coding is remarkably robust: If run over sufficiently large batch sizes K, it achieves the maximum feasible rate of a given coding subgraph, with only assumption (5.1) on the arrival of received packets on a link. Assumption (5.1) makes no claims on loss correlation or lack thereof—all we require is that a long-run average exists. This fact is particularly important in wireless packet networks, where slow fading and collisions often cause packets not to be received in a steady stream.

To see why Theorem 5.2 is true, consider first the simplest non-trivial case: that of a single unicast connection over two links in tandem (see Figure 5.6).

Suppose we wish to establish a connection of rate arbitrarily close to R packets per unit time from node 1 to node 3. Suppose further that the coding scheme is run for a total time Δ, from time 0 until time Δ, and that, in this time, a total of N packets is received by node 2. We call these packets $v_1, v_2, \ldots, v_N$.

Any received packet y in the network is a linear combination of $v_1, v_2, \ldots, v_N$, so we can write

$$y = \sum_{n=1}^{N} \beta_n v_n.$$

Since v_n is formed by a random linear combination of the message packets $w_1, w_2, \ldots, w_K$, we have

$$v_n = \sum_{k=1}^{K} \alpha_{nk} w_k$$

for $n = 1, 2, \ldots, N$. Hence

$$y = \sum_{k=1}^{K} \left(\sum_{n=1}^{N} \beta_n \alpha_{nk} \right) w_k,$$

and it follows that the kth component of the global encoding vector of y is given by

$$\gamma_k = \sum_{n=1}^{N} \beta_n \alpha_{nk}.$$

We call the vector β associated with y the *auxiliary encoding vector* of y, and we see that any node that receives $\lfloor K(1+\varepsilon) \rfloor$ or more packets with linearly-independent auxiliary encoding vectors has $\lfloor K(1+\varepsilon) \rfloor$ packets whose global encoding vectors collectively form a random $\lfloor K(1+\varepsilon) \rfloor \times K$ matrix over $\mathbb{F}_q$, with all entries chosen uniformly. If this matrix has rank K, then node 3 is able to recover the message packets. The probability that a random $\lfloor K(1+\varepsilon) \rfloor \times K$ matrix has rank K is, by a simple counting argument, $\prod_{k=1+\lfloor K(1+\varepsilon) \rfloor - K}^{\lfloor K(1+\varepsilon) \rfloor} (1 - 1/q^k)$, which can be made arbitrarily close to 1 by taking K arbitrarily large. Therefore, to determine whether node 3 can recover the message packets, we essentially need only to determine whether it receives $\lfloor K(1+\varepsilon) \rfloor$ or more packets with linearly-independent auxiliary encoding vectors.

Our proof is based on tracking the propagation of what we call *innovative* packets. Such packets are innovative in the sense that they carry new, as yet unknown, information about $v_1, v_2, \ldots, v_N$ to a node. It turns out that the propagation of innovative packets through a network follows the propagation of jobs through a queueing network, for which fluid flow models give good approximations. We present the following argument in terms of this fluid analogy.

Since the packets being received by node 2 are the packets $v_1, v_2, \ldots, v_N$ themselves, it is clear that every packet being received by node 2 is innovative. Thus, innovative packets arrive at node 2 at a rate of z_{122}, and this can be approximated by fluid flowing in at rate z_{122}. These innovative packets are stored in node 2's memory, so the fluid that flows in is stored in a reservoir.

Packets, now, are being received by node 3 at a rate of z_{233}, but whether these packets are innovative depends on the contents of node 2's memory. If node 2 has more information about $v_1, v_2, \ldots, v_N$ than node 3 does, then it is likely that new information will be described to node 3 in the next packet that it receives. Otherwise, if node 2 and node 3 have the same degree of information about

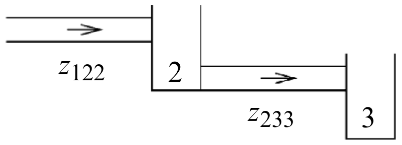

Figure 5.7. Fluid flow system corresponding to two-link tandem network.

$v_1, v_2, \ldots, v_N$, then packets received by node 3 cannot possibly be innovative. Thus, the situation is as though fluid flows into node 3's reservoir at a rate of z_{233}, but the level of node 3's reservoir is restricted from ever exceeding that of node 2's reservoir. The level of node 3's reservoir, which is ultimately what we are concerned with, can equivalently be determined by fluid flowing out of node 2's reservoir at rate z_{233}.

We therefore see that the two-link tandem network in Figure 5.6 maps to the fluid flow system shown in Figure 5.7. It is clear that, in this system, fluid flows into node 3's reservoir at rate $\min(z_{122}, z_{233})$. This rate determines the rate at which packets with new information about $v_1, v_2, \ldots, v_N$—and, therefore, linearly-independent auxiliary encoding vectors—arrive at node 3. Hence the time required for node 3 to receive $\lfloor K(1+\varepsilon)\rfloor$ packets with linearly-independent auxiliary encoding vectors is, for large K, approximately $K(1+\varepsilon)/\min(z_{122}, z_{233})$, which implies that a connection of rate arbitrarily close to R packets per unit time can be established provided that

$$R \leq \min(z_{122}, z_{233}). \tag{5.3}$$

If, as in Theorem 5.2, there exists a flow vector $x^{(3)}$ such that $x^{(3)}_{122} = x^{(3)}_{233} = R$, $x^{(3)}_{122} \leq z_{122}$, and $x^{(3)}_{233} \leq z_{233}$, then it is clear that condition (5.3) is satisfied, and the connection can be established.

From this simple case of a single unicast connection over two links in tandem, it is in fact not difficult to extend to the general case described by Theorem 5.2. We first extend from two links in tandem to arbitrarily many links in tandem, which is quite straightforward. From arbitrarily many links in tandem, we then consider any unicast connection by decomposing the hypergraph flow into a set of paths, each of which can be considered as a unicast connection over a number of links in tandem. Extending to multiple sources is straightforward, as is extending to multiple sinks, where, because the coding scheme is quite oblivious to the flow vectors $x^{(t)}$ for each $t \in T$, each flow behaves more or less independently of the others. For a formal proof of Theorem 5.2, see [Lun et al., 2005b].

4. Cost minimization

We now turn to the subgraph selection problem, which we see is the problem of finding a coding subgraph z of minimum cost satisfying (5.2). Thus, the subgraph selection problem, for a single session, equates to the following optimization problem.

$$\begin{aligned}
&\text{minimize } f(z) \\
&\text{subject to } z \in Z, \\
&\sum_{j \in K} x_{iJj}^{(t)} \leq \sum_{\{L \subset J | L \cap K \neq \emptyset\}} z_{iJL}, \qquad \forall\, (i,J) \in \mathcal{A}, K \subset J, t \in T, \\
&x^{(t)} \in F^{(t)}, \qquad \forall\, t \in T,
\end{aligned} \tag{5.4}$$

where $x^{(t)}$ is the vector consisting of $x_{iJj}^{(t)}$, $(i,J) \in \mathcal{A}$, $j \in J$, and $F^{(t)}$ is the bounded polyhedron of points $x^{(t)}$ satisfying the conservation of flow constraints

$$\sum_{\{J|(i,J)\in\mathcal{A}\}} \sum_{j \in J} x_{iJj}^{(t)} - \sum_{\{j|(j,I)\in\mathcal{A}, i \in I\}} x_{jIi}^{(t)} = \sigma_i^{(t)}, \qquad \forall\, i \in \mathcal{N},$$

and non-negativity constraints

$$x_{iJj}^{(t)} \geq 0, \qquad \forall\, (i,J) \in \mathcal{A}, j \in J,$$

In the lossless case, problem (5.4) simplifies to the following optimization problem.

$$\begin{aligned}
&\text{minimize } f(z) \\
&\text{subject to } z \in Z, \\
&\sum_{j \in J} x_{iJj}^{(t)} \leq z_{iJ}, \qquad \forall\, (i,J) \in \mathcal{A}, t \in T, \\
&x^{(t)} \in F^{(t)}, \qquad \forall\, t \in T.
\end{aligned} \tag{5.5}$$

As an example, let us return to the wireless network shown in Figure 5.4. The network is lossless, and we have $a(1) = 2$, $a(2) = 3$, and $T = \{6, 7\}$. We wish to achieve unit rate to both sinks, so $R_6 = R_7 = 1$. We suppose that $Z = [0,1]^{|\mathcal{A}|}$ and $f(z) = \sum_{(i,J)\in\mathcal{A}} z_{iJ}$. An optimal solution to problem (5.5) is shown in Figure 5.8. We have flows $x^{(6)}$ and $x^{(7)}$, each with half a unit of flow originating from each of the sources and going to their respective sinks. For each hyperarc (i,J), $z_{iJ} = \max(\sum_{j\in J} x_{iJj}^{(6)}, \sum_{j \in J} x_{iJj}^{(7)})$, as we expect from the optimization. To achieve the optimal cost, we can apply distributed random network coding to the subgraph defined by z.

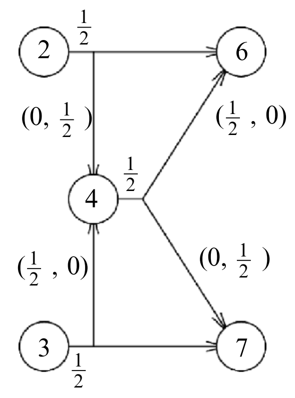

Figure 5.8. Cost optimization of a wireless network with multicast. Each hyperarc is marked with z_{iJ} at the start and with the pair $(x_{iJj}^{(6)}, x_{iJj}^{(7)})$ at the ends.

Neither problem (5.4) nor (5.5) as they stand are easy to solve. But they are very general. Their complexities improve if we assume that the cost function is separable and convex, or even linear; *i.e.* if we suppose $f(z) = \sum_{(i,J)\in A} f_{iJ}(z_{iJ})$, where f_{iJ} is a convex or linear function, which is a very reasonable assumption in many situations. Their complexities also improve if we make some assumptions on the form of the constraint set Z.

A simplification can also be made if we assume that, when nodes transmit in a lossless network, they reach all nodes in a certain area, with cost increasing as this area is increased. More precisely, suppose that we have separable cost, so $f(z) = \sum_{(i,J)\in\mathcal{A}} f_{iJ}(z_{iJ})$. Suppose further that each node i has M_i outgoing hyperarcs $(i, J_1^{(i)}), (i, J_2^{(i)}), \ldots, (i, J_{M_i}^{(i)})$ with $J_1^{(i)} \subsetneq J_2^{(i)} \subsetneq \cdots \subsetneq J_{M_i}^{(i)}$. (We assume that there are no identical links, as duplicate links can effectively be treated as a single link.) Then, we assume that $f_{iJ_1^{(i)}}(\zeta) < f_{iJ_2^{(i)}}(\zeta) < \cdots < f_{iJ_{M_i}^{(i)}}(\zeta)$ for all $\zeta \geq 0$ and nodes i. For $(i,j) \in \mathcal{A}' := \{(i,j)|(i,J) \in A, J \ni j\}$, we introduce the variables

$$\hat{x}_{ij}^{(t)} := \sum_{m=m(i,j)}^{M_i} x_{iJ_m^{(i)}j}^{(t)},$$

where $m(i,j)$ is the unique m such that $j \in J_m^{(i)} \setminus J_{m-1}^{(i)}$ (we define $J_0^{(i)} := \emptyset$ for all $i \in \mathcal{N}$ for convenience). Then, provided that $a_{iJ_1^{(i)}} < a_{iJ_2^{(i)}} < \cdots < a_{iJ_{M_i}^{(i)}}$ for all nodes i, problem (5.5) can be reformulated as the following optimization problem, which has substantially fewer variables.

$$\begin{aligned}
&\text{minimize } \sum_{(i,J)\in\mathcal{A}} f_{iJ}(z_{iJ}) \\
&\text{subject to } z \in Z, \\
&\sum_{k\in J^{(i)}_{M_i}\setminus J^{(i)}_{m-1}} \hat{x}^{(t)}_{ik} \leq \sum_{n=m}^{M_i} z_{iJ^{(i)}_n}, \qquad \forall\, i \in \mathcal{N},\, m = 1,\dots,M_i,\, t\in T, \\
&\hat{x}^{(t)} \in \hat{F}^{(t)}, \qquad \forall\, t \in T,
\end{aligned} \tag{5.6}$$

where $\hat{F}^{(t)}$ is the bounded polyhedron of points $\hat{x}^{(t)}$ satisfying the conservation of flow constraints

$$\sum_{\{j|(i,j)\in\mathcal{A}'\}} \hat{x}^{(t)}_{ij} - \sum_{\{j|(j,i)\in\mathcal{A}'\}} \hat{x}^{(t)}_{ji} = \sigma^{(t)}_i, \qquad \forall\, i \in N,$$

and non-negativity constraints

$$0 \leq \hat{x}^{(t)}_{ij}, \qquad \forall\, (i,j) \in \mathcal{A}'.$$

PROPOSITION 5.1 *Suppose that $f(z) = \sum_{(i,J)\in\mathcal{A}} f_{iJ}(z_{iJ})$ and that $f_{iJ^{(i)}_1}(\zeta) < f_{iJ^{(i)}_2}(\zeta) < \cdots < f_{iJ^{(i)}_{M_i}}(\zeta)$ for all $\zeta \geq 0$ and $i \in \mathcal{N}$. Then problem (5.5) and problem (5.6) are equivalent in the sense that they have the same optimal cost and z is part of an optimal solution for (5.5) if and only if it is part of an optimal solution for (5.6).*

Proof: Suppose (x, z) is a feasible solution to problem (5.5). Then, for all $(i, j) \in \mathcal{A}'$ and $t \in T$,

$$\begin{aligned}
\sum_{m=m(i,j)}^{M_i} z_{iJ^{(i)}_m} &\geq \sum_{m=m(i,j)}^{M_i} \sum_{k\in J^{(i)}_m} x^{(t)}_{iJ^{(i)}_m k} \\
&= \sum_{k\in J^{(i)}_{M_i}} \sum_{m=\max(m(i,j),m(i,k))}^{M_i} x^{(t)}_{iJ^{(i)}_m k} \\
&\geq \sum_{k\in J^{(i)}_{M_i}\setminus J^{(i)}_{m(i,j)-1}} \sum_{m=\max(m(i,j),m(i,k))}^{M_i} x^{(t)}_{iJ^{(i)}_m k} \\
&= \sum_{k\in J^{(i)}_{M_i}\setminus J^{(i)}_{m(i,j)-1}} \sum_{m=m(i,k)}^{M_i} x^{(t)}_{iJ^{(i)}_m k} \\
&= \sum_{k\in J^{(i)}_{M_i}\setminus J^{(i)}_{m(i,j)-1}} \hat{x}^{(t)}_{ik}.
\end{aligned}$$

Hence $(\hat{x}, z)$ is a feasible solution of problem (5.6) with the same cost.

Now suppose $(\hat{x}, z)$ is an optimal solution of problem (5.6). Since $f_{iJ_1^{(i)}}(\zeta) < f_{iJ_2^{(i)}}(\zeta) < \cdots < f_{iJ_{M_i}^{(i)}}(\zeta)$ for all $\zeta \geq 0$ and $i \in \mathcal{N}$ by assumption, it follows that, for all $i \in \mathcal{N}$, the sequence $z_{iJ_1^{(i)}}, z_{iJ_2^{(i)}}, \ldots, z_{iJ_{M_i}^{(i)}}$ is given recursively, starting from $m = M_i$, by

$$z_{iJ_m^{(i)}} = \max_{t \in T} \left\{ \sum_{k \in J_{M_i}^{(i)} \setminus J_{m-1}^{(i)}} \hat{x}_{ik}^{(t)} \right\} - \sum_{m'=m+1}^{M_i} z_{iJ_{m'}^{(i)}}.$$

Hence $z_{iJ_m^{(i)}} \geq 0$ for all $i \in \mathcal{N}$ and $m = 1, 2, \ldots, M_i$. We then set, starting from $m = M_i$ and $j \in J_{M_i}^{(i)}$,

$$x_{iJ_m^{(i)}j}^{(t)} := \min \left(\hat{x}_{ij}^{(t)} - \sum_{l=m+1}^{M_i} x_{iJ_l^{(i)}j}, z_{iJ_m^{(i)}} - \sum_{k \in J_{M_i}^{(i)} \setminus J_{m(i,j)}^{(i)}} x_{iJ_m^{(i)}k}^{(t)} \right).$$

It is now not difficult to see that (x, z) is a feasible solution of problem (5.5) with the same cost.

Therefore, the optimal costs of problems (5.5) and (5.6) are the same and, since the objective functions for the two problems are the same, z is part of an optimal solution for problem (5.5) if and only if it is part of an optimal solution for problem (5.6). ■

One specific problem of interest is that of minimum-energy multicast (see, for example, [Wieselthier et al., 2002; Liang, 2002]). In this problem, we wish to achieve minimum-energy multicast in a lossless wireless network without explicit regard for throughput or bandwidth, so the constraint set Z can be dropped altogether. Moreover, the cost function is separable and linear, *i.e.* $f(z) = \sum_{(i,J) \in \mathcal{A}} a_{iJ} z_{iJ}$, where a_{iJ} represents the energy required to transmit a packet to nodes in J from node i. Hence problem (5.6) becomes a linear optimization problem with a polynomial number of constraints, which can therefore be solved in polynomial time. By contrast, the same problem using traditional routing-based approaches is NP-complete—in fact, the special case of broadcast in itself is NP-complete, a result shown by [Ahluwalia et al., 2002; Liang, 2002]. The problem must therefore be addressed using polynomial-time heuristics such as the MIP algorithm by [Wieselthier et al., 2002]. Even if an optimal routing solution is found, it is in general worse than an optimal coding solution because coding subsumes routing. Thus coding promises to significantly outperform routing for practical multicast, and, indeed, simulation results reported by [Lun et al., 2005a] show significant reductions in the average total energy of random multicast connections in random wireless networks of varying size as a result of coding as opposed to routing with the MIP algorithm.

It is, however, not sufficient to have polynomial-time algorithms. For practical applications, it is usually important that solutions can be computed in a distributed manner, with each node making computations based only on local knowledge and knowledge acquired from message exchanges. Thus, we seek distributed algorithms to solve optimization problems (5.4), (5.5), and (5.6), which, when paired with distributed random network coding gives us a fully distributed approach for establishing minimum-cost connections in wireless networks. To this end, we simplify the problem by assuming that the objective function is of the form

$$f(z) = \sum_{(i,J)\in\mathcal{A}} f_{iJ}(z_{iJ}),$$

where f_{iJ} is a monotonically increasing, convex function, and by assuming that, as z_{iJ} is varied, z_{iJK}/z_{iJ} is constant for all $K \subset J$. Therefore,

$$b_{iJK} := \frac{\sum_{\{L\subset J | L\cap K\neq\emptyset\}} z_{iJL}}{z_{iJ}}$$

is a constant. We also drop the constraint set Z, noting that separable constraints, at least, can be handled by making f_{iJ} approach infinity as z_{iJ} approaches its upper constraint. Moreover, in energy-limited scenarios where energy is the principal concern, the rate of the multicast connection can always be dropped so that the constraint set Z is not restrictive; we discuss bandwidth-limited scenarios in a later section.

Hence problem (5.4) becomes

$$\begin{aligned} \text{minimize } & \sum_{(i,J)\in\mathcal{A}} f_{iJ}(z_{iJ}) \\ \text{subject to } & \sum_{j\in K} x_{iJj}^{(t)} \leq z_{iJ} b_{iJK}, \qquad \forall\, (i,J)\in\mathcal{A}, K\subset J, t\in T, \\ & x^{(t)} \in F^{(t)}, \qquad \forall\, t\in T. \end{aligned} \tag{5.7}$$

Since the f_{iJ} are monotonically increasing, the constraint

$$\sum_{j\in K} x_{iJj}^{(t)} \leq z_{iJ} b_{iJK}, \qquad \forall\, (i,J)\in\mathcal{A}, K\subset J, t\in T \tag{5.8}$$

gives

$$z_{iJ} = \max_{K\subset J, t\in T} \left\{ \frac{\sum_{j\in K} x_{iJj}^{(t)}}{b_{iJK}} \right\}. \tag{5.9}$$

Expression (5.9) is, unfortunately, not very useful for algorithm design because the max function is difficult to deal with, largely as a result of it not being everywhere differentiable. One way to overcome this difficulty is to approximate z_{iJ} by replacing the max in (1.9) with an l^n-norm (see [Deb and Srikant, 2004]), *i.e.* to approximate z_{iJ} with z'_{iJ}, where

$$z'_{iJ} := \left(\sum_{K \subset J, t \in T} \left(\frac{\sum_{j \in K} x_{iJj}^{(t)}}{b_{iJK}} \right)^n \right)^{1/n}.$$

The approximation becomes exact as $n \to \infty$.

Now the relevant optimization problem is

$$\begin{aligned} &\text{minimize } \sum_{(i,J) \in \mathcal{A}} f_{iJ}(z'_{iJ}) \\ &\text{subject to } x^{(t)} \in F^{(t)}, \qquad \forall\, t \in T, \end{aligned}$$

which is no more than a convex multicommodity flow problem. There are many algorithms for convex multicommodity flow problems (see [Ouorou et al., 2000] for a survey), some of which (*e.g.* the algorithms by [Bertsekas, 1980; Bertsekas et al., 1984]) are well-suited for distributed implementation. Thus, there exists a significant number of distributed algorithms for the subgraph selection problem. We present two: The first, which we call the subgradient method, does not reformulate the problem as a convex multicommodity flow problem and attempts to deal with the constraint (5.8) directly, while the second, which we call the primal-dual method, applies a particular method for solving convex multicommodity flow problems to our problem.

Subgradient method

We present the subgradient method for linear cost functions, though, with some modifications, it may be made to apply to convex ones. Thus, we assume that the objective function f is of the form

$$f(z) := \sum_{(i,J) \in \mathcal{A}} a_{iJ} z_{iJ},$$

where $a_{iJ} > 0$.

Consider the Lagrangian dual of problem (5.7):

$$\begin{aligned} &\text{maximize } \sum_{t \in T} q^{(t)}(p^{(t)}) \\ &\text{subject to } \sum_{t \in T} \sum_{K \subset J} p_{iJK}^{(t)} = a_{iJ} \qquad \forall\, (i,J) \in \mathcal{A}, \\ &p_{iJK}^{(t)} \geq 0, \qquad \forall\, (i,J) \in \mathcal{A}, K \subset J, t \in T, \end{aligned} \tag{5.10}$$

where

$$q^{(t)}(p^{(t)}) := \min_{x^{(t)} \in F^{(t)}} \sum_{(i,J)\in\mathcal{A}} \sum_{j\in J} \left(\sum_{\{K\subset J | K \ni j\}} \frac{p^{(t)}_{iJK}}{b_{iJK}} \right) x_{iJj}. \tag{5.11}$$

In the lossless case (optimization problem (5.5)), the dual problem defined by equations (5.10) and (5.11) simplifies somewhat, and we require only a single dual variable $p^{(t)}_{iJJ}$ for each hyperarc (i, J). In the case of optimization problem (5.6), the dual problem simplifies more still, as there are fewer primal variables associated with it. Specifically, we obtain, for the Lagrangian dual,

$$\begin{aligned} &\text{maximize } \sum_{t\in T} \hat{q}^{(t)}(p^{(t)}) \\ &\text{subject to } \sum_{t\in T} p^{(t)}_{iJ^{(i)}_m} = s_{iJ^{(i)}_m}, \qquad \forall\, i \in \mathcal{N}, m = 1, \dots, M_i, \\ &\qquad p^{(t)}_{iJ} \geq 0, \qquad \forall\, (i,J) \in \mathcal{A}, t \in T, \end{aligned} \tag{5.12}$$

where

$$s_{iJ^{(i)}_m} := a_{iJ^{(i)}_m} - a_{iJ^{(i)}_{m-1}},$$

and

$$\hat{q}^{(t)}(p^{(t)}) := \min_{\hat{x}^{(t)} \in \hat{F}^{(t)}} \sum_{(i,j)\in\mathcal{A}'} \left(\sum_{m=1}^{m(i,j)} p^{(t)}_{iJ^{(i)}_m} \right) \hat{x}^{(t)}_{ij}. \tag{5.13}$$

In all three cases, the dual problems are very similar, and essentially the same algorithm can be used to solve them. We present the subgradient method for the case of optimization problem (5.6) and its associated dual (5.12) with the understanding that straightforward modifications can be made for the other cases.

We outline the subgradient method below. [Lun et al., 2005c] give a proof that the algorithm does indeed converge to an optimal solution for appropriate choices of the parameters $\{\theta[n]\}$ and $\{\mu_l[n]\}$. We later explain how to choose these parameters.

1 Each node $i \in N$ computes s_{iJ} for its outgoing hyperarcs and initializes $p_{iJ}[0]$ to a point in the feasible set of (5.12). We take, for our purposes,

$$p^{(t)}_{iJ}[0] := \frac{s_{iJ}}{|T|}.$$

The values of s_{iJ} and $p_{iJ}[0]$ are then sent over hyperarc (i, J).

2 In the nth iteration, use $p^{(t)}[n]$ as the hyperarc costs, and run a distributed shortest path algorithm (*e.g.* distributed Bellman-Ford) to determine

$\hat{x}^{(t)}[n]$ for all $t \in T$. This can be done because subproblem (5.13) is in fact a standard shortest path problem.

3 Based on the $\hat{x}[n]$ obtained, we calculate a subgradient of the dual function with respect to $p^{(t)}_{iJ^{(i)}_m}[n]$, namely $g^{(t)}_{iJ^{(i)}_m}[n] := \sum_{k \in J^{(i)}_{M_i} \setminus J^{(i)}_{m-1}} \hat{x}^{(t)}_{ik}[n]$. We then compute, at node i,

$$p_{iJ}[n+1] := \arg\min_{v \in P_{iJ}} \sum_{t \in T} (v^{(t)} - (p^{(t)}_{iJ}[n] + \theta[n] g^{(t)}_{iJ}[n]))^2$$

for each $(i, J) \in \mathcal{A}$, where P_{iJ} is the $|T|$-dimensional simplex

$$P_{ij} = \left\{ v \,\middle|\, \sum_{t \in T} v^{(t)} = s_{iJ}, v \geq 0 \right\},$$

and $\theta[n] > 0$ is an appropriate step size. In other words, $p_{iJ}[n+1]$ is the Euclidean projection of $p_{iJ}[n] + \theta[n] g_{iJ}[n]$ onto the feasible set P_{iJ}. This projection can be done using the method described by [Lun et al., 2005c]. The value of $p_{iJ}[n+1]$ is sent over hyperarc (i, J).

4 At the end of each subgradient iteration, nodes recover a primal solution $\{\tilde{x}[n]\}$ by setting

$$\tilde{x}[n] := \sum_{l=1}^{n} \mu_l[n] \hat{x}[l], \tag{5.14}$$

where $\{\mu_l[n]\}_{l=1,\ldots,n}$ is an appropriate sequence of convex combination weights.

5 Finally, we need to compute the current coding subgraph $z[n]$ based on the recovered primal solution $\tilde{x}[n]$. Therefore, for each $i \in \mathcal{N}$, we set

$$z_{iJ^{(i)}_m}[n] := \max_{t \in T} \left\{ \sum_{k \in J^{(i)}_{M_i} \setminus J^{(i)}_{m-1}} \tilde{x}^{(t)}_{ik}[n] \right\} - \sum_{m'=m+1}^{M_i} z_{iJ^{(i)}_{m'}}[n]$$

recursively, starting from $m = M_i$ and proceeding through to $m = 1$.

6 Steps (2) to (5) are repeated until the primal solution has converged.

We see that the subgradient method is indeed a distributed algorithm, though it does, unfortunately, operate in synchronous rounds. We expect that, in practice, this synchronicity can be slightly relaxed.

For the choice of $\{\theta[n]\}$ and $\{\mu_l[n]\}$, we first define

$$\gamma_{ln} := \frac{\mu_l[n]}{\theta[n]}, \quad l = 1, \ldots, n, \, n = 0, 1, \ldots,$$

and

$$\Delta\gamma_n^{\max} := \max_{l=2,\ldots,n}\{\gamma_{ln} - \gamma_{(l-1)n}\}.$$

Now, if the step sizes $\{\theta[n]\}$ and convex combination weights $\{\mu_l[n]\}$ are chosen such that

1 $\gamma_{ln} \geq \gamma_{(l-1)n}$ for all $l = 2, \ldots, n$ and $n = 0, 1, \ldots$,

2 $\Delta\gamma_n^{\max} \to 0$ as $n \to \infty$, and

3 $\gamma_{1n} \to 0$ as $n \to \infty$ and $\gamma_{nn} \leq \delta$ for all $n = 0, 1, \ldots$ for some $\delta > 0$,

then the subgradient method converges to an optimal solution.

The required conditions on the step sizes and convex combination weights are satisfied by the following choices ([Sherali and Choi, 1996, Corollaries 2–4]):

1 step sizes $\{\theta[n]\}$ such that $\theta[n] > 0$, $\lim_{n\to 0}\theta[n] = 0$, $\sum_{n=1}^{\infty}\theta_n = \infty$, and convex combination weights $\{\mu_l[n]\}$ given by $\mu_l[n] = \theta[l]/\sum_{k=1}^{n}\theta[k]$ for all $l = 1, \ldots, n$, $n = 0, 1, \ldots$;

2 step sizes $\{\theta[n]\}$ given by $\theta[n] = a/(b + cn)$ for all $n = 0, 1, \ldots$, where $a > 0$, $b \geq 0$ and $c > 0$, and convex combination weights $\{\mu_l[n]\}$ given by $\mu_l[n] = 1/n$ for all $l = 1, \ldots, n$, $n = 0, 1, \ldots$; and

3 step sizes $\{\theta[n]\}$ given by $\theta[n] = n^{-\alpha}$ for all $n = 0, 1, \ldots$, where $0 < \alpha < 1$, and convex combination weights $\{\mu_l[n]\}$ given by $\mu_l[n] = 1/n$ for all $l = 1, \ldots, n$, $n = 0, 1, \ldots$.

Moreover, for all three choices, we have $\mu_l[n+1]/\mu_l[n]$ independent of l for all n, so primal iterates can be computed iteratively using

$$\begin{aligned}\tilde{x}[n] &= \sum_{l=1}^{n}\mu_l[n]\hat{x}[l] \\ &= \sum_{l=1}^{n-1}\mu_l[n]\hat{x}[l] + \mu_n[n]\hat{x}[n] \\ &= \phi[n-1]\tilde{x}[n-1] + \mu_n[n]\hat{x}[n],\end{aligned}$$

where $\phi[n] := \mu_l[n+1]/\mu_l[n]$.

Primal-dual method

For the primal-dual method, we assume that the cost functions f_{iJ} are strictly convex, as this guarantees a unique optimal solution to problem (5.7). We present the algorithm for the lossless case, with the understanding that it can be straightforwardly extended to the lossy case. Thus, the optimization problem we address is

$$\begin{aligned} &\text{minimize} \sum_{(i,J)\in\mathcal{A}} f_{iJ}(z'_{iJ}) \\ &\text{subject to } x^{(t)} \in F^{(t)}, \qquad \forall\, t \in T, \end{aligned} \tag{5.15}$$

where

$$z'_{iJ} := \left(\sum_{t\in T} \left(\sum_{j\in J} x^{(t)}_{iJj} \right)^n \right)^{1/n}.$$

Let $(y)^+_a$ denote the following function of y:

$$(y)^+_a = \begin{cases} y & \text{if } a > 0, \\ \max\{y, 0\} & \text{if } a \le 0. \end{cases}$$

To solve problem (5.15) in a distributed fashion, consider the following primal-dual algorithm:

$$\dot{x}^{(t)}_{iJj} = -k^{(t)}_{iJj}(x^{(t)}_{iJj}) \left(\frac{\partial f_{iJ}(z'_{iJ})}{\partial x^{(t)}_{iJj}} + q^{(t)}_{ij} - \lambda^{(t)}_{iJj} \right), \tag{5.16}$$

$$\dot{p}^{(t)}_i = h^{(t)}_i(p^{(t)}_i)(y^{(t)}_i - \sigma^{(t)}_i), \tag{5.17}$$

$$\dot{\lambda}^{(t)}_{iJj} = m^{(t)}_{iJj}(\lambda^{(t)}_{iJj}) \left(-x^{(t)}_{iJj} \right)^+_{\lambda^{(t)}_{iJj}}, \tag{5.18}$$

where

$$\begin{aligned} q^{(t)}_{ij} &:= p^{(t)}_i - p^{(t)}_j, \\ y^{(t)}_i &:= \sum_{\{J|(i,J)\in\mathcal{A}\}} \sum_{j\in J} x^{(t)}_{iJj} - \sum_{\{j|(j,I)\in\mathcal{A}, i\in I\}} x^{(t)}_{jIi}, \end{aligned}$$

and $k^{(t)}_{iJj}(x^{(t)}_{iJj}) > 0$, $h^{(t)}_i(p^{(t)}_i) > 0$, and $m^{(t)}_{iJj}(\lambda^{(t)}_{iJj}) > 0$ are non-decreasing continuous functions of $x^{(t)}_{iJj}$, $p^{(t)}_i$, and $\lambda^{(t)}_{iJj}$ respectively.

It can be shown that the above primal-dual algorithm is globally, asymptotically stable (see [Lun et al., 2005c]). Such stability of the algorithm implies that no matter what the initial choice of (x, p) is, the primal-dual algorithm will converge to the unique solution of problem (5.15). We have to choose λ, however, with non-negative entries as the initial choice.

We associate a processor with each node. We assume that the processor for node i keeps track of the variables $\{p^{(t)}_i\}_{t\in T}$, $\{\lambda^{(t)}_{iJj}\}_{t\in T}$, and $\{x^{(t)}_{iJj}\}_{t\in T}$. With such an assignment of variables to processors, the algorithm can be shown to be distributed in the sense that a node exchanges information only with its neighbors at every iteration of the primal-dual algorithm.

In implementing the primal-dual algorithm, we must bear the following points in mind.

- The primal-dual algorithm in (5.16)–(5.18) is a continuous time algorithm. To discretize the algorithm, we consider time steps $m = 1, 2, \ldots$ and replace the derivatives by differences:

$$\begin{aligned} x_{iJj}^{(t)}[m+1] =& x_{iJj}^{(t)}[m] \\ & - \alpha_{iJj}^{(t)}[m] \left(\frac{\partial f_{iJ}(z'_{iJ}[m])}{\partial x_{iJj}^{(t)}[m]} + q_{ij}^{(t)}[m] - \lambda_{iJj}^{(t)}[m] \right), \\ p_i^{(t)}[m+1] =& p_i^{(t)}[m] + \beta_i^{(t)}[m](y_i^{(t)}[m] - \sigma_i^{(t)}), \\ \lambda_{iJj}^{(t)}[m+1] =& \lambda_{iJj}^{(t)}[m] + \gamma_{iJj}^{(t)}[m] \left(-x_{iJj}^{(t)}[m] \right)^+_{\lambda_{iJj}^{(t)}[m]}, \end{aligned}$$

  where

$$\begin{aligned} q_{ij}^{(t)}[m] &:= p_i^{(t)}[m] - p_j^{(t)}[m], \\ y_i^{(t)}[m] &:= \sum_{\{J|(i,J)\in\mathcal{A}\}} \sum_{j\in J} x_{iJj}^{(t)}[m] - \sum_{\{j|(j,I)\in\mathcal{A}, i\in I\}} x_{jIi}^{(t)}[m], \end{aligned}$$

  and $\alpha_{iJj}^{(t)}[m] > 0$, $\beta_i^{(t)}[m] > 0$, and $\gamma_{iJj}^{(t)}[m] > 0$ can be thought of as step sizes.

- While the algorithm is guaranteed to converge to the optimal solution, the value of the variables at any time instant m is not necessarily a feasible solution. A start-up time is required before a feasible solution is computed.

Bandwidth-limited scenarios and medium access control

While constraints on feasible coding subgraphs are not needed in the energy-limited case, they are central to operation in bandwidth-limited scenarios, *e.g.* optimizing throughput or congestion. The set of feasible vectors of instantaneous and average rates of hyperarcs in $\mathcal{A}$ is determined by physical layer modulation and coding, channel characteristics, and constraints on transmit power. Given the modulation, coding and channel characteristics, medium access control allocates channels or power among interfering transmitters, determining the realizable coding subgraphs.

The number of possible coding subgraphs grows exponentially with the number of nodes and power levels. Although obviously inefficient subgraphs can be eliminated from consideration, optimal medium access in the bandwidth-limited case is generally very complex, even for centralized methods. The only known way to guarantee an optimal solution in general is to specify all feasible subgraphs and optimize over this set. Since this is computationally prohibitive, heuristics are used to reduce the number of subgraphs considered.

To illustrate two existing approaches, we consider the problem of supporting a set $\mathcal{C}$ of multicast sessions on a network, where each session $c \in \mathcal{C}$ consists of a source node $a_c \in \mathcal{N}$ at which exogenous data arrives with average rate R_c and is required to be multicast to a set $T_c \subset \mathcal{N}$ of sinks.

Iterative optimization. One approach, due to [Jain et al., 2003; Wu et al., 2005], starts with a set $\mathcal{Z}$ of K feasible coding subgraphs $z^{(1)}, \ldots, z^{(K)}$, called *elementary capacity graphs*, or ECGs for short. Each ECG is formed by adding unit rate hyperarcs using the following random greedy procedure. A typical communication range R and interference range $R' > R$ are chosen. At each step, a node i is chosen randomly from among those not within distance R' of any end nodes of hyperarcs previously added to the ECG. Let J be the set of nodes within distance R of i but not within distance R' of previously added transmitters. The hyperarc (i, J) is added if power can feasibly be allocated to support unit rate on (i, J) and each previously added hyperarc. This is repeated until no further hyperarcs can be added.

Given $\mathcal{Z} = \{z^{(k)}\}$, we solve the following linear optimization problem.

$$\begin{aligned}
\text{minimize } & \sum_{k=1}^{K} \lambda_k \\
\text{subject to } & \lambda_k \geq 0, \qquad \forall\, k = 1, \ldots, K, \\
& \sum_{j \in J} x_{iJj}^{(tc)} \leq y_{iJ}^{(c)}, \qquad \forall\, (i, J) \in \mathcal{A}, c \in \mathcal{C}\,, t \in T_c, \\
& \sum_{c \in \mathcal{C}} y_{iJ}^{(c)} \leq \sum_{k=1}^{K} \lambda_k z_{iJ}^{(k)}, \qquad \forall\, (i, J) \in \mathcal{A}, \\
& x^{(tc)} \in F^{(tc)}, \qquad \forall\, c \in \mathcal{C}, t \in T_c.
\end{aligned} \tag{5.19}$$

where $F^{(tc)}$ is the bounded polyhedron of points $x^{(tc)}$ forming a flow solution of rate R_c from source node a_c to sink t of session c.

If the solution satisfies $\sum_k \lambda_k \leq 1$, then variables λ_k specify a valid time-sharing among the ECGs in $\mathcal{Z}$ that supports the connection requirements. Otherwise, we carry out an iterative algorithm that alternates between heuristically modifying the set $\mathcal{Z}$ and solving the linear optimization problem (5.19). The set

$\mathcal{Z}$ is modified as follows. For some large integer Q, say 200, the new set $\mathcal{Z}$ is initialized with $\lceil \lambda_k Q \rceil$ copies of ECG $z^{(k)}$, for $k = 1, \ldots, K$. Based on the calculated flows $\{x^{(tc)}\}$, the algorithm removes from these ECGs any hyperarcs that are not needed, and shrinks the destination sets of remaining hyperarcs where possible. It then tries to remove one ECG at a time from $\mathcal{Z}$ by shifting all of its remaining hyperarcs into other ECGs. Each ECG removed reduces the objective function $\sum_k \lambda_k$ by $1/Q$. The iterative algorithm produces a monotonically non-increasing sequence of objective function values, and is carried out until the objective function cannot be further reduced. For more details, see [Jain et al., 2003; Wu et al., 2005].

Dynamic operation. Another approach, due to [Ho and Viswanathan, 2005], dynamically controls medium access, packet scheduling, routing and network coding, without using knowledge of long-term average rates. This algorithm extends to multicast a *back pressure* approach for multi-commodity flow in which routing and flow prioritization are locally determined by gradients in packet queue length. One difference is that the multicast network coding algorithm uses virtual queues to keep track of information intended for each sink. All control decisions are locally made, except for medium access control among interfering transmitters, which is guided by the virtual queue lengths. If all feasible subgraphs are considered, the algorithm stably supports any set of source rates stabilizable with intra-session network coding.

More precisely, we consider a dynamic network model with time slots of length τ_0. We assume that channel state is described by a vector $\underline{S}(\tau)$ that is constant over each time slot $[\tau, \tau + \tau_0)$, takes values from a finite set and is ergodic. Control decisions are made at most once a slot. For simplicity, we assume fixed length packets and link transmission rates that are restricted to integer multiples of the packet-length/time-slot quotient, *i.e.* an integer number of packets can be transmitted in each slot. In each time slot $[\tau, \tau + \tau_0)$, the coding subgraph z takes a value $z(\tau)$, determined by the medium access control policy, from a set $Z(\underline{S}(\tau))$ of feasible subgraphs that depends on the channel state vector $\underline{S}(\tau)$.

Each node i maintains a virtual queue Q_i^{ct} for each sink t of each session c, whose length is denoted U_i^{ct}. The queues are called virtual as the same actual data may be associated with more than one virtual queue. Exogenous session c data arriving at source node a_c undergoes random linear coding to produce coded packets at $(1 + \epsilon)$ times the exogenous arrival rate, which are then associated with queues $Q_{a_c}^{ct}, t \in T_c$. In each time slot $[\tau, \tau + \tau_0)$, the following are carried out:

- Scheduling: For each hyperarc (i, J), one session

$$c^*_{iJ} = \arg\max_{c \in \mathcal{C}} \left\{ \sum_{t \in T_c} \max\left(\max_{b \in J} \left(U_i^{ct} - U_b^{ct} \right), 0 \right) \right\}$$

  is chosen, and link weights

$$w^*_{iJ} = \sum_{t \in T_{c^*_{iJ}}} \max\left(\max_{b \in J} \left(U_i^{c^*_{iJ}t} - U_b^{c^*_{iJ}t} \right),\ 0 \right)$$

  are defined.

- Medium access control: The state $\underline{S}(\tau)$ is observed, and a coding subgraph

$$z(\tau) = \arg\max_{z \in Z(\underline{S}(\tau))} \sum_{(i,J) \in \mathcal{A}} w^*_{iJ} z_{iJ} \tag{5.20}$$

  is chosen.

- Network coding: For each hyperarc (i, J), a random linear combination of data corresponding to each (session, sink) pair $(c^*_{iJ}, t \in T_{c^*_{iJ}})$ for which $\max_{b \in J} \left(U_i^{c^*_{iJ}t} - U_b^{c^*_{iJ}t} \right) > 0$ is sent at the rate $z_{iJ}(\tau)$ determined by the medium access control, decreasing $U_i^{c^*_{iJ}t}$ by an amount $\min \left\{ U_i^{c^*_{iJ}t}(\tau)\ ,\ \tau_0 z_{iJ}(\tau) \right\}$ and increasing $U_d^{c^*_{iJ}t}$ by the same amount, where $d = \arg\max_{b \in J} \left(U_i^{c^*_{iJ}t} - U_b^{c^*_{iJ}t} \right)$.

If the optimizations in the algorithm are done exactly, we have the following theorem.

THEOREM 5.3 *Let Λ be the set of all exogenous arrival rate vectors (R_c) such that the connection requirements can be stably supported by some control algorithm with full knowledge of future events. If $((1+\epsilon)R_c)$ is strictly interior to Λ, the probability that not all sinks are able to decode their respective information decreases exponentially in the length of the code.*

In practice, the medium access control optimization (5.20) can be done heuristically using a greedy approach similar to that in the static case, but with the added guidance of weights w^*_{iJ} for prioritization among candidate hyperarcs (i, J).

5. Further directions and results

Network coding has come a long way in the last few years and it has been extended in many different ways. While we have focused in the previous sections on random coding and optimization problems for network coding in the wireless multicast scenario, it seems appropriate to shine a light also on other applications and research directions that have been inspired by this new and exciting techniques. Some of these advances reside most naturally in wireline networks and we will describe them in this context when so indicated. Nevertheless, almost all results can be without much effort be translated into a wireless context.

Network code construction

The multicast case. Random network coding is an extremely useful tool for multicast operation. However, any randomly chosen network solution does not come with *guarantees* or a certificate indicating that a "good" solution was found. Moreover, it may be possible to significantly reduce the complexity of the network coding itself by finding solution that operate in finite fields of minimal allowable size. Clearly, a network code that can operate over a binary field, *i.e.* utilizing only XOR operations on bits would be far easier to implement than finite field arithmetic over, say, $GF(2^{12})$. An efficient, *i.e.* running in time polynomial in the description of the network, algorithm to find network coding solutions with guaranteed performance was given by [Jaggi et al., 2005b]. The algorithm operates in a centrally controlled manner by first identifying a set of disjoint paths between the source and each of the individual receivers. In subsequent steps a network coding solution is iteratively grown starting from the source node to each of the intended receivers by considering one edge at the time. The encoding function for each edge is here chosen so as to guarantee that the information flowing on any cut in the network is as rich and different as possible.[1] Since all we have to guarantee in order to find a valid network coding solution is that random processes of maximal entropy have to reach the receivers in order to satisfy the multicast requirements this approach will be successful. This realization also led to the notion of a network code as a "linear dispersion" by [Li et al., 2005].[2] A distinguishing feature of the iterative process is that it is possible to keep the field size at its minimum (which often means binary operations). The construction algorithm, originally formulated for acyclic graphs, was later extended by [Erez and Feder, 2005] to the case of graphs with cycles so that we can now avail of efficient algorithms for practically all interesting scenarios. In particular, the extension to constructing wireless network coding solutions with guaranteed performance is straight forward once a set of source/sink flows is known for each receiver. For this purpose the algorithm of Section 4 can be effectively combined with the iterative encoding function choice of [Jaggi et al., 2005b]. Nevertheless, we want to emphasize that the added complexity of constructing a highly efficient network code is only justified for very stable and well known network topologies.[3] Wireless networks would typically operate very efficiently with random network coding.

The construction of network codes has spawned further research on low complexity solutions. A natural question is the effect of limiting the number of nodes or edges in a network that are capable of performing network coding operations (see [Bhattad et al., 2005]). However, it appears that these considerations are far more important in a wireline network where one might be confronted with, *e.g.* converting optical signals into electrical signals in order to perform network coding. The added cost of network coding in a wireless network are likely to be relatively minor when compared to the already necessary modulation and demodulation of RF signals.

The non-multicast case. Constructing codes for the non-multicast case is considerably more difficult than the corresponding task in the multicast setup. The main problem we are now facing is that network coding solutions have to make sure that the right information is delivered to each node. In particular, this renders random network coding as an almost useless tool. Indeed, at present we do not know of an efficient, structured approach to solve the network coding problem optimally. For linear network coding the algebraic characterization of valid solutions given by [Koetter and Médard, 2003] leads to an algorithm that, in principle, can decide if a given network problem has a solution or not. However, the running time of this algorithm is not bounded by a polynomial function in the size of the network.[4] The situation is exacerbated by the fact that linear network coding does not achieve the capacity of networks with arbitrary demands (see [Dougherty et al., 2005]). We find ourselves in a situation where, in practice, we have to resort to various ad-hoc solutions that can be used to demonstrate network coding advantages.

Ad hoc approaches to network coding

Finding optimal solutions to network coding problems involving more than one multicast session is a difficult problem and, at present, there is no practical approach known to achieve the *optimal* throughput in a network. Nevertheless, it is clear that this problem is of prime importance and a number of suboptimal solutions have been investigated. In many of these ad hoc schemes network coding operations are restricted to binary addition, which seems to realize a large portion of the possible gains. Opportunistic network coding in a wireless context has been described by [Katti et al., 2005] in a wireless 802.11 environment. This approach to network coding in a wireless network is motivated by the idea to start from a given wireless system and to opportunistically identify and exploit opportunities for improvement, such as the one mentioned in Section 1. The scheme proposed by [Katti et al., 2005] is characterized by an assumed knowledge of a node of a list of packets that its direct neighbors have. Moreover, network coding is used only if combining packets can give some immediate advantage for the reception of the direct neighbors of a node. The scheme was implemented in an 802.11 network and shows surprisingly large throughput advantages. For further details and for a quantification of the gains, see [Katti et al., 2005]. A fairly similar thinking has been applied by [Hausl et al., 2005] in the context of relay aided up/downlink improvement in a cellular network. Here a relay provides a sum of an uplink and a downlink package to both the mobile and the base station. Owing to the ability of mobile and base station to subtract its own packet from the relay signal, both can improve their reception link quality in a turbo coded system setup; see [Hausl et al., 2005] for further details.

Security aspects

While the main thrust of network coding research aims at increasing bandwidth and/or energy efficiency it also can greatly increase security of transmission. The main idea here is that in random network coded systems transmissions are typically mixtures of many portions of a set of data. Thus no individual packet reveals any information about the individual packets contributing to its makeup. Moreover, a wire-tapper or attacker cannot undo the random linear combination of the observed packets unless he/she has opportunity to observe enough information to retrieve the entire data. In other words, there exists a threshold behavior to the secrecy of the packets which is similar to more traditional wire-tapper or secret sharing schemes. The interested reader is referred to [Ho et al., 2004] for an application of network coding to Byzantine security and to [Jaggi et al., 2005a] for the situation where an attacker's resources are limited. Further references include [Cai and Yeung, 2002; Feldman et al., 2004; Bhattad and Narayanan, 2005].

Notes

1. Loosely speaking we try to maximize the degrees of freedom on each cut
2. Indeed, Jaggi et al.'s algorithm readily solves the problem of constructing a "linear dispersion" as well.
3. Network coding has, *e.g.* been proposed in a VLSI context to distribute multicast signals on a chip. Such a extremely stable environment seems predestined for a very careful construction of the encoding function.
4. In fact it has been shown by [Rasala Lehman and Lehman, 2003] that this question is NP-hard.

References

Ahlswede, Rudolf, Cai, Ning, Li, Shuo-Yen Robert, and Yeung, Raymond W. (2000). Network information flow. *IEEE Trans. Inform. Theory*, 46(4): 1204–1216.

Ahluwalia, Ashwinder, Modiano, Eytan, and Shu, Li (2002). On the complexity and distributed construction of energy-efficient broadcast trees in static ad hoc wireless networks. In *Proc. 2002 Conference on Information Sciences and Systems (CISS 2002)*.

Bertsekas, Dimitri P. (1980). A class of optimal routing algorithms for communication networks. In *Proc. 5th International Conference on Computers and Communication (ICCC '80)*, pages 71–76.

Bertsekas, Dimitri P., Gafni, Eli M., and Gallager, Robert G. (1984). Second derivative algorithms for minimum delay distributed routing in networks. *IEEE Trans. Commun.*, 32(8):911–919.

Bhattad, Kapil and Narayanan, Krishna R. (2005). Weakly secure network coding. In *Proc. WINMEE, RAWNET and NETCOD 2005 Workshops*.

Bhattad, Kapil, Ratnakar, Niranjan, Koetter, Ralf, and Narayanan, Krishna R. (2005). Minimal network coding for multicast. In *Proc. 2005 IEEE International Symposium on Information Theory (ISIT 2005)*, pages 1730–1734.

Cai, Ning and Yeung, Raymond W. (2002). Secure network coding. In *Proc. 2002 IEEE International Symposium on Information Theory (ISIT 2002)*, page 323.

Cruz, R. L. and Santhanam, Arvind V. (2003). Optimal routing, link scheduling and power control in multi-hop wireless networks. In *Proc. IEEE Infocom 2003*, volume 1, pages 702–711.

Deb, Supratim and Srikant, R. (2004). Congestion control for fair resource allocation in networks with multicast flows. *IEEE/ACM Trans. Networking*, 12(2):274–285.

Dougherty, Randall, Freiling, Christopher, and Zeger, Kenneth (2005). Insufficiency of linear coding in network information flow. In *Proc. 2005 IEEE International Symposium on Information Theory (ISIT 2005)*, pages 264–267.

Erez, Elona and Feder, Meir (2005). Convolutional network codes for cyclic networks. In *Proc. WINMEE, RAWNET and NETCOD 2005 Workshops*.

Feldman, Jon, Malkin, Tal, Servedio, Rocco A., and Stein, Cliff (2004). On the capacity of secure network coding. In *Proc. 42nd Annual Allerton Conference on Communication, Control, and Computing*.

Hausl, Christoph, Schreckenbach, Frank, Oikonomidis, Ioannis, and Bauch, Gerhard (2005). Iterative network and channel decoding on a tanner graph. In *Proc. 43rd Annual Allerton Conference on Communication, Control, and Computing*.

Ho, Tracey, Leong, Ben, Koetter, Ralf, Médard, Muriel, Effros, Michelle, and Karger, David R. (2004). Byzantine modification detection in multicast networks using randomized network coding. In *Proc. 2004 IEEE International Symposium on Information Theory (ISIT 2004)*, page 144, Chicago, IL.

Ho, Tracey, Médard, Muriel, Shi, Jun, Effros, Michelle, and Karger, David R. (2003). On randomized network coding. In *Proc. 41st Annual Allerton Conference on Communication, Control, and Computing*.

Ho, Tracey and Viswanathan, Harish (2005). Dynamic algorithms for multicast with intra-session network coding. In *Proc. 43rd Annual Allerton Conference on Communication, Control, and Computing*.

Jaggi, Sidharth, Langberg, Michael, Ho, Tracey, and Effros, Michelle (2005a). Correction of adversarial errors in networks. In *Proc. 2005 IEEE International Symposium on Information Theory (ISIT 2005)*, pages 1455–1459.

Jaggi, Sidharth, Sanders, Peter, Chou, Philip A., Effros, Michelle, Egner, Sebastian, Jain, Kamal, and Tolhuizen, Ludo M. G. M. (2005b). Polynomial time algorithms for multicast network code construction. *IEEE Trans. Inform. Theory*, 51(6):1973–1982.

Jain, Kamal, Padhye, Jitendra, Padmanabhan, Venkata N., and Qiu, Lili (2003). Impact of interference on multi-hop wireless network performance. In *MobiCom '03: Proc. 9th Annual International Conference on Mobile Computing and Networking*, pages 66–80.

Johansson, Mikael, Xiao, Lin, and Boyd, Stephen (2003). Simultaneous routing and power allocation in CDMA wireless data networks. In *Proc. 2003 IEEE International Conference on Communications (ICC 2003)*, volume 1, pages 51–55.

Katti, Sachin, Katabi, Dina, Hu, Wenjun, Rahul, Hariharan, and Médard, Muriel (2005). The importance of being opportunistic: Practical network coding for wireless environments. In *Proc. 43rd Annual Allerton Conference on Communication, Control, and Computing*.

Kodialam, Murali and Nandagopal, Thyaga (2005). Characterizing achievable rates in multi-hop wireless mesh networks with orthogonal channels. *IEEE/ACM Trans. Networking*, 13(4):868–880.

Koetter, Ralf and Médard, Muriel (2003). An algebraic approach to network coding. *IEEE/ACM Trans. Networking*, 11(5):782–795.

Li, Shuo-Yen Robert, Cai, Ning, and Yeung, Raymond W. (2005). On theory of linear network coding. In *Proc. 2005 IEEE International Symposium on Information Theory (ISIT 2005)*, pages 273–277.

Liang, Weifa (2002). Constructing minimum-energy broadcast trees in wireless ad hoc networks. In *Proc. 3rd ACM International Symposium on Mobile Ad Hoc Networking & Computing (MOBIHOC '02)*, pages 112–122.

Lun, Desmond S., Médard, Muriel, and Koetter, Ralf (2005a). Efficient operation of wireless packet networks using network coding. In *Proc. International Workshop on Convergent Technologies (IWCT) 2005*. Invited paper.

Lun, Desmond S., Médard, Muriel, Koetter, Ralf, and Effros, Michelle (2005b). Further results on coding for reliable communication over packet networks. In *Proc. 2005 IEEE International Symposium on Information Theory (ISIT 2005)*, pages 1848–1852.

Lun, Desmond S., Ratnakar, Niranjan, Koetter, Ralf, Médard, Muriel, Ahmed, Ebad, and Lee, Hyunjoo (2005c). Achieving minimum-cost multicast:

A decentralized approach based on network coding. In *Proc. IEEE Infocom 2005*, volume 3, pages 1608–1617, Miami, FL.

Motwani, Rajeev and Raghavan, Prabhakar (1995). *Randomized Algorithms*. Cambridge University Press, Cambridge.

Ouorou, A., Mahey, P., and Vial, J.-Ph. (2000). A survey of algorithms for convex multicommodity flow problems. *Manage. Sci.*, 46(1):126–147.

Rasala Lehman, April and Lehman, Eric (2003). Complexity classification of network information flow problems. In *Proc. 41st Annual Allerton Conference on Communication, Control, and Computing*.

Sherali, Hanif D. and Choi, Gyunghyun (1996). Recovery of primal solutions when using subgradient optimization methods to solve Lagrangian duals of linear programs. *Oper. Res. Lett.*, 19:105–113.

Wieselthier, Jeffrey E., Nguyen, Gam D., and Ephremides, Anthony (2002). Energy-efficient broadcast and multicast trees in wireless networks. *Mobile Networks and Applications*, 7:481–492.

Wu, Yunnan, Chou, Philip A., Zhang, Qian, Jain, Kamal, Zhu, Wenwu, and Kung, Sun-Yuan (2005). Network planning in wireless ad hoc networks: A cross-layer approach. *IEEE J. Select. Areas Commun.*, 23(1):136–150.

Xiao, Lin, Johansson, Mikael, and Boyd, Stephen (2004). Simultaneous routing and resource allocation via dual decomposition. *IEEE Trans. Commun.*, 52(7):1136–1144.