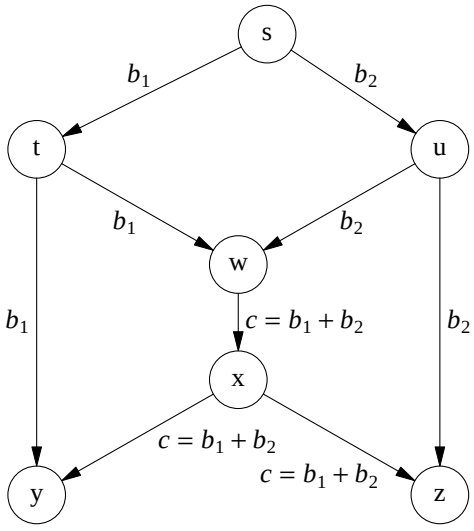
Chapter III: Network Coding

1. Wired Networks

Observation: It is sometimes possible to get more information between a source and destination by sending linear combinations of the bits over some links than by sending the messages.

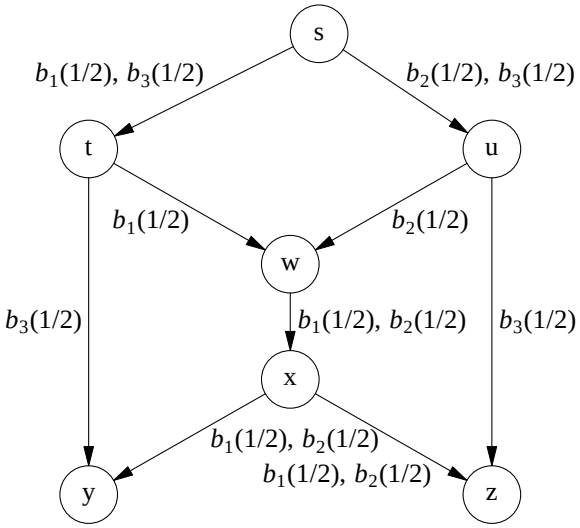
Example: 1

- Multicast messages from s to destinations y and z
- The capacity of each link is the same. (For simplicity, the capacity is 1 unit per second)



- b_1 and b_2 are streams at 1 unit/sec.
- $c = b_1 + b_2$ is the bit-by-bit modulo 2 sum of b_1 and b_2. The i^{th} bit $c_i = b_{1,i} + b_{2,i}$
- Both nodes y and z can receive both messages b_1 and b_2.
  - At node y, $b_2 = c + b_1$,
  - At node z, $b_1 = c + b_2$
- The throughput to each node is 2 units/sec.

- Without using the linear combination, the throughput to each destination would only be 1.5 units/sec..
  We show one way to receive 1.5 units/sec.:
  - Each of the three flows is 1/2 units/sec..
  - The flow on each link is ≤ one units/sec.
  - There are other ways to achieve 1.5 units/sec., without linear coding, but no ways to achieve 2 units/sec.



Note: The increase in capacity may be misleading. Network coding increased the flow between node s and nodes y and z. If this was the only flow in the network, the network throughput has increased. However, the capacity on links t-w, u-w, t-y, and u-z has increased from 1/2 to 1. If there were other flows on the network these links may carry flows between other sources and destinations. If the flows between other sources and destinations are taken into account, the network throughput may decrease when we use network coding.

2. Wireless Networks - Opportunistic Messages

On a broadcast channel the message is heard by many receivers. [2]

2.1 Retransmissions:

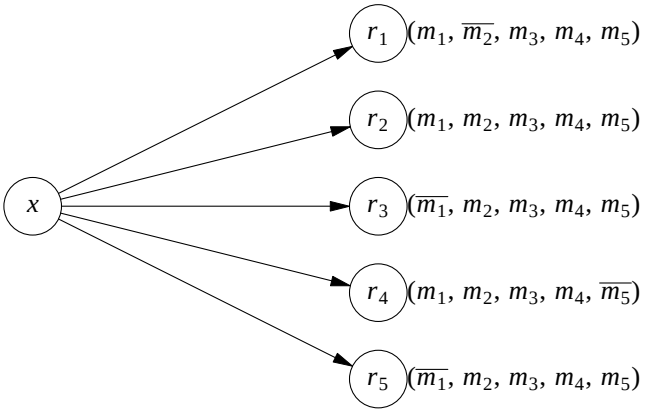
If we know which receivers have successfully received which messages, network coding allows us to retransmit a smaller number of linear combinations of messages, instead of retransmitting all of the messages that were missed by any of the receivers.

2.1.1 In Local Broadcast or on a Multicast Tree:

Example 1, a source, x, multicasts 5 messages, (m_1, m_2, m_3, m_4, m_5) to 5 receivers $r_1, \cdots, r_5$. Each of the receivers must receive all 5 messages.

- r_1 misses m_2, but receives the other messages
- r_2 receives all of the messages
- r_3 misses m_1 but receives the other messages
- r_4 misses m_5 but receives the other messages
- r_5 misses m_1 but receives the other messages.

The missing messages are depicted at each receiver in the figure by a bar



In a conventional system, x would retransmit 4 messages:

- m_2 to r_1
- m_1 to r_3
- m_5 to r_4 and
- m_1 to r_5

By realizing that the channel is a broadcast channel, x only has to retransmit m_1 once, since both r_1 and r_5 can hear the retransmission. This reduces the number of retransmissions from 4 to 3.

By realizing that the receivers have successfully received some messages, and using linear coding, the source can reduce the number of retransmitted messages from 3 to 1.

The source retransmits $m_r = m_1 + m_2 + m_5$.

Where "+" is the bit by bit exclusive-or of the bits in the messages

For instance, the i^{th} bit of $b_{r,i} = b_{1,i} + b_{2,i} + b_{3,i}$

At the receivers, the missing message is reconstructed by taking the bit-by-bit exclusive-or of m_r and the messages that are included in m_r which the receiver has previously received.

In particular:

- At r_1, $m_2 = m_r + m_1 + m_5$
- At r_3, $m_1 = m_r + m_2 + m_5$
- At r_4, $m_5 = m_r + m_1 + m_2$
- At r_5, $m_1 = m_r + m_2 + m_5$

The source has used its knowledge of which messages a source already has to construct a packet that provides the missing information to all of the recievers.

The technique is erasure correction, because the reciever knows which packet, and hence which bit in the code, it is missing.

The code in this example has 4 bit code words: $m_{1,i}$, $m_{2,i}$, $m_{5,i}$, $m_{r,1}$

The first three bits are transmitted during the original multicast, and the fourth bit is transmitted during the retransmission interval.

Note that the code isn't decided until we determine which messages were missed during the original broadcast.

The code is a simple parity check code the minimum Hamming distance between code words is 2, so that the erasure correcting capability is 1

For a binary code, the Hamming distance is the minimum number of bits in which the code words differ.

When the minimum distance between code words is d, the code can correct up to $d - 1$ erasures.

If the receiver did not know which bits were missing, it would have to use an error correcting procedure.

An error correcting code can only correct up to $\frac{d-1}{2}$ errors.

The distance 2, parity check code cannot correct any errors.

The simple parity check code works as long as a receiver is missing at most 1 packet.

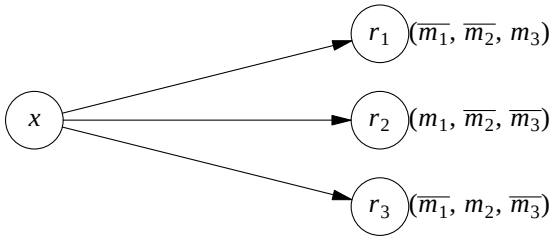
What if r_1 is also missing m_1?

- r_1 cannot determine both m_1 and m_2 when m_r is received.
- We need two independent retransmissions to determine two missing messages at m_1. If we could decode both messages from a single transmission, we would not have had to transmit both messages to any receiver.

  Note: Messages that are transmitted in a network are not always independent. In schemes, such as dispersity routing, which is discussed in a later chapter, we sometimes intentionally transmit redundant messages, that are linear combinations of other messages, to survive network failures or avoid links with long instantaneous delays. For instance: In a network with 5 paths betweeen a source and destination we may send independent messages on 4 of the paths and the fifth message might be the parity check of the first four messages, so that any single missing message can be determined from the others. $m_5 = \sum_{i=1}^{4} m_i$ at the source, and $m_j = \sum_{i \neq j} m_i$ for any missing message m_j at the receiver, so that we only have to receive messages on 4 of the 5 paths.

- In this example we could transmit $m_{r1} = m_1$ and $m_{r2} = m_2 + m_5$. When these two retransmissions are received, every receiver can recover its missing messages.
- In general, if the receiver that is missing the largest number of messages, is missing k *independent* messages we need at least k retransmitted messages m_{r1}, $m_{r,2}$, $\cdots$, m_{rk}.
- k is a lower bound on the number of retransmissions.
- When multiple receivers are missing several messages, it is not always easy, or possible to find the appropriate linear combinations on the binary field that provides a sufficient set of independent equations at every node.
  - — When we discuss random coding, we will show how these conditions can be satisfied more easily on a non-binary field
- Example 2: The source sends 3 messages to 3 receivers, and each misses two different messages.

  Find the set of 2 retransmitted messages that makes it possible for each receiver to recover both messages.



- Answer: $m_{r1} = m_1 + m_2$, $m_{r2} = m_2 + m_3$
  - — At r_1, $m_2 = m_{r2} + m_3$, then $m_1 = m_{r1} + m_2$
  - — At r_2, $m_2 = m_{r1} + m_1$, then $m_3 = m_{r2} + m_2$
  - — At r_3, $m_1 = m_{r1} + m_2$, $m_3 = m_{r2} + m_2$.

2.1.2 Retransmissions - General:

The reduction in the number retransmissions for one point-to-many point communications through a node "x" can be extended to several point-to-point communications through node "x".

In wireless networks, a message is heard by many receivers in a broadcast region. In a conventional nework, the messages that are not intended for a receiver are discarded. With network coding, a receiver retains messages that are not intended for it, in order to reduce the number of messages that must be retransmitted when several receivers miss their intended messages.

In example 1, x is a forwarding node.

- m_1 and m_2 are being forwarded to r_1,
- m_3 and m_4 are being forwarded to r_2
- m_5 is being forwarded to r_4.
- The receivers miss the messages, as specified in example 1.

In a conventional network m_2 and m_5 are both retransmitted. With network coding, all receivers temporarilly save all of the messages that they can receive, rather than discarding the messages that are not intended for them, and a single message $m_r = m_2 + m_5$ is retransmitted.

- r_1 uses message m_5, which it would have discarded to recover m_2
- r_4 uses m_2, which it would have discarded, to recover m_5.
- note that even though m_1 was missed by r_3 and r_5, it is not included in the linear combination since the message was not intended for either of these receivers.

2.2 Initial Transmissions:

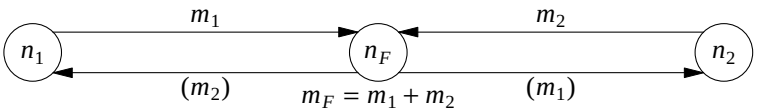
In most networks losses are an uncommon event, occurring with probabiliy between 10^{-2} and 10^{-6} in wireless networks, depending on the transmission environment. Network coding provides a greater advantage when it can be used to reduce the number of original transmitted messages.

2.2.1 Transmitter Knowledge:

A forwarding node knows that the transmitter that sent a message to be forwarded is in its transmission range, and that the transmitter has the message. Normally, the transmitter would discard the message after it is recived by the forwarding node, but in network coding, the transmitter retains the message so that the forwarding node can reduce the number of messages that it needs to transmit.

For instance, in the figure:

- n_1 transmits message m_1 to n_2 through n_F, and
- n_2 transmits message m_2 through n_2 through n_F



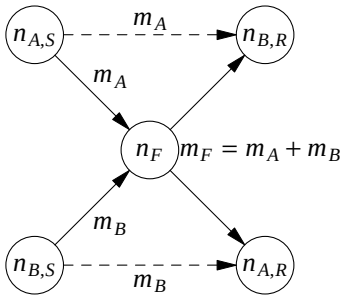
- In a conventional network n_F would forward two meassages, m_1, and m_2.
- With network coding, n_F forwards a single message, $m_F = m_1 + m_2$.
  - n_1 recovers message $m_2 = m_F + m_1$, and
  - n_2 recovers $m_1 = m_F + m_2$
- In this 2-hop network, the number of messages between the source and destination is reduced from 4 to 3, each of the 2 messages is transmitted an average of 1.5 times in the transmission region of node n_F, instead of being transmitted twice in that region..
- In a k-hop network, the number of messages between a source and destination is reduced from 2*k to k+1. (Show how in problem 1)
- In general there are more than 2 nodes that communicate with a forwarding node.
  - — If messages are being exchanged with more than one node in an area, the messages being received by any of the nodes can only be reduced by the number of messages that it has.
  - — When we use overhearing, described in the next section, we can do more.

2.2.2 Extend Overhearing:

- In addition to the source of a message, other receivers within the range of the forwarding node also overhear the transmissions from the source.
- Normally, a receiver does not keep messages that are not intended for it.
- However, this information can be used to reduce the number of messages that are forwarded.

For instance, in the figure

- $n_{A,S}$ transmits message m_A to $n_{A,R}$ through n_F.
  m_A is overheard by $n_{B,R}$
- $n_{B,S}$ transmits message m_B to $n_{B,R}$ through n_F



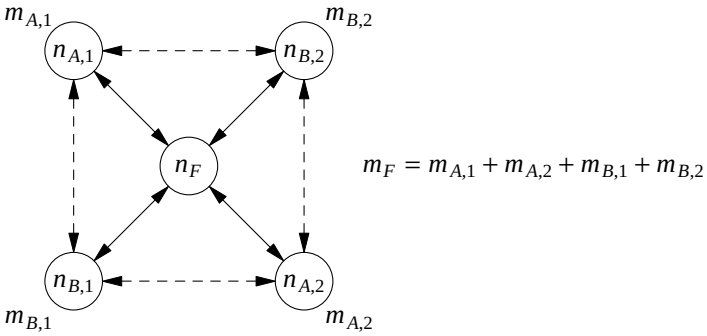
- n_F forwards one message $m_F = m_A + m_B$ instead of 2 messages
- $n_{A,R}$ has over heard m_B and decodes $m_A = m_F + m_B$
- Likewise, $n_{B,R}$ decodes m_B.

In this example, the receivers have 1 message that they have overheard, so the message forwarded at n_F can combine that message with the message that must be forwarded to that receiver. In this example, the number of messages forwarded by n_F is cut in half by taking advantage of the opportunistic receptions.

In general, if a receiver has k messages, all of those messages may be included in the forwarded message, and the number of forwarded message can be cut by up to $1/k$.

For example, in the figure:

- $n_{A,1}$ transmits $m_{A,1}$ to $n_{A,2}$, and the message is overheard by $n_{B,1}$ and $n_{B,2}$.
- $n_{A,2}$ transmits $m_{A,2}$ to $n_{A,1}$ and it is overheard by 2 other nodes as shown.
- $n_{B,1}$ and $n_{B,2}$ exchange message $m_{B,1}$ and $m_{B,2}$ and the messages are overheard as shown.



- Each receiver has the three messages that it does not need, the message that it transmitted and two messages that it overheard.
- It uses these three messages and m_F to decode the message that it needs to receive.
- The number of messages that a receiver has is an upper bound on the number of messages that can be included in a single forwarded message, in addtion to the message that it must receive.

- In general, a forwarding node, transmits M messages. The M messages are a linear combination of N messages.
  - The messages for a receiver R_i are in a subset M_i of M, that contain a subset N_i of the N messages.
  - Receiver R_i has K_i of the N_i messages, either because it transmitted those messages, or because it opportuistically received those messages, and has $N_i - K_i$ unknowns.
  - The unknowns at R_i include all of the messages that are destined for this receiver and some of the messages for other receivers.
  - If the M_i received linear combinations have $N_i - K_i$ independent equations in the unknowns, then the receiver can extract the messages destined for it.
  - The objective is to design the the linear combinations M so that every receiver can extract its messages.
- This opportunistic message overhearing has proven to be useful in connected LAN's.
  - Based on previous receptions we can predict which nodes have overheard a message.
  - Occasionally there are transmission errors and the node doesn't overhear the message.
  - This results in retransmissions.

2.3 Summary of Coding

- At the receiver:

  We know which message we are missing

  Erasure correction, rather than error correction - Simpler

- At the transmitter

  Select the code after we know what is missing at each receiver

  As opposed to protecting against all possible combinations of loses

- The minimum number of messages we must retransmit equals the maximum number of messages missing at any receiver

  At each receiver we must form independent equations to solve for the missing messages

  Not always possible with binary codes

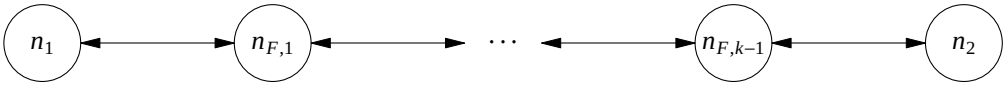
  Random coding on a higher order field

  Correcting K errors -> solving k equations in k unknowns

  Random Coding provides the right number of independent equations at every receiver

2.4 Home Work

Consider a k-hop network, where the nodes can communicate with one another as shown.



- N_1 has a long sequence of messages $m_{1,1}$, $m_{1,2,} \cdots$ destined for N_2
- N_2 has a long sequence of messages $m_{2,1}$, $m_{2,2,} \cdots$ destined for N_1

A. After an initial message sequence, how can each node reduce the number of messages that it forwards?

B. Construct the messages forwarded by each node

C. How many messages are forwarded at each node?

3. Randomly Generated Codes

- In several of the previous examples we have had the problem of finding the smallest number of linear combinations of codewords that result in a sufficient number of linear independent equations to solve for all of the missing messages at each node.
- Solving this problem on the binary field is often difficult, and may result in a larger number of linear combinaions than we can find on other fields.
- In this section we look into solving these equations on a higher order field.
  - — The higher we make the order of the field, the larger the number of possible solutions.
- In this section we will generate and check random codes. We pick coefficients for forwarding equations at random then check to see if the equations actually result in the necessary number of independent equations at every node.
  - — If the forwarding equations don't result in the necessary number of independent equations at every node, we pick another set of forwarding equations.
  - — The higher the order of the field, the more likely that we succeed.

3.1 Basic Idea

- A source sends k messages, S_1, S_2, $\cdots$, S_k.
- Message S_i contains characters, $s_{i,t}$ that are calculated from the characters in the original messages, M_1, M_2, $\cdots$, M_l as

  $s_{i,t} = a_{i,1}m_{1,t} + a_{i,2}m_{2,t} + \cdots + a_{i,l}m_{l,t}$

  where $m_{j,t}$ is the t^{th} character in message M_j

  and, $a_{i,j}$ are coefficients in the equations and are selected randomly.
- The same equation is used to calculate each character of the message S_i

  If the messages M_j have W characters, the transmitted message, S_i has W answers calculated by the equation $s_{i,t}$.
- At the receiver we know the $a_{i,j}$
- If we are missing at most K messages, and receive K messages S_i that have linearly independent coefficients for the K missing messages, we can solve for the missing messages.
- If we randomly pick the $a_{i,j}$ on the field of real numbers, which has infinite precision, the equations are always independent,

  Since there are an infinite number of $a_{i,j}$, the probability of picking a set that are linearly related goes to zero.

  but $a_{i,j}$ takes an infinite number of bits to represent,

  as does each $s_{i,t}$,

  and hence the S_i take an infinite number of bits to transmit
- Instead we use finite (Galois) fields with 2^k elements

  The $a_{i,j}$ take k bits to represent

  The characters in the messages M_j are k bits long, and

  The answers $s_{i,t}$ are k bits long

- On a finite field, the message of answers, S_i has the same number of bits as the M_i
- The bigger we make the finite field, the more likely the equations that we pick are linearly independent

3.2 Generating a Higher Order Field:

- A field has 2 operations, usually addition and multiplication, that are closed on the field

  When the operations are performed on members of the field, the answers are also on the field.

  When we perform addition and multiplication on the field of real numbers, the answers are real numbers.

  The operations have inverse operations, division and subtraction

  and identity operators, zero for addition and one for multiplication

- A finite field has a finite number of elements.

  We have to use an operation to fold the elements back on the field

  Consider the binary field, with elements 0 and 1

  1 + 1 modulo 2 = 0, so that the results are always 0 or 1

  Multiplication always gives an answer on the field

- Galois fields have polynomials of the form:

  $f(x) = a_0 + a_1 x + a_2 x^2 + \cdots a_{k-1} x^{k-1}$

  The a_i are taken from a field with b elements. This is the base field.

  The total number of elements in the Galois field is b^k

  For this work we will only consider the base field with 2 elements (the binary field).

- The two operations are polynomial addition and polynomial multiplication.

  The operations on the base field are modulo 2 addition and multiplication.

  Polynomial addition is modulo 2 addition of the terms of the polynomial

  $$f_1(x) + f_2(x) = \left((a_{1,0} + a_{2,0}) \bmod 2\right) + \left((a_{1,1} + a_{2,1}) \bmod 2\right)x + \cdots + \left((a_{1,k-1} + a_{2,k-1}) \bmod 2\right)x^{k-1}$$

  polynomial addition is is closed on the field.

  Ordinary polynomial multiplication has terms of the form

  $f(x) = f_1(x) * f_2(x) = a_0 + a_1 x + a_{2(k-1)} x^{2(k-1)}$, and has elements that are not on the original field.

  In order to bring the answer back onto the original field, we define polynomial multiplication as $f(x) = (f_1(x) * f_2(x))$ modulo $p(x)$, where $p(x)$ is a primitive polynomial whose order (highest power coefficient) is k.

  The modulo operation is the remainder of the polynomial division and reduces the order of the polynomial to at most k-1, which brings the polynomial back into the field.

- A polynomial, $p(x) = x^k + p_{k-1} x^{k-1} + \cdots + p_1 x + 1$, where $p_j = 0, 1$, of order k, is defined to be a primitive polynomial when $\left(x^j c(x)\right) \bmod p(x)$, for $j = 0, 1, \cdots, 2^k - 2$. generates all of the non-zero, binary words, of the form $c(x) = c_{k-1} x^{k-1} + \cdots + c_1 x + c_0$, of the field

  — The starting word, $c(x)$ can be any none zero code word, but is usually selected as $c(x) = 1$ and referred to as α^0.

- $\alpha^j = x^j$ modulo $p(x)$

— Primitive polynomials have played an important role in the development of algebraic, error correcting codes, such as the BCH codes.

- Large numbers of primitive polynomials have been found and tabulated. as an appendix in reference [3].
- The table of primitive polynomials, and polynomials with other characteristics, have been copied to many locations on the WEB. For instance: http://www.cs.utk.edu/ plank/plank/papers/CS-07-593/primitive-polynomial-table.txt

- To generate random codes, we will use Galois fields, with 2^k elements, referred to as $GF(2^k)$.
- The Galois field has two representations.

1. A vector representation, with k bits, that we will use for addition, (and subtraction - which is the same as addition on the binary base field.).
   - A polynomial $a(x) = a_{k-1}x^{k-1} + \cdots + a_1 x + a_0$ is represented as $A = (a_{k-1}, \cdots, a_1, a_0)$, where $a_j = 0, 1$.
   - $C = A + B$ implies than $c_j = a_j + b_j$ modulo 2
2. A logarithmic representation, of the form α^i, that we will use for multiplication and division.
   - $a(x) = \alpha^{i_a} \bmod p(x)$
   - $\alpha^0 = \alpha^{2^k-1} = 1$
   - $$\begin{aligned} c(x) &= (a(x) * b(x)) \bmod p(x) \\ &= \left(\alpha^{i_a} * \alpha^{i_b}\right) \bmod p(x) \\ &= \left(\alpha^{i_a+i_b}\right) \bmod p(x) \\ &= \left(\alpha^{(i_a+i_b) \bmod (2^k-1)}\right) \bmod p(x) \end{aligned}$$
   - The final reduction of the exponent is possible because $\alpha^{2^k-1} = 1$, and guarantees that multiplication always results in an exponent between 0 and $2^k - 2$.
   - As usual, 0*a = 0.
   - We have reduced multiplication to addition of the exponents modulo $2^k - 1$
   - Division is subtraction of the exponents modulo $2^k - 1$

— Many of the calculations that we will perform require many additions and multiplications. We could use the primitive polynomial each time. However, for fields with reasonable numbers of elements it is easier to construct a table to convert between the two representations.

3.3 Example of the representations of a Galois Field

- A primitive polynomial for $GF(2^4)$ is $p(x) = x^4 + x + 1$
  - The vector representation for the polynomial is (10011)
- Multiplication by x on the binary field:
  - $A(x) = a_3x^3 + a_2x^2 + a_1x + a_0$
    $A = (a_3, a_2, a_1, a_0)$
  - Multiplication by x -> $xA(x) \bmod p(x)$
  - To multiply A by x, shift the binary representation left and find the remainder when dividing by (10011)
    - A = (1100)
    - B(x) = xA(x) mod p(x); B=(11000) mod (10011) = (1011)
- Starting with A = (0001) we can generate the table that maps beween the binary and exponential representations
  - A=(0000) is not generated by multiplication

Exponential	binary
0	0000
α^0	0001
α^1	0010
α^2	0100
α^3	1000
α^4	0011
α^5	0110
α^6	1100
α^7	1011
α^8	0101
α^9	1010
α^{10}	0111
α^{11}	1110
α^{12}	1111
α^{13}	1101
α^{14}	1001
$\alpha^{15} = \alpha^0$	0001

3.4 Forming messages as linear combinations of other messages

To form the linear combination $m_F = a_1m_1 + a_2m_2$, where a_1 and a_2 are elements of $GF(2^4)$:

1. Break m_1 and m_2 into 4 bit segments and translate the 4 bit segments into the exponetial represention
   - $m_1 = (0010001100011101) = (0010, 0011, 0001, 1101) = (\alpha^1, \alpha^4, \alpha^0, \alpha^{13})$
   - $m_2 = (1001010011110001) = (1001, 0100, 1111, 0001) = (\alpha^{14}, \alpha^2, \alpha^{12}, \alpha^0)$
   - Select $a_1 = (1001) = \alpha^{14}$, $a_2 = (1010) = \alpha^9$
   - $a_1m_1 = (\alpha^0, \alpha^3, \alpha^{14}, \alpha^{12}) = (0001, 1000, 1001, 1111)$

— $a_2 m_2 = (\alpha^8, \alpha^{11}, \alpha^6, \alpha^9) = (0101, 1110, 1100, 1010)$

— $m_F = (0100, 0110, 0101, 0101)$

3.5 An example of a randomly generated code:

- Consider the example in opportunistic receptions, where three messages are transmitted and each of three receivers miss two different messages.
- We found a binary representation that allowed us to decode the messages when the forwarding node sent two combinations of the message, rather than retransmitting the three messages.
  However, finding the right representation wasn't obvious.
- With random coding we construct two equations from the three messages and select the coefficients randomly from the higher order field.
  - — We check that at each node the two equations result in two independent equations in two unknowns.
  - — If the equations aren't linearly independent, we pick the coefficients at random again.
  - — The more elements in the higher order field, the more likely that we can successfully pick the coefficients on the first try.
- For example, select the coefficients on $GF(2^4)$:
  - — I threw 4 pennies at a time to generate the binary representation and resulted in the following two equations:
    1. $m_{F1} = (0111)m_1 + (1000)m_2 + 1011m_3 = \alpha^{10}m_1 + \alpha^3 m_2 + \alpha^7 m_3$
    2. $m_{F2} = (1101)m_1 + (1101)m_2 + (1000)m_3 = \alpha^{13}m_1 + \alpha^{13}m_2 + \alpha^3 m_3$
  - — When m_{F1} and m_{F2} are received at the node that has received m_3, but missed m_1 and m_2
    The node constructs two equations in the two missing unknowns:
    1. $\alpha^{10}m_1 + \alpha^3 m_2 = m_{F1} + \alpha^7 m_3$
       (note that addition and subtraction are the same on the binary field)
    2. $\alpha^{13}m_1 + \alpha^{13}m_2 = m_{F2} + \alpha^3 m_3$
  - — The node can recover the missing messages as long as the two equations are independent.
    - We can use Gaussian row reduction, with our two operators to determine if the following equations are independent: $\begin{bmatrix} \alpha^{10} & \alpha^3 \\ \alpha^{13} & \alpha^{13} \end{bmatrix}$
    - multiply the first row by α^5
      - — $\alpha^{10}\alpha^5 = \alpha^{15} = \alpha^0 = 1$
      - — $\alpha^3\alpha^5 = \alpha^8$

      $\begin{bmatrix} \alpha^{10} & \alpha^3 \\ \alpha^{13} & \alpha^{13} \end{bmatrix} \rightarrow \begin{bmatrix} \alpha^0 & \alpha^8 \\ \alpha^{13} & \alpha^{13} \end{bmatrix}$
    - Multiply the first row by α^{13}, and subtract it (add it) to the second row.
      - — $\alpha^{13} - \alpha^{13} = 0$
      - — $\alpha^8 * \alpha^{13} = \alpha^{21} = \alpha^6 = (1100)$
      - — $\alpha^{13} + \alpha^6 = (1101) + (1100) = (0001) = \alpha^1$

      $\begin{bmatrix} \alpha^0 & \alpha^8 \\ \alpha^{13} & \alpha^{13} \end{bmatrix} \rightarrow \begin{bmatrix} \alpha^0 & \alpha^8 \\ 0 & \alpha^1 \end{bmatrix}$

- Multiplying the second row by α^{14}, then subtracting α^8 times the second row from the first row, we get $\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$
- This shows that the node can determine m_1 and m_2 from the two equations.

— The node that received m_2, and missed the other two messages, has the equations:

1. $\alpha^{10} m_1 + \alpha^7 m_3 = m_{F1} + \alpha^3 m_2$
2. $\alpha^{13} m_1 + \alpha^3 m_3 = m_{F2} + \alpha^{13} m_2$

- This node can decode m_1 and m_2 because there are 2 independent equations in 2 unknowns.
- By Gaussian rwo reduction $\begin{bmatrix} \alpha^{10} & \alpha^7 \\ \alpha^{13} & \alpha^3 \end{bmatrix} \rightarrow \begin{bmatrix} \alpha^0 & \alpha^{12} \\ 0 & \alpha^{12} \end{bmatrix} \rightarrow \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$

— The node that received m_1 and missed m_2 and m_3 has the equations:

1. $\alpha^3 m_2 + \alpha^7 m_3 = m_{F1} + \alpha^{10} m_1$
2. $\alpha^{13} m_2 + \alpha^3 m_3 = m_{F2} + \alpha^{13} m_1$

- This node can decode m_1 and m_2 because there are 2 independent equations in 2 unknowns.
- By Gaussian rwo reduction $\begin{bmatrix} \alpha^3 & \alpha^7 \\ \alpha^{13} & \alpha^3 \end{bmatrix} \rightarrow \begin{bmatrix} \alpha^0 & \alpha^4 \\ 0 & \alpha^6 \end{bmatrix} \rightarrow \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$

- Actually determining the missing messages requires solving 2 simultaneous equations in 2 unknowns for each 4 bits of the message.

— For instance:
For 4 bits of the message a node has $m_3 = (1011) = \alpha^7$, $m_{F1} = (1111) = \alpha^{12}$, and $m_{F2} = (1100) = \alpha^6$.

— The simultaneous equations are: $\begin{bmatrix} \alpha^{10} & \alpha^3 \\ \alpha^{13} & \alpha^{13} \end{bmatrix} \begin{bmatrix} m_1 \\ m_2 \end{bmatrix} = \begin{bmatrix} m_{F1} + \alpha^7 m_3 \\ m_{F2} + \alpha^3 m_3 \end{bmatrix} = \begin{bmatrix} \alpha^5 \\ \alpha^7 \end{bmatrix}$

— By Gaussian row reduction: $\left[\begin{array}{cc|c} \alpha^{10} & \alpha^3 & \alpha^5 \\ \alpha^{13} & \alpha^{13} & \alpha^7 \end{array}\right] \rightarrow \left[\begin{array}{cc|c} 1 & \alpha^8 & \alpha^{10} \\ 0 & 1 & \alpha^{11} \end{array}\right] \rightarrow \left[\begin{array}{cc|c} 1 & 0 & \alpha^2 \\ 0 & 1 & \alpha^{11} \end{array}\right]$

3.6 Home Work

1. Consider 6 nodes on the corners of a hexagon, and a forwarding node in the center. Each node on a corner transmits a message, through the forwarding node to a node on the opposite edge of a diagonal. The transmission is overheard by the two nearest nodes in the circumference of the hexagon.

   A. Find the smallest set of forwarded messages that consist of the binary sum of messages and describe how the receivers decode the messages that they must receive.

   B. Repeat part A using random codes on a higher order field with 16 elements.

2. A receiver is missing messages m_1 and m_2 and has message m_3. A source sends two messages, m_{s1} and m_{s2} whose 4-bit nibbles are $m_{s1} = \alpha^{12}m_1 + \alpha^5 m_2 + \alpha^9 m_3$ and $m_{s2} = \alpha^9 m_1 + \alpha^9 m_2 + \alpha^5 m_3$

   Where the operations are on the Galois field, $GF(2^4)$ generated with the promitive polynomial $p(x) = x^4 + x + 1$. (The correspondence between the exponential and binary representation of the field elements is given in the notes.)

   The i^{th} nibble of $m_3(i) = \alpha^1$, the i^{th} nibbles of the received messages are $m_{s1}(i) = \alpha^{13}$, and $m_{s2}(i) = \alpha^8$.

   What are the i^{th} nibbles $m_1(i)$ and $m_2(i)$?

REFERENCES

[1] R. Koetter, M. Medard, "Beyond Routing: An Algebraic Approach to network Coding," Infocom 2002, 23-27 June 2002, pp. 122 - 130.

[2] S. Katti, H. Rahul, W. Katabi, M. Medard, J. Crowcroft, "XORs in The Air: Practical Wireless Network Coding," SigComm 2006, Sept. 11-15, 2006, Pisa, Italy.

[3] W. W. Peterson, E. J. Weldon, **Error-Correcting Codes**, M.I.T. Press, 1972.