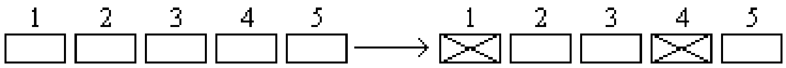
CS 70 Discrete Mathematics for CS
Fall 2006 Papadimitriou & Vazirani **Lecture 13**

Error Correcting Codes

Erasure Errors

We will consider two situations in which we wish to transmit information on an unreliable channel. The first is exemplified by the internet, where the information (say a file) is broken up into packets, and the unreliability is manifest in the fact that some of the packets are lost during transmission, as shown below:



Suppose that the message consists of n packets and suppose that k packets are lost during transmission. We will show how to encode the initial message consisting of n packets into a redundant encoding consisting of $n+k$ packets such that the recipient can reconstruct the message from any n received packets. Note that in this setting the packets are labelled and thus the recipient knows exactly which packets were dropped during transmission.

We can assume without loss of generality that the contents of each packet is a number modulo q, where q is a prime. This is because the contents of the packet might be a 32-bit string and can therefore be regarded as a number between 0 and $2^{32}-1$. Thus we can choose q to be any prime larger than 2^{32}. The properties of polynomials over $GF(q)$ (i.e. with coefficients and values reduced modulo q) are perfectly suited to solve this problem and are the backbone to this error-correcting scheme. To see this, let us denote the message to be sent by $m_1,\ldots,m_n$ and make the following crucial observations:

1) There is a unique polynomial $P(x)$ of degree $n-1$ such that $P(i)=m_i$ for $1\le i\le n$ (i.e. $P(x)$ contains all of the information about the message, and evaluating $P(i)$ gives the contents of the *i-th* packet).

2) The message to be sent is now $m_1=P(1),\ldots,m_n=P(n)$. We can generate additional packets by evaluating $P(x)$ at points $n+j$ (remember, our transmitted message must be redundant, i.e. it must contain more packets than the original message to account for the lost packets). Thus the transmitted message is $c_1=P(1),c_2=P(2),\ldots,c_{n+k}=P(n+k)$. Since we are working modulo q, we must make sure that $n+k\le q$, but this condition does not impose a serious constraint since q is very large.

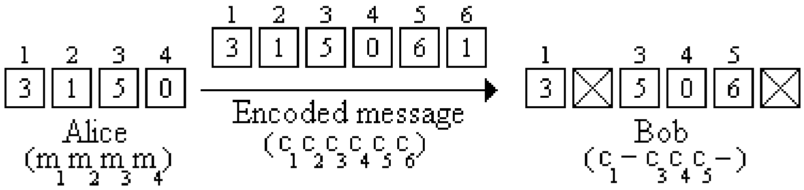
3) We can uniquely reconstruct $P(x)$ from its values at any n distinct points, since it has degree $n-1$. This means that $P(x)$ can be reconstructed from any n of the transmitted packets. Evaluating this reconstructed polynomial $P(x)$ at $x=1,\ldots,n$ yields the original message $m_1,\ldots,m_n$.

Example

Suppose we are working over $GF(7)$ (i.e. all coefficients and numbers can take on values between 0 and 6), and the number of packets in the message is $n=4$. Suppose the message that Alice wants to send to Bob is $m_1=3$, $m_2=1$, $m_3=5$, and $m_4=0$. The unique degree $n-1=3$ polynomial described by these 4 points is $P(x)=x^3+4x^2+5$ (verify that $P(i)=m_i$ for $1\le i\le 4$).

Now, suppose that Alice wishes to guard against $k=2$ lost packets. In order to do this, she must evaluate

$P(x)$ at 2 extra points: $P(5) = 6$ and $P(6) = 1$. Now, Alice can transmit the encoded message which consists of $n+k = 6$ packets, where $c_j = P(j)$ for $1 \leq j \leq 6$. So $c_1 = P(1) = 3$, $c_2 = P(2) = 1$, $c_3 = P(3) = 5$, $c_4 = P(4) = 0$, $c_5 = P(5) = 6$, and $c_6 = P(6) = 1$. Suppose packets 2 and 6 are dropped, in which case we have the following situation:



From the values that Bob received (3, 5, 0, and 6), he uses Lagrange interpolation and computes the following delta functions:

$$\begin{aligned}
\Delta_1(x) &= \frac{(x-3)(x-4)(x-5)}{-24} \\
\Delta_3(x) &= \frac{(x-1)(x-4)(x-5)}{4} \\
\Delta_4(x) &= \frac{(x-1)(x-3)(x-5)}{-3} \\
\Delta_5(x) &= \frac{(x-1)(x-3)(x-4)}{8}.
\end{aligned}$$

He then reconstructs the polynomial $P(x) = (3)\Delta_1(x) + (5)\Delta_3(x) + (0)\Delta_4(x) + (6)\Delta_5(x) = x^3 + 4x^2 + 5$. Bob then evaluates $m_2 = P(2) = 1$, which is the packet that was lost from the original message. We can extend this example to a more general case and say that, no matter which 2 packets were dropped, Bob still could've reconstructed $P(x)$ and thus the original message, and that is because of the remarkable properties of polynomials over $GF(q)$.

Let us consider what would happen if Alice sent one fewer packet. If Alice only sent c_j for $1 \leq j \leq n+k-1$, then with k erasures, Bob would only receive c_j for $n-1$ distinct values j. Thus, Bob would not be able to reconstruct $P(x)$ (since there are exactly q polynomials of degree at most $n-1$ that agree with the $n-1$ packets which Bob received). This error-correcting scheme is therefore optimal—it can recover the n characters of the transmitted message from any n received characters.

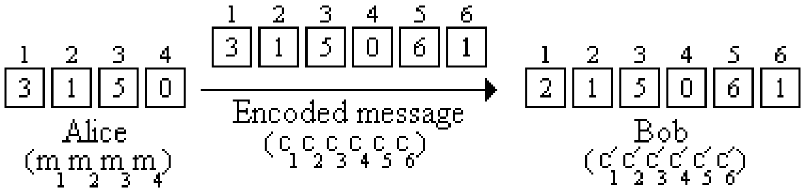
General Errors

Let us now consider a much more challenging scenario. Now Alice wishes to communicate with Bob over a noisy channel (say via a modem). Her message is $m_1, \ldots, m_n$, where we will think of the m_i's as characters (either bytes or characters in the English alphabet). The problem now is that some of the characters are corrupted during transmission due to channel noise. So Bob receives exactly as many characters as Alice transmits. However, k of them are corrupted, and Bob has no idea which k. Recovering from such general errors is much more challenging than erasure errors, though once again polynomials hold the key.

Let us again think of each character as a number modulo q for some prime q (for the English alphabet q is some prime larger than 26, say $q = 29$). As before, we can describe the message by a polynomial $P(x)$ of degree $n-1$ over $GF(q)$, such that $P(1) = m_1, \ldots, P(n) = m_n$. As before, to cope with the transmission errors Alice will transmit additional characters obtained by evaluating $P(x)$ at additional points. To guard against k general errors, Alice must transmit $2k$ additional characters (as opposed to just k additional packets, which was the case with erasure errors). Thus the encoded message is $c_1, \ldots, c_{n+2k}$ where $c_j = P(j)$ for

$1 \le j \le n+2k$, and $n+k$ of these characters that Bob receives are uncorrupted. As before, we must put the mild constraint on q that it be large enough so that $q \ge n+2k$.

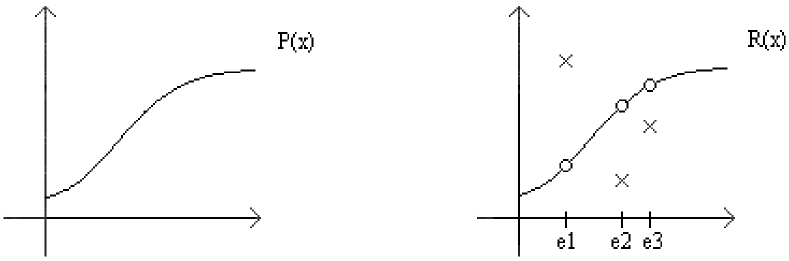
For example, if Alice wishes to send $n = 4$ characters to Bob via a modem in which $k = 1$ of the characters is corrupted, she must redundantly send an encoded message consisting of 6 characters. Suppose she wants to transmit the same message as above, and that c_1 is corrupted and changed to $c_1' = 2$. This scenario can be visualized in the following figure:



From Bob's viewpoint, the problem of reconstructing Alice's message is the same as reconstructing the polynomial $P(x)$ from the $n+2k$ received characters $R(1), R(2), \ldots, R(n+2k)$. In other words, Bob is given $n+2k$ values modulo q, $R(1), R(2), \ldots, R(n+2k)$, with the promise that there is a polynomial $P(x)$ of degree $n-1$ over $GF(q) : R(i) = P(i)$ for $n+k$ distinct values of i between 1 and $n+2k$. Bob must reconstruct $P(x)$ from this data (in the above example, $n+k = 5$ and $R(2) = P(2) = 1$, $R(3) = P(3) = 5$, $R(4) = P(4) = 0$, $R(5) = P(5) = 6$, and $R(6) = P(6) = 1$).

But does Bob have sufficient information to reconstruct $P(x)$? Our first observation shows that the answer is yes. There is a unique polynomial that agrees with $R(x)$ at $n+k$ points. Suppose that $P'(x)$ is any polynomial of degree $n-1$ that agrees with $R(x)$ at $n+k$ points. Then among these $n+k$ points there are at most k errors, and therefore on at least n points x_i: $P'(x_i) = P(x_i)$. But a polynomial of degree $n-1$ is uniquely defined by its values at n points, and therefore $P(x) = P'(x)$ (for all x).

But how can Bob quickly find such a polynomial? The issue at hand are the locations of the k errors. Let $e_1, \ldots, e_k$ be the k locations at which errors occurred. Note that $P(e_i) \ne R(e_i)$ for $1 \le i \le k$:



We could try to guess where the k errors lie, but this would take too long (it would take exponential time, in fact). Consider the error-locator polynomial $E(x) = (x-e_1)(x-e_2)\cdots(x-e_k)$, which has degree k (since x appears k times).

Let us make a simple but crucial observation: $P(i)E(i) = R(i)E(i)$ for $1 \le i \le n+2k$ (this is true at points i at which no error occurred since $P(i) = R(i)$ and trivially true at points i at which an error occurred since $E(i) = 0$). This observation forms the basis of a very clever algorithm invented by Berlekamp and Welch. Looking more closely at these equalities, we will show that they are $n+2k$ linear equations in $n+2k$ unknowns, from which the locations of the errors and coefficients of $P(x)$ can be easily deduced.

Let $P(x)E(x) = Q(x)$, which is a polynomial of degree $n+k-1$, and is therefore described by $n+k$ coefficients. The error-locator polynomial $E(x) = (x-e_1)\cdots(x-e_k)$ has degree k and is described by $k+1$

coefficients, but the leading coefficient (coefficient of x^k) is always 1. So we have:

$$Q(x) = a_{n+k-1}x^{n+k-1} + \cdots + a_1x + a_0$$
$$E(x) = x^k + b_{k-1}x^{k-1} + \cdots + b_1x + b_0$$

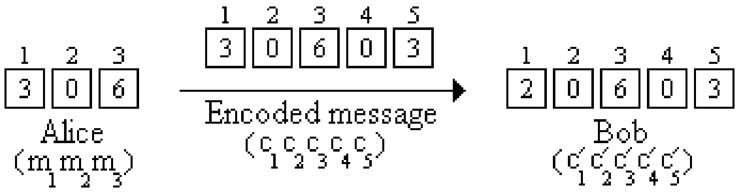
Once we fix a value i for x, the received value $R(i)$ is fixed. Also, $Q(i)$ is now a linear function of the $n+k$ coefficients $a_{n+k-1}\ldots a_0$, and $E(x)$ is a linear function of the k coefficients $b_{k-1}\ldots b_0$. Therefore the equation $Q(i) = R(i)E(i)$ is a linear equation in the $n+2k$ unknowns $a_{n+k-1},\ldots,a_0$ and $b_{k-1},\ldots,b_0$. We thus have $n+2k$ linear equations, one for each value of i, and $n+2k$ unknowns. We can solve these equations and get $E(x)$ and $Q(x)$. We can then compute the ratio $\frac{Q(x)}{E(x)}$ to obtain $P(x)$.

Example. Suppose we are working over $GF(7)$ and Alice wants to send Bob the $n = 3$ characters "3," "0," and "6" over a modem. Turning to the analogy of the English alphabet, this is equivalent to using only the first 7 letters of the alphabet, where $a = 0,\ldots,g = 6$. So the message which Alice wishes for Bob to receive is "dag". Then Alice interpolates to find the polynomial

$$P(x) = x^2 + x + 1,$$

which is the unique polynomial of degree 2 such that $P(1) = 3$, $P(2) = 0$, and $P(3) = 6$.

Suppose that $k = 1$ character is corrupted, so she needs to transmit the $n+2k = 5$ characters $P(1) = 3$, $P(2) = 0$, $P(3) = 6$, $P(4) = 0$, and $P(5) = 3$ to Bob. Suppose $P(1)$ is corrupted, so he receives 2 instead of 3 (i.e. Alice sends the encoded message "dagad" but Bob instead receives "cagad"). Summarizing, we have the following situation:



Let $E(x) = (x - e_1)$ be the error-locator polynomial—remember, Bob doesn't know what e_1 is yet since he doesn't know where the error occurred—and let the degree 3 polynomial $Q(x) = R(x)E(x)$ for $x = 1$ to 5. Now Bob just substitutes $x = 1, x = 2, \ldots, x = 5$ to get five linear equations in five unknowns (recall that we are working modulo 7 and that $R(i) = c_i'$ is the value Bob received for the *i-th* character):

$$\begin{aligned}
a_3 + a_2 + a_1 + a_0 + 5b_0 &= 2\\
a_3 + 4a_2 + 2a_1 + a_0 &= 0\\
6a_3 + 2a_2 + 3a_1 + a_0 + b_0 &= 4\\
a_3 + 2a_2 + 4a_1 + a_0 &= 0\\
6a_3 + 4a_2 + 5a_1 + a_0 + 4b_0 &= 1
\end{aligned}$$

Bob then solves this linear system and finds that $a_3 = 1$, $a_2 = 0$, $a_1 = 0$, $a_0 = 6$, and $b_0 = -e_1 = -1 \equiv 6 \mod 7$ (as a check, we expect e_1 to be the index at which the error occurred. Indeed, this is the case since the first character was corrupted from a "d" to a "c"). This gives him the polynomials $Q(x) = (x^3+6)$ and $E(x) = (x-1)$. He can then find $P(x)$ by computing the quotient $P(x) = \frac{Q(x)}{E(x)} = \frac{x^3+6}{x-1} = x^2+x+1$. Bob notices that the first character was corrupted (since $e_1 = 1$), so now that he has $P(x)$, he just computes $P(1) = 3 =$ "d" and obtains the original, uncorrupted message "dag".

Finer Points

Two points need further discussion. How do we know that the $n+2k$ equations are consistent? What if they have no solution? This is simple. The equations must be consistent since $Q(x) = P(x)E(x)$ together with the error locator polynomial $E(x)$ gives a solution.

A more interesting question is this: how do we know that the $n+2k$ equations are independent, i.e., how do we know that there aren't other spurious solutions in addition to the real solution that we are looking for? Put more mathematically, how do we know that the solution $Q'(x)$ and $E'(x)$ that we reconstruct satisfy the property that $E'(x)$ divides $Q'(x)$ and that $\frac{Q'(x)}{E'(x)} = \frac{Q(x)}{E(x)} = P(x)$? To see this notice that $Q(x)E'(x) = Q'(x)E(x)$ for $1 \leq x \leq n+2k$. This holds trivially whenever $E(x)$ or $E'(x)$ is 0, and otherwise it follows from the fact that $\frac{Q'(x)}{E'(x)} = \frac{Q(x)}{E(x)} = R(x)$. But the degree of $Q(x)E'(x)$ and $Q'(x)E(x)$ is n + 2k - 1. Since these two polynomials are equal at $n+2k$ points, it follows that they are the same polynomial, and thus rearranging we get that $\frac{Q'(x)}{E'(x)} = \frac{Q(x)}{E(x)} = P(x)$.