

Lecture 8

Algorithms for multicast network code design

As it was shown in the previous lecture, a network coding solution does exist for a directed acyclic graph with unit capacities, h sources, and a set of terminals T as long as:

- The multicast property is satisfied.
- The field size is greater than $|T|^1$.

The overhead associated with network coding is attributed to two sources:

1. (Computational overhead): due to linear coding operations at intermediate nodes and decoding operations at the terminal nodes.
3. (Communication overhead): because of the coefficient vectors sent with each data packet.

Apparently, both the above mentioned sources of overhead depend on the *field size*. Therefore, we aim in this lecture to show how network codes can be designed efficiently with the minimum amount of overhead.

The design of network codes can be either *centralized* or *distributed*. In the case of *centralized* design, the network graph $G(V,E)$, the set of source nodes, and the set of terminal nodes should be provided as input. On the other hand, the *distributed* design requires that each node only knows its neighbors. Decentralized (*distributed*) network coding is favored over centralized coding in large networks or networks with changing topologies and/or dynamically varying connections as it does not require a central entity to coordinate code assignment.

Random Linear Coding

In *Random linear coding*, the coding process operate, in a distributed fashion, over a large field of size “ q ” such that the network code coefficients, i.e. α ’s and β ’s, are chosen from $GF(q)$ independently and uniformly at random, i.e. $\Pr(\beta_{e'_i e} = x) = \frac{1}{q}$. We shall show that with high probability and for large enough q , the demands of the terminals T will be met (if q is large enough, this probability is ≈ 1).

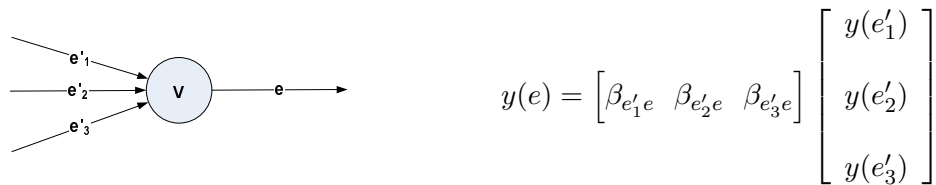


Figure 1: Node v chooses the coefficients $\beta_{e'_1 e}$, $\beta_{e'_2 e}$, and $\beta_{e'_3 e}$ at random from $GF(q)$

¹Note that this condition is sufficient, but not necessary.

Assuming that \mathbf{A} , \mathbf{F} , and \mathbf{B}_i^T , $T_i \in T$, are known in terms of indeterminates, the transfer function from the sources to terminal T_i is:

$$\mathbf{M}_i = \mathbf{A}(\mathbf{I} - \mathbf{F})^{-1}\mathbf{B}_i^T.$$

The following condition should be satisfied for the network code to be valid:

$$\det(\mathbf{M}_i) \neq 0 \forall i \equiv \prod_{i=1}^{|T|} \det(\mathbf{M}_i) \neq 0$$

Recall:

$$\det(\mathbf{M}_i) = \pm \det \left(\begin{bmatrix} \mathbf{A} & 0 \\ (\mathbf{I} - \mathbf{F}) & \mathbf{B}_i^T \end{bmatrix} \right)$$

$\prod_{i=1}^{|T|} \det(\mathbf{M}_i)$ is a polynomial in $\{\alpha_{l,e}\}, \{\beta_{e',e}\}, \{\varepsilon_{e,i}\}$ that we denote,

$$f(\{\alpha_{l,e}\}, \{\beta_{e',e}\}, \{\varepsilon_{e,i}\})$$

As mentioned earlier, the coefficients are independently and randomly chosen, so the question we seek to answer is “*Is it guaranteed that using any random choice of coefficients, the function f does not evaluate to zero?*” To answer this question, let us first simplify the function f . \mathbf{B}_i^T is not critical to ensuring that $\det(\mathbf{A}(\mathbf{I} - \mathbf{F})^{-1}\mathbf{B}_i^T) \neq 0$.

Let $\vec{e}_i = [0 \dots 1 \dots 0]_{1 \times h}$, i.e., the i^{th} element is 1 and all other elements are 0’s. Then, the term $\mathbf{A}(\mathbf{I} - \mathbf{F})^{-1}\mathbf{B}_i^T$ can be written as:

$$\mathbf{A}(\mathbf{I} - \mathbf{F})^{-1} \begin{bmatrix} 0 \\ \vdots \\ \vec{e}_1 \\ \vdots \\ 0 \\ \vec{e}_2 \\ \vdots \\ \vec{e}_3 \\ \vdots \end{bmatrix}_{|E| \times h} \quad [\mathbf{B}'_i^T]_{h \times h} = \mathbf{A}(\mathbf{I} - \mathbf{F})^{-1} \mathbf{N} \mathbf{B}'_i^T$$

As $\det(\mathbf{B}'_i^T)$ can be chosen to be non-zero by terminal T_i , we only need to consider the polynomial:

$$f(\{\alpha_{l,e}\}, \{\beta_{e',e}\})$$

which has the following two properties:

1. The maximum degree of any variable is $|T|$.

2. The maximum degree of every term² in the polynomial is bounded by $|E| \times |T|$ (since there are at most $|E|$ variables and each can have degree at most $|T|$).

Lemma 1 (Schwartz-Zippel). *Let $f(\alpha_1, \dots, \alpha_\eta)$ be a polynomial of a maximum total degree d over a field \mathbf{F} . Let S be a finite subset of \mathbf{F} and let r_1, \dots, r_η be selected randomly from S . Then,*

$$\Pr[f(r_1, \dots, r_\eta) = 0] \leq \frac{d}{|S|}$$

Proof. *Induction on the number of variables η .* For $\eta = 1$, the polynomial is univariate and it follows from basic algebra that the function has at most d roots.

Let us now consider the general case of η . If f does not depend on α_1 , i.e. $f(\alpha_1, \dots, \alpha_\eta) = f'(\alpha_2, \dots, \alpha_\eta)$, then by induction f' has at most d roots. So we write f as,

$$f(\alpha_1, \dots, \alpha_\eta) = \sum_{i=0}^k \alpha_1^i f_i(\alpha_2, \dots, \alpha_\eta),$$

where $k > 0$ is the maximum degree of the variable α_1 , and the f_i 's are polynomials in $\alpha_2, \dots, \alpha_\eta$ and are not identically zero. Let us define two events, E_1 and E_2 such that:

- E_1 be the event that $f_i(r_2, \dots, r_\eta) = 0 \quad \forall i$.
- E_2 be the event that $f(r_1, \dots, r_\eta) = 0$.

Then,

$$\Pr[E_2, E_1] = \Pr[E_1] \Pr[E_2|E_1] \leq \Pr[E_1] = \Pr[f_i(r_2, \dots, r_\eta) = 0],$$

We have by induction that $\Pr[f_i(r_2, \dots, r_\eta) = 0] = \frac{d-k}{|S|}$, therefore,

$$\Pr[E_2, E_1] \leq \frac{d-k}{|S|}.$$

On the other hand,

$$\Pr[E_2, \bar{E}_1] = \Pr[\bar{E}_1] \Pr[E_2|\bar{E}_1] \leq \Pr[E_2|\bar{E}_1] \leq \frac{k}{|S|}.$$

Therefore,

$$\Pr[E_2] = \Pr[E_2, E_1] + \Pr[E_2, \bar{E}_1] \leq \frac{d-k}{|S|} + \frac{k}{|S|} = \frac{d}{|S|}.$$

which concludes the proof. ■

Lemma 2 shows how we can a tighter bound on the probability $\Pr[f = 0]$ by considering the maximum degree of any variable in f instead of the maximum total degree considered in lemma 1.

²The degree of a term is the sum of exponents of the individual elements in that term. For example, the term $x_2^2 x_3^3$ has a degree of $2 + 3 = 5$.

Lemma 2. Let $f(\alpha_1, \dots, \alpha_\eta)$ be a η -variate non-zero polynomial with the maximum degree of any variable be $\leq d$ (Note that the d in 2 is the maximum degree of any variable whereas d in 1 is the maximum total degree of f). If $\alpha_1, \dots, \alpha_\eta$ are chosen iid uniform from $GF(q)$ such that $q > d$. Then,

$$\Pr[f(r_1, \dots, r_\eta) = 0] \leq 1 - (1 - d/q)^\eta$$

Proof. By induction on number of variables. For the case $\eta = 1$, f is a univariate polynomial with degree at most $d \Rightarrow \Pr[f = 0] \leq \frac{d}{q} = 1 - (1 - \frac{d}{q})$.

Hypothesis: the lemma is true for all polynomials with number of variables at most $\eta - 1$. The function $f(\alpha_1, \dots, \alpha_\eta)$ can be written as:

$$f(\alpha_1, \dots, \alpha_\eta) = \alpha_\eta^{d_1} f_1(\alpha_1, \dots, \alpha_{\eta-1}) + f_2(\alpha_1, \dots, \alpha_\eta),$$

where $d_1 \leq d$. Assume that the variables $\alpha_1, \dots, \alpha_{\eta-1}$ were chosen from $GF(q)$ such that $f(r_1, \dots, r_{\eta-1}, \alpha_\eta)$ is a univariate function in the variable α_η whose maximum degree is d . Let that choice be $r_1, \dots, r_{\eta-1}$. Then,

$$\Pr[f(r_1, \dots, r_{\eta-1}, \alpha_\eta) = 0 | f_1(r_1, \dots, r_{\eta-1})] = \frac{d}{q}.$$

Now,

$$\begin{aligned} \Pr[f = 0] &= \Pr[f = 0, f_1 = 0] + \Pr[f = 0, f_1 \neq 0] \\ &\leq \Pr[f_1 = 0] + \Pr[f_1 \neq 0] \Pr[f = 0 | f_1 \neq 0] \\ &= \Pr[f_1 = 0] + (1 - \Pr[f_1 = 0]) \Pr[f = 0 | f_1 \neq 0] \\ &\leq \Pr[f_1 = 0] + (1 - \Pr[f_1 = 0]) \frac{d}{q} \\ &= \Pr[f_1 = 0] \left(1 - \frac{d}{q}\right) + \frac{d}{q} \\ &\leq \left(1 - \left(1 - \frac{d}{q}\right)^{\eta-1}\right) \left(1 - \frac{d}{q}\right) + \frac{d}{q} \\ &= 1 - \left(1 - \frac{d}{q}\right)^\eta \end{aligned}$$

■

This means that the probability that a randomly chosen set of coefficients in a multicast network code does not work is given as,

$$1 - \left(1 - \frac{|T|}{q}\right)^{|E|}$$

Note that the bound above does not depend on the multicast rate. Moreover this probability can be made arbitrarily small by choosing q large enough. Thus, if we are willing to work with large field sizes, then the probability that a random network code does not work is negligible.

Example: Consider a network with $|E|=2^8$, $q=2^{16}$, and $|T|=1$. Then, the receiver will be able to decode the data (equivalently, the coefficients matrix at the receiver will have full rank) with probability at least $(1 - 2^{-16})^{2^8} = 0.9961$.

Practical Implementations of Random Network Codes

Up to this point, we restricted our analysis by the following assumptions:

1. The nodes choose coefficients at random and the information about these choices is somehow communicated to the terminals.
2. *Information travels in a synchronous and delay free manner.* In real networks, however, information travels asynchronously in form of packets that are subject to delays and losses.
3. *The network graph is known.* But in real networks it might be difficult to obtain centralized knowledge about the network.
4. *A directed acyclic graph.* However, in real networks cycles are frequent.

In this lecture we will consider an implementation of random network codes that considers the practical aspects of real networks.

Basically, each edge $e \in E$ going out of node v carries a symbol $y(e)$ which is a linear combination of the symbols $y(e'_i)$ that are carried on the edges e'_i entering v , i.e. $y(e) = \sum_{e':out(e')=v} m_e(e')y(e')$. The vector $\mathbf{m}(e) = [m_e(e')]_{e':out(e')=v}$ is called the *local encoding vector* and it represents the encoding function at node v along edge e .

If v is the a source node, then we introduce artificial edges e'_1, \dots, e'_h entering v , and carrying the h symbols $y(e'_i) = x_i$, $i = 1, \dots, h$. We also introduce the vector $\mathbf{g}(e) = \sum_{e':out(e')=v} m_e(e')\mathbf{g}(e')$ where $\mathbf{g}(e)$ is initialized to the i^{th} unit vector. $\mathbf{g}(e)$ is known as the *global encoding vector*. A receiver t that receives, along its incoming edges e_1, \dots, e_h , the symbols

$$\begin{bmatrix} y(e_1) \\ \vdots \\ y(e_h) \end{bmatrix} = \begin{bmatrix} g_1(e_1) & \dots & g_h(e_1) \\ \vdots & \ddots & \vdots \\ g_1(e_h) & \dots & g_h(e_h) \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_h \end{bmatrix} = G_t \begin{bmatrix} x_1 \\ \vdots \\ x_h \end{bmatrix}$$

can recover the source symbols x_1, \dots, x_h as long as G_t has rank h . This will be true with high probability, as we have seen in lemma 1, if the local encoding vectors are chosen randomly from a finite field that is sufficiently large.

In reality, networks are packetized, meaning that the data travels in form of packets. So symbols can be grouped into packets. In Internet for example, the maximum packet size is around $N = 1400$ symbols (bytes) using $GF(2^8)$. Therefore, a packet can be treated as a vector over $GF(2^8)$. Thus, we write:

$$\begin{bmatrix} \vec{y}(e_1) \\ \vdots \\ \vec{y}(e_h) \end{bmatrix} = G_t \begin{bmatrix} \vec{x}_1 \\ \vdots \\ \vec{x}_h \end{bmatrix}$$

Now the idea is to make the global encoding vectors needed to invert the code at any receiver obtainable from the arriving packets themselves. We can do this by prepending the i^{th} unit vector to the source vector \vec{x}_i , $i = 1, \dots, h$.

$$\begin{bmatrix} \vec{y}(e_1) \\ \vdots \\ \vec{y}(e_h) \end{bmatrix} = G_t \begin{bmatrix} 1 & \dots & 0 & \vec{x}_1 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \dots & 1 & \vec{x}_h \end{bmatrix}$$

The algorithm will be discussed in detail in the next lecture.