Simplified Random Network Codes for Multicast Networks

by

Anna H. Lee

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degrees of

Bachelor of Science in Electrical Science and Engineering

and

Master of Engineering in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

May 2005 [June 2005]

Author .....
Department of Electrical Engineering and Computer Science
May 19, 2005

Certified by .....
Muriel Médard
Esther and Harold E. Edgerton Associate Professor of Electrical Engineering & Computer Science
Thesis Supervisor

Accepted by .....
Arthur C. Smith
Chairman, Department Committee on Graduate Theses

Simplified Random Network Codes for Multicast Networks

by

Anna H. Lee

Submitted to the Department of Electrical Engineering and Computer Science
on May 19, 2005, in partial fulfillment of the
requirements for the degrees of
Bachelor of Science in Electrical Science and Engineering
and
Master of Engineering in Electrical Engineering and Computer Science

Abstract

Network coding is a method of data transmission across a network which involves coding at intermediate nodes. Network coding is particularly attractive for multicast. Building on the work done on random linear network codes, we develop a constrained, simplified code construction suitable for multicast in wireless networks. We analyze bounds on sufficient code size and code success probability via an algebraic framework for network coding. We also present simulation results that compare generalized random network codes with our code construction. Issues unique to the simplified code are explored and a relaxation of the code to improve code performance is discussed.

Thesis Supervisor: Muriel Médard
Title: Esther and Harold E. Edgerton Associate Professor of Electrical Engineering & Computer Science

Acknowledgments

I would like to thank my advisor Professor Muriel Médard for her guidance and support. I am privileged to have such a talented and dedicated person guide my growth as a researcher. I would also like to thank Todd Coleman for being an invaluable mentor. His enthusiasm and work ethic set the standards for the kind of researcher I hope to be. Finally I would like to thank my family and friends for their unconditional support and understanding.

This research was supported by the National Reconnaissance Office.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The purpose of a multicast network is to simultaneously transmit data from one source to multiple receivers. The *multicast connection problem* concerns the ability to transmit data from a specific source node to a set of receiver nodes at a specified rate. The constraints of the problem are the topology of the network and the capacity of the links in the network. Conventionally data is transmitted across a network by routing, where intermediate nodes only receive and forward data towards their destination. For the multicast problem, [4] showed that higher rates can be achieved if intermediate nodes are allowed to perform coding. It has also been shown that linear codes are sufficient for multicast [5]. An example of a network that requires network coding was first presented by [4], shown in figure 1-1. In this network, we wish to transmit both bits b_1 and b_2 to the two receivers simultaneously. If each link can only transmit one bit at a time, this is impossible by routing; only one of the receivers can receive both bits if nodes can only forward them. A solution exists if network coding is allowed. The interior node sends the sum (modulo 2) of the two bits, and the two receivers are able to recover the second bit by adding (modulo 2) their inputs.

The advantages of network coding, however, are not limited to simple networks such as this one. Reference [1] showed the benefits of randomized coding over routing on randomly generated networks, and the utility of randomized coding in dynamic networks. Network coding is particularly attractive for multicast. Multicast rout-

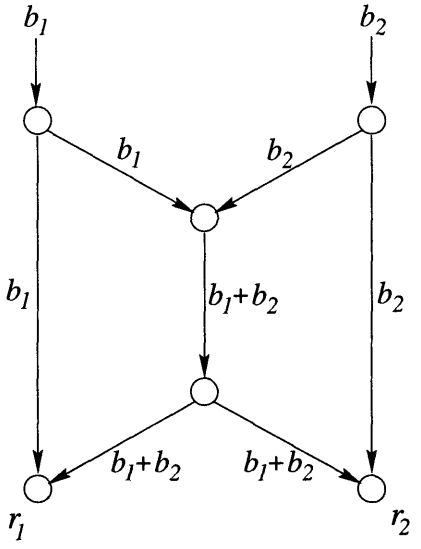


Figure 1-1: A network that requires coding. Using routing, only one receiver can receive both bits. Both receivers can recover both bits if the internal node sends $b_1 \oplus b_2$.

ing requires calculating a Steiner-tree or tree packing (when there are no buffers in the interior of the network), which are NP-complete problems. On the other hand, polynomial-time deterministic algorithms for constructing network codes have been developed [6]. Furthermore, multicast routing is infeasible for networks that require decentralized coordination (e.g. wireless ad-hoc networks), but randomly generated network codes can be constructed and operated in a distributed setting. Thus network coding can not only provide higher rates than routing, but can solve the network connection problem in situations where routing cannot.

In a randomly generated linear network code, nodes send along their output links a random linear combination of their inputs, by choosing random coefficients. The network code can be generated in a distributed fashion because each node can generate its own random code–a mapping of inputs to outputs–independently of other nodes and without knowledge of the network topology. Data is successfully transmitted from source to receivers if the collection of individual codes yields a valid solution to the network connection problem. One disadvantage of randomly generated codes is that there is a probability of failure to form a successful network code. However, the probability of failure decays exponentially with increasing code length. Reference [1]

explains random network codes in more detail.

We motivate our work by considering networks in a wireless setting, such as sensor networks. Several issues particular to wireless networks must be addressed. First, a distributed method for multicast is particularly necessary for such networks, as a centralized approach is often infeasible. The second issue is the node broadcast constraint. In a wireless network, nodes are only able to communicate within their transmitting radius. Nodes with omnidirectional antennas can only broadcast to all other nodes within their transmitting radius, and not to a selective set of its receivers. Unlike a point-to-point network where a nodes can send disparate information on each of its outgoing links, all receiving nodes of a particular transmitting node in a wireless network must receive the same information. A third issue concerns scalability. Sensor networks can have hundreds of identical nodes that are mass-produced and must be preprogrammed before they are deployed. Rather than construct a separate random code for each node, a more simple approach is to preprogram the same code into all nodes.

We address these issues by proposing a simplified version of a random network code. In this code construction, all nodes use the same code involving a single random coefficient, and send the same output on all outgoing links. In chapter 2 we define the network model and describe the construction of this code. In chapter 3 we adapt the analysis on general random codes in [1] to obtain analytical results for this code construction. Using the algebraic framework presented in [2], we refer to the unconstrained random code as a *multivariate code* and the simplified code construction as a *univariate code*. These definitions will become clear in the analysis of the code. In chapter 4 we provide simulation results for randomly generated networks to compare the performance of multivariate and univariate codes. The univariate code introduces some issues not present for multivariate codes, such as the characteristic of the finite field and the choice of the single coefficient. We discuss these issues and a relaxation of the univariate code to improve its performance in chapter 5.

Chapter 2

Definitions

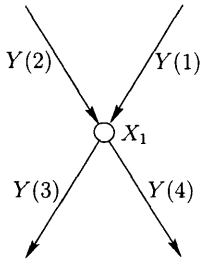
2.1 Network Model

We use the network coding model presented in [2]. A network is represented as a directed acyclic graph, where vertices represent the nodes of the network, and edges represent communication links. There are r discrete random source processes $X_1, X_2, ..., X_r$ that are observable at one or more source nodes. There are $d \geq 1$ receiver nodes, and each receiver β has output processes denoted $Z(\beta, i)$. The network has ν links, where each link l carries the random process $Y(l)$. We assume the random processes generate binary sequences, which are partitioned into vectors of bits. All source processes produce one vector per unit time, and all links transmit one vector per unit time. We wish to construct a network code that enables all the input processes to be transmitted to every receiver node. Operations in the code occur over a finite field $\mathbb{F}$ by viewing the bit vectors as scalar elements of the field. In our simulations, we used both finite fields of characteristic 2, i.e. $\mathbb{F}_{2^m}$ and prime finite fields $\mathbb{Z}_p$, which have characteristic p. The characteristic of a field is the smallest integer n such that every element added to itself n times equals zero. For the analysis of bounds in chapter 3, the type of field is irrelevant, but in chapter 5 we show how the characteristic of the field can affect the code.

2.2 Code Construction

We describe three different codes: the unconstrained multivariate code, the multivariate broadcast code, and the univariate code. The unconstrained multivariate code is the most general one applicable to any network. The code is constructed as follows. For each outgoing link of a non-receiver node v, a random coefficient from $\mathbb{F}$ is chosen for every source process observed at v and for every incoming link of v. Thus every outgoing link of v carries a random linear combination of the processes observed at v and the processes on incoming links of v. If the random coefficients are chosen properly, the process on each outgoing link is different from that of other outgoing links of the same node. Each receiver node collects the signals on its incoming links and decodes them by taking the necessary linear combinations to recover the input signals. The multivariate broadcast code is similar to the unconstrained multivariate code, but every outgoing link of a given node sends the same linear combination of the node's inputs. Thus each node needs to choose only one coefficient for each source process and each incoming link.

Now we describe the univariate code construction. A single coefficient for the entire network α is randomly chosen from $\mathbb{F}$. For each link l let $v = tail(l)$. Each outgoing link carries the sum of all source processes observed at v, and the processes on incoming links of v multiplied by α. Thus not only does the univariate code meet the broadcast constraint, but every node uses the same code. The decoding method at the receiver nodes is the same as for the multivariate code. Figure 2-1 depicts the various codes for an individual node.



	$Y(3)$	$Y(4)$
multivariate	$a_{1,3}X + f_{1,3}Y(1) + f_{2,3}Y(2)$	$a_{1,4}X + f_{1,4}Y(1) + f_{2,4}Y(2)$
multivariate broadcast	$a_1X_1 + f_1Y(1) + f_2Y(2)$	$a_1X_1 + f_1Y(1) + f_2Y(2)$
univariate	$X_1 + \alpha Y(1) + \alpha Y(2)$	$X_1 + \alpha Y(1) + \alpha Y(2)$

Figure 2-1: Codes for an individual node. Note that for the multivariate broadcast code and the univariate code, the function on all output links are equivalent.

The overall network code (for any code type) is represented as the triple (A, F, B). A is an $r \times \nu$ matrix where $a_{i,j}$ is nonzero if X_i is observed at $link(j)$ and 0 otherwise. F is a $\nu \times \nu$ matrix where $f_{i,j}$ is nonzero if $link(i)$ is the incoming link and $link(j)$ is the outgoing link of the same node, and 0 otherwise. The links are numbered so that F is upper triangular, which is possible since the graph is acyclic. B consists of a column of d matrices B_β, one for each receiver. Each B_β is a $r \times \nu$ matrix that represents the linear operation β performs on its inputs to recover the signals. Each B_β has nonzero entries in the columns corresponding to incoming links of receiver node β. The product $A(I - F)^{-1}B_\beta^T$ is the transfer matrix from source processes to output processes of β.

For the unconstrained multivariate code, nonzero entries of A and F are indeterminate variables $a_{i,j}$ and $f_{i,j}$, respectively, hence the name *multivariate code.* In the univariate code, the nonzero entries of A are all 1, and the nonzero entries of F are the same variable α, hence the name *univariate code.* The multivariate broadcast code is represented similarly to the unconstrained multivariate code, except the same variable is used for all nonzero entries in the same row. The entries of the B_β matrices are not determined by the code, but for analysis purposes the entries are indeterminate variables $b_{\beta_{i,j}}$ for all code types. As an illustration, the A, F, and B_β matrices for the three code types are given for the network in figure 1-1, reproduced in figure 2-2 with the sources, links, and receivers numbered.

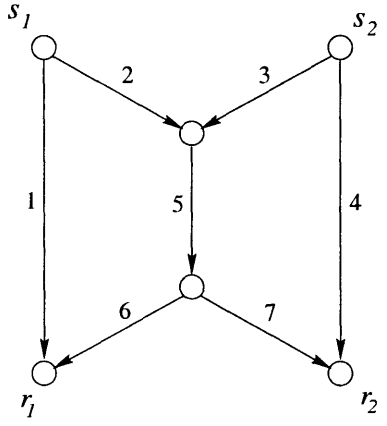


Figure 2-2: A network indicating the sources and receivers. The links are numbered in topological order.

Unconstrained Multivariate Code:

$$A = \begin{pmatrix} a_{1,1} & a_{1,2} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & a_{2,3} & a_{2,4} & 0 & 0 & 0 \end{pmatrix}, \quad F = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & f_{2,5} & 0 & 0 \\ 0 & 0 & 0 & 0 & f_{3,5} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & f_{5,6} & f_{5,7} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Broadcast Constrained Multivariate Code:

$$A = \begin{pmatrix} a_1 & a_1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & a_2 & a_2 & 0 & 0 & 0 \end{pmatrix}, \quad F = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & f_2 & 0 & 0 \\ 0 & 0 & 0 & 0 & f_3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & f_5 & f_5 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Univariate Code:

$$A = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 \end{pmatrix}, \quad F = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \alpha & 0 & 0 \\ 0 & 0 & 0 & 0 & \alpha & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \alpha & \alpha \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

B Matrices (identical for all code types):

$$B_1 = \begin{pmatrix} b_{1,1} & 0 & 0 & 0 & 0 & b_{1,6} & 0 \\ b_{2,1} & 0 & 0 & 0 & 0 & b_{2,6} & 0 \end{pmatrix}, \quad B_2 = \begin{pmatrix} 0 & 0 & 0 & b_{1,4} & 0 & 0 & b_{1,7} \\ 0 & 0 & 0 & b_{2,4} & 0 & 0 & b_{2,7} \end{pmatrix}$$

The analysis of a point-to-point network uses the link adjacency matrix F because different outgoing links of a node can carry disparate signals. But in the broadcast scenario, each node has only one outgoing signal. This allows an alternate representation of the two broadcast codes. We replace F with the node adjacency matrix G, where $g_{i,j}$ is nonzero if there exists a link that connects node i to node j. A and B are modified so that columns correspond to nodes instead of links. The nonzero entries of G for the multivariate broadcast code are all independent indeterminate variables $g_{i,j}$, like the F matrix for the unconstrained multivariate code. The nonzero entries of A and G of the univariate code are constructed in the same manner as before. The transfer matrix is the same as the previous case, with G replacing F.

Chapter 3

Performance Bounds

The network code (A, F, B) solves the multicast problem if and only if the transfer matrix $A(I - F)^{-1}B_\beta^T$ is full rank for each receiver β [2]. This is equivalent to the requirement that $A(I - F)^{-1}B_\beta^T$ have a nonzero determinant. Note that for a multivariate code, the random coefficients are distinct variables $a_{i,j} \in A$ and $f_{i,j} \in F$, and the determinant polynomial is a multivariate polynomial in the ring $\mathbb{F}[x]$. The univariate code only has one variable, α, resulting in a univariate polynomial. We do not consider the coefficients in B because those coefficients are chosen in order to recover source processes, and do not affect the design of the code. The code length is determined by the size of the finite field, and is proportional to the computational complexity of the code. Thus is it desirable to find the smallest field size which contains a non-zero solution to the determinant polynomial. Given that a solution exists in a given field, we also want to know the probability of randomly finding a non-zero solution. We apply the analysis technique used in [1] to obtain bounds for sufficient field size and probability of successful code construction for the univariate code. Reference [1] makes the connection between network coding and network flows, then recognizes that the network coding problem has a corresponding bipartite matching formulation, since network flow problems can be reduced to bipartite graph matching problems. Furthermore, the network coding problem can be analyzed using the Edmonds matrix formulation for checking for a perfect matching in a bipartite graph [3].

3.1 Bound on Code Length and Construction

Lemma 1. *Let (A, F, B) be the univariate network code for a specific connection problem. Let d be the number of receivers and ν be the number of links in the network.*

(i) For a feasible multicast connection problem using the univariate code, a solution exists in any finite field $\mathbb{F}_q$, where $q > d\nu$.

(ii) The probability of a successful code construction by randomly choosing α is at least $1 - \frac{d\nu}{q}$.

Proof. It can be shown [1] that

$$det(A(I-F)^{-1}B_\beta^T) = (-1)^{r(\nu+1)} det(\begin{bmatrix} A & 0 \\ I-F & B_\beta^T \end{bmatrix}), \tag{3.1}$$

where $\begin{bmatrix} A & 0 \\ I-F & B_\beta^T \end{bmatrix}$ is the Edmonds matrix corresponding to the flow graph of the source nodes and receiver β.

(i) Consider the determinant polynomial of the Edmonds matrix in terms of the variables α and $b_{\beta;i,j}$ (the coefficients of B_β). Every term of the determinant is a product of elements such that no element is from the same row or column. Thus no product term in the determinant can have a $b_{\beta_{i,j}}$ with exponent greater than 1. The exponent of α in any product term cannot be greater than ν, since α only occurs in the first ν columns of the matrix. Therefore the largest exponent of any variable is at most ν. If we take the product of d polynomials (one for each receiver), the largest exponent of any variable is at most $d\nu$. By applying the argument used in [1], we can show that there exists a nonzero solution to the polynomial in $\mathbb{F}_q$, $q > d\nu$. The argument uses induction to show that values can be assigned to each variable in such a way that the polynomial is nonzero.

(ii) The determinant polynomial of the corresponding Edmonds matrix has degree $\leq \nu$ in terms of α. There are d such polynomials, one for each receiver. The

requirement that each polynomial is nonzero is equivalent to requiring the product of the polynomials to be nonzero. Therefore the degree of the polynomial we wish to consider is at most $d\nu$. The polynomial has at most $d\nu$ distinct roots in a field $q > d\nu$. If α is randomly chosen from $\mathbb{F}_q$, the probability that the determinant is nonzero (i.e. a root is not chosen) given the determinant is not identically zero is $> 1 - \frac{d\nu}{q}$, by the Schwartz-Zippel Theorem [3].

□

3.2 An Alternate Bound

We give an alternate lower bound on the field size necessary for a solution to exist.

Lemma 2. *Let (A, F, B) be the univariate network code for a specific connection problem. Let d be the number of receivers and r be the number of distinct source processes. Define p to be one less than the number of links in the longest path between source and receiver nodes.*

(i) For a feasible multicast connection problem using the univariate code, a solution exists in any finite field $\mathbb{F}_q$, where $q > dpr$.

(ii) The probability of a successful code construction by randomly choosing α is at least $1 - \frac{dpr}{q}$.

Proof. All possible paths between nodes are represented in the series $(I + F + F^2 + F^3 + \ldots)$. Since F is nilpotent there exists some n such that F^n is the all zero matrix, and $(I + F + F^2 + F^3 + \ldots) = (I - F)^{-1}$ [2]. It is easy to verify that $F^{p+1} = 0$ and $(I - F)^{-1} = (I + F + F^2 \ldots + F^p)$. This means that $(I - F)^{-1}$ contains entries that are polynomials in α of degree no more than p. Then the determinant polynomial of $A(I - F)^{-1}B_\beta^T$ in terms of α and $b_{\beta_{i,j}}$ has a maximal degree $\leq pr$, and a nonzero solution for all receivers exists in $\mathbb{F}_q$, $q > dpr$. Applying the Schwartz-Zippel Theorem, the probability the determinant is nonzero for a randomly chosen α is $> 1 - \frac{dpr}{q}$. □

3.3 Bounds Using the Node Adjacency Matrix

Bounds for the broadcast codes can be obtained using the node adjacency matrix representation, yielding a third bound for the univariate code.

Lemma 3. *Let (A, G, B) (node adjacency matrix represenation) be the univariate network code for a specific connection problem. Let d be the number of receivers and n be the number of nodes in the network.*

(i) For a feasible multicast connection problem using the univariate code, a solution exists in any finite field $\mathbb{F}_q$, where $q > dn$.

(ii) The probability of a successful code construction by randomly choosing α is at least $1 - \frac{dn}{q}$.

Proof. The proof is exactly the same as in Lemma 1, by replacing the link adjacency matrix F with the node adjacency matrix G, and constructing the appropriate A and B matrices. □

The node adjacency matrix representation also allows us to analyze the multivariate broadcast code. Since the variables are identical to the unconstrained multivariate code representation, by the proof in [1] for unconstrained multivariate codes, the bound for the multivariate broadcast code is d, the number of receivers.

3.4 Comparison of Bounds

The lower bound of required field size for a multivariate code is d, the number of receivers [1]. Clearly the three univariate bounds are greater than the multivariate bound. The multivariate bound only depends on the number of receivers, and the univariate bounds are dependent on the topology of the graph. In particular, the univariate bounds can grow as the number of nodes in the network increases. However the code length will not be adversely affected, because code length grows logarithmically in relation to field size. Depending on the network, one bound will be smaller than the other, and either bound can be used to choose an appropriate field size. A

Code type	multivariate	multivariate broadcast	univariate
bound	d	d	$d\nu$ dpr dn

Table 3.1: Bounds for various code types.

network that is very dense may contain many links, but consequently have a small maximum path length. Conversely, a sparse, wide-spread network can contain long path lengths between sources and receivers, but may have few links. Furthermore, use of the node adjacency matrix tightens the bound from $d\nu$ to dn, because in general the number of links in a network exceeds the number of nodes, and can be $O(n^2)$ in a dense graph. Table 3.1 summarizes the various bounds.

We showed that the bound for the multivariate broadcast code is identical to the unconstrained multivariate code. In other words, there is no penalty, in terms of bounds, for enforcing the broadcast constraint on the multivariate code. Another way to analyze the multivariate broadcast code is to model the broadcast constraint by creating a broadcast graph by inserting virtual nodes and links at every real node that has multiple outgoing links, and constructing an unconstrained multivariate code on the new graph. Figure 3-1 illustrates how this is done. If an unconstrained multivariate code cannot be constructed on the broadcast graph, then a multivariate broadcast code cannot be constructed on the original graph, providing a way to determine the feasibility of a broadcast constrained code. The broadcast graph also makes it possible to apply the min-cut max-flow theorem for network information flow [4] to broadcast constrained networks.

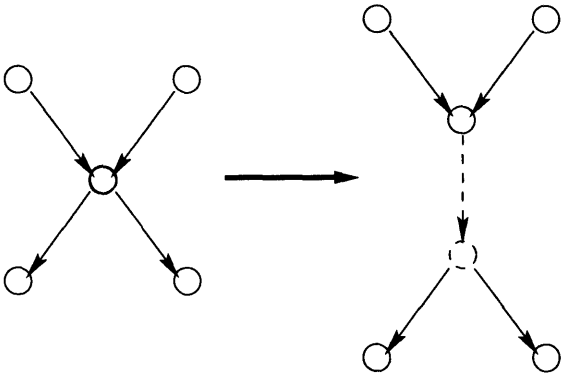


Figure 3-1: The graph on the right shows how the graph on the left is augmented to model the broadcast restraint. A multivariate broadcast code on the left graph is equivalent to an unconstrained multivariate code on the right graph.

Chapter 4

Simulations

Simulations were run on various network configurations to compare the performance of the univariate code with the multivariate codes. The three different code types were constructed on randomly generated network graphs, and the rank of the transfer matrices for each receiver was evaluated to determine the success of a code. For each code type, codes were constructed in finite fields $\mathbb{Z}_p$ starting with $p = 11$, and three different random codes were generated and tested before increasing the field size. Four different field sizes were tried before declaring failure. Trials were also conducted using fields with characteristic 2, and they yielded similar results to trials using fields with odd characteristic.

The three main parameters for the networks were number of nodes n, transmission range ρ, and number of source (r) and receiver (d) nodes. n node locations were chosen uniformly within a unit square by randomly generating x- and y-coordinates. Nodes were sorted by increasing x-coordinate value, and nodes within range ρ were connected by directed links (the tail at the lower-numbered node) to produce acyclic graphs. Source and receiver nodes were assigned to the r lowest- and d highest-numbered nodes, respectively. Note that the broadcast constraint precludes source nodes with a rate greater than one. This is due to the fact that a single node cannot output distinct processes within the broadcast constraint. Thus, the rate for each network is equal to the number of source nodes.

Figure 4-2 provides some simulation results. Each plot corresponds to a different

source-receiver configuration. For a constant number of nodes (10 for the results in 4-2), the plot shows success rates for different transmission ranges. Included with the success rates of the three code types is the success rate for the connectivity of the network. Connectivity in this case is defined as the existence of a path from each source to each receiver, a requisite for any code to work. As expected, the connectivity success rate upper bounds the codes' success rates, the unconstrained multivariate code upper bounds the constrained version, which in turn upper bounds the univariate code. This is because univariate codes are a subset of multivariate broadcast codes, which is a subset of general multivariate codes. In the cases that the unconstrained multivariate code succeeded but the multivariate broadcast code failed, the problem was essentially a bottleneck in the network caused by the broadcast constraint. In the cases where the multivariate broadcast code succeeded but the univariate code failed the problem was that the processes at the receiver nodes were linearly dependent, preventing decoding of the source processes. Figure 4-1 contains examples of networks for which only a subset of the codes succeed. While the unconstrained multivariate code works for network (a), the broadcast codes fail because links 3 and 4 transmit the same signal, so the signals on the receiver's terminal links are not linearly dependent. The multivariate codes work for network (b) but the univariate code fails. The determinant polynomial of the transfer matrix is identically zero for the univariate code of this network.

The unconstrained multivariate code success rate is the best that can be done in any network, and the multivariate broadcast code success rate is the best that can be done in a wireless network. While the results for the multivariate codes track well with the connectivity, the univariate code success rates deviate from the multivariate results. The gap diminishes when the number of nodes is increased (see figure 4-3 for $n = 15$ and figure 4-4 for $n = 20$). Also note for the $n = 20$ case that the energy cost (from increase in transmission range) of switching from a multivariate broadcast code to a univariate code is small. Furthermore, chapter 5 discusses a relaxation of the univariate code which bridges the performance gap between the multivariate broadcast code and the univariate code.

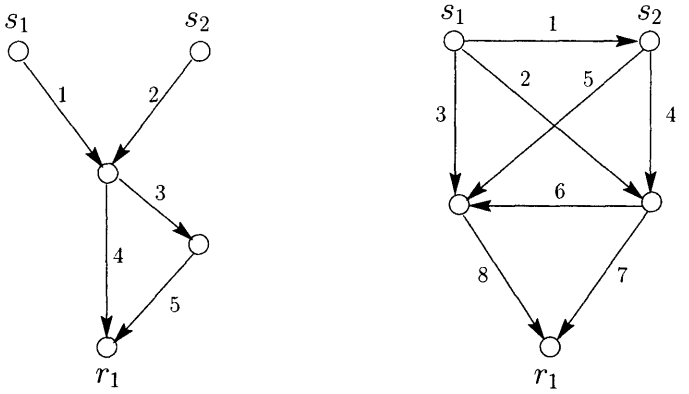


Figure 4-1: Networks for which only a subset of the codes succeed. (a) The unconstrained multivariate code works for this network but the broadcast codes fail. (b) The multivariate codes work for this network but the univariate code fails.

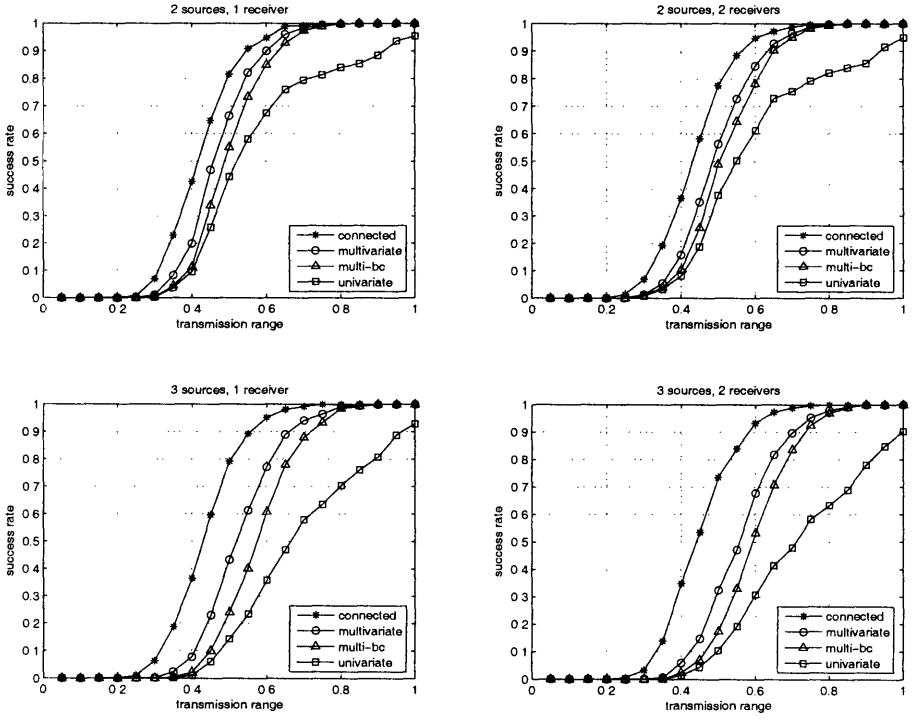


Figure 4-2: Simulation results for $n = 10$.

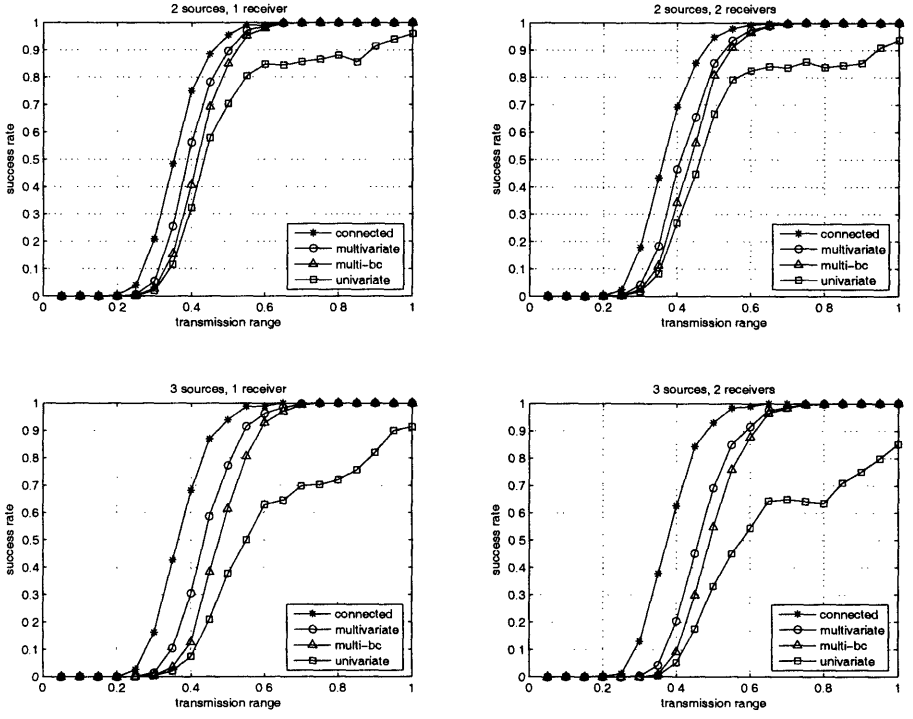


Figure 4-3: Simulation results for $n = 15$.

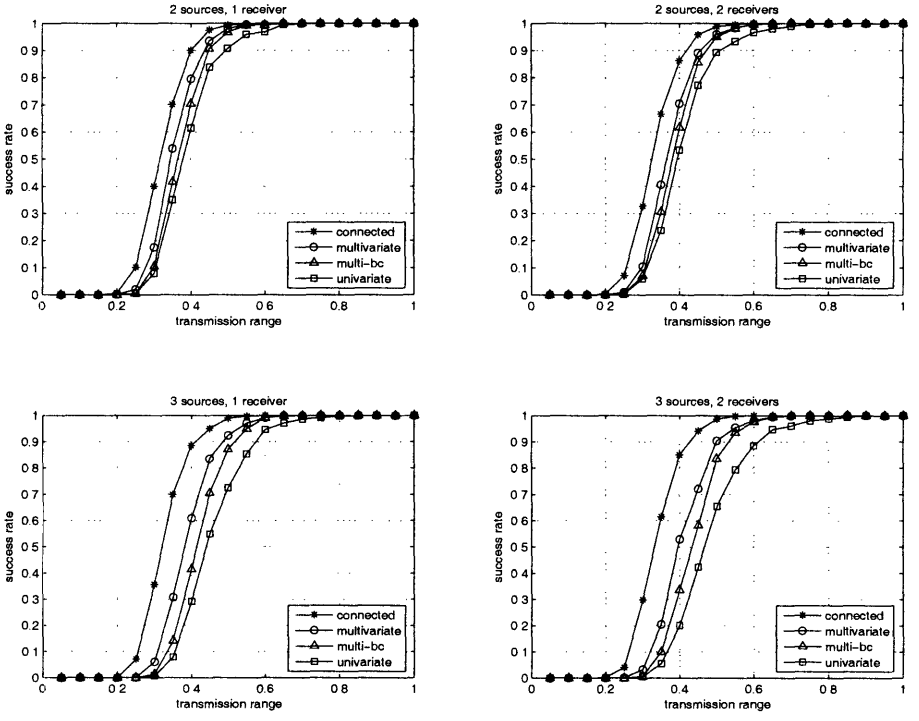


Figure 4-4: Simulation results for $n = 20$.

Chapter 5

Analysis of Univariate Code

5.1 Effect of Field Characteristic

In simulations for networks with a single source, code failures should have only occurred from lack of connectivity in the graph, resulting in equal success rates for all code types. However, occasionally the univariate code failed when the multivariate code succeeded. The reason was found to be caused by the characteristic of the field, 2 in this case. For a finite field of characteristic 2, any value added to itself results in zero. There were instances where nodes received inputs that summed to zero, regardless of the value of α, causing the code to fail unilaterally. This situation can be easily avoided by choosing fields with odd characteristic, and illustrates how the characteristic of the field can affect the code. Figure 5-1 gives a simple graph exhibiting this type of situation. This graph is the simplest case; the situation where signals are cancelled out can arise in an unlimited number of different network configurations.

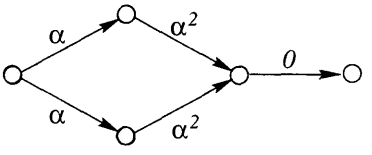


Figure 5-1: This network illustrates the simplest case where using a field with characteristic 2 causes signals to be cancelled out in using the univariate code. The intermediate nodes receive the same data from the left-most node. The summing node receives equal inputs, resulting in a zero output.

field	1	2	3	4	5	6	7	8	9	10	11 - 15
$\mathbb{F}_4$	0.70	0.99	0.99								
$\mathbb{F}_8$	0.69	0.99	0.99	0.99	0.99	0.99	0.99				
$\mathbb{F}_{16}$	0.69	0.99	0.99	0.99	0.99	0.98	0.98	0.99	0.98	0.99	0.99
$\mathbb{Z}_3$	0.97	0.72									
$\mathbb{Z}_5$	0.98	0.97	0.95	0.70							
$\mathbb{Z}_7$	0.98	0.99	0.98	0.99	0.95	0.70					
$\mathbb{Z}_{11}$	0.96	0.96	0.96	0.96	0.95	0.96	0.96	0.97	0.95	0.66	

Table 5.1: Univariate code success rate for specific values of α.

5.2 Choice of α

We examined how the actual value chosen for the variable α affects code success. Using various network configurations, the univariate code was tested with every nonzero value in the finite field substituted for α. Ignoring the trials where the code failed for all possible values, the success rate for each value was recorded. Three finite fields with even characteristic and four finite fields with odd characteristic were tested. The results are provided in table 5.1.

For most values of α, the univariate code is successful essentially all the time. At first glance this is surprising, because for a given network connection, the determinant polynomial of the transfer matrix is not affine, and thus there is some value α for which the polynomial evaluates to 0. On closer inspection, a receiver node had many more terminal links than the number of sources r, and it only required r linearly independent links to decode. Thus, with high probability, there existed some set of links that were linearly independent for any value of α.

However, the success rate for $\alpha = 1$ in $\mathbb{F}_{2^m}$ and $\alpha = p - 1$ in $\mathbb{Z}_p$ is around 70%. To explain this we examine the properties of these elements in their respective fields. The $r \times \nu$ matrix $M = A(I - F)^{-1}$ is the transfer matrix from source processes to each link in the network. The elements of M are polynomials in the variable α. Let P represent an entry of M. Let M_β be the matrix created by taking the columns of M that correspond to the terminal links of a receiver node β. The source processes can be recovered at β if $rank(M_\beta) = r$.

In $\mathbb{F}_{2^m}$,

$$P = \sum_{i=1}^{2^m-1} g_i \alpha^i, \; g_i \in \{0,1\}. \tag{5.1}$$

If $\alpha = 1$, then

$$P = \sum_{i=1}^{2^q-1} g_i \tag{5.2}$$

$$= 0 \text{ or } 1. \tag{5.3}$$

Thus every element in M_β evaluates to 0 or 1 when $\alpha = 1$. This restricts the degrees of freedom of the entries of M_β, making it less likely for $rank(M_\beta)$ to equal r.

In $\mathbb{Z}_p$, the order of the element $p-1$ is 2, i.e. $(p-1)^2 = 1 \bmod p$. This means the powers of α evaluate to either 1 or -1, restricting the degrees of freedom of P and the possible values it can evaluate to.

Overall the results for $\mathbb{F}_{2^m}$ were better than that for $\mathbb{Z}_p$. This helps the case for choosing $\mathbb{F}_{2^m}$ over $\mathbb{Z}_p$, along with the observation that operations over $\mathbb{F}_{2^m}$ are more straightforward to implement for vectors of bits. The caveat is that the cancellation effect of the characteristic explained in the previous section will not be avoided.

5.3 Relaxing the Code

In the univariate code, every link coefficient is the same value α. We diversify the code by allowing some of the link coefficients to be a different value. At each node, a certain fraction of incoming links f are assigned the coefficient α and the leftover $1-f$ fraction of links are assigned the value 1. While the link fraction f is fixed for the entire network, each node chooses the specific link assignments randomly. The broadcast constraint is maintained. Figure 5-2 contains the univariate code and multivariate broadcast code results from 4-2, but includes the results for relaxed codes. For each transmission range, nine different values of f from 0.1 to 0.9 were tested. As the results show, this simple modification can make the univariate code comparable to the multivariate broadcast code. Figure 5-3 provides the results of the

relaxed code for $n = 15$. The performance of the relaxed code closely meets the bound of the multivariate broadcast code. Recall the simulation results provided in chapter 4. The performance of the univariate code deviated from that of the other codes for $n = 10$ and $n = 15$, but not for $n = 20$. Note that the relaxed code performance for $n = 10$ and $n = 15$ is similar to the univariate code performance for $n = 20$. Therefore, diversifying the code is an alternate solution to improving performance, instead of increasing the number of nodes.

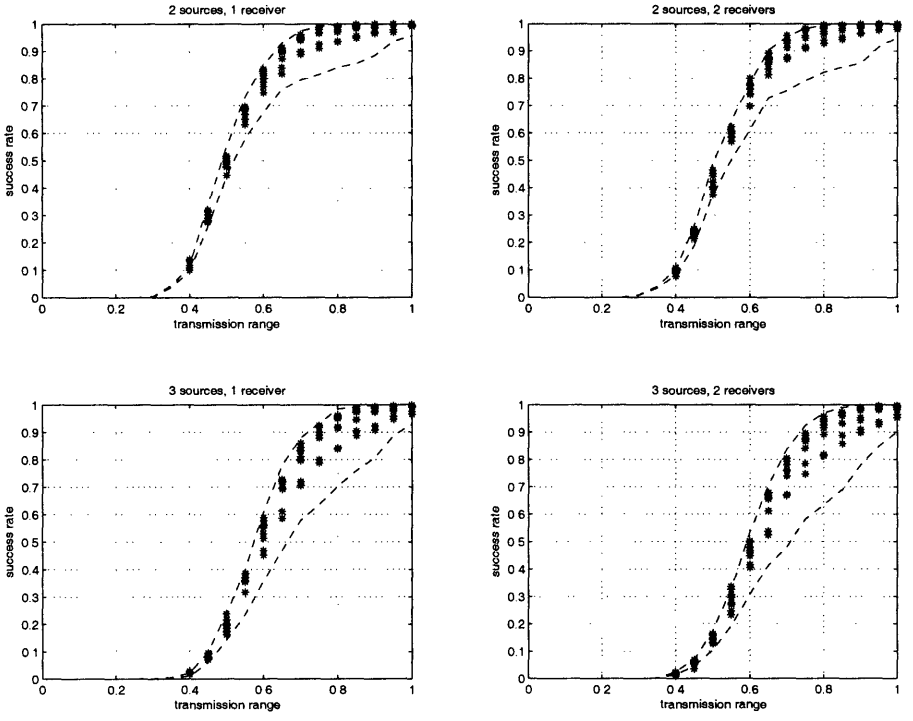


Figure 5-2: Simulation results for the relaxed univariate code. $n = 10$. The two dotted lines outline the success rate of the multivariate broadcast code and the univariate code. The points indicate different realizations of the relaxed code by varying the link fraction parameter (the fraction of links using α as the link coefficient).

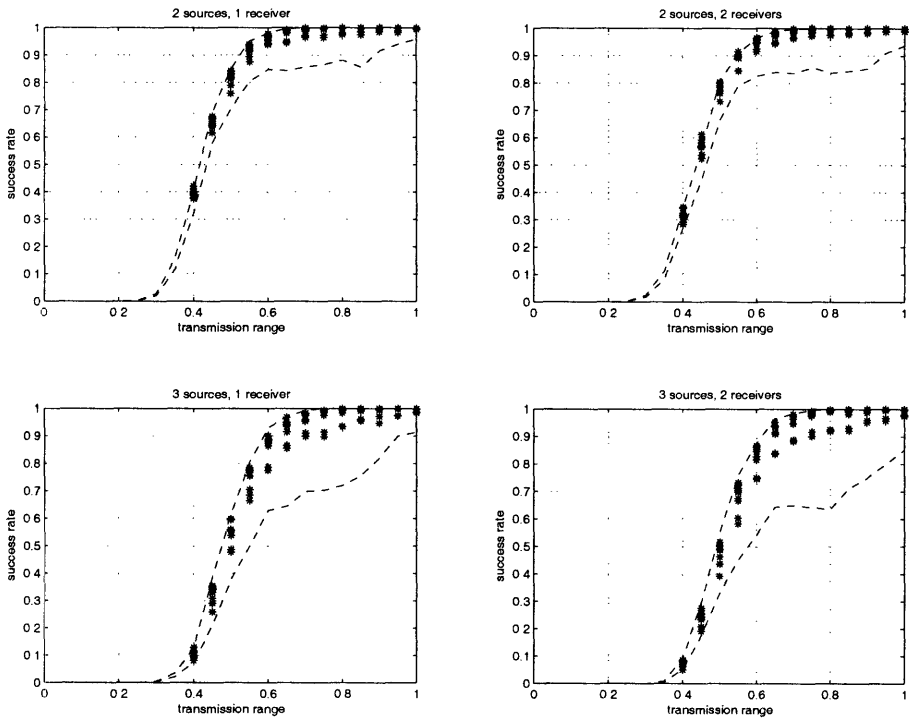


Figure 5-3: Simulation results for the relaxed univariate code. $n = 15$. The two dotted lines outline the success rate of the multivariate broadcast code and the univariate code. The points indicate different realizations of the relaxed code by varying the link fraction parameter (the fraction of links using α as the link coefficient).

Chapter 6

Conclusion

Multicast network coding can be applied to any type of network and clearly has its advantages over multicast routing. However there are additional issues that must be addressed for multicast in wireless sensor networks. These issues include the necessity of a distributed method of operation, the node broadcast constraint, and concerns of scalability. We have presented a simplified network code motivated by wireless sensor networks. The code derives from placing constraints on a randomly generated code to meet the challenges of a wireless network. The first issue is solved by the use of a randomly generated network code. The other two issues are addressed by constraining the random network code to meet the broadcast constraint and requiring all nodes to use the same, simple code. This single code facilitates mass production of nodes and only requires one parameter to be programmed into all nodes before deployment. We have adapted the analysis used for random network codes to determine bounds on the required code length and code construction success probability.

Simulation results indicate that the univariate code performance is comparable to multivariate codes. Further work is needed to fully characterize the tradeoffs between number of nodes, transmission range, and node complexity. Different network configurations will benefit from deploying additional nodes versus increasing transmission range, or diversifying the code. Each method has its own cost in term of computational and transmission energy. The question is, when can a network connection problem be solved by adding more nodes to the network, and when will it only be

solved by a more complicated code? Other further work is to look at the robustness of the code to link erasures and dynamic networks; is there a penalty in terms of robustness for using univariate codes?

We introduced one method to add diversity to the code and showed that the performance gap between the broadcast codes could be recovered. There are several other simple methods of adding diversity to the code. Another idea is to have two codes; some nodes use the coefficient α and other nodes use another coefficient α'. The same, simple node can be used for both codes because the coefficient is programmed as a parameter.

There are many more problems to solve in order to implement the univariate code as a working method for data transmission in wireless networks. For one, the model does not deal with interference, an unavoidable aspect of wireless communications. Either we need to include the effects of interference into the model, or explore a model that is not adversely affected by interference, bursty traffic for example. An actual implementation will have to address synchronization issues, and how to deal with failed network connections. But as the history of coding theory has shown, abstract models and theoretical bounds are an important and essential first step towards developing practical algorithms.

Bibliography

[1] T. Ho, R. Koetter, M. Médard, M. Effros, J. Shi, and D. Karger, "Toward a Random Operation of Networks", *IEEE Transactions on Information Theory*, submitted (2004). `http://web.mit.edu/trace/www/itrandom5.pdf`

[2] R. Koetter and M. Médard, "An Algebraic Approach to Network Coding", *IEEE/ACM Transactions on Networking*, to appear.

[3] R. Motwani and P. Raghavan, *Randomized Algorithms*, Cambridge Univeristy Press, 1995.

[4] R. Ahlswede, N. Cai, S.-Y. Li, and R. Yeung. "Network Information Flow," *IEEE Transactions on Information Theory*, 46:1204-1216, 2000.

[5] S.-Y. R. Li, R. W. Yeung, and N. Cai. "Linear Network Coding," *IEEE Transactions on Information Theory*, 49:371-381, 2003.

[6] S. Jaggi, P. Sanders, P. A. Chou, M. Effros, S. Egner, K. Jain, L. Tolhuizen. "Polynomial Time Algorithms for Multicast Network Code Construction", *IEEE Transactions on Information Theory*, submitted.